Agile hopping robot for lunar exploration targeting data gathering and sample collection

João Francisco Formigão e Santos

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisor(s): Prof. Rodrigo Martins de Matos Ventura

Examination Committee
Chairperson: Prof. José Fernando Alves da Silva
Supervisor: Prof. Rodrigo Martins de Matos Ventura
Member of the Committee: Prof. Rita Maria Mendes de Almeida Correia da Cunha

October 2019
Dedicated to my mother
Acknowledgments

This thesis is the product of one semester of hard work, passion for this project and will to learn and improve. All of this could not have been accomplished without the support and contribution of so many people I found throughout this journey. This section is dedicated to those who contributed to it and to whom I am thankful for.

For proposing and accepting my application to this topic, I have to thank my supervisor Prof. Rodrigo Ventura who believed I could carry every task proposed along these months. His insights and suggestions throughout this project were fundamental to the consistency and success of this thesis.

I would also like to thank my colleagues Francisco Gomes and Rui Correia who work alongside me in ISR and with whom I shared my struggles during these months. Without our long talks, our weeks and weekends at ISR but especially their support, the success of this thesis could not have been accomplished.

For reading and reviewing my entire thesis, I would like to thank my colleague Rute Luz. Her suggestions and support together with her constant smile significantly improved the quality of this work.

For his dedication and helpfulness, I thank Prof André Marta who dedicated his time to discuss the various approaches for developing the equations of motion.

An honest thanks to my family, especially my mom, for providing me with a great education and all the resources I needed to accomplish my objectives. But most importantly, for all the unconditional love and support throughout these years.

Finally, a huge thanks to all my friends who had to put up with my complaining during these months and were able to turn this whining into laughs and joy.
Resumo

Nas últimas décadas, um crescente número de missões espaciais levou à aplicação do conceito de hopping robots para a locomoção autônoma em ambientes de exploração planetária e de pequenos corpos. Em geral a capacidade de salto destes robots permite uma locomoção em terrenos acidentados, com uma grande eficiência.

O maior desafio dos hopping robots está relacionado com impactos com o solo resultantes de aterragens falhadas. Para resolver este problema, esta tese explora a aplicação de um giroscópio de controlo de momento de dois eixos e velocidade variante para o controlo de um hopping robot após descolagem do solo. Durante o tempo de salto, este atuador deve garantir controlo sobre a atitude e velocidade angular de modo que o robô chegue ao solo em segurança. Deve garantir ainda a execução de manobras que permitam a recolha de dados a uma determinada orientação durante um salto.

Para garantir estes objetivos, esta tese apresenta o desenvolvimento do modelo em espaço de estados do robô utilizado para o estudo - o Moonhopper. De seguida, um estudo da controlabilidade é conduzido e três arquiteturas de controladores são apresentadas. Entre as quais, uma estratégia envolvendo otimização de trajetória.

De modo a comparar o desempenho, quatro manobras foram desenhadas para serem seguidas pelos controladores. Os resultados apresentam uma vantagem da estratégia baseada em otimização de trajetória tendo em conta a eficiência. Como controlador final, é proposta uma arquitectura que combina esta otimização com um controlador em anel fechado. No final, são executados testes com este controlador sendo os resultados comparados com os anteriores.

**Palavras-chave:** hopping robots, controlo de atitude, giroscópio de controlo de momento, optimização de trajetória
Abstract

The growing number of space missions has led to the application of the concept of hopping robots for autonomous locomotion in planetary and small-body environments. The strong jumping ability of these robots allows them to move in such complex environments with high efficiency.

The main challenge to this concept is related to impacts with the ground resulting from failed landings. To solve this issue, this thesis explores the application of a double gimbal variable speed control moment gyroscope for attitude control of a hopping robot during a ballistic flight. During flight, this actuator must be able to guarantee an attitude and angular velocity of the robot such that, when the robot hits the ground, it has a safe position. Additionally, in flight, the robot must be able to perform observation maneuvers that require the robot to achieve a certain attitude with zero angular velocity.

To achieve these objectives, this thesis starts with the development of a state-space model of the robot Moonhopper. Then, a broad study of the controllability is performed culminating in the design of three control architectures to be tested. Among these, a strategy using Trajectory Optimization is developed and applied.

To compare the performance of these controllers, four attitude trajectories are designed to be followed by these controlled systems. The results show an advantage of the trajectory optimization-based controllers in terms of energy efficiency. The final proposed strategy is a closed-loop controller that follows these optimized trajectories, combining efficiency with robustness. The results are presented and compared with the previous solutions revealing an advantage in the use of this architecture.

Keywords: hopping robots, attitude control, control moment gyroscope, trajectory optimization
## Contents

Acknowledgments ................................................................. v  
Resumo ........................................................................ vii  
Abstract ........................................................................... ix  
List of Tables ..................................................................... xv  
List of Figures ..................................................................... xvii  
Nomenclature ..................................................................... xix  

1 Introduction ........................................................................ 1  
1.1 Motivation ...................................................................... 1  
1.2 Objectives ...................................................................... 2  
1.3 Major Contributions ....................................................... 2  
1.4 Thesis Outline ............................................................... 3  

2 Background and State of the Art ........................................ 5  
2.1 Hopping Robots ........................................................... 5  
2.1.1 Stages of the jumping cycle ....................................... 5  
2.1.2 Jumping Metrics ....................................................... 7  
2.2 Moonhopper ................................................................... 8  
2.3 Trajectory Optimization ................................................ 10  
2.4 State of the art .............................................................. 12  
2.4.1 Hopping Robots ....................................................... 12  
2.4.2 Actuators .................................................................. 17  
2.4.3 Control Laws for Double Gimbal Variate Speed CMG 19  
2.4.4 Trajectory Optimization .......................................... 20  

3 System Dynamics .............................................................. 23  
3.1 Translation motion .......................................................... 23  
3.2 Rotational motion .......................................................... 25  
3.2.1 Notation .................................................................. 25  
3.2.2 Definitions ............................................................... 25  
3.2.3 Dynamic model ......................................................... 29
# List of Tables

6.1 Structural parameters of the robot ........................................... 48
6.2 Results of real prototype translation ......................................... 49
6.3 Estimated metrics for moon surface ......................................... 50
6.4 Safety landing conditions ...................................................... 57
6.5 Objectives for locomotion test - 1 .......................................... 59
6.6 Objectives for locomotion test - 2 .......................................... 59
6.7 Objectives for observation test - 1 ......................................... 59
6.8 Objectives for observation test - 2 .......................................... 60
6.9 State and input constraints .................................................... 66
6.10 Boundary constraints - Locomotion test 1 ............................... 67
6.11 Boundary constraints - Locomotion test 2 ............................... 68
6.12 Boundary constraints - Observation test 1 ............................. 70
6.13 Boundary constraints - Observation test 2 ............................. 70
6.14 Energy - Trajectory Optimization (Ideal Model) ....................... 75
6.15 Energy - Classic Control (Ideal Model) ................................ 75
6.16 Energy - Trajectory Optimization (Friction Model) .................. 75
6.17 Energy - Classic Control (Friction Model) ............................. 76
6.18 Summary of the comparison of the three controllers in Observation maneuver 2 ........................................ 78

D.1 Ideal conditions at take off .................................................. 94
D.2 Hyperparameters .............................................................. 95
List of Figures

2.1 The Moonhopper ................................................. 9
2.2 Flight with oscillatory motion of pole, using one traction foot ............... 12
2.3 Raibert’s planar hopping machines .................................. 13
2.4 Three generations of hopping robots .................................. 14
2.5 First and second generations Minimalist hopping robots ..................... 14
2.6 Phobos Hopper, PROP-F, model ..................................... 16
2.7 Hopping Robot Minerva ............................................ 16
2.8 Concept operation of the proposed hopping robot .......................... 18
2.9 Shooting methods vs Direct collocation methods .......................... 22

3.1 Detail of Moonhopper leg .......................................... 23
3.2 Representation of the DGVSCMG .................................... 26

4.1 Open Loop Architecture ........................................... 41
4.2 Attitude hold architecture .......................................... 41
4.3 Nonlinear control architecture ...................................... 42

6.1 Results from MOCAP position and velocity of Moonhopper ................ 49
6.2 $u_W$ step test - Input torques .................................... 52
6.3 $u_W$ step test - Orientation in Euler Angles ............................ 52
6.4 $u_W$ step test - Angular velocity .................................. 52
6.5 $u_W$ step test - Joints rates ....................................... 53
6.6 $u_W$ step test - Gimbal axes angles ................................ 53
6.7 Outer gimbal-Inner gimbal step test - Orientation in Euler Angles ........ 53
6.8 Outer gimbal-Inner gimbal step test - Angular velocity .................... 54
6.9 Outer gimbal-Inner gimbal step test - Joints rates ........................ 54
6.10 Outer gimbal-Inner gimbal step test - Gimbal axes angles ............... 54
6.11 Outer gimbal-Inner gimbal step test - Gimbal axes angles ............... 55
6.12 Angular momentum vector variation .................................. 56
6.13 Kinetic Energy - State space model .................................. 56
6.14 Locomotion vs Observation ......................................... 58
6.15 Locomotion test 1: Classic attitude controller - Attitude and angular velocity .. 60
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.16</td>
<td>Locomotion test 1: Classic attitude controller - Gimbal Axes and Wheel velocity</td>
<td>61</td>
</tr>
<tr>
<td>6.17</td>
<td>Locomotion test 1: Classic attitude controller - Gimbal inputs</td>
<td>61</td>
</tr>
<tr>
<td>6.18</td>
<td>Locomotion test 1: Classic attitude controller - Wheel inputs</td>
<td>61</td>
</tr>
<tr>
<td>6.19</td>
<td>Locomotion test 2: Classic attitude controller - Attitude and angular velocity</td>
<td>62</td>
</tr>
<tr>
<td>6.20</td>
<td>Locomotion test 2: Classic attitude controller - Gimbal Axes and Wheel velocity</td>
<td>62</td>
</tr>
<tr>
<td>6.21</td>
<td>Locomotion test 2: Classic attitude controller - Gimbal inputs</td>
<td>63</td>
</tr>
<tr>
<td>6.22</td>
<td>Locomotion test 2: Classic attitude controller - Wheel inputs</td>
<td>63</td>
</tr>
<tr>
<td>6.23</td>
<td>Observation test 1: Classic attitude controller - Attitude and angular velocity</td>
<td>64</td>
</tr>
<tr>
<td>6.24</td>
<td>Observation test 1: Classic attitude controller - Gimbal Axes and Wheel velocity</td>
<td>64</td>
</tr>
<tr>
<td>6.25</td>
<td>Observation test 1: Classic attitude controller - Gimbal inputs</td>
<td>64</td>
</tr>
<tr>
<td>6.26</td>
<td>Observation test 1: Classic attitude controller - Wheel inputs</td>
<td>65</td>
</tr>
<tr>
<td>6.27</td>
<td>Observation test 2: Classic attitude controller - Attitude and angular velocity</td>
<td>65</td>
</tr>
<tr>
<td>6.28</td>
<td>Observation test 2: Classic attitude controller - Gimbal Axes and Wheel velocity</td>
<td>65</td>
</tr>
<tr>
<td>6.29</td>
<td>Observation test 2: Classic attitude controller - Gimbal inputs</td>
<td>66</td>
</tr>
<tr>
<td>6.30</td>
<td>Observation test 2: Classic attitude controller - Wheel inputs</td>
<td>66</td>
</tr>
<tr>
<td>6.31</td>
<td>Locomotion test 1: Trajectory optimization - Attitude and angular velocity</td>
<td>68</td>
</tr>
<tr>
<td>6.32</td>
<td>Locomotion test 1: Trajectory optimization - Gimbal Axes and Wheel velocity</td>
<td>68</td>
</tr>
<tr>
<td>6.33</td>
<td>Locomotion test 1: Trajectory optimization - Gimbal inputs</td>
<td>69</td>
</tr>
<tr>
<td>6.34</td>
<td>Locomotion test 1: Trajectory optimization - Wheel inputs</td>
<td>69</td>
</tr>
<tr>
<td>6.35</td>
<td>Locomotion test 2: Trajectory optimization - Attitude and angular velocity</td>
<td>69</td>
</tr>
<tr>
<td>6.36</td>
<td>Locomotion test 2: Trajectory optimization - Gimbal Axes and Wheel velocity</td>
<td>70</td>
</tr>
<tr>
<td>6.37</td>
<td>Locomotion test 2: Trajectory optimization - Gimbal inputs</td>
<td>70</td>
</tr>
<tr>
<td>6.38</td>
<td>Locomotion test 2: Trajectory optimization - Wheel inputs</td>
<td>71</td>
</tr>
<tr>
<td>6.39</td>
<td>Observation test 1: Trajectory optimization controller - Attitude and angular velocity</td>
<td>71</td>
</tr>
<tr>
<td>6.40</td>
<td>Observation test 1: Trajectory optimization controller - Gimbal Axes and Wheel velocity</td>
<td>72</td>
</tr>
<tr>
<td>6.41</td>
<td>Observation test 1: Trajectory optimization controller - Gimbal inputs</td>
<td>72</td>
</tr>
<tr>
<td>6.42</td>
<td>Observation test 1: Trajectory optimization - Wheel inputs</td>
<td>72</td>
</tr>
<tr>
<td>6.43</td>
<td>Observation test 2: Trajectory optimization controller - Attitude and angular velocity</td>
<td>73</td>
</tr>
<tr>
<td>6.44</td>
<td>Observation test 2: Trajectory optimization - Gimbal Axes and Wheel velocity</td>
<td>73</td>
</tr>
<tr>
<td>6.45</td>
<td>Observation test 2: Trajectory optimization - Gimbal inputs</td>
<td>73</td>
</tr>
<tr>
<td>6.46</td>
<td>Observation test 2: Trajectory optimization - Wheel inputs</td>
<td>74</td>
</tr>
<tr>
<td>6.47</td>
<td>Observation test 2: Lyapunov + Trajectory optimization controller - Attitude and angular velocity</td>
<td>76</td>
</tr>
<tr>
<td>6.48</td>
<td>Observation test 2: Lyapunov + Trajectory optimization - Gimbal Axes and Wheel velocity</td>
<td>77</td>
</tr>
<tr>
<td>6.49</td>
<td>Observation test 2: Lyapunov + Trajectory optimization - Gimbal inputs</td>
<td>77</td>
</tr>
<tr>
<td>6.50</td>
<td>Observation test 2: Lyapunov + Trajectory optimization - Wheel inputs</td>
<td>77</td>
</tr>
<tr>
<td>D.1</td>
<td>Reaction Force as function of leg compression</td>
<td>94</td>
</tr>
</tbody>
</table>
Nomenclature

Acronyms

CAD  Computer-aided design.
CMG  Control Moment Gyroscope.
CoM  Center of Mass.
DGVSCMG  Double Gimbal Variable Speed Control Moment Gyroscope.

Greek symbols

$\eta$  Jump efficiency.
$\Omega$  Spinning Wheel velocity.
$\omega$  Angular velocity.
$\Psi$  State boundaries.
$\psi$  Rotation angle of the outer gimbal.
$\rho$  Path Constraints.
$\theta$  Rotation angle of the inner gimbal.

Others

$(\cdot, i)$  Matrix column i.
$(i, :)$  Matrix line i.
$\hat{g}_i$  Unit vector in frame G oriented with the i-axis.
$\mathbf{S}()$  Cross product operation.
$\mathbf{N}d/dt$  Time derivative in frame N.
$[\mathbf{F}\mathbf{G}]$  Direction cosine matrix of the frame F relative to the frame G.
$[\mathbf{I}_A]$  Tensor of inertia of body A.

Roman symbols
a  Upper leg length.
b  Lower leg length.
c  Distance between robot foot gears.
g  Gravitational acceleration.
H  Angular velocity.
h  Height.
h_{max}  Maximum height of a jump.
J  Boundary objective function.
k  Spring constant.
l_0  Spring length at rest.
m_{robot}  Robot mass.
q  Quaternion unit vector.
t_F  Duration of a jump.
T_r  Kinetic energy rotation.
T_t  Kinetic energy translation.
W  Work of resultant force - ideal.
w  Path objective function.
W_{TO}  Work of resultant force - real at take off.
x_{max}  Maximum horizontal distance of a jump.
u  Input vector.
v  Tangential velocity.
z  Decision variables.
x  State variables vector.

Subscripts

1, 2, 3  Frame axis (x,y,z).
B  Quantity relative to body B - robot body.
F/N  Rotation of frame F with respect to frame N.
G  Quantity relative to body G - outer gimbal.
\( G_c \) Quantity relative to centered rotation of body G.

\( G_{off} \) Quantity relative to an off centered rotation of body G.

\( H \) Quantity relative to body H - inner gimbal.

\( L \) Quantity relative to body L - robot legs and foot.

\( LBS \) Quantity relative to body LBS - robot structure.

\( S \) Quantity relative to body S - CMG holder.

\( W \) Quantity relative to body W - spinning wheel.

**Superscripts**

\( F \) Quantity represented in frame F.

\( G \) Quantity represented in frame G.

\( H \) Quantity represented in frame H.

\( N \) Quantity represented in frame N.

\( T \) Transpose.
Chapter 1

Introduction

1.1 Motivation

Jumping or leaping is a form of locomotion or movement in which an organism or non-living mechanical system propels itself through the air along a ballistic trajectory \[1\]. Both jumping and running can be described by the previous definition. The main difference between both resides in the duration of the aerial phase and the takeoff angle of the jump. Inside the concept of jumping there is hopping. According to the dictionary \[2\], hopping can be applied to describe the movement of jumping on one foot or be used to characterize the type of locomotion of small animals such as grasshoppers, frogs or rabbits. In both definitions, the hop has the connotation of a small jump which is part of a high-frequency sequence of small jumps. When applied to robotics, the concept of hopping robots is used to describe robots that present jumping capabilities. Usually inspired by these animals, examples of such robots can be found in \[3\], and \[4\]. All of these robots have in common this ability of locomotion.

As stated in \[5\], \[4\] hopping or jumping robots have the ability to cross obstacles several times its height. Some of them can jump a distance of up to 30 times its body size \[5\], without any risk associated with its integrity. The strong jumping ability of this type of robot makes it possible to move in a complex environment with large obstacles \[3\]. All of this is accomplished with higher efficiency when compared to wheeled robots \[4\]. As a consequence, these robots have potential application value in many fields such as disaster rescue and military reconnaissance \[5\].

With the recent trend toward more frequent space missions to Mars and other celestial bodies, such as moons, asteroids, and comets, \[6\] the field of hopping robotics achieved a new application. These celestial bodies are usually characterized by a low to medium gravitational environment and unstructured terrain \[6\]. For exploration in such complex scenarios, an interest in multi-functional vehicles, capable of providing high mobility for scientific packages has sparked the interest of the space exploration community in this field. Examples such as in \[4\] propose the usage of multiple small hopping robots to perform simple distributed tasks, previously designed for the single-vehicle paradigm. Another example is found in the recent asteroid sample-return mission Hayabusa 2. This mission carried four small rovers to investigate the asteroid surface \textit{in situ}. Two of them move by hopping in the low gravitational field, using a
torque generated by rotating masses within the rovers [7]. The success of this recent mission presents one of the biggest contributions to this field.

In order to perform as desired, the hopping robot requires a robust control system responsible for ensuring the integrity and the continuity of the robot’s operations. This control system can be active or passive. Passive control is performed with no actuation and relies on the body configuration to regain its original position before the jump [8]. Active control is performed using attitude actuation to control the robot orientation [9]. Inspired by satellite attitude actuators, these active control systems can be performed using jet propulsion or momentum exchange devices, such as reaction-wheels [9]. The main advantage of the last is the increased lifespan of operations as they do not rely on a limited reserve of chemical propellant. Momentum exchange systems can be designed to work with electrical energy collected by solar panels. For this reason, the efficient design of the control system and trajectory planning is of extreme importance for mission sustainability.

1.2 Objectives

The objective of this thesis is to study the solution for an attitude controller based on the use of Double Gimbal Variable Speed Control Moment Gyroscope (DGVSCMG). This study is based on the application of the actuation system on a hopping robot prototype developed at ISR (Institute for Systems and Robotics): the Moonhopper. Thus, the objectives to be achieved by this thesis are:

**Objective 1**: Develop a state space model of the Moonhopper and validate the results;

**Objective 2**: Design and implement a control system for the DGCVSCMG actuated hopping robot.

**Objective 3**: Investigate and apply trajectory optimization methods for attitude trajectory planning

1.3 Major Contributions

The major contributions of this thesis are:

1. The development of an attitude and position model of a DGVSCMG actuated hopping robot. Aside from an hopping robot, the attitude model is expected to be applied to describe the attitude dynamics of a small DGVSCMG actuated satellite.

2. The development of a simulation of the Moonhopper prototype in Gazebo/ROS framework. This simulation is able to receive control torque inputs and output the expected states

3. A study of the controllability properties of the attitude dynamics and a study and application of several attitude control strategies for control of the robot orientation after take off.

4. A study of the application of trajectory optimization methods to attitude control of hopping robots. A discussion of this process and respective implementation is found in this work.
1.4 Thesis Outline

Chapter 2 Background and State of the Art presents a brief review of the background and the state of the art. Along with the background, a description of the Moonhopper is provided together with the parameters that characterize this model.

Chapter 3 System Dynamics starts with the presentation of the position and velocity model of the robot after takeoff. The second part of this chapter is dedicated to the construction of the state space model for the attitude dynamics of the actuated hopping robot.

Chapter 4 Controllability, Singularities and Control Architecture presents a study of the controllability properties of the system described in the previous chapter, accounting with the singularities presented in the actuation. In the end, it presents the control architectures implemented in this work.

Chapter 5 Trajectory Optimization is dedicated to the application of trajectory optimization to the control problem described earlier. The chosen method is described and the trajectory optimization problem presented.

Chapter 6 Results and Discussion reports the results of the various experiments conducted in this thesis. It is divided into three parts: characterization of the translation motion, validation of the attitude model and testing the control systems.

Chapter 7 Conclusions presents the conclusions of this thesis and a few suggestions for future work.
Chapter 2

Background and State of the Art

Chapter 2 is divided into two parts: (1) the Background, which comprises an analysis of the main concepts of this research, and (2) the State of the Art, which presents the latest stage developments in this topic. The purpose of this section is to provide insights into the matters in which this thesis is supported. The main topics explored in this chapter comprise hopping robots, attitude control and trajectory optimization.

2.1 Hopping Robots

2.1.1 Stages of the jumping cycle

In general, the jumping process of a robot can be divided into three stages: takeoff, air posture adjustment, and landing [5]. This segmentation of the process is useful for control and actuation purposes. By separating into three stages the control can be applied separated for each phase. This way the takeoff phase will have a proper actuation designed to maximize the jump height and/or horizontal distance. The air posture will be designed to efficiently control the attitude of the robot while in flight. Finally, the landing buffer will be designed to protect the robot from the impact with the ground.

The takeoff stage starts when the robot is in equilibrium on the ground and ends when the robot takeoff from the ground. It is during this phase that occurs the conversion of potential energy (stored in the actuator) into kinetic energy. The takeoff process can determine the takeoff speed, jumping height, and jumping distance. In general, these quantities depend on the amount of potential energy that is converted and the amount of energy that is lost.

The design of the takeoff actuator must take into account the main purpose of the robot. The following list presents the most common types of actuators found in literature and a brief description of each mechanism.

Reazione-propulsion Drive: The first hopping robots to be explored made use of chemical propellant to perform the jump. In general, these robots are propelled as result of chemical reaction performed in its take off mechanism. Examples of this are found in [10] and [11].
**Pneumatic Drive:** Pneumatic drive hopping robots make use of a series of pneumatic structures to accomplish the hop. In general, these pneumatic actuators are used to mimic the behavior of the biological musculoskeletal system of certain mammals [5]. These robots are characterized by a good compliance, large driven forces, powerful and highly dynamic. An example of this is the robot Mowgly [12].

**Spring Drive:** Spring driven hopping robots achieve vertical hopping motions from the release of a spring system [4]. The spring has the advantage of high energy storage, fast energy release, simple structure, and simple control. Spring driven robots are more frequent than any other type [5]. Examples of such are [3], [4], [13], .

**Flexible Material Drive:** The principle behind these robots is to take advantage of elastic properties of certain materials such as fiberglass [14] or shape memory alloy [15] to store potential energy to perform the jump. So far, only small robots are driven by this type of hopping system. This is because the takeoff phase of large robots requires an amount of stored energy larger than the rupture point of these materials.

**Momentum Exchange Drive:** An example of such is found in [7] with hopping robot Minerva. Using a torquer (rotating wheel) inside the robot, this actuator starts rotating and creates a reaction force against the surface that makes the robot hop with a significant horizontal velocity. This results in a small jumping height (on Earth’s surface), and for that reason it was only applied to locomotion on asteroids.

Some robots also combine two of these characteristics. With the advance of material science it is common to see the application of flexible materials to other types of takeoff mechanisms. An example is found in the robot Spacebok [16].

The second stage of the cycle corresponds to the air posture adjustment in the flight phase. In nature, when the posture in the air is unstable, most of the animals can change their body posture while in the air to achieve a smooth landing [5]. In general, they perform that in one of two ways: using wings to achieve a safe position or swinging other parts of the body, such as spines, tail, and abdomen. The first is more common with birds and insects. The second is used by mammals and reptiles.

These solutions have been recreated to achieve an adequate air posture of bioinspired robots. An example is the recreation of the tail motion of the *Galago senegalensis* in SALTO [3]. This small animal is famous for having the highest vertical jumping ability amongst all mammals and, for this reason, researchers tried to mimic this hopping mechanism in order to improve the mobility characteristics of SALTO.

Solutions for air posture adjustment of non-bioinspired robots can be found in the literature[9][10][11]. In general, these solutions consist of an appropriation of attitude control actuators, usually applied to small satellites. Most of these solutions make use of reaction propulsion mechanisms to perform the jump. However, in [9] the authors propose a solution that consists of the addition of two reaction wheels to the top of the robot. This attitude mechanism is explored in the next section.
Simpler robots can also present no attitude controller. This strategy is found in robots with a special structure design. This special structure design enables the robots to right itself when pushed at an angle just like a roly-poly toy. An example of these robots is found in the first-generation design of [4] and in [8]. To protect their structural integrity, the focus of the design of these robots lies in the landing stage.

Finally, the landing stage is the last stage of the cycle. This phase starts when the robot touches the ground and finishes when it reaches a static equilibrium. In nature, many creatures can achieve a stable landing buffer through real-time control of body postures and muscle forces when falling from high places. According to [5], this is accomplished by anticipatory control over the leg muscles, which provide a mechanically stable landing. This strategy was implemented in several hopping robots such as the kangaroo robot [12]. A common practice is to use the same mechanism responsible for takeoff to perform the safe landing. For spring drive robots, the metal hoop springs allow energy to be stored in a stable material that can drive the robot to jump and avoid rigid collision with the ground during landings. When this is not possible (ex. for the majority of reaction propulsion drive robots), usually the focus is on the development of an independent buffering mechanism able to protect the structure from the rigid collision with the ground.

2.1.2 Jumping Metrics

In order to classify and compare the hopping robots mentioned throughout the introduction, some jumping metrics are presented. The purpose of this presentation is to provide some of the characteristics that are commonly used to quantify the performance of a hopping robot.

The trajectory of a general hopping robot after take-off is ballistic (when the aerodynamic drag is negligible). The only force of significance that acts on the robot is gravity, which acts downward, thus imparting to the object a downward acceleration. With a horizontal velocity different from zero in take-off (constant throughout this stage), the result is a parabolic motion trajectory [3]. Since all forces are conservative (or negligible) the total energy of the system is constant. Meaning that during this time, kinetic and potential energy are converted into each other. After reached the top of the ballistic trajectory, where the potential energy reaches the maximum value, the energy is converted into kinetic until the robot reaches the ground. Depending on the horizontal distance from the starting point, \( x_{\text{max}} \), and the maximum height reached during the trajectory it is possible to estimate the total energy of the system. In this document, the letter \( h \) is reserved to represent the height of the robot in the trajectory and \( h_{\text{max}} \) will be used to represent the maximum height.

With only the values of \( h_{\text{max}} \) and \( x_{\text{max}} \), and assuming a ballistic trajectory, it is possible to estimate the take off speed in both directions and the initial kinetic energy. For this reason, they are used as metrics to evaluate the jumping performance. For some hopping robots, it is possible to estimate the initial velocity before the jump using information about the take off actuator state. This property is often found in spring drive robots but it can also be extended to other types with more complex formulas. The idea is that the potential energy created by the contraction of the actuator can be computed as a function of the spring displacement. Due to the existence of friction, not all the potential energy is converted into
kinetic, thus it is introduced the efficiency metric $\eta$. Efficiency is the relation between the potential energy stored in the springs and the kinetic power during take off. Let $T_{TO}$ be the hopper kinetic energy at take off and $T_t$ the energy stored in the compressed member (ideally equal to the kinetic energy at TO), then, the efficiency is given by

$$\eta = \frac{T_{TO}}{T_t} \quad (2.1)$$

The efficiency $\eta$ can also be computed assuming conservation of the total energy and using the two metrics described earlier: $h_{max}$ and $x_{max}$). The Eq. 2.1 is introduced in [4] and it is a useful comparison metric between the three generation robots developed by the authors.

In conclusion, this first part of Chapter 2 presented an overview of the important concepts when working with hopping robots. Recalling the objectives of this thesis, the focus is on the air posture adjustment phase. To understand the conditions and attitude control problem it is required to characterize the whole jumping cycle and all the concepts associated with it. These concepts here presented, are used to characterize the Moonhopper in the next section.

### 2.2 Moonhopper

The Moonhopper is a hopping robot developed by ISR (Institute for Systems and Robotics) with the main objective of serving as a first iteration design for this project. Figure 2.1 presents a picture of this robot. Dividing into four parts, the robot is composed by (from top to bottom):

- **Double Gimbal Control Moment Gyro (DGCMG)** - this device is used as the attitude control actuator. A DGCMG is composed of a spinning rotor and two motorized gimbals (inner and outer gimbals) that tilt the wheel spinning axis. As the axis tilts, the changing angular momentum causes a gyroscopic torque that rotates the body, due to the effects of conservation of angular momentum. The wheel is speed is controlled by a hard drive motor (HDD motor).

- **Robot body - 3D printed**, it serves as a support for the DGCMG, for the electric components, the batteries and the motor responsible for taking off.

- **Robot Legs** - a geared six-bar spring/linkage system similar to the second generation design found in [4]. Figures 2.1 show pictures of its mechanical implementation in both its compressed and uncompressed state. Displacements in the y-direction induce, through the linkage, displacements in the linear spring along the x-direction. As a result, the linkage creates a non-linear spring from a linear spring.

- **Robot foot - also 3D printed** it supports all the other components. It is connected to the robot body through a wire that is locked on this part. To contract the legs for takeoff, this wire is retracted by a rotor through a system of gears. The role of the foot is to maintain the position of the robot throughout this series of events.

Based on section 2.1 and the information about the robot it is concluded that, regarding the take-off
system, the robot is classified as spring-driven. The main reason for this choice lies in the robustness of these systems, compared to other solutions.

An important feature is the springs arrangement. Instead of contracting, as it is more commonly found, these springs stretch when the wire is retracted. The main advantage of this system is described in [4]. In this article, the authors prove the advantages of this arrangement (second generation robot) when compared to a single spring compression (first generation robot). More about this system will be covered in the next section.

To characterize the robot, jumping metrics and other characteristics are calculated. For control purposes the most important parameters are \( h_{\text{max}} \), \( x_{\text{max}} \) and the respective time of flight (duration of jump), the efficiency \( \eta \). The procedure for computing these parameters is described in Chapter 6. Regarding other characteristics, such as the inertia matrices and location of the center of mass (CoM), these are computed through numerical models integrated into computer-aided design (CAD) software, used to design the robot (In this case SolidWorks).

The Moonhopper will serve as a platform for testing the attitude controllers developed with this work. Since the controllers used are classified as model-based, a simulation of this robot was also developed considering the information that resulted from the measurements. The comparison between the simulation and the theoretical model will be presented in the section dedicated to model verification in Chapter 6.

Finally, it should be noted that this robot lacks a take off angle mechanism. As a result, the direction of the jump of this robot will be defined only by the inclination of the platform from which the robot takes off. Other features such as pos landing self-righting mechanisms (introduced in [4], are systems that recover the initial position of the robot, from any configuration resultant from landing) are also not included in this prototype. The reader should recall that this robot is only the first iteration prototype and features such as these will be integrated in following iterations.
2.3 Trajectory Optimization

The term trajectory optimization refers to a set of methods that are used to find the best choice of trajectory, typically by selecting the inputs to the system, known as controls, as functions of time [17]. In this definition, trajectory can mean the trajectory of a spacecraft between two planets, but in general, it refers to the sequence of states during a time interval, starting from an initial state and ending in a desired state.

The origin of the concept trajectory optimization can be found together with the introduction of the Brachystochrone problem [18] as a possible approach to solve it. This is a well known problem where the goal is to find the shape of a wire such that a bead sliding along it will move between two points in the minimum time. The most famous solution to this problem was accomplished using calculus of variations, the concept from which trajectory optimization grew out of.

Trajectory optimization evolved with the field of optimal control, integrating solutions based on the Pontryagin’s maximum principle and solutions that require heavy computational performance.[18] Following this evolution, in the 1950s, much of the work in trajectory optimization was focused on computing rocket thrust profiles and related problem in the aerospace industry. This early research discovered many basic principles that are still used today, and that are presented here and in chapter 4 [18].

According to [19], a trajectory optimization problem can be formulated as a collection of \( N \) phases of trajectory. In general, the independent variable \( t \) for phase \( k \) is defined in the region \( t_0^{(k)} \leq t \leq t_f^{(k)} \). For many applications the independent variable \( t \) is time, and the phases are sequential, that is \( t_0^{(k+1)} = t_f^{(k)} \). Within phase \( k \) the dynamics of the system are described by a set of dynamic variables

\[
z = \begin{bmatrix} x^k(t) \\ u^k(t) \end{bmatrix} \tag{2.2}
\]

made up of the \( n_x^{(k)} \) state variables and the \( n_u^{(k)} \) control variables, respectively. Some systems may also contain \( n_p^{(k)} \) parameters \( p \). These parameters \( p \) can represent the inertia of the body, for example, and are constant in time. This feature represents one of the main advantages of trajectory optimization, as it can be used to determine the optimal configuration of the system in addition to the trajectory to be followed. Multi phase system, such as a two phase rocket, require different dynamics to be implied in each phase. In the case of this hopping robot, the system dynamics do not change in the scope of this research. For this reason, from this moment on, the notation with \( k \) is dropped and it is assumed that the system is single phase.

The dynamics of the system to be controlled can be defined by a set of ordinary differential equations written in explicit form, which are referred to as the state equations

\[
\dot{x} = f [x(t), u(t), p, t] \tag{2.3}
\]

where \( x \) is the \( n_x \) dimension state vector. In trajectory optimization problems, these equations are interpreted as constraints to the subsequent optimization problem. This way, the optimization algorithm to be implemented is enforced to output solutions that respect the system dynamics.
The next step is to implement the constraints of the system. Following the same procedures of an optimization problem development, the constraints regarding the states, inputs and time are implemented. Assuming fixed initial conditions at time $t_0$, these are defined by $\Psi [x(t_0), u(t_0), p, t_0] \equiv \Psi_0$. The terminal conditions at the final time $t_f$ are defined by

$$
\Psi_{fl} \leq \Psi [x(t_f), u(t_f), p, t_f] \leq \Psi_{fu}
$$

(2.4)

where $\Psi [x(t_f), u(t_f), p, t_f] \equiv \Psi_f$, $\Psi_{fl}$ are the lower bounds and $\Psi_{fu}$ the upper bounds. In addition, the solution must satisfy algebraic path constraints of the form

$$
\rho_l \leq \rho [x(t), u(t), p, t] \leq \rho_u
$$

(2.5)

where $\rho$ is a vector of size $n_\rho$. Path constraints can be applied to describe some dynamics constraints, as opposed to boundary constraints. When applying trajectory optimization to legged robots, a path constraint is used to ensure that the swing foot remains above the ground during the step. Boundary constraints on the state variables and control variables are defined as

$$
x_l \leq x(t) \leq x_u, \quad u_l \leq u(t) \leq u_u
$$

(2.6)

State boundaries are used, for example, to represent joint limits in a robot arm whereas control boundaries can describe maximum torques on motors. Notice that an equality constraint can be imposed if the upper and lower bounds are equal, e.g. $u_l = u_u$. In addition, it is often important to include specific limits on the initial and final time and state.

$$
t_0_l \leq t_0 \leq t_0_u, \quad t_f_l \leq t_f \leq t_f_u
$$

(2.7)

These might be used to ensure that the solution to a path planning problem reaches the goal within some desired time window.

The final piece of the optimization problem description is the definition of the objective function. In fact, the basic problem of optimal control is to determine the $n_u$-dimensional control vectors $u(t)$ and parameters $p$ to minimize an objective function. In general, this function can include two terms: a boundary objective $J(.)$ and a path integral along the entire trajectory, with the integrand $w(.)$. A problem with both terms is said to be in Bolza form. A problem with only the integral term is said to be in Lagrange form, and a problem with only a boundary term is said to be in Mayer form

$$
\min_{t_0, t_f, x(t_0), u(t)} J(t_0, t_f, x(t_0), u(t_f)) + \int_{t_0}^{t_f} w(\tau, x(\tau), u(\tau))d\tau
$$

(2.8)

The boundary objective function $J(.)$ is often set for optimization problems in which the system is required to achieve a certain final state. It is also used to require the minimum time to achieve the same objective [17]. $w(.)$ is often applied to trajectory optimization problems when energy and force are required to be optimized throughout the trajectory [17]. For this case, it is common to find functions
such as $\int_{t_0}^{t_F} u^2(\tau) d\tau$ usually applied for optimization problems whose objective is to minimize the applied force.

The trajectory optimization problem is fully defined when all the above mentioned features are set. The result would follow the structure defined in 2.9. After that, the problem is converted into an optimization problem following one of the methods found in [19].

$$
\min_{t_0, t_F, x(t_0), u(t_0)} J(t_0, t_F, x(t_0), u(t_F)) + \int_{t_0}^{t_F} w(\tau, x(\tau), u(\tau)) d\tau
$$

Variables = \{x(t), u(t)\}

Dynamics : $\dot{x} = f(x, u, t)$

Constraints : $\rho_l \leq \rho(x(t), u(t), p, t) \leq \rho_u$

Boundary constraints : $\Psi_{fl} \leq \Psi(t_F, x_F, u_F) \leq \Psi_{fu}$

(2.9)

2.4 State of the art

2.4.1 Hopping Robots

Hopping systems for planetary mobility were first proposed as a promising transportation concept for astronauts in a lunar environment [10]. The proposed vehicle consisted of a single-seat device propelled by a gas actuated leg hinged under the astronaut seat and stabilized by four elastic legs. The work of Kaplan was never developed but served as a base concept for many other contemporary concepts such as the Lunar Pogo Stick. This device was a man-carrying cabin designed for locomotion on the lunar surface. This concept was developed in [11] and consisted of jumping pole with an attached cabin. In a typical case, a man-carrying cabin would accelerate at an average rate of 1 earth g for a distance of 30 ft by "climbing" the pole inclined at 45 deg with the lunar surface. The pole would be picked up and turned relative to the gyroscopically stabilized cabin during the 12 sec of coasting flight to 110 ft apogee and 400 ft range. Upon contact, the cabin would decelerate at 1 g along the pole, which would "flip" to the launching position again, using residual momentum and/or internal torques. Figure 2.2 presents the operational mode description.

![Figure 2.2: Flight with oscillatory motion of pole, using one traction foot. Source:[11]](image)

Although never being explored, these two researches published important results. Based on data from the Apollo missions, [11] presents a comparison of different approaches to lunar transportation and concludes that hopping can be an efficient form of transportation in a low-gravity environment. More
precisely, the authors of [10] and [11] point out that other solutions such as wheeled devices are slow and sensitive to terrain irregularities. A slow movement of these wheeled devices leads to an inefficient exploration of the terrain, which could represent an increase on costs of operation. As opposed to these slower devices, the authors prove that hopping systems can explore the terrain more efficiently.

Few years later, an also important reference to the development of this field is found in the works of Marc H. Raibert [20], published in 1986. In [20], the author presents a profound research on legged machines. Motivated by the mobility advantages that legs provide to locomotion, Raibert presents a series of legged robots as support of his research on running machines. To study running in its simplest form, the author built a machine that ran on just one leg. This machine had two main parts: a body and a leg [20]. The body carried the actuators and instrumentation needed for the machine operation. The leg could telescope to change length, was springy along the telescoping axis, and could pivot with respect to the body at a simple hip. Sensors measured the pitch angle of the body, the angle of the hip, the length of the leg, the tension in the leg spring, and contact with the ground. This machine with only one leg was the first milestone to non propulsive hopping robots. In fact, it was the first hopping robot with continuous motion and dynamic stability.

Figure 2.3: Raibert’s planar hopping machine (right) and three dimensional hopping machine (left). Source: [20]

A second generation of this legged machine is also presented in [20] and shown in Figure 2.3. Distinct from the first generation, this machine was able to perform locomotion in three dimensions. In order to accomplish this, it had an additional joint at the hip to permit the leg to move sideways as well as fore and aft, with no external mechanical support. These two generations of hopping robots served as reference for many other more recent prototypes such as [3].

The closest relevant work to the design of the Moonhopper comes from [4]. In this article from 2003, the authors adapt the concept developed so far to extraterrestrial exploration describing a novel approach to the design and deployment of small and minimally actuated hopping robots. As result of this research, three generations of small hopping robots were developed.

The first generation design is shown in Figure 2.4. It is composed by a clear polycarbonate shell that surrounds the mechanism, and is attached to the body at the upper support and lower plate. Inside are the components: motor, camera, spring, batteries and micro controller, as in Figure 2.9. Vertical hopping motions are generated by the release of a simple linear spring, which is compressed after each jump via a ball screw that is driven by the motor. Besides jumping this prototype is able to control the take
off angle and regain its original attitude after every jump. This self-righting capability is implemented passively in this design by creating a low center of mass.

Figure 2.4: Three generations of hopping robots (first - left, third - right). Source: [4]

The experiments performed to the first generation prototype show an efficiency of only 20%, meaning that 80% of the energy stored in the spring was not converted into kinetic energy. Instead, this energy was dissipated by friction and wasted motions of the mass–spring take off system. The three main reasons for these losses presented in [4] were:

- The abrupt stop of the foot due to an elastic impact with a mechanical stop, at the end of decompression phase.
- Premature lift-off characteristic of linear springs - a phenomena known as surge.
- The fact that the motor’s design torque is required to compress the spring in a regime where it does little good.

In addition to the inefficient hopping, the first generation also showed a fragile steering and self righting mechanisms.

Figure 2.5: First (left) and second (right) generations hopping robots. Source: [4]

The second generation is developed with the objective of solving these three major shortcomings. The prototype is presented in Figure 2.4 - middle. The main difference between the first and second generation lies on the spring-drive take off actuator designed as a spring/linkage mechanism.
In fact, the new take off mechanism presents force/displacement profile (see [4]) proves that this structure reduces the likelihood of premature lift-off due to the shocks inherent in initial spring release. Furthermore, since the peak force realized during displacement is reduced, the motor’s peak design torque is reduced as compared with the linear spring leg. In addition, when this leg is nearly fully compressed, very little force is required to maintain the compressed state, allowing for a small lock mechanism. As a result, experiments with this system have shown that this leg design realizes a 70% mechanical energy conversion efficiency.

Figure 2.9 presents the schematics of the first two generations. In both, the direction of the jump can be controlled by the movement of the center of mass. In the first generation, this is accomplished by the rotation of the camera around the support. In the second, this is accomplished by the rotation of the pinion gear when the robot is fully compressed. Besides these, the second generation includes a self-righting mechanism. This mechanism is responsible for re-positioning the robot after landing. This is seen in the triangular structure in 2.4.

The third generation is developed to overcome two major problems of the second generation device: the lack of an adjustable take-off angle and the lack of fine mobility. An adjustable take-off enables the robot to predict its aerial trajectories for specific obstacles. The mobility issue is caused by a lack of means for fine adjustment of the robot's position on the terrain and can be overcome by implementing wheels or threads.

The third generation device contained components that addressed these two shortcomings. It retained the six-bar thrusting mechanism (although with a new compression driver), while adding two driven wheels and a mechanism to adjust the take-off angle. Figure 2.4-right shows the hopper. The main mechanical component of the hopper is the gear-box to compress the thrusting springs. In the second generation, the spring was compressed by a rigid power screw. In this prototype, the spring is compressed by winding a cable on a capstan. The cable’s retraction compresses the spring.

So far, all hoppers mentioned were developed to operate on the Moon’s, Mars’ or on Earth’s surface - low gravity environments. However, hopping have its main advantage over any other type of locomotion in micro gravitational environment. This environment can be found on small moons, asteroids or comets. This advantage is mainly because rovers and legged robots can hardly be operated in micro-g conditions due to lack of sufficient wheel-soil/foot-soil friction [21]. The first hopper designed to operate in micro gravity was part of the Soviet Phobos 2 Mission in the 1980s. This hopper, PROP-F, had a mass of about 41 kg and would have been delivered from the main spacecraft at close distance to the surface of the Martian moon Phobos if the communication with the Phobos spacecraft had not been lost before delivering the hopper to the moon surface [21].

Figure 2.6 shows a model of PROP-F. On top, there is the quasi-spherical main body over a structure designed to damp the impact energy at touch down and prevent multiple bouncing or rolling. The hopping is accomplished by releasing an inclined piston with a “foot”. An electric drive with planetary and worm gears tensions a pushing spring by means of a cord. After that, the pushing spring is released leading to take off.

After landing, the self-righting mechanism works so that robot regains its original attitude. Initially
located inside the hopper housing, two fixed boom-like structures (whiskers) and two rotational (active) whiskers, activated by an electric drive adjust the robot position according to the procedure presented in Figure 2.6. The overall operations time was limited to about 4 h and a maximum of 10 jumps. Communications with the main spacecraft had a maximum distance of 300 km. Even though the deployment was a failure, the concept and experience gained within this program in the 1980s was still valid and served as base for future works [22], [23] and [21].

Up to date the only successful deployment of a jumping robot on an asteroid surface was accomplished by JAXA with mission Hayabusa2[22]. Hayabusa2 was launched on 3 December 2014 and rendezvoused with near-Earth asteroid 162173 Ryugu on 27 June 2018. It is in the process of surveying the asteroid for a year and a half, departing in December 2019, and returning to Earth in December 2020.

The robot Minerva [22], [23] is much smaller than PROP-F. The mass of Minerva is only 591 g. It is equipped with three CCD cameras (one single and one stereo pair), sun sensors and thermometers as payload. The robot has a diameter of 120 mm and jumps by means of an internal torquer. This innovative mechanism is described in [7]. When the torquer rotates, a reaction force is developed against the surface making the robot hop with a significant horizontal velocity. After hopping into the free space, the robot moves ballistically and goes back to the surface again. In addition to the torquer, the robot also has a DC motor that rotates the table on which the torquer is placed. This turntable is
rotated very slowly and is mainly used to control the hopping direction. The hop by rotating the turntable is also possible.

Figure 2.8 presents the jumping cycle as it was described above. This mechanism has several significant advantages namely, both torquers can be used for attitude control during the ballistic orbit, no actuators are necessary outside the robot, and in general, the system is light weighted.

This chapter ends with a short remark about this field and the robot prototype used in this thesis. The Moonhopper resembles the second generation of [4]. The reason lies on the efficiency of the robot versus its structure simplicity when compare to other jumping robots. When compared to other designs such as Minerva or [21], Moonhopper has the advantage of generating greater thrust on a planet's surface to overcome obstacles. When compared to [20] and other robots such as [3] designed to operate on Earth and perform specific tasks, Moonhopper's structure often simpler than these. A more complex design usually means an increased chance of system failures, representing greater costs to the project. Nevertheless, it should be recalled that Moonhopper is the first iteration prototype of this project. Meaning that future iterations on the design will be performed later on. For this reason, the overview of the most important designs here presented is fundamental to be documented, hence the objective of this extensive description.

2.4.2 Actuators

One of the main criticisms of the concept of hopping robots has been related to crash-landings [9]. In fact, it is believed that crash-landings can have a degrading effect on the robot's structure and instrumentation. In literature, two ways of approaching this problem are proposed: one way to solve it is to make a structure more capable of enduring such crash-landings [8] and first generation [4]. Another way is to have an active control over the attitude trajectory in air [9]. In this section, a short survey over the attitude actuation strategies in the field of hopping robots is presented and discussed. The main objective is to present the arguments that support the decisions taken for the actuation mechanism design of the Moonhopper.

Non actuated devices in this subject are defined as bodies that can regain its original attitude and position after a collision with the ground. For that reason they do not require attitude control during the trajectory in air. This property is accomplished by either an active self righting mechanisms (second generation [4]) or by a passive mechanism based on the shape and mass distribution of the robot. In article [8] presents a concept of the second type

The structure consists of an elastic cage composed of multiple bended metal strips arranged as the ribs of a Chinese lantern. Since it is also able to provide thrust, at the take off phase, the cage is compressed along the vertical axis, properly oriented and suddenly released. The peculiarity of this design lies on a small foot and a large head: a small foot allows choosing the take off angle at jumping, so that the optimal parabolic trajectory can be approached; a large head enables carrying a large payload, protects its inner parts, and allows for an efficient jump since most of the robot mass is concentrated in the head. At the landing phase, the elastic metal strips of the cage can partially absorb the impact
energy at landing, thus achieving the desired landing protection. In order to regain its original attitude, the design is able to ensure the movement towards the original position, with the help of small legs located at the bottom of the robot [8].

Other robots such as the JollBot [24] and the Canadian Hopping Robot [25] follow the same principles, differing only in the take-off actuation principle. This concept follows a passive control allowing for an exploration with a minimal expense of energy. At the same time, it focuses on the protection of the systems and instrumentation located in the robot’s body.

One of the main arguments against the previous designs is the fact that the introduction of such systems increases the weight of the robot and limits the accessibility of science instrumentation to the environment. One way to avoid unnecessary weight while guaranteeing integrity is by introducing control over the attitude of the robot.

In 2011 the article that lead to the motivation for the second half of this thesis was published. In [9], the authors conceptualize an actuation mechanism based on momentum exchange devices. In addition, they develop a control algorithm for the proposed system. This algorithm allows the robot to aim its internal camera to a desired attitude. After reaching the desired attitude, the controller is able to restore the robot’s initial attitude before landing. The robot that is used for simulation is the second generation of [4], already described in previous sections.

The proposed actuation consists of two reaction wheels aligned with the y and z axes of the body. To understand this choice, the authors point out two facts:

- When the system’s initial angular momentum is zero, two reaction wheels can provide the system full attitude control.
- Disturbance torques during the robot’s lift off phase create angular velocity. In this case, it is possible only to achieve control over the direction of the axis that is perpendicular to the axes of the two reaction wheels [9]

For these reasons, the axes of the reaction-wheels should be chosen so that the system is most robust towards the likeliest axis of initial angular velocity caused by disturbance torques during the take-off, in this case, the x axis.

After developing the model of the actuated robot using Newton-Euler formulation, the authors introduce the spin-axis control law. This proposed law ensures that the system converges to the desired
attitude in any conditions. Additionally, the results for an initially non zero angular velocity, show that for holding an attitude, the actuators that have to constantly accelerate and decelerate the wheel.

The results for the application of this method were tested with a simulation of the robot. The first test to be performed was a Single Rotation. In this test, the most pessimistic rotation was simulated. This corresponds to a rotation of 179 degrees around the axis [1 1 0] from the origin followed by a return to the initial orientation. This was tested with different initial conditions. The second test that was performed was a camera tracking and landing maneuver. Right after take-off, the attitude control system was set to start tracking the camera direction to a target 5 meters to the right of the projected hop trajectory. 2 seconds before the expected landing, the attitude reference was set to prepare for landing. This time was chosen based on the settling time of the pessimistic rotation scenarios described in the Single Rotation test. The results showed the feasibility of attitude control with certain initial conditions and possible mission scenario of tracking a target with camera during flight and achieving preferable landing attitude.

Finally, a short remark on the impact of [9] in this project. The results of this article prove the advantages of having attitude controllers in flight phase. Also, they present a solution involving momentum exchanging devices proving not only the feasibility of this solution, but also concluding that the weight and power requirements of the additional actuators is not significant. Those two facts represent the motivation behind this research and the design of the Moonhopper. The implementation of a DGVSCMG replaces the actuation of two reaction wheels. In fact, DGVSCMG offers considerably more torque at a fraction of the power, being one of the main reason for this replacement [26].

2.4.3 Control Laws for Double Gimbal Variate Speed CMG

In this section, the reader is presented with an overview of steering laws for satellites controlled by these momentum exchange systems.

[27], [28], [29] and many other articles and books approach this control problem. In general, the classical control strategy is divided into two steps. The first step is to compute the required torque to be produced by the actuator in order for the body to follow a specific trajectory. The second step is to compute the required inputs in terms of gimbal acceleration or gimbal rates and wheel acceleration. There is also a third step where the torques that are directly applied by the servos are computed from the results of the gimbal rates or gimbal accelerations. In literature, this step is usually not addressed as currently available CMGs typically require the gimbal rate vector as the input, not the actual physical torque vectors [27].

The described strategy follows a logical procedure for the actuation over the body. In fact is base on the conservation of angular momentum described by the Euler’s equation.

\[
\dot{H}_s + \dot{H}_{CMG} = 0
\]  

Whether the body (spacecraft or hopping robot) is required to follow a trajectory or to maintain a desired attitude, the error between the actual and commanded attitude and rate are computed. Defining
functions $\hat{\epsilon}$ as a generic attitude error and $\hat{\omega}$ as a generic attitude rate error a proportional derivative controller (PD) is developed as

$$u(t) = K_P \hat{\epsilon} + K_D \hat{\omega}$$

(2.11)

where $K_P$ and $K_D$ are scalar proportional and derivative gains. Here $u(t)$ is the correspondent torque applied to the body by the CMG.

From this result, the values of the inputs (gimbal acceleration or gimbal rates and wheel acceleration) are computed using an accurate model inversion of the dynamics of the system. This approach is developed later on this document and for that reason is not explicitly presenter in this section.

Based on this approach, two control modes are usually developed. First, the momentum management mode, where the controller finds and follows torque equilibrium attitudes (a special combination of states at which the disturbance torques are balanced). Then, the attitude hold mode, used to maintain the commanded attitude, regardless of whether the respective attitude corresponds to a torque equilibrium attitude [29]. The first control mode, can only be applied to a specific set of satellites whose attitude can be passively controlled by gravity gradient. For the case of hopping robots, only the second mode is important.

An example of a control law for DGVSCMG can be found in [30]. This article describes a nonlinear control algorithm for spacecraft attitude control using the referred momentum exchange actuator. The article presents the equations of motion of an actuated spacecraft. This deduction uses Newton’s formulation and is based on the conservation of angular momentum.

[30] proceeds with the presentation of the nonlinear steering law. Instead of applying a double-gimbal CMG control law that uses a desired momentum vector tracking with feedback control to determine gimbal rates, a nonlinear Lyapunov control law is developed. This control law is based on the concept of Lyapunov stability. By developing a valid Lyapunov function, the control law can be obtained assuming that the conditions for the Lyapunov stability are met. The resulting Lyapunov condition for the actuation requires to solve a nonlinear matrix equation. The authors then propose a solution via a Newton–Raphson (N–R) iteration to find the solution. The result of this nonlinear equation is the input set that is sent to the actuators. In Appendix B, the rules are presented.

The results of the implementation of this control law show that by using this device it is possible to fully control a spacecraft attitude. However, for some maneuvers, the control law can encounter singularities. When singularities where the gimbal frames line up with each other are avoided, the nonlinear control algorithm achieves very accurate tracking of a reference attitude trajectory.

### 2.4.4 Trajectory Optimization

The trajectory optimization spectrum can be divided into global search methods and local methods. Global search methods include motion planning techniques, such as asymptotically optimal sampling-based motion planning (SBP) algorithms. Due to its enormous computational time [31] (when applied to high order dynamic systems) they are considered out of the scope of this review. On the other hand,
local can be divided in methods direct and indirect methods.

Indirect methods analytically construct the necessary and sufficient conditions for optimality then discretize these conditions and solve them numerically. The necessary conditions for a problem in Bolza form are described by the Euler-Lagrange equations. These methods solve the control equations, implied by the Euler-Lagrange equations, by applying the Pontryagin Maximum Principle. These conditions are then discretized and solved using methods such as root finding.

Based on the nomenclature described in the previous section. Assume a trajectory optimization problem as described in 2.9, with state equations \( \dot{x} = f[x(t), u(t)] \), with only equal constraints and with boundary conditions \( \psi[x(t_f), u(t_f), t_f] = 0 \). Defining the Lagrangian function

\[
\hat{J} = [\phi + \nu^T \psi]_{t_f} + \int_{t_0}^{t_f} w(t) + \lambda^T(t)\{f[y(t), u(t)] - \dot{y}\} dt
\]

(2.12)

where the discrete constraint multipliers are \( \nu \) and the \( \lambda \) are the adjoint or costate variables for the continuous (differential equation) constraints. Defining the Hamiltonian \( H = w(t) + \lambda(t)^T f[x(t), u(t)] \) and the auxiliary function \( \Phi = \phi + \nu^T \psi \). The necessary conditions referred to as the Euler-Lagrange equations which result from setting the first variation to zero in addition to

\[
\begin{align*}
\dot{\lambda} &= -H_x^T \\
0 &= H_u^T \\
\lambda(t_f) &= \Phi_y^T \big|_{t=t_f} \\
0 &= (\Phi_t + H) \big|_{t=t_f} \\
0 &= \lambda(t_0)
\end{align*}
\]

(2.13)

With the first set of equations called adjoint equations, followed by the control equations on the left, and on the right the transversality conditions [19]. The partial derivatives \( H \), are considered considered row vectors. The control equations are an application of the Pontryagin Maximum Principle. This set of equations are solved by the indirect methods. Depending on the algorithm that is used, they are classified as such.

For low order (number of states is small) trajectory problems, indirect methods are often used since in general return a more accurate optimal solution. For high order problems, these methods tend to become less efficient. In fact, the region of convergence of an indirect method tends to be smaller for indirect than for direct methods. This means that an indirect method usually requires a better initialization to converge. In addition, the initialization of an indirect method is complicated by the need to initialize the adjoint variables, which are not used in a direct method. The biggest issue that must be addressed when using an indirect method is the derivation of the necessary conditions themselves. For realistic trajectory simulations the differential equations and the functions that define path constraints and objectives may all be complicated expressions. In order to impose the optimality conditions, it is necessary to analytically differentiate these mathematical expressions which can become complex.

Direct methods discretize the trajectory optimization problem itself, typically converting the original trajectory optimization problem into a non-linear program.

This way the optimization problem can be solved using a NLP solver such as fmincon() or GPOPS. In general, direct methods are much easier to set up and solve. At the same time, they do not have a built-in accuracy metric and are less accurate.
since the solution can include discretization errors. As a result, direct methods are more widely used to
solve more complex problems. In this research, the optimization methods are direct because of the high
dimension of the state.[17]

Most trajectory optimization methods can also be classified into shooting methods or collocation
methods. Shooting methods for trajectory optimization are originated from shooting methods for solving
boundary value problems. From an initial approximation, these methods simulate the behaviour of the
system. Then they evaluate the results, comparing with the boundary conditions and objective functions.
From the error these methods iterate until convergence. The name shooting methods is a reference to
the canon shooting procedure: from an initial set of inputs (canon power and orientation) the canon
ball is shot. If the canon ball went too high from the target, the power and orientation of the canon are
reduced and another shot is fired. The reverse if the canon ball went too low. This procedure goes on
until the target is hit. [17]

Shooting methods tend to be more efficient for problems that do not require complex control expres-
sions. In fact, if the dynamic behavior of the control functions \( u(t) \) cannot be represented using a limited
number of variables (polynomial combinations) the success of a shooting method can be degraded
significantly.[19]

Collocation methods on the other hand, are based on function approximation. These methods use
approximations for the dynamics of the systems, converting the differential equations into algebraic
equations. This process is called transcription and the result is a system of equations with a set of
optimization variables to be optimized.[19]

In general, collocation methods are better for problems with complicated control expressions. In fact,
these methods do not require an a priori specification of the control expressions. Instead they are defined
by the method as a combination of smooth functions such as polynomials or splines. Nevertheless, the
nonlinear programming problem which results from this transcription is large, leading to a large number
of NLP variables with a similar number of constraints. For this method to be efficient it can require a
good initialization.

![Diagram of Shooting methods vs Direct collocation methods](image_url)

Figure 2.9: Shooting methods vs Direct collocation methods. Source: [32]
Chapter 3

System Dynamics

In this section the equations of motion that describe the dynamics of Moonhopper are developed. This section is divided into two parts: in the first part, the focus is on the translation motion of the body and its relation the spring deformation; in the second part the equations that describe the attitude of the body in air are derived. The objective it to have a full characterization of the robot movement in the flight phase.

3.1 Translation motion

As it was mentioned in previous chapters, Moonhopper was inspired in the second generation of [4]. In fact, the take off actuator of Moonhopper resembles the one from this hopping robot with a difference on the retraction mechanism. This means that the legs are closely similar and that the same model for energy storage in the springs can be applied as it can be seen in Figure 3.1.

Following the same model, the equations that relate the legs displacement with the force applied during take off are presented in 3.1 and [4]. For deriving an expression for the energy at the take off, it helps to perspective this force to be exchanging energy with the floor in the form of work. In fact, the sum of forces in this system is given by
The work produced by the leg extension is determined by the integral of this expression, over the trajectory of the center of mass. This trajectory starts with the fully compressed state of the robot and when ends when the body leaves the ground.

\[
W = \int_{y_0}^{y_F} \left( ky \left( \frac{(c - l_0) + \sqrt{4a^2 - y^2}}{\sqrt{4a^2 - y^2}} \right) - m_{\text{robot}}g \right) dy
\]  

where \( k \) is the springs constant, \( y \) is the displacement of the CoM (center of mass) during the take off phase, \( m_{\text{robot}} \) corresponds to the robots mass and \( g \) to the gravitational acceleration (note that this is not necessarily Earth's one). \( a, c \) and \( l_0 \) are parameters that describe the legs and are defined in figure 3.1.

Since no closed form solution was found for solving 3.2, the results were computed numerically and are presented and discussed in Chapter 6. From this point it is possible to compute an estimate for the take off velocity, based on the definition of kinetic energy and assuming no rotation during take off [33].

\[
W_{TO} = \frac{1}{2} m_{\text{robot}} v_{TO}^2 \iff v_{TO} = \sqrt{\frac{2W_{TO}}{m_{\text{robot}}}}
\]  

Notice the difference between \( W \) and \( W_{TO} \). \( W_{TO} \) differs from \( W \) according to the relation

\[
\eta_W = \frac{W_{TO}}{W}
\]  

and accounts for the losses due to friction during the springs extension. The variable \( \eta_W \) is often called "efficiency" and it is used in Chapter 6 to characterize the prototype with respect to the theoretical model. Finally, from the initial velocity, and assuming a ballistic trajectory, it is possible to compute the jumping metrics based on the expressions of the vertical ballistic trajectory

\[
y(t) = y_{TO} + v_{TO}t - \frac{1}{2}gt^2
\]

\[
v(t) = v_{TO} - gt
\]

\( h_{max} \) is the value of \( y(t) \) when \( v(t) = 0 \) and when \( t = t_F, y(t_F) = 0 \). From these definitions the expressions for these metrics are

\[
h_{max} = \frac{v_{TO}^2}{2g}, \quad t_F = \frac{2v_{TO}}{g}
\]  

The main objective of these theoretical results is to obtain an approximate estimate for the time of flight of the robot on the moon. It is important to mentioned that the moon locomotion serves as a motivation. In fact, this expression (as all this work) can be applied for other planets or small bodies. In this case, the difference resides on the gravitational acceleration.

This result is important as it represents a time constraint for the optimization problem described in Chapter 5. In order to compute the time of flight, first an estimate is computed for a jump on Earth. Then
experiments are performed in order to determine the real values for $h_{\text{max}}$ and $t_f$. With these values, an estimate of the energy at take off is obtained following the principle of conservation of the total energy of the body in air (assuming the drag cause by the atmosphere is negligible for the low velocities the robot reaches). By comparing the expected energy at take off, from the model, and the real energy at take off, from the experiment results, the result is a value for $\eta_W$. Finally, new values for $h_{\text{max}}$ and $t_f$ are predicted using $g_{\text{moon}}$ and $\eta_W$ and following the same procedure.

3.2 Rotational motion

The model developed in this section is a modification of the model developed in [30] of the attitude of an actuated satellite. In contrast with other models of bodies actuated by CMGs, one of the main features of the approach is that it takes into account the moments of inertia of the two gimbal axes. This feature is shown to be of great importance since the two gimbal axes have a significant influence on the inertia of the whole system. Another important feature of this deduction is the use of the transport theorem instead of inertial derivatives computation. This way the calculations of the Euler’s equation are simplified.

For understanding the derivation, this section provides an introduction to the notation followed by a few important definitions. In addition to the information presented in this chapter, Appendix A will provide the reader with the most exhausting parts of this derivation. The reader is encourage to consult these additional chapters for a more clear presentation of the results.

3.2.1 Notation

For this derivation a specific notation is used. This section presents and describes the notation, so it can be applied in the following sections.

Starting with coordinate frames, these are denoted using uppercase like in frame N or frame G. Here, $[AB]$ is the direction cosine matrix that defines the orientation of the A frame relative to the B frame, and $[M_i(\alpha)]$ denotes a rotation by angle $\alpha$ about the $i$-th axis.

A unit vector representing an axis that is fixed to a frame is represented with the lowercase of the frame’s designation - $\hat{g}_i$ is fixed to frame G. $i$ can be equal to $\{1, 2, 3\}$ representing the x-axis, y-axis and z-axis of the correspondent frame, respectively.

A vector $v$ defined in frame A will be represented as $^A v$. The same notation is used for tensors. The tensor of inertia of body A, $[I_A]$ defined in frame A is represented as $^A [I_A]$.

When it comes to angular velocities between two bodies, the notation used in this document is $\omega_{A/B}$. This variable represents the angular velocity of body A with respect to body B.

3.2.2 Definitions

A representation of the DGVSCMG is presented in Figure 3.2. According to this representation, the DGVSCMG consists of three structures that rotate along specific axes. The outer gimbal, the round
structure, is fixed to reference frame G and rotates along the axis $\hat{f}_1$, relative to the robot body. The inner gimbal is fixed to reference frame H and rotates along the axis $\hat{g}_2$ relative to the outer gimbal. Finally, the wheel’s rotation axis is fixed to frame H and is oriented towards the z-axis $\hat{h}_3$ its axis of rotation. Figure 3.2 (a) presents three different configurations of the double gimbal. In the first, the three frames $\hat{f}, \hat{g}, \hat{h}$ are parallel. In the second, a rotation of G and H of 90° about the axis $\hat{f}_1$. The frame $\hat{f}$ is fixed and $\hat{g}, \hat{h}$ are parallel. In the third, frames F and G are fixed and H rotates 90° over $\hat{g}_2$. Axes $\hat{f}$ and $\hat{g}$ are parallel and $\hat{h}$ is rotated.

From this description it is necessary to define four reference frames necessary to the description of this system’s dynamics.

Picturing a hopping motion, the inertial reference frame is defined outside the robot and is fixed to the ground. In this document, is defined with the letter N. $\hat{n}_3$ axis is perpendicular to the ground and is represented with the color blue. Axes $\hat{n}_1$ and $\hat{n}_2$ are contained in the ground plane and are represented with the colors red and green respectively (Figure 3.2). Reference frame F stays fixed with respect to the robot. It is used to define the orientation of the entire device relative to frame N. When the attitude of the robot is 0 for the pitch, yaw and roll angles, reference F and N have the same orientation.

Frame G is connected to the outer gimbal. Since it rotates $\psi$ about the $\hat{f}_1$ axis fixed to the robot’s body, it is possible to conclude that $\hat{f}_1 \equiv \hat{g}_1$. The same can be concluded for the rotation of frame H of $\theta$, fixed to the inner gimbal, with respect to frame G: $\hat{g}_2 \equiv \hat{h}_2$. Figure 3.2 presents this definitions.

Using the notation described earlier for the rotation matrices, the following (orthogonal) matrices are defined:

$$[FG] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & -\sin(\psi) \\ 0 & \sin(\psi) & \cos(\psi) \end{bmatrix} \quad [GH] = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

$$[FH] = [FG][GH], \quad [HF] = [FH]^T$$
With these matrices defined, writing a vector in a different reference frame is done by multiplying the respective rotation matrix. Thus the following vectors are computed

\[
F^1 \hat{f}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad F^2 \hat{g}_2 = FG, \quad G^2 \hat{g}_2 = FG, \quad F^1 \hat{h}_1 = FH, \quad \dot{F}^1 \hat{h}_1 = [FH] \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\] (3.7)

The angular rates are defined as the velocity of rotation of a body with respect to a reference. According to [34] they are defined from the time derivative of the respective rotation matrices. In this system, the angular velocities to be considered are:

\[
\omega_{F/N} = \dot{\psi}^1, \quad \omega_{G/F} = \dot{\psi}^2, \quad \omega_{H/G} = \dot{\theta}, \quad \omega_{W/H} = \dot{\Omega}^3.
\]

The notation represents the angular velocity of frame A with respect to frame B. As an example, \(\omega_{F/N}\) is used to describe the angular velocity of the robot (frame F) with respect to the inertial frame N. All these vectors follow the relations

\[
A \omega_{A/B} = -B \omega_{B/A}, \quad B \omega_{A/B} = -A \omega_{B/A},
\]

\[
A \omega_{A/B} = [AB]B \omega_{A/B}, \quad A \omega_{A/C} = A^{A} \omega_{A/B} + B^{A} \omega_{B/C}
\] (3.8)

The moment of inertia measures the extent to which an object resists rotational acceleration about a particular axis [33]. In a way, it determines the torque needed for a desired angular acceleration about a rotational axis and depends on the body’s mass distribution and the axis chosen. As result, larger moments require more torque to change the body’s rotation rate. For bodies free to rotate in three dimensions, their moments of inertia are described by a symmetric 3-by-3 matrix [33].

In this system, the moments of inertia to be defined are \([I_S], [I_G], [I_H]\) and \([I_W]\), and correspond to the moment of inertia matrices of the robot’s body, the outer gimbal, the inner gimbal and the wheel, respectively. Geometrically, these bodies considered regular shapes with several axes of symmetry. For simplification, \([I_G], [I_H]\) and \([I_W]\) are assumed to be diagonal matrices when defined in the the respective reference frame. \(G[I_G] = \text{diag}([I_{Gxx}, I_{Gyy}, I_{Gzz}]), H[I_H] = \text{diag}([I_{Hxx}, I_{Hyy}, I_{Hzz}]), H[I_W] = \text{diag}([I_{Wxx}, I_{Wyy}, I_{Wzz}])\). For changing coordinates, these matrices are multiplied in both sides by the rotation matrices, following the form \(F[I_G] = [FG][I_G][GF]\).

Finally, the angular momentum vector is defined. Recalling the definition of the moment of a vector \(m.v\) and denoting by \(r\) the position vector of a point \(P\) with respect to a point \(O\). The angular momentum of a particle about \(O\) is defined as

\[
H_O = r \times m.v \quad (3.9)
\]

where \(\times\) corresponds to the cross product.

For a rigid body rotating about its center of mass the definition is extended and the concept of moment of inertia \(I\) of the body about its center of mass (CoM) is introduced as \(H_{CoM} = [I] \omega\). For a body that rotates around a point different from its center of mass, an additional term that quantifies the rotation of its center of mass must be included. This new term corresponds to the angular momentum of the center
of mass that caused by the off centred rotation.

\[ H = [I]\omega + r \times m.v \]  \hspace{1cm} (3.10)

Where \( m \) is the mass of the body, \( v \) corresponds to the velocity of the CoM around the axis of rotation and \( r \) is the distance between the CoM and the point on the axis about which the body is rotating. According to the definition of tangential velocity [33], in this case \( v = \omega \times r \). Introducing this relation in the previous equation the result is

\[ H = [I]\omega + r \times (m\omega \times r) = [I]\omega - r \times (mr \times \omega) \]

\[ = ([I] - m\tilde{r})\omega \]

\( \tilde{r} \) is the skew symmetric matrix defined by vector \( r \) that represents the cross product. The result shows that, in the case of constant \( r \) the inertia of the body over the new axis corresponds to \( ([I] - m\tilde{r}) \).

This result is often described as the Steiner’s Theorem [33] and is often directly applied to compute the angular momentum of an off centered body rotation.

In order to define the angular momentum of the robot, its structure is divided into four parts: Robot body (LBS) - composed by legs (L), body (B) and CMG holder (S); Outer Gimbal (G); Inner Gimbal (H) and Reaction Wheel (W).

For each of these four parts, the angular momentum vector \( H_i \) is defined according to equation 3.10, taking into account the rotation of each COM with respect to the COM of the robot:

\[ H_{LBS} = H_L + H_B + H_S \]

\[ = F[I_L]\omega + r_L \times m_Lv_L + F[I_B]\omega + r_B \times m_Bv_B + F[I_S]\omega + r_S \times m_Sv_S \]

\[ = (F[I_L] + F[I_B] + F[I_S])\omega - m_L\tilde{r}_L\tilde{r}_L\omega - m_B\tilde{r}_B\tilde{r}_B\omega - m_S\tilde{r}_S\tilde{r}_S\omega \]

\[ = ([I_{LBS}] - m\tilde{r})\omega \]  \hspace{1cm} (3.12)

\[ H_G = H_{G_c} + H_{G_{off}} \]

\[ = F[I_G]\omega_{G/N} + r_G \times m_Gv_G = F[I_G](\omega + \dot{\psi}_1) - m_G\tilde{r}_G\tilde{r}_G\omega \]  \hspace{1cm} (3.13)

\[ H_H = H_{H_c} + H_{H_{off}} \]

\[ = F[I_H]\omega_{H/N} + r_H \times m_Hv_H = F[I_H](\omega + \dot{\psi}_1 + \dot{\theta}_2) - m_H\tilde{r}_H\tilde{r}_H\omega \]  \hspace{1cm} (3.14)

Due to the fact that the position of the reaction wheel is shifted from the center of mass of the inner and outer gimbals, the definition of the angular momentum is different. It must account for the velocity caused by the rotation of these two bodies that move independently, in addition to the velocity caused by \( \omega \).
\[ H_W = H_{W_0} + H_{W_{off}} \]
\[ H_{W_0} = [I_W] \omega_{W/N} = [I_W] \left( \omega + (\dot{\theta}\hat{g}_1 + \dot{\theta}\hat{g}_2 + \Omega\hat{h}_3) \right) \]
\[ H_{W_{off}} = H_{W_{off1}} + H_{W_{off2}} + + H_{W_{off3}} \]
\[ = r_W \times (m_W \dot{v}_\theta) + r_W \times (m_W \dot{v}_\phi) + r_W \times (m_W \omega) \]
\[ = r_W \times (m_W (\dot{\psi}\hat{f}_1 + \dot{\theta}\hat{g}_2) \times r_W/H) + r_W \times (m_W \omega \times r_W) \] (3.15)

Where \( r_{W/H} = [FH][0 0 r_{w_2}]^T \) is the wheel position relative to the position of \( H \) and \( r_W = r_{W/H} + r_H \) is the position relative to the robot CoM.

The readers should note that the rotation of the gimbals generates a change in the position of the wheel in the body reference frame. In theory, this displacement of the wheel induces a change in the position of the robot CoM. However, for the sake of simplicity and since the wheel’s mass is significantly lower when compared to the robot’s entire mass, the robot CoM is considered to remain static with respect to the body frame.

### 3.2.3 Dynamic model

In order to develop a dynamic model of the robot attitude, the Euler’s equation is applied. In classical mechanics, Euler’s rotation equations are a vectorial first-order ordinary differential equation describing the rotation of a rigid body, using a rotating reference frame with its axes fixed to the body[33]. Euler’s equation expresses the conservation of angular momentum in a closed system and follows

\[ \dot{H} = 0 \] (3.16)

Where \( \dot{H} = N dH/dt \) is the time derivative in the inertial reference frame. According to the definitions stated earlier, equation 3.16 can be written as the sum of the time derivatives of its elements

\[ \dot{H} = \dot{H}_{LBS} + \dot{H}_G + \dot{H}_H + \dot{H}_W = 0 \] (3.17)

From this equation, time derivatives of the angular momentum vectors must be computed with respect to frame \( N \). In order to do so, equation 18.22 of [33] is applied. This equation gives a direct expression for the computation of the time derivative with respect to a reference frame \( N \)

\[ F \frac{dA_v}{dt} = A \frac{d^A v}{dt} + \omega_{A/F} \times ^A v \] (3.18)

Assuming that frame \( A \) is rotating with respect to frame \( F \) with velocity \( \omega_{A/F} \), the time derivative of some vector \( v \) defined in \( A \) in frame \( F \) is equal to the time derivative of the same vector \( v \) in frame \( A \), plus an additional term. A direct application of this equation can be performed to compute the time derivative of the angular momentum of the robot body.
\[
\dot{H}_{LBS} = \frac{d}{dt}\left(\frac{N}{d}H_{LBS}\right) = \frac{N}{d}d\left(F[I_{LBS}]\omega\right) = F\frac{d}{dt}(F[I_{LBS}]\omega) + \omega_{F/N} \times F[I_S]\omega
\]
\[
= F[I_S]\dot{\omega} + \omega \times F[I_S]\omega
\]

(3.19)

In order to compute the remaining derivatives equation 3.18 is applied recursively as it can be seen in equation 3.20.

\[
\dot{H}_{G_c} = \frac{\partial}{\partial t}\left(F[I_{G}]\omega_{G/N}\right) + \omega_{G/N} \times \left(F[I_{G}]\omega_{G/N}\right)
\]
\[
= F[I_G]\left(h\dot{f}_1 + \omega \times (\dot{f}_1 + \dot{\theta}g_2) + \dot{\theta}g_2\right) + \omega_{G/N} \times \left(F[I_G]\omega_{G/N}\right)
\]
\[
= F[I_G]\left(\dot{\omega} + \dot{\psi}f_1 + (-\dot{\psi}f_1 + (\omega + \dot{\psi}f_1) + \omega_{G/N} \times (F[I_G]\omega_{G/N})
\]
\[
= F[I_G]\left(\dot{\omega} + \dot{\psi}f_1 + \omega \times (\dot{\psi}f_1)\right) + \omega_{G/N} \times (F[I_G]\omega_{G/N})
\]

(3.20)

\[
\dot{H}_{G_{off}} = \frac{\partial}{\partial t}\left(-m_G\bar{r}_{G_{off}}\omega\right) = F\frac{d}{dt}\left(-m_G\bar{r}_{G_{off}}\omega\right) + \omega \times (-m_G\bar{r}_{G_{off}}\omega)
\]
\[
= -m_G\bar{r}_{G_{off}}\dot{\omega} + \omega \times (-m_G\bar{r}_{G_{off}}\omega)
\]

(3.21)

Applying the same procedure to \(\dot{H}_H\) and \(\dot{H}_W\), the result is 3.22, and |

\[
\dot{H}_H = F[I_H]\left(\dot{\omega} + \dot{\psi}f_1 + \dot{\theta}g_2 + \omega \times (\dot{\psi}f_1 + \dot{\theta}g_2) + \dot{\theta}g_2\right) + \omega_{H/N} \times (F[I_H]\omega_{H/N}) - m_{H\bar{r}_H}\bar{r}_H\dot{\omega} + \omega \times (-m_{H\bar{r}_H}\bar{r}_H\omega)
\]

(3.22)

\[
\dot{H}_W_c = F[I_W]\left(\dot{\omega} + \dot{\psi}f_1 + \dot{\theta}g_2 + \Omega\dot{h}_3 + \omega \times (\dot{\psi}f_1 + \dot{\theta}g_2 + \Omega\dot{h}_3) + \dot{\theta}g_2 + \omega \times (\dot{\theta}g_2 + \Omega\dot{h}_3)
\]
\[
\left(\dot{\theta}g_2\right) + \omega_{W/N} \times (F[I_W]\omega_{W/N})
\]

(3.23)

\[
\dot{H}_{W_{off}} = D^F_r \times \left(m_W (\dot{\psi}f_1 + \dot{\theta}g_2) \times r_{W/H} + \omega \times r_W\right)
\]
\[
+ r_W \times \left(m_W (\dot{\psi}f_1 + \dot{\theta}g_2 + \omega_{G/F} \times (\dot{\theta}g_2)) \times r_{W/H} + m_w \dot{\omega} \times r_w + m_w (\dot{\psi}f_1 + \dot{\theta}g_2 + \omega) \times D^F_r\right)
\]
\[
+ \omega \times \left(r_W \times (m_W (\dot{\psi}f_1 + \dot{\theta}g_2) \times r_{W/H}) + r_W \times (m_w \omega \times r_W)\right)
\]

(3.24)

Where \(D^F_r\) is the time derivative of \(r_w\) in frame F. All these results are added according to equation 3.17 and the result is an equation with states \(\omega, \psi, \theta, \Omega, \dot{\psi}, \dot{\theta}\) representing the angular velocity of the robot, the angle displacement of the outer gimbal and inner gimbal, the wheel velocity and the rates of the outer and inner gimbal, respectively. The objective from this point is to obtain an expression for \(\dot{\omega}\). Introducing the modified angular momentum derivatives \(\dot{H}'\) and combined moments of inertia \([I_{stat}]\) and \([I]\), defined in 3.25.
\[ \dot{H}_i = \dot{H}_i - [I_i] \dot{\omega} - (m_i \vec{r}_i \dot{r}_i \dot{\omega} - \omega \times (m_i \vec{r}_i \dot{r}_i \omega)) \quad [I_{stat}] = [I_{LBS}] - m_G \ddot{r}_g - m_H \ddot{r}_h - m_w \ddot{r}_w \ddot{r}_w \\
[I] = [I_{stat}] + F[I_G] + F[I_H] + F[I_W] \]

(3.25)

The expression of \( \dot{\omega} \) is given by (according to the structure presented in [30])

\[ \dot{\omega} = [I]^{-1} \left( -\omega \times [I_{stat}] \omega - \dot{H}_G - \dot{H}_H' - \dot{H}_W' \right) \]

(3.26)

Since the DGVSCMG is commanded by torques, an additional relation regarding the inputs and the states of the system must be added. The goal of this second derivation is to obtain an equation of \( \ddot{\psi} \), \( \ddot{\theta} \) and \( \ddot{\Omega} \) as function of the states and input torques. For this purpose, the relations between the time derivatives of the angular momentum of the parts of the CMG and the respective input torques are introduced. Following [30] the relation is given by

\[ [HF] F \left( \dot{H}_{W_c} \right) = \begin{bmatrix} \sim \\ \sim \\ u_W \end{bmatrix}, [GF] F \left( \dot{H}_{W_c} + \dot{H}_{W_{off}} + \dot{H}_{H_c} \right) = \begin{bmatrix} \sim \\ u_H \\ \sim \end{bmatrix} \]

(3.27)

\[ F \left( \dot{H}_{W_c} + \dot{H}_{W_{off}} + \dot{H}_{W_{off}} + \dot{H}_{H_c} + \dot{H}_{G_c} \right) = \begin{bmatrix} u_G \\ \sim \\ \sim \end{bmatrix} \]

Notice the introduction of \( \dot{H}_{W_{off}} \) and \( \dot{H}_{W_{off}} \). In fact, this new variable (defined in the appendix) is important for the derivation of these equations. It expresses the time derivative of the offset part of the angular momentum vector with respect to the inner gimbal center of mass. So far these variables had not occurred yet as the analogous were defined with respect to the robot COM.

The rest of this derivation is found in Appendix. The idea behind this derivation is to invert the equations derived earlier. The first step is to obtain a function of \( \ddot{\psi} \) and separate the terms with \( \ddot{\psi}, \ddot{\theta} \) and \( \ddot{\Omega} \). The following step is to repeat the process for the expressions of \( \dot{H}_{W_c}, \dot{H}_{W_{off}} + \dot{H}_{H_c} \) and \( \dot{H}_{G_c} \). From the results three different expressions are generated that follow 3.27. From this point, the expressions are organized as

\[ \begin{bmatrix} \left( \dot{H}_{W_c} + \dot{H}_{W_{off}} + \dot{H}_{W_{off}} + \dot{H}_{H_c} + \dot{H}_{G_c} \right) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T = \begin{bmatrix} u_G \\ u_H \\ u_W \end{bmatrix} \]

From here it is possible to organize the previous equation and obtain an expression in the form

\[ [M] \begin{bmatrix} \ddot{\psi} \\ \ddot{\theta} \\ \ddot{\Omega} \end{bmatrix} + B = \begin{bmatrix} u_G \\ u_H \\ u_W \end{bmatrix} \iff \begin{bmatrix} \ddot{\psi} \\ \ddot{\theta} \\ \ddot{\Omega} \end{bmatrix} = M^{-1} \begin{bmatrix} u_G \\ u_H \\ u_W \end{bmatrix} - B \]

(3.28)
Matrices M and B are defined in Appendix A.

State Space System Equations

This section presents the final state space system of the Moonhopper attitude dynamics. To complete the previous derivation, the attitude kinematics must be defined. In Appendix A these are defined for Euler angles and quaternions. Applying the quaternion vector, the states of this system are $q, \omega, \dot{\psi}, \dot{\theta}$ in total 12 variables using the quaternions representation, 11 if Euler’s angles are used. The result is

\[
\dot{q} = \frac{1}{2} \begin{bmatrix}
q_w & -q_z & q_y \\
n_z & q_w & -q_x \\
-q_y & q_x & q_w \\
-q_x & -q_y & -q_z
\end{bmatrix} \omega
\]

\[
\dot{\omega} = [I]^{-1} \left( -\omega \times [I_{stat}] \omega - \dot{H}_G - \dot{H}_H - \dot{H}_W \right)
\]

(3.29)

\[
\dot{\psi} = \dot{\theta} = \dot{\theta}
\]

with $u = [u_G, u_H, u_W]^T$. From this point, and having the attitude dynamics, the next step is to perform model verification. For this, a model of the system was developed in ROS/Gazebo. ROS (Robot Operating System) is a collection of software frameworks for robot software development and Gazebo is an open-source 3D robotics simulator. The dynamics of the robot are simulated using this software and physics simulator ODE. Information regarding the states and inputs is collected and compared with the state space model variables. The results are presented in Chapter 6.

Another way to perform model verification is to compute some important physical quantities that describe the system. Computing the time derivative of angular momentum and obtaining 0 for all $t$ is an indicative of a valid system. The kinetic energy (3.30) is also an important quantity to be monitored as it represents the influence of the input torques over the system. For this reason these two quantities are also computed and compared for model verification.

\[
T = \frac{1}{2} \omega^T [I_{LBS}] \omega + \frac{1}{2} \omega^T_{G/N} [I_G] \omega_{G/N} + \frac{1}{2} m_G v^2_G
\]

+ \frac{1}{2} \omega^T_{H/N} [I_H] \omega_{H/N} + \frac{1}{2} m_H v^2_H + \frac{1}{2} \omega^T_{W/N} [I_G] \omega_{W/N} + \frac{1}{2} m_W v^2_W
\]

(3.30)

with $v_G = r_G \times \omega, v_H = r_H \times \omega$ and $v_W = r_W \times \omega + r_{W/H} \times (\dot{\psi} \hat{f}_1 + \dot{\theta} \hat{g}_2)$.

In conclusion, in this section the translation and the rotation models were developed. The objective from this point is first to validate these models. Following validation, the rotation model can then be used to develop the attitude controllers that are seen in the next chapter as well as the be used as constraint dynamics in the trajectory optimization problem.
Chapter 4

Controllability, Singularities and Control Architecture

The principal objective of this research is to control the Moonhopper attitude during its trajectory after take off. But before designing the controller, in this section, a broad study of the controllability and singularities is provided. The objective is not to prove that the system is controllable in the manifold in which is defined \((SO(3) \times \mathbb{R}^8)\) but instead, if given an initial state, the system is able to reach any desired attitude with a specific angular velocity \((SO(3) \times \mathbb{R}^3)\) [35] - in this case, the output controllability.

In the final section, information regarding dynamics and controllability is used to design the control architecture. Based on mission objectives plus the characteristics described, three controllers are designed and discussed.

4.1 Controllability

In general, controllability can be defined as the ability to steer a system from a given initial state to any final state, in finite time, using available controls. Introducing the concept of manifold, to prove that a system is controllable, it is required to prove that the control affine system is globally controllable from any initial point \(x \in M\) - curved space that defines the space of configurations of the system. Assuming a general affine dynamical system described by differential equation

\[
\dot{x} = f_0(x) + \sum_{i=1}^{p} f_i(x)u_i, \quad x \in M
\]

where \(t \rightarrow x(t)\) is a curve in a state manifold \(M\). The vector field \(f_0\) is the drift vector field, describing the dynamics of the system in the absence of controls, and vector fields \(f_1, \ldots, f_m\) are the input vector fields or control vector fields, indicating how the system reacts to actuation. \(u_i\) are the control inputs \(u : [0, T] \rightarrow U\) and are integrable and \(U\) is a subset of \(\mathbb{R}^m\).

A control affine system is described by a triple \(\Sigma = (M, F = (f_1, \ldots, f_m), U)\) and a control trajectory for \(\Sigma\) is a pair \((c, u)\) where \(u\) takes admissible values and where \(c : [0, T] \rightarrow M\) is defined so that
\[ c'(t) = f_0(c(t)) + \sum_{i=1}^{P} f_i(c(t))u_i \]

For \( x \in M \) and \( T > 0 \) the reachable sets are defined as

\[
R_{\Sigma}(x, T) = \{ c(T) \mid c(c, u) \text{ is a controlled trajectory for } \Sigma \text{ with } c(0) = x \}
\]

and

\[
R_{\Sigma}(x, \leq T) = \bigcup_{t \in [0, T]} R_{\Sigma}(x, t), \quad R_{\Sigma}(x) = \bigcup_{t \geq 0} R_{\Sigma}(x, t)
\]

Based on these, it is possible to define the various types of controllability of nonlinear systems:

**Definition 4.1.1.** Let \( \Sigma = (M, \mathcal{F}, U) \) be a control affine system and let \( x \in M \):

1. \( \Sigma \) is accessible from \( x \) if \( \text{int}(R_{\Sigma}(x)) \neq \emptyset \)
2. \( \Sigma \) is strongly accessible from \( x \) if \( \text{int}(R_{\Sigma}(x)) \neq \emptyset \) for each \( T > 0 \)
3. \( \Sigma \) is locally controllable from \( x \) if \( x \in \text{int}(R_{\Sigma}(x)) \)
4. \( \Sigma \) is small-time locally controllable (STLC) from \( x \) if there exists \( T > 0 \) so that \( x \in \text{int}(R_{\Sigma}(x, \leq t)) \) for each \( t \in [0, T] \)
5. \( \Sigma \) is globally controllable from \( x \) if \( R_{\Sigma}(x) = M \)

For more detailed explanations and examples of systems with this characteristics, readers are encouraged to consult [36]. These definitions are cited from this article.

In [35], these definitions are used to study the controllability of a spacecraft carrying one or more CMGs. Although Moonhopper is not a spacecraft, while in air it can be seen as a free falling body actuated by a specific CMG. From this perspective it can be interpreted as a spacecraft, and the results can be applied.

In order to prove the controllability of the system, the authors of [35] define a generic affine mode representing the kinematics and dynamics of the system. They state that since the inertial components of the total angular momentum are constant along the motion of the spacecraft, for every initial inertial angular momentum \( H \), the dynamics evolve on the constant angular momentum sub manifold set \( M_H = Q(H) \subseteq SO(3) \times \mathbb{R}^3 \times \mathbb{R}^m \). This subset of \( M \) represents the constraint derived from the conservation of angular momentum of the actuated system.

The first result obtained by the authors in [35] is enough to prove that the system is not globally controllable according to definition 4.1.1. However, it does not mean it cannot be controlled inside this subset of manifold \( M \). In order to prove this, authors prove that for every \( H \in \mathbb{R}^3 \) the system is strongly accessible and controllable.

In this section, the goal is to use the same approach described in [35] applied to a free falling DGVSCMG actuated body. Instead of using the system described in Chapter 3, in this section, the
The system that is used is a simplified version of that. Found and described in [37], the system consists of a body plus wheel, where the velocity of the last is controlled (by setting $\dot{\Omega}$) as well as its axis (in two directions $\psi$ and $\theta$) - ignoring the angular momentum caused by the gimbals.

The angular momentum of the resulting body is defined as

$$H = I_{\text{stat}}\omega + H_{\text{cmg}}(\psi, \theta, \Omega) = I_{\text{stat}}\omega + \begin{bmatrix} I_{wzz}\Omega\sin(\theta) \\ -I_{wzz}\Omega\sin(\psi)\cos(\theta) \\ I_{wzz}\Omega\cos(\theta)\cos(\psi) \end{bmatrix}$$ \hspace{1cm} (4.1)$$

Applying the time derivative to the expression determined above, the result is

$$[I_{\text{stat}}\dot{\omega} + \dot{H}_{\text{CMG}} + \omega \times ([I_{\text{stat}}\omega + H_{\text{CMG}}] = 0$$

The expression of $\dot{H}_{\text{CMG}}$ is obtained using the procedure followed earlier for the complete system. The result is given by

$$\dot{H}_{\text{CMG}} = \frac{F_d}{I_{wzz}} (I_{wzz}\Omega \dot{h}_3) = \frac{H_d}{I_{wzz}} (I_{wzz}\Omega \dot{h}_3) + \frac{\omega_H}{F} \times (I_{wzz}\Omega \dot{h}_3)$$

$$= I_{wzz}\Omega \dot{h}_3 + (\dot{\psi} \tilde{f}_1 + \dot{\theta} \tilde{g}_2) \times (I_{wzz}\Omega \dot{h}_3)$$

$$= I_{wzz}\Omega \begin{bmatrix} \sin(\theta) \\ -\sin(\psi)\cos(\theta) \\ \cos(\theta)\cos(\psi) \end{bmatrix} + I_{wzz}\Omega \begin{bmatrix} 0 & \cos(\theta) \\ -\cos(\theta)\cos(\psi) & \sin(\theta)\sin(\psi) \\ \cos(\theta)\sin(\psi) & -\sin(\theta)\cos(\psi) \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ \dot{\theta} \end{bmatrix}$$ \hspace{1cm} (4.2)$$

Describing the attitude kinematics using a matrix $R \in SO(3)$ that represents the attitude of the the body with respect to a reference inertial frame, the resulting equation follows

$$\dot{R}(t) = R(t)S(\omega(t))$$ \hspace{1cm} (4.3)$$

where $\omega$ is the instantaneous body-frame components of the angular velocity of the body $\omega \in \mathbb{R}^3$ and $S()$ represents the cross product operation.

The system is then described as
\[ R(t) = R(t)S(\omega(t)) \]

\[
[I_{\text{stat}}] \dot{\omega} = -\omega \times ([I_{\text{stat}}] \dot{\omega} + H_{\text{cmg}}(\psi, \theta, \Omega)) - H'_{\text{cmg}} \begin{bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{bmatrix}
\]

\[ \dot{\psi} = u_1 \]
\[ \dot{\theta} = u_2 \]
\[ \dot{\Omega} = u_3 \]

(4.4)

To investigate the controllability of the system described above on the constant angular momentum manifold, suppose \( H \in \mathbb{R}^3 \) to be the angular momentum of the entire system (body+DGVSCMG). As stated in [35], on the angular momentum level \( M_H \) the attitude kinematics equation can be written in terms of the conservation of angular momentum \( (R^T H = I_{\text{stat}} \omega + H_{\text{cmg}}(\psi, \theta, \Omega)) \) as

\[ \dot{R}(t) = R(t)S(I_{\text{stat}}^{-1} R^T (t)H - H_{\text{cmg}}(\psi, \theta, \Omega)) \]  

(4.5)

Since 4.5 combines the kinematics and dynamics equation of 4.4, the system is fully defined on \( M_H \) by including \( \dot{\psi} = u_1, \dot{\theta} = u_2, \dot{\Omega} = u_3 \) to equation . The resulting equation define a control affine system of the form

\[ \dot{y}(t) = f_H(y(t)) + g_1(y(t))u_1(t) + g_2(y(t))u_2(t) + g_3(y(t))u_3(t) \]  

(4.6)

on the manifold \( N \), where \( y = (R, \psi, \theta, \Omega) \in N \) represents the body attitude and the DGVSCMG configuration. In this system

\[ f_H(R, \psi, \theta, \Omega) = (R(t)S(I_{\text{stat}}^{-1} R^T (t)H - H_{\text{cmg}}(\psi, \theta, \Omega)), 0) \]
\[ g_1(R, \psi, \theta, \Omega) = (0, e_1) \]
\[ g_2(R, \psi, \theta, \Omega) = (0, e_2) \]
\[ g_3(R, \psi, \theta, \Omega) = (0, e_3) \]  

(4.7)

where \( e_i \in \mathbb{R}^3 \) is a vector whose i-th element is 1 and the remaining 0.

The resulting system is relatively similar to the one from [35] regarding a spacecraft controlled by three single gimbal control moment gyros. In order to prove the controllability of system 4.6 on the respective manifold, two results must be proven: Proposition 4.1: For every \( H \in \mathbb{R}^3 \) the vector field \( f_H \) is weakly positively Poisson stable on \( N \); and Theorem 4.1: For every \( H \in \mathbb{R}^3 \), the system 4.6 is strongly accessible and controllable on \( N \).

For the first result, the proof presented in [35] can be directly applied to this case as it does not depend on the function \( v(\theta) \), in this case \( H_{\text{cmg}}(\psi, \theta, \Omega) \). For the second result, the authors start by proving the strong accessibility of system 4.6. For this case, the result still applies, but is rather different.

Let \( H \in \mathbb{R}^3 \). Let \( \xi_1 = [g_1, f_H] \), \( \xi_2 = [g_1, \xi_1] \) and \( \xi_3 = [\xi_1, \xi_2] \) be defined as the Lie Brackets and treating \( f_H \) and \( g_i \) as vector fields on \( \mathbb{R}^{3 \times 3} \times \mathbb{R}^3 \). Using the expression (1) of [35] to compute the Lie Brackets, the resulting expression of \( \xi_{11} \) is given by.

36
\[ \xi_{11} = [g_1, f_H] (R, \psi, \theta, \Omega) = \frac{d}{dh} |_{h=0} \left[ f_H((R, \psi, \theta, \Omega) + hg_1(R, \psi, \theta, \Omega)) - \hat{g}_1((R, \psi, \theta, \Omega) + h f_H(R, \psi, \theta, \Omega)) \right] \]
\[ = \frac{d}{dh} |_{h=0} \left[ f_H(R, \psi + h, \theta, \Omega) - (0, e_1) \right] = \frac{d}{dh} |_{h=0} \left( R(t) S(I_{stat}^{-1} \{ R^T (t) H - H_{cmg}(\psi + h, \theta, \Omega) \}), -e_1 \right) \]
\[ = - (R(t) S(I_{stat}^{-1} \partial H_{cmg}/\partial \psi(\psi, \theta, \Omega)), 0) \]

\[ \text{The remaining Lie Brackets are computed in the Appendix B.} \]

\[ \xi_{12} = (R(t) S(I_{stat}^{-1} \partial^2 H_{cmg}/\partial \psi^2(\psi, \theta, \Omega)), 0) \]
\[ \xi_{13} = (-RS(I_{stat}^{-1} \partial H_{cmg}/\partial \psi(\psi, \theta, \Omega)) S(I_{stat}^{-1} \partial^2 H_{cmg}/\partial \psi^2(\psi, \theta, \Omega)) + \]
\[ RS(I_{stat}^{-1} \partial^2 H_{cmg}/\partial \psi^2(\psi, \theta, \Omega)) S(I_{stat}^{-1} \partial H_{cmg}/\partial \psi(\psi, \theta, \Omega)), 0) \]
\[ = (RS(I_{stat}^{-1} \partial^2 H_{cmg}/\partial \psi^2(\psi, \theta, \Omega) \times I_{stat}^{-1} \partial H_{cmg}/\partial \psi(\psi, \theta, \Omega)), 0) \]

As in [35], \( \xi_{11} \) and \( \xi_{12} \) are linearly independent as the vectors \( \partial H_{cmg}/\partial \psi \) and \( \partial^2 H_{cmg}/\partial \psi^2 \) are perpendicular. \( \xi_{13} \) is linearly independent as it is also perpendicular to the others.

An analogous result can be obtained for the Lie Brackets regarding \( g_2 \) with the partial derivative with respect to \( \theta \) instead of \( \psi \). For \( g_3 \) the result is slightly different. In this case only \( \xi_{13} \) is defined as \( \xi_{13} = [g_3, f_H] \) since \( \xi_{23} = 0 \) and \( \xi_{13} = 0 \).

\[ \xi_{13} = [g_3, f_H](R, \psi, \theta, \Omega) = \frac{d}{dh} |_{h=0} \left[ f_H((R, \psi, \theta, \Omega) + hg_3(R, \psi, \theta, \Omega)) - \hat{g}_3((R, \psi, \theta, \Omega) + h f_H(R, \psi, \theta, \Omega)) \right] \]
\[ = (-R(t) S(I_{stat}^{-1} \partial H_{cmg}/\partial \Omega(\psi, \theta, \Omega)), 0) \]

This result is clearly linear independent from the other results \( \xi_{ij} \) as well as from \( g_3 \).

All Lie Brackets and functions \( g_i \) are linear independent between each other and contained in the strong accessibility algebra [35]. As result, the system 4.6 is strongly accessible. Strong accessibility implies accessibility, and assuming the weak positive Poisson stability of 4.6, the system is controllable in \( M_H \) following [35].

This result implies that, for every \( H \in \mathbb{R}^3 \), the set of states reachable from every \( z \in M_H \subseteq SO(3) \times \mathbb{R}^3 \times \mathbb{R}^3 \) is \( M_H \). This means that system 4.4 can be steered between any two states lying on the same constant angular momentum sub manifold. As expected, this result matches the one in [35]. In fact, it the result found in this article is independent from the number or arrangement of the CMGs in the CMG array, being for this case applied to a DGVSCMG.

The previous result can be applied to comprehend the ability of this system to steer to a desired combination of attitude and angular velocity - output controllability. In [35], Proposition 4.2 states the conditions for this characteristic to be applied. Defining the momentum volume of the CMG array as the set \( V = v(\mathbb{R}^9) \subseteq \mathbb{R}^3 \) of all possible angular momentum vectors of the CMG array, Proposition 4.2 states that \( "H \subseteq \mathbb{R}^3, \Theta \in \mathbb{R} \) and \( \gamma \in S^2 \) are such that \( V \) contains a sphere of radius \( |H| \) centered at \( -\Theta I \gamma \), then, for every \( \zeta \in S^2 \), every state on \( M_H \) can be steered to a state in which the body rotates instantaneously at the rate \( \Theta \) about a unit vector whose body components are given by \( \gamma \) and inertial
components are given by $\zeta^\pi$. This proof can be applied to system 4.4 with the conditions differing only in the definition of $V$. In the case of system 4.4, $V = H_{cmg}(\psi, \theta, \Omega)$ and assuming a general $H$, it follows that if $R^TH - \Omega I_{stat} \gamma \in H_{cmg}(\psi, \theta, \Omega)$, there exists $(\psi, \theta, \Omega) \in \mathbb{R}^3$ such that $R(\Omega I_{stat} \gamma + H_{cmg}(\psi, \theta, \Omega)) = H$.

To understand this, let’s assume a scenario where a system 4.4 is in an environment with no gravity and with its wheel spinning at an initial rate of $\Omega_0$. In this case $H = [0, 0, I_{wzz} \Omega_0]$. The set of possible combinations of attitude and angular velocity is given by

$$\Theta I_{stat} \gamma = R^T \begin{bmatrix} 0 \\ 0 \\ I_{wzz} \Omega_0 \end{bmatrix} - \begin{bmatrix} -I_{wzz} \Omega \sin(\psi) \\ I_{wzz} \Omega \cos(\psi) \\ -I_{wzz} \Omega \cos(\psi) \end{bmatrix}$$

Unfortunately, this analysis does not include state and input constraints. It is expected that the same results do not apply for that case. For example, assume a given initial $x_0$, characterized, among others variables, by a small wheel velocity $\Omega$. It is expected that this system might not be able to perform a $2\pi$ turn over one of its body axes $x$ or $y$. The main reason is found in the limitations on the gimbal axes rotations.

### 4.2 Singularities

In general, singularities are defined as a condition caused by the collinear alignment of two or more robot axes resulting in unpredictable robot motion and velocities [38]. In fact, when the system reaches a singularity state, the robot loses (at least) one of its degrees of freedom. As a consequence, its controllability properties change losing its global controllability.

This situation is well presented when working with double gimbal control moment gyros. It occurs when the outer gimbal axis is collinear with the reaction wheel axis [30]. In this case, an acceleration in the wheel produces the same effect as the acceleration in the outer gimbal axis, leading to a lost of one degree of freedom.

It is a complex task to analytically represent the singularity based only on the entire system presented in Chapter 3 (see [30]). For this reason, and for illustration purposes, the simplified system used for studying the controllability 4.4 is used.

When $\theta$ equals $\frac{\pi}{2}$ or $\frac{-\pi}{2}$, matrix 4.2 degenerates and looses one rank, resulting in

$$\begin{bmatrix} 0 & 0 & I_{wzz} \\ 0 & I_{wzz} \Omega \sin(\psi) & 0 \\ 0 & -I_{wzz} \Omega \cos(\psi) & 0 \end{bmatrix}$$

In a way, when the spin axis is parallel to the outer gimbal axis, the conventional DGCMG will degenerate into SGCMG and lose one degree of freedom control torque. That is to say, when DGVSCMG falls into the singularity state, the outer gimbal angular rate does not generate any control torque, no matter how large it is.
Finally, it is important to mention that the controllability of the system is not compromised. In [35], the authors prove that, for a system like this, the controllability in $M_H$ exists despite the existence of singularities. In addition, this state is almost impossible to reach, as the actuators of the robot have a finite area of actuation, being only able to rotate between $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$.

### 4.3 Control Architecture

At this point, the system dynamics are set, the constraints of the system are described and the controllability was studied. The objective from here is to design a controller able to perform the desired maneuvers. As it was established in the first chapter, the controller must be able to perform two maneuvers: locomotion and observation. In both maneuvers there exists a large steering movement from an initial attitude to a desired attitude.

#### 4.3.1 Nonlinear Dynamic Inversion

The first step for the design concerns the torque/gimbal acceleration relation. Presented in section 3 and the Appendix A, these equations are extensive and highly nonlinear. One solution for dealing with this system is to control the system by directly computing the inputs in torques from the objective (i.e., to include the entire system in the trajectory optimization solver). As result, the controller/solver will require more time to compute these inputs and errors can accumulate. Another solution is to include a Nonlinear Dynamic Inversion block [39] and separate the control problem.

Using the NDI controller, the new outputs of the controller are given in terms of $\ddot{\psi}, \dot{\theta}$ and $\Omega$. The advantage of this procedure is that the controller/solver needs only to consider the system (3.31). In addition, it separates the control problem into two smaller problems: one solved by the NDI approach (the computation of $u$), and the other solvable by a conventional attitude controller for DGVSCMG (the computation of gimbal and wheel accelerations).

Nonlinear dynamics inversion, in the input output perspective is obtained by following a set of steps. The first part concerns an evaluation of the relative degree $n_k$ of each output variable to be controlled.

According to the [39], the output $y_i = h_i(x)$ is said to have a relative degree $k_i$ if

$$
\left( L_{g_1} L_{f}^{j} h_i(x), L_{g_2} L_{f}^{j} h_i(x), \ldots, L_{g_m} L_{f}^{j} h_i(x) \right) = (0, 0, \ldots, 0)
$$

for $j = 0, 1, \ldots, k_i - 2$ but

$$
\left( L_{g_1} L_{f}^{k_i-1} h_i(x), L_{g_2} L_{f}^{k_i-1} h_i(x), \ldots, L_{g_m} L_{f}^{k_i-1} h_i(x) \right) \neq (0, 0, \ldots, 0)
$$

where,

$$
L_{g} h(x) = \frac{\partial h(x)}{\partial x} g(x) \quad L_{f} h(x) = \frac{\partial h(x)}{\partial x} f(x) \quad L_{g} L_{f}^{j} h(x) = \frac{\partial L_{f}^{j} h(x)}{\partial x} g(x)
$$

After computing the relative degree of each $h_i(x)$, the resulting expressions are organized in the following way.

39
The objective is to have this vector of output derivatives to equal the virtual input vector \( v \). This is only possible if \( A(x) \) is invertible for all \( x \). Rearranging the expression for the input \( u \), the result is

\[
    u = -A^{-1}(x)N(x) + A^{-1}(x)v
\]

Applying this formulation to the system defined in Chapter 3 with the output matrix \( h(x) \) is defined as

\[
    h(x) = [\dot{\psi}, \dot{\theta}, \Omega]^T
\]

Assuming that \( g_1(x), g_2(x) \) and \( g_3(x) \) follow from the definition of affine dynamical system and are defined for system (3.37) by

\[
    [g_1(x), g_2(x), g_3(x)] = M^{-1}
\]

The first step is to compute the relative degree. Due to the relation presented above plus the definition of relative degree, it is possible to determine that all outputs in \( h(x) \) have relative degree 1. In addition, expression 4.14 is obtained from the state space equation 3.33 with \( N(x) = B \) and \( A(x) = M \).

The nonlinear dynamics inversion expression is then

\[
    u = M\left([\ddot{\psi}, \ddot{\theta}, \dot{\Omega}]^T + B\right)
\]

A common practice to NDI is to determine the hidden state dynamics (states that are not directly involved in this controller) resulting from the implementation of this controller. Fortunately the new states are exactly the same as the states prior to the implementation of the NDI controller meaning that the hidden dynamics do not change.

The NDI controller is a common feature to all controllers developed in this work, as it will be seen in the following sections.

### 4.3.2 Open Loop

With an open loop system the control action from the controller is independent of the "process output" (the state variables that are being controlled). As result, system inputs are computed a priori regardless of the state of the system. In general, these controllers have an established control set, defined in time, and establishing a maneuver for the system to perform. In order to compute this set of inputs, a knowledge about the systems dynamics is often mandatory.

For Moonhopper, an open loop controller is developed that receives a set of inputs which result in a certain attitude maneuver. As it was stated above, this set of inputs are described as signals of \( \ddot{\psi}, \ddot{\theta} \) and \( \dot{\Omega} \). Later on, this open loop system receives the inputs resultant from the trajectory optimization solvers, The formulation is presented in the next chapter. For now, the open loop system is presented in the figure 4.1.

### 4.3.3 Close Loop - First approach

In chapter 2, two controllers for a DGVSCMG actuated body were presented. Both controllers are developed to follow pre defined trajectories, and so the first step is to develop a trajectory to be followed...
The first controller, here presented as a "classical approach" is described as PD controller with inverse dynamics. Based on the reference trajectory (defined by $q_d$ and $\omega_d$ at each time stamp) and the current state, this controller starts by computing a desired torque to be applied to the robot by the DGVSCMG. The torque is computed following the relation:

$$\dot{H}_{stat} + \dot{H}_G + \dot{H}_H + \dot{H}_W = \dot{H}_{stat} + \dot{H}_{CMG} = 0.$$  

$\dot{H}_{CMG}$ corresponds to the torques produced by the control moment gyroscope and is obtained by

$$\dot{H}_{CMG}(t) = K_P e_q(t) + K_D e_\omega(t) \quad (4.15)$$

with

$$e_q = \begin{bmatrix} q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} -q_{dx} & q_{dw} & q_{dz} \\ -q_{dy} & -q_{dz} & q_{dx} \\ -q_{dz} & q_{dy} & -q_{dx} \end{bmatrix} \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix}, e_\omega = \omega_d - \omega \quad (4.16)$$

the expression for $e_q$ corresponds to the error quaternion vector part. Finally, the computation of input values is performed following a dynamic inversion of system. This expression is presented in Appendix B. Based on the values for $\dot{H}_{CMG}$, and the relation $\dot{H}_{CMG} = \dot{H}_G + \dot{H}_H + \dot{H}_W$, the objective is to have an expression $\dot{H}_{CMG}$ that is a function of the input accelerations $[\dot{\psi} \ \dot{\theta} \ \dot{\Omega}]^T$ and the current state variables $[\omega \ \psi \ \theta \ \dot{\psi} \ \dot{\theta} \ \dot{\Omega}]^T$. This relation is described based on the expressions determined in the previous chapter. The result follows the structure $M_{H_{CMG}}(x)[\dot{\psi} \ \dot{\theta} \ \dot{\Omega}]^T + B_{H_{CMG}}(x)$, where the variable $x$ is used to represent the state. The schematic of the controller is presented in Figure 4.2. Inverting the dynamics results in an expression that has as inputs the state and the $\dot{H}_{CMG}$ and outputs the accelerations. It should be noted that $M_{H_{CMG}}$ is essentially determined by the inertia of the gimbals and wheel. As they are small, by inverting this matrix, it should be possible to expect high amplitude signals for the accelerations.
This way, it is used to maintain the attitude of the robot either for a desired view or for a safe landing.

As it is mentioned, this controller is a first approach to the problem and is used later as a way to compare the approach using trajectory optimization. The objective is to have a reference for the classical approaches, hence the denomination, and to compare the results with the optimized results.

4.3.4 Close Loop - Nonlinear Lyapunov Controller

Inspired by the variable speed single gimbal CMG (VSCMG) Lyapunov controller presented in [29], article [30] presents a different approach to the control problem. Defining a Lyapunov function for the control dynamics

\[ V(\delta \omega, \sigma) = \frac{1}{2} \delta \omega^T [I] \delta \omega + 2K \ln (1 + \sigma^T \sigma) \]

the authors state the conditions in which this function is valid:

\[ -[I] \dot{\omega} - \frac{1}{2} \frac{d}{dt} [I] \delta \omega = K \sigma + [P] \delta \omega - [I] (\dot{\omega}_r - \omega \times \omega_r) \]  

(4.17)

the definitions are according to the nomenclature presented in this document. In [30] the authors use \( \sigma \), the modified Rodrigues parameters (MRPs) of the current orientation relative to the desired orientation. For simplification the quaternions of the attitude system are converted into MRPs using a predefined function. Also \( \delta \omega = \omega - \omega_d \). In [30], equation 4.17 - control stability constraint - is modified and written in terms of the desired trajectory, current states and input acceleration. The resulting expression (presented in Appendix B) is organized as

\[ \dot{\Omega}_{des}a + \dot{\psi}_{des}b + \dot{\theta}_{des}c + \dot{\psi}_{des}\dot{\theta}_{des}d + \ddot{\psi}_{des}e = L_r, \]

with \( a-e \) being 3x1 vectors that are functions of the states. \( L_r \) is a function of the orientation error \( \sigma \), the angular velocity \( \omega \), the desired \( \omega_r \) and \( \dot{\omega}_r \) and the angular velocity of the wheel \( \Omega \), as well as the inertia matrices (see Appendix B). The control algorithm solves this nonlinear matrix equation using a Newton–Raphson (N–R) iteration, finding the values for \( [\dot{\psi} \dot{\theta} \Omega]^T \). The values of \( \dot{\psi} \dot{\theta} \) are obtained by a proportional controller:

\[ \ddot{\psi}_{srv} = -K \dot{\psi} (\dot{\psi} - \dot{\psi}_{des}) + \ddot{\psi}_{des} \quad \text{and} \quad \ddot{\theta}_{srv} = -K \dot{\theta} (\dot{\theta} - \dot{\theta}_{des}) + \ddot{\theta}_{des}. \]

\( \ddot{\psi}_{des} \) and \( \ddot{\theta}_{des} \) are the first order derivative of \( \dot{\psi}_{des} \) and \( \dot{\theta}_{des} \).

![Figure 4.3: Nonlinear control architecture](image)

This controller is a different approach to the PD invert dynamics. The reason for this approach to be chosen lies on the fact that it does not require matrices inversion. For this reason it is expected to have lower values for the acceleration inputs.

This controller is dependent of vector \( K \), matrix \( [P] \) and \( K_{\dot{\psi}} \), \( K_{\dot{\theta}} \). \( K \) is related to the attitude error, matrix \( [P] \), often diagonal, is related to the angular velocity error and \( K_{\dot{\psi}} \), \( K_{\dot{\theta}} \) are related to the proportional controllers that output the gimbal accelerations.
Chapter 5

Trajectory Optimization

The organization of this chapter follows a natural sequence of the trajectory optimization solving procedure, starting with choosing a method from Chapter 2. The next step is to describe the selected method and apply it to the Moonhopper system. The chapter ends with the presentation of the optimal trajectory problem and the transcripted optimization problem associated.

5.1 Direct Collocation Method

When it comes to trajectory optimization solvers there is a wide variety of methods, as it was seen in Chapter 2. Based on the characteristics described, for attitude control of the Moonhopper the chosen method is a direct collocation method. In fact, this class of methods is appealing for complicated applications and promises versatility. Not only because they can be applied without explicitly deriving the necessary conditions (i.e. adjoint, transversality, maximum principle), but also because they do not impose simple control expressions as solutions [40].

Direct transcription or direct collocation methods are defined by a prior discretization of the trajectory (transcription) followed by an optimization problem formulation [17]. The trajectory is discretized in a finite number of points \((t,x,u)\) that represent a finite number of decision variables. The transcription is done by representing the state vector by its values at specific points in time. These are known as collocation points.

\[
\begin{align*}
  t &\rightarrow t_0, t_1, \ldots, t_{N_{grid}} \\
  x &\rightarrow x_0, x_1, \ldots, x_{N_{grid}} \\
  u &\rightarrow u_0, u_1, \ldots, u_{N_{grid}}
\end{align*}
\]  

(5.1)

The next step is to convert the continuous system dynamics into a set of constraints that can be applied to the state and control at the collocation points. The key idea is that the change in state between two collocation points is equal to the integral of the system dynamics. This procedure is different, depending on the type of direct collocation method that is used. For the trapezoid method, the trapezoid rule is applied. When the Hermite Simpson method is used the Simpson’s rule for integration is used. For illustration purposes, from now on, the direct collocation trapezoidal method is applied and described [17].
\[
\int_{t_k}^{t_{k+1}} \dot{x} \, dt = \int_{t_k}^{t_{k+1}} f(x, u) \, dt \quad \Rightarrow \quad x_{k+1} - x_k \approx \frac{1}{2}(h_k)(f(x_{k+1}, u_{k+1}) + f(x_k, u_k)) \tag{5.2}
\]

\( h_k \) is the time interval. As a result of this numerical approximation, a set of equations that allow an approximation of the dynamics between each of the collocation points. This set of equations are enforced on every segment of the trajectory. For this reason a new set of constraints must be added to the optimization problem which is described by these equations. These are known as collocation constraints [17].

In the transcription step, all constraints of the trajectory optimization problems are converted into constraints of the NLP. In addition to the collocation constraints, there can also be limits on the state and control, path constraints, and boundary constraints. These constraints are all handled by enforcing them at specific collocation points [17].

The final step of the transcription phase is the approximation of the objective function. For trapezoidal collocation, trapezoid quadrature is use (Equation 5.3). For boundary objectives this step is not taken into account as the transcription for an optimization problem is straightforward. For path objectives, it is essential. This step is applied by converting the integral into a sum of the control effort at each collocation point.

\[
\min_{u(t)} \int_{t_0}^{t_N} u^2(\tau) \, \tau \approx \min_{u_0, \ldots, u_N} \sum_{k=0}^{N-1} \frac{1}{2}(h_k)(u_k^2 + u_{k+1}^2) \tag{5.3}
\]

From this point, it is possible to define the NLP optimization problem from the set of constraints plus objective function

\[
\min_{u_0, \ldots, u_N} \sum_{k=0}^{N-1} \frac{1}{2}(h_k)(u_k^2 + u_{k+1}^2)
\]

\[
\text{Variables} = \{x(t_0), \ldots, x(t_{N_{grid}}), \ u(t_0), \ldots, u(t_{N_{grid}})\}
\]

\[
\text{Collocation Constraints} : x_{k+1} = x_k + \frac{1}{2}(h_k)(f(x_{k+1}, u_{k+1}) + f(x_k, u_k))
\]

\[
\text{Constraints} : g_{\text{min}} \leq g(t, x, u) \leq g_{\text{max}}
\]

\[
\text{Boundary constraints} : \Psi_{\text{min}} \leq \Psi(t, x, u) \leq \Psi_{\text{max}}
\]

The result of the transcription presented above results from the trapezoidal collocation method. Other direct collocation methods vary in the function used for discretization, such as the Hermite–Simpson, the Orthogonal or the Pseudospectral Collocation methods [32]. In this work, only the trapezoidal collocation was used. The reason lies in the fact that this method tends to be faster than the rest to compute each iteration, allowing for a grid with more points [32].

After being defined, the optimization problem 5.4 is inserted in a NLP solver, such as Matlab’s fmincon(1), and a solution is provided. This solution corresponds to input values defined at each point of the grid. These points are interpolated using polynomial functions according to the method implied (for

1 https://nl.mathworks.com/help/optim/ug/fmincon.html
trapezoidal, a linear polynomial is used). These interpolated functions are defined in $t \in [t_0, t_F]$

5.1.1 Remarks - Quaternions discretization

Before applying the direct collocation method to Moonhopper system dynamics, a few corrections must be performed. The reader should recall that the attitude kinematics defined in Chapter 3 were represented in Euler angles and quaternions. When using Euler angles, no correction is required. However, for quaternions this is not the case.

The quaternions representation is a SO(3) universal cover and follows a specific (Clifford) algebra. Following this algebra, the quaternion subtraction (error quaternion) is obtained according to 5.5, where $q_{d} = [q_{dx}, q_{dy}, q_{dz}, q_{dw}]$ is the desired attitude quaternion and $q = [q_{x}, q_{y}, q_{z}, q_{w}]$ is the current quaternion.

$$
\begin{bmatrix}
q_{e_{w}} \\
q_{e_{x}} \\
q_{e_{y}} \\
q_{e_{z}}
\end{bmatrix}
= 
\begin{bmatrix}
q_{dw} & q_{dx} & q_{dy} & q_{dz} \\
-q_{dx} & q_{dw} & q_{dz} & -q_{dy} \\
-q_{dy} & -q_{dz} & q_{dw} & q_{dx} \\
-q_{dz} & q_{dy} & -q_{dx} & q_{dw}
\end{bmatrix}
\begin{bmatrix}
q_{w} \\
q_{x} \\
q_{y} \\
q_{z}
\end{bmatrix}
$$

(5.5)

For this reason 5.2 cannot be applied to the usage of quaternion representation. The discretization of these variables do not follow that algebra. In order to obtain a more accurate discretization of the quaternions to substitute in 5.2, [41] presents a First Order Quaternion Integrator. According to this, the quaternion propagation is obtained by 5.6.

$$
q(t_{k+1}) = \left( \exp \left( \frac{1}{2} \Omega(\dot{\omega}) \Delta t \right) + \frac{1}{48} \left( \Omega(\omega(t_{k+1})) \Omega(\omega(t_k)) - \Omega(\omega(t_k)) \Omega(\omega(t_{k+1})) \right) \Delta t^2 \right) q(t_k)
$$

(5.6)

The derivation of this expression can be found in [41]. It assumed a linear evolution of $\omega$ during the integration interval $\Delta t$. In this expression, $\exp()$ is the matrix exponential and $\Omega(\omega)$ is defined by the expression

$$
\Omega(\omega) = 
\begin{bmatrix}
0 & \omega_z & -\omega_y & \omega_x \\
-\omega_z & 0 & \omega_x & \omega_y \\
-\omega_y & -\omega_x & 0 & \omega_z \\
-\omega_x & \omega_y & \omega_z & 0
\end{bmatrix}
$$

(5.7)

The quantities $\Omega(\dot{\omega})$ and $\Omega(\ddot{\omega})$ are expressed as

$$
\Omega(\dot{\omega}) = \Omega \left( \frac{\omega(t_{k+1}) - \omega(t_k)}{\Delta t} \right), \quad \Omega(\ddot{\omega}) = \Omega(\omega(t_k)) + \frac{1}{2} \Omega(\dot{\omega}(t_k)) \Delta t
$$

(5.8)

5.2 Trajectory optimization problem

From Chapter 3, the dynamics are introduced in the trajectory problem, together with the modifications created by the NDI controller presented in Chapter 4. All combined, the resulting trajectory problem is

45
\[
\min_{\mathbf{u}(t)} \int_{0}^{T} (u_\psi(\tau))^2 + u_\theta(\tau)^2 + u_\Omega(\tau)^2 d\tau
\] (5.9)

\[
\var = \begin{bmatrix} q_x(t) & q_y(t) & q_z(t) & q_w(t) & \omega_x(t) & \omega_y(t) & \omega_z(t) & \psi(t) & \theta(t) & \Omega(t) & \dot{\psi}(t) & \dot{\theta}(t) \end{bmatrix}
\]

\textit{System dynamics}
\[
\begin{align*}
\dot{q}_x &= q_w \omega_x - q_z \omega_y + q_y \omega_z, & \quad \dot{q}_y &= q_z \omega_x + q_w \omega_y - q_x \omega_z \\
\dot{q}_z &= -q_y \omega_x + q_x \omega_y + q_w \omega_z, & \quad \dot{q}_w &= -q_z \omega_x - q_y \omega_y - q_x \omega_z \\
\dot{\omega} &= I^{-1} \left( -\omega \times [I_{max}] \omega - H'_G - H'_H - H'_W \right) \\
\dot{\psi} &= \ddot{\psi} & \dot{\theta} &= \ddot{\theta} & \dot{\Omega} &= u_\Omega \\
\dddot{\psi} &= u_\psi & \dddot{\theta} &= u_\theta
\end{align*}
\]

\textit{subject to}

\textit{Constraints}
\[-\psi_l \leq \psi(t) \leq \psi_u \quad -\theta_l \leq \theta(t) \leq \theta_u \quad \dot{\psi}_l \leq \dot{\psi}(t) \leq \dot{\psi}_u \quad \dot{\theta}_l \leq \dot{\theta}(t) \leq \dot{\theta}_u \forall t\]

\textit{Boundary constraints}
\[
q(t_0) = q_0 \quad \omega(t_0) = \omega_0 \quad \psi(t_0) = \psi_0 \quad \theta(t_0) = \theta_0 \quad \Omega(t_0) = \Omega_0 \quad \dot{\psi}(t_0) = \dot{\psi}_0 \quad \dot{\theta}(t_0) = \dot{\theta}_0
\]
\[
q_{t_F} \leq q(t_f) \leq q_{t_F u} \quad \omega_{t_F} \leq \omega(t_f) \leq \omega_{t_F u}
\]

The reader should note that the boundaries constraints present the initial state, enforced by the state right after take off. Because of an uneven balance, it is possible that the robot would start a jump with an initial angular velocity \(\omega\). Also, the initial attitude, here represented by \(q_0\) correspond to the inclination of the plane. The boundary constraints for \(t_F\) are concerned with the objective state the robot is supposed to achieve and is represented by a desired attitude and a desired angular velocity. All these features are covered in Chapter 6.

As a final remark, it should be noted that the resulting trajectory optimization problem is introduced in a already existing software named OptimTraj [17]. It implemented directly the algorithms described in [19] and was able to introduce some changes regarding the quaternions formulation. Additionally it is referred to present results such as in [17]. In fact, there are several other solvers as it is mentioned in [31] and their exploration can be a future development.

Finally, the trajectory optimization problem is discretized and rewritten as an optimization problem. The resulting problem is presented in Appendix C.

The trajectory optimization based controller, already mentioned, corresponds to an open loop controller that follows the inputs determined by the solution to this control problem. As it is presented in Chapter 6, for each maneuver an optimization problem is defined and solved. The results are in the form of an input set (correspondent to the optimized trajectory) that is introduced in the open loop NDI controller. The objective in this wok to understand the advantages of the application of a trajectory optimization. For that a set of tests are introduced in the next section.
Chapter 6

Results and Discussion

This chapter presents the experiments performed in this work. For each experiment, a description of the tests is presented together with the respective results. The chapter is divided in three parts starting with a characterization of the translational motion of the robot. Following this description, the second part comprises the verification of the state space model developed in Chapter 3. These two sections follow the first objective of this thesis which is to obtain an accurate model of the robot Moonhopper. The third section is dedicated to control and focuses on two objectives: first, to present the results of the application of trajectory optimization to the problem of attitude control; and second, and most important, the development of a efficient and robust attitude controller for the Moonhopper.

6.1 Translation

In order to predict the motion on the Moon’s surface, the respective behaviour of the robot on Earth’s surface was first predicted. The result was used to recreate the same conditions with the real prototype using ISR MOCAP system\(^1\). With this installation, different measurements of position and velocity of the CoM were performed with the robot departing from the same initial state (same deflection of the springs). These results were compared with the initial prediction resulting in the value of \(\eta\) for this prototype. With a fully description of the motion, an extrapolation for an environment on the moon is performed, obtaining the estimates for the robot time of flight upon take off. This value is of extreme importance, as it is one of the key parameters for the formulation of the trajectory optimization problem.

Prediction

In Chapter 3, the equations describing the release of the springs during the take off phase were presented. These equations described the reaction force performed by the ground during this phase. An expression for the work produced by the forces applied to the robot (ground reaction and gravity force) was also introduced, together with the relation between the energy, the maximum height and time in air.

\(^1\)http://welcome.isr.tecnico.ulisboa.pt/isrobonet/
\[ W = \int_{y_0}^{y_F} \left( k y \left( \frac{(c - l_0) + \sqrt{4a^2 - y^2}}{\sqrt{4a^2 - y^2}} \right) - m_{\text{robot}} g \right) dy \]

In order to predict the velocity of the Moonhopper, the first step is to compute the energy that is transmitted by the spring compression during the take off phase. Numerical methods were applied to solve this integral. Assuming the values for the Moonhopper in Table 6.1 (Notation in Chapter 3),

\[
\begin{align*}
&k(Nm^{-1}) & l_0(m) & a(m) & c(m) & m_{\text{robot}}(kg) & g(ms^2) & y_0(m) & y_F(m) \\
&982(\times 6) & 0.052 & 0.08 & 0.01 & 1.359 & 9.82 & 0.04 & 0.15
\end{align*}
\]

Table 6.1: Structural parameters of the robot

the result of the integration value of the energy deployed in this phase is \( W = 34.60 J \).

In ideal conditions, assuming \( \eta \approx 1 \), this value would result in a jumping height of 2.59m and a time of flight of 1.45s. This of course, is assumed in a vertical trajectory and following the expressions

\[
v_0 = \sqrt{\frac{2W}{m_{\text{robot}}}}, \quad h_{\text{max}} = \frac{v_0^2}{2g}, \quad t_F = \frac{2v_0}{g}
\]

Since this is not the case, the next step is to determine an estimate for \( \eta \). In Appendix a detailed analysis of the force curve is presented together with more results of this first prediction.

**Estimation of \( \eta \)**

The value of \( \eta \) represents the efficiency of the system. Due to disturbance forces such as friction and aerodynamic drag, and also due to model mismatch there are energy losses that are not account in this model. In fact, the model here presented is quite simple and does not take into account the mechanical system used for take off. As result, the value of \( \eta \) is expected to include this mismatch.

For this estimation, experiments with the real prototype were performed. These experiments consisted in measuring the jumping height the prototype could reach in several jumps. In order to perform these measurements, the robot legs were fully compressed. Reaching the state of full compression, the locking mechanism would be set on. When the system was unlocked, springs would compress and the robot would take off. Since the robot does not have a steering mechanism to control its attitude, the robot would have to be manually caught before reaching the ground.

In order to measure the position of the robot during the jump, the ISR MOCAP system was used. MOCAP systems are devices that record the movement of objects or people a in a 3D environment. The most usual setup for optical MOCAP systems is composed by calibrated infrared cameras (with a minimum of two, but typically more then six) arranged around the area to be captured, while passive (reflective) markers, with diameter between 9 to 25 mm, are placed on the surface of the object [42]. The results from the MOCAP system are obtained via ROS and treated using Matlab. Upon the application of an average filter, the results for the position and velocity were obtained. These are shown in Figure 6.1.

As it can be seen 6.1 the resulting jumping height on Earth is expected to be around 0.9m. Performing
an average over 5 jumps, the average was 0.916 m. Inverting the expression of $h_{\text{max}}$ and $v_0$ the results for the initial velocity and energy at take off are presented in Table 6.2.

<table>
<thead>
<tr>
<th>$h_{\text{max}}$ (m)</th>
<th>$v_0$ (ms$^{-1}$)</th>
<th>$t_F$ (s)</th>
<th>$W_{TO}$ (J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.916</td>
<td>3.6</td>
<td>0.97</td>
<td>12.242</td>
</tr>
</tbody>
</table>

Table 6.2: Results of real prototype translation

From this result, $\eta$ is obtained from $\eta = W_{TO}/W$ and the value is around 0.3538 (35%). The value is considerably lower than expected, specially when compared to the robot developed in [4] ($\approx 70\%$). The main differences between these two robots lie in the compression and locking mechanisms. The Moonhopper uses two gears connected to each other. One gear is connected to a motor shaft that transmits the torque responsible for the wire retraction and legs compression. The other gear is inserted on a bearing and rotates together with the first gear to curl the wire around itself. The locking mechanism is responsible for maintaining the two gears together blocking the gear with the wire. When the mechanism is unlock, the first gear disconnects from the second which rotates freely as the wire unrolls and the legs are decompressed.

During the experience, it was noticed that the bearing of one of the gears offers a considerable resistance to rotation due to friction. This fact is expected to be the main cause for the low jumping efficiency $\eta$. In addition, leg joints friction and model mismatch are expected also play key roles on this low value, by creating resistance during the decompression phase.

Moon’s surface estimates

Based on the result obtained in the previous section for $\eta$, an estimate for the $t_F$ and $h_{\text{max}}$ on the Moon’s surface is finally possible. Computing the value $W$ using the same values for the parameters presented in table 6.1, with the exception of $g = 1.62 m/s^2$, the result is $W = 35.78$. Applying the definition of $\eta$ the result is $W_{TO} = W\eta = 12.52$. From this, the values for $v_0$, $h_{\text{max}}$ and $t_F$ are computed following the respective expressions. The final result is shown in Table D.1
Discussion

The first observation that should be pointed out, is the fact that the result is a rough and conservative estimate of $t_F$ and $h_{\text{max}}$. In fact, the system presents a considerable complexity that was ignored. Instead, a simpler model was considered as the development of a complex model was out of the scope of this approach. In this case, the more conservative the estimate, the lesser the time the controller has to perform the maneuver. On the other hand, the more ambitious the estimate, the greater the chance the robot reaches the ground without the desired attitude. For this reason, a simpler and conservative estimate is enough for this stage of the project and for the objectives of this thesis.

Besides the simplicity of the model, another limitation of the results is its accuracy, which is compromised by the lack of diversity of experiments. One way of introducing diversity would be accomplished by performing tests with different degrees of leg compression. The reason for not including these tests also lies in two arguments. The first argument can be explained by the profile of the curve of reaction force (Appendix D). As it can be seen, the force to be applied to retract the wire is lower when the system is almost totally compressed allowing for a weak locking mechanism. In fact, when compressing the system half way, the locking mechanism struggled to act. The second argument is simpler and is due to the nature of operations. Since it is only a first generation prototype, the robot is only expected to perform full compression jumps on Earth and on the Moon. For this reason, only full compression jumps were tested.

Finally, the results show an unexpected low $\eta$ and consequently a low $t_F$. When the result was presented three reasons for the unexpected value were appointed: the friction on the bearing, the friction on the joints and a model mismatch. It is expected that by acting on these three causes, the values will increase. However, it should be also important to rethink the concept here described. Not only because the system lacks mobility, but specially, because the system requires too much energy to compress the legs. In the section dedicated to future work, a few suggestions are exposed.

### 6.2 Rotation

In this section, a verification of the state space model for the rotation motion is performed. This procedure consists on the application of methods that ensure the computer program of the computerized model and its implementation are correct [43]. The first method is a comparison to other models. It is used when various results (e.g., outputs) of the simulation model are compared to results of other models. This procedure is often described as a validation technique when the model to be used for comparison is already validated. Since this is not the case (the model developed in Gazebo was not validated with the real system), it is classified as a model verification technique. The second technique is the computation
of physical quantities used to verify the results of the system. These quantities are the total angular momentum $H$ and the the kinetic energy $T$.

### 6.2.1 ROS/Gazebo model verification

In a simple way, a Gazebo model is defined by a set of links and joints commanded by transmissions. In addition, a series of plugins can be included to perform measurements as sensors, or vision capabilities as cameras. Other features are also available.

A link contains the physical properties of one body of the model. It is defined by its collision properties, visual elements and inertial properties. For simplicity, collision properties were removed and most part of the visual elements regarding the robot body are represented by boxes. Regarding inertial properties, this element describes the dynamic properties of the link, such as mass and rotational inertia matrix. For mass, each element of the robot was weighted separately and the value inserted in the model. For the inertia matrices and location of the CoM, the CAD model of the robot was consulted. This model was developed in SolidWorks and follows the structure of the real prototype. As most parts of the prototype were developed using this software and 3D printed, only a few corrections had to be performed to account for the real mass. The matrices are presented in the Appendix A.

A joint connects two links. It is defined by a parent and child link relationship along with other parameters such as the axis of rotation, and joint limits. For this model all links that describe the robot body (legs, body and CMG frame) are connected using the 'fixed' joints. Joints that connect the gimbals (CMG frame - outer gimbal; outer gimbal - inner gimbal and inner gimbal - wheel) are all continuous and their direction corresponds to the definition in Chapter 3.

The continuous joints are each controlled by a transmission that enables controllers to act on the model. The transmission element is used to describe the relationship between an actuator and a joint. For these experiments, the transmission type that was used for all joints was 'effort joint'. This property simulates the inner torques applied to the links, as a servo motor would act.

Gazebo uses ODE (Open Dynamics Engine) solver to simulate the dynamics of the model. Several experiments were performed for model verification. In this thesis only two tests are presented and described. For each test the results are presented in terms of attitude (in Euler Angles), angular velocity, gimbal angles and rates and wheel velocity. For each variable the following sections present the results of the simulation of the model in Gazebo and the simulation of the state space model.

**Wheel step test**

The wheel step test is the response of the system to a step in the wheel joint, with no other torques applied. The amplitude of the step input is $0.05 \text{Nm}$ of $u_W$. The following figures (6.3 - 6.2) represent the evolution of the states as response to the input signal.
Outer gimbal - Inner gimbal step with rotating wheel test

The second test here presented corresponds to a maneuver that involves the three input torques ($u_G$, $u_H$ and $u_W$). The maneuver starts with an acceleration of the wheel caused by a step in $u_W$. When
Figure 6.5: $u_W$ step test - Joints rates

Figure 6.6: $u_W$ step test - Gimbal axes angles

$t = 5\, s$ the input signals for $u_H$ and $u_W$ are started (see figure 6.11). Figures (6.7 - 6.11) present the states and inputs related to this maneuver.

Figure 6.7: Outer gimbal-Inner gimbal step - Orientation in Euler Angles. The red line represents values of the orientation output from the state space model. The blue line represents values from simulation
Figure 6.8: Outer gimbal-Inner gimbal step test - Angular velocity.

Figure 6.9: Outer gimbal-Inner gimbal step test - Joints rates

Figure 6.10: Outer gimbal-Inner gimbal step test - Gimbal axes angles
Discussion

As a result of these experiments it is possible to observe that, in general, the outputs of the model match with the simulation outputs. In fact, it is seen that the errors are quite small for most of the cases. The only mismatch found for the orientation in the second test. It is seen that for the $\psi_{roll}$ and $\theta_{pitch}$ angle signals, the relative error is considerably high after $t = 10s$. However, the shape of the signal remains similar although offset by a fixed error. Also, the reader should observe that these values where the error occurs are small (no more than 2 deg) and more prominent when sudden changes appear.

This test was deliberately chosen to document some errors that might be associated with the model in Gazebo (assuming the model is correct). In fact, it is important to mention that both models were forced to have friction introduced to the joints as the framework was not expected to perform well with ideal joints (no friction). In addition to this unexpected behaviour, it is logical to think that this framework values efficiency more than accuracy. For this reason some errors due to simplifications and truncation are expected. It is true that this software might not be the best choice for model verification, however, for the future it is a useful resource for simulating other systems, hence the choice.

6.2.2 Physical quantities verification

Due to problems with the framework described above, additional verification of the model is performed using physical quantities. This section is divided in two parts with each subsection dedicated to one physical quantity. The results used to compute these quantities were obtained from the last experience (Outer gimbal-Inner gimbal step test) mentioned in the last section.

Conservation of angular momentum

As the title suggests, in this section, the angular momentum of the system is computed. As the robot starts the simulation still the initial value for the angular momentum is 0 in all directions. As result of this experiment, the angular momentum vector is expected to remain constant throughout time. The results are shown in figure 6.14.
As expected the angular momentum vector remains constant despite the apparent motion of the robot. Regarding the simulation, the results show a visible variation of the angular momentum vector. This result is used to show that conservation principle does not occur in the simulation. This can be the cause of the errors presented in the simulation.

**Kinetic energy**

As stated in [30] good way to understand if the simulation is running correctly is to compare the numerical derivative of the analytical expression of the kinetic energy

\[
T = \frac{1}{2} \omega^T [I_{LBS}] \omega + \frac{1}{2} \omega^T [I_G] \omega_G/N + \frac{1}{2} m_G v_G^2
+ \frac{1}{2} \omega^T [I_H] \omega_H/N + \frac{1}{2} m_H v_H^2 + \frac{1}{2} \omega^T [I_W] \omega_W/N + \frac{1}{2} m_W v_W^2
\]  

(6.1)

and the analytical expression for the kinetic energy derivative

\[
\dot{T} = \omega^T L + \dot{\psi} u_G + \dot{\theta} u_H + \Omega u_W
\]  

(6.2)

In this case \(L = [0, 0, 0]\). Both expressions are compared and the results are presented in Figure 6.13.

The error is only considerable at \(t = 21\) s. It is shown that this error is due to the computation of the numerical derivative as by reducing the sampling time leads to the reduction of this value. Nevertheless
its relative value considered negligible.

**Discussion**

As it was seen in the first part of this section, the angular momentum remains constant throughout the experience, proving the conservation of angular momentum. This is not the case for the Gazebo model, whose angular momentum is not conserved. This experience proves that this principle is valid for this system, corresponding to the expected. For the kinetic energy, both the numerical derivative and the analytical expression are equal during the experience (hence the blue line does not appear). In addition, the variation of the kinetic energy is concluded to be related to the actuation.

### 6.3 Moonhopper Control

This section is dedicated to the solution for the Moonhopper control problem. First, this section focuses on the application of trajectory optimization methods, as well as understanding the advantages of this procedure when compared to a classical approach. For this purpose the two controllers (PD+inversion) and the open loop trajectory optimization based are applied and compared. Four trajectories that represent the normal functioning of this system are defined and followed by these two controllers: two locomotion maneuvers and two observation maneuvers. The goal is to compare the performance of both controlled systems in terms of the objectives achievement as well as energy consumption.

With the results from this comparison, the final controller is presented. Ideally it combines robustness of a closed loop system as well as efficiency of a trajectory optimization.

#### 6.3.1 Locomotion vs Observation

As it was described, the Moonhopper must be able to perform two maneuvers: locomotion and observation. The locomotion maneuver presents the required attitude maneuver for the robot to move from one point to another. When performing locomotion, the robot starts with an initial attitude defined by the inclination of the terrain and an initial angular velocity caused by disturbances (rough terrain, off centered location of the CoM). The objective of this first maneuver is to achieve a safe state before reaching the ground (at $t = 5s$). The safe state is defined by an attitude and an angular velocity safety interval. This interval is an estimate based on the dimensions of the robot and represents the conditions for a safety landing. It is presented in Table 6.4.

<table>
<thead>
<tr>
<th>$\phi_{roll}(^\circ)$</th>
<th>$\phi_{roll_u}(^\circ)$</th>
<th>$\theta_{pitch}(^\circ)$</th>
<th>$\theta_{pitch_u}(^\circ)$</th>
<th>$\psi_{yaw}(^\circ)$</th>
<th>$\psi_{yaw_u}(^\circ)$</th>
<th>$\omega_l$(rad.s$^{-1}$)</th>
<th>$\omega_u$(rad.s$^{-1}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3</td>
<td>3</td>
<td>-3</td>
<td>3</td>
<td>-</td>
<td>-</td>
<td>-0.02</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Table 6.4: Safety landing conditions

If the system reaches the ground in these conditions, it is considered a safety landing. These values are supposed conditions and serve only for illustrative purposes.
The observation maneuver makes use of the jumping capabilities of the robot to observe a target located in an area close to the body. Upon take off the robot departs with an attitude of \([0, 0, 0]\) (roll, pitch and yaw) and an arbitrary initial angular velocity caused by disturbance torques. Before reaching the highest point of its trajectory, the robot must be able to achieve an orientation that enables an observation over the target area. Using an attitude hold controller, the actuators hold this orientation during one second (enough time to perform measurements and collect images), while the robot reaches the highest point and starts the downward trajectory. During this final phase, the system must be able to achieve a safety attitude described by the safety conditions presented above.

These two maneuvers here presented represent the conceptual functioning of this robot. Hopping from point A to point B (locomotion) requires special care to the attitude maneuvers as it can compromise the mobility of the robot. Additionally, the robot can reach a considerable jumping height (with a respective large time interval) on a low gravitational environment. With this characteristic, the attitude control can take advantage for monitoring the space around the robot.

### 6.3.2 Experiments description

The experiments applied in this work are illustrative of the missions to be performed under the conditions of the Moon’s surface. These conditions are mainly represented in the form of a different gravity acceleration and consequently a different time of jump (approximately 5 seconds of jump - result from translation tests). In addition, disturbances are introduced to represent the terrain roughness. Also, effects caused by aerodynamic drag are ignored.

In these experiments, both modes (locomotion and observation) are tested. For each mode two trajectory objectives are suggested. For the locomotion mode, the trajectory objectives are defined by an initial state. The controlled system should be able to reach a safety attitude before reaching the ground (in 5s). These two initial states are defined in tables 6.5 and 6.6.
The first initial attitude corresponds to the quaternions initial conditions \( q_x = 0.342, q_y = 0, q_z = 0, \)
\( q_w = 0.940. \) The second is \( q_x = 0.350, q_y = 0.078, q_z = 0.416, q_w = 0.835. \)

For the observation mode, the trajectory objectives are defined by an initial state and a desired
attitude for the robot to be oriented. As explained, the Moonhopper must be able to reorient itself and
reach the desired orientation. Then, the system must be able to return to a safety state before reaching
the ground. These trajectory objectives defined in tables 6.7 and 6.8.

In both modes, the robot starts with the wheel spinning at \( 565 \text{ rad.s}^{-1}, \) the value for a regular hard
disc. The next sections present the results for each control architecture. Special attention is devoted to
the implementation of the trajectory optimization system and description of the optimization problem.

### 6.3.3 Classic attitude controller

The first controller to tested is the PD+model inversion controller. Given a (pre-defined) desired trajec-
tory connecting an initial state to a desired state, the torques to be applied by the actuator at each time
\( t \) are firstly computed. From these values, inputs in the form of \( \ddot{\psi}, \ddot{\theta} \) and \( \dot{\Omega} \) are obtained using dynamics
inversion. These signals are then converted to \( u_G, u_H \) and \( u_W \) using the NDI controller.

**Locomotion mode**

The first locomotion maneuver is presented in figures (6.15-6.18). The complete set of results for each
maneuver comprises the evolution of the states - Euler angles, \( \omega, \psi, \theta, \Omega, \dot{\psi} \) and \( \dot{\theta} \) - and the inputs as
gimbal accelerations - \( \ddot{\psi}, \ddot{\theta} \) and \( \dot{\Omega} \) - and as torques - \( u_G, u_H \) and \( u_W \). Although the results were obtained
using quaternion formulation, the Euler angles were used for this representation due to its more intuitive
interpretation. This structure is followed for the next maneuvers.

The second locomotion maneuver is presented in figures (6.19-6.22).
Table 6.8: Objectives for observation test - 2

<table>
<thead>
<tr>
<th>$\phi_{roll}$ ($^\circ$)</th>
<th>$\theta_{pitch}$ ($^\circ$)</th>
<th>$\psi_{yaw}$ ($^\circ$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-55</td>
<td>40</td>
<td>40</td>
</tr>
</tbody>
</table>

Observation mode

The presentation of the results regarding the two observation maneuvers are presented in figures 6.23-6.26 for the first maneuver and 6.27-6.30 for the second maneuver. The following figures follow the standard presentation defined for the locomotion maneuvers and present the states and inputs. Figures 6.23 and 6.27 lack the reference signal. This mainly due to the simplification purposes as too many lines would not be desirable. In this approach, the PD+Dynamics inversion controller is responsible for the sub trajectories between $t = [0, 2]$ and $t = [3, 5]$. During the 1 second maneuver between these, the reference angular velocity is set to 0 and the reference attitude is set to be the desired attitude. This corresponds to the attitude hold controller.

Comments on the results

The first observation to be mentioned is the accomplishment of all objectives. As it can be seen both locomotion and observation maneuvers are accomplished with success, with only the second observation maneuver $\phi_{roll}$ with a small deviation from the safety interval (Figure 6.27). In fact, in all observation maneuvers, the desired attitude is obtained and hold with a small angular velocity, almost zero. These conditions are ideal for capturing images.

Regarding the gimbal axes it is clear the impact they have on the attitude control. However, this is done at a high cost in terms of velocity of the gimbal axes which are translated in high amplitude maneuvers over a short period of time. This feature is more easily perceived for the observation maneuvers. Nevertheless, in general, the angular velocity remains inside the available range (see optimization.

Figure 6.15: Locomotion test 1: Classic attitude controller - Attitude and angular velocity. On the left image, the blue line represents the $\psi_{yaw}$ angle, red line is the $\theta_{pitch}$ angle and the yellow line is $\phi_{roll}$ angle. This code of colors is used for the classic control results.
Figure 6.16: Locomotion test 1: Classic attitude controller - Gimbal Axes and Wheel velocity

Figure 6.17: Locomotion test 1: Classic attitude controller - Gimbal inputs

Figure 6.18: Locomotion test 1: Classic attitude controller - Wheel inputs

boundaries). The other notable observation is the high gimbal accelerations requests, specially during the attitude hold phase. The reason for that is discussed in [27], and is related to the small inertia matrices of the gimbals. Inverting these inertia matrices leads to high acceleration inputs which causes this type of behaviour.
Although gimbal accelerations values are quite high, the input torque are actually small in comparison, reaching values between -0.4 and 0.2 N.m for the worse case scenario. In fact, this specially due to the topic discussed in the previous paragraph and the fact that all other variables are several degrees of order lower than this value.

Finally, the role of the wheel is critical for attitude control. In fact, as [27] (page 415) points out, the wheel play two different roles in this control: reaction wheel, which is the reason for this high amplitude oscillations, and torque amplification, which is explored with the gimbal axes motion. It is clear that both roles are being played, but such a high oscillation can be mean that the reaction wheel effect is prominent. This might be explained by the way the control architecture is designed: following a specific attitude trajectory requires precision, and the torque amplification, although important, is not ideal for stabilization as much as it is for pushing the robot to perform great maneuvers. Thus, enter the action of the wheel that improves precision. As soon as the robot reaches the vicinity of the desired attitude and
the attitude stabilizes, the wheel velocity drops.

6.3.4 Open loop trajectory optimization controller

The section presents the results for the proposed trajectories applying the trajectory based control. The section starts with a few implementation details. Then the results are presented. These trajectories were obtained from the simulation of the state space system subject to the input sets obtained from the trajectory optimization solver (OptimTraj) for each proposed maneuver.

Implementation details

As presented, the implementation of the trajectory optimization solver requires the definition of the problem and the choice of method. The definition of the problem is done by choosing the cost function and introducing the dynamics and respective constraints. Recalling chapter 3, the system dynamics
model that was introduced was the one that uses the NDI controller. Regarding the cost function, for this
Figure 6.26: Observation test 1: Classic attitude controller - Wheel inputs

Figure 6.27: Observation test 2: Classic attitude controller - Attitude and angular velocity

Figure 6.28: Observation test 2: Classic attitude controller - Gimbal Axes and Wheel velocity

system two were used:

\[
\int_{t_0}^{t_f} \left(u_G^2 + u_H^2 + u_W^2\right) \, d\tau, \quad \int_{t_0}^{t_f} \left(\ddot{\psi}^2 + \ddot{\theta}^2 + \ddot{\Omega}^2\right) \, d\tau
\]
The first cost function directly penalizes the torques and consequently, the energy consumed. The second cost function directly penalizes the input accelerations and is applied to the system using the inner loop NDI controller. Both are related as the gimbal accelerations are functions of the inputs. As it was discussed earlier, the torque input is computed using an highly nonlinear expression. When applied to the solver, the introduction of this expression increases significantly the time of computation of each iteration. In this perspective, the use of the second cost function reaches the results faster and for that reason was preferred. An other advantage of using the second cost function is that it enforces the solution to have a smooth behaviour leading to less oscillations while penalizing high energy cost trajectories.

Regarding the constraints, only time, state and input boundaries were introduced to the system. The unit quaternion constraint was not introduced because it is already enforced by the quaternion propagation equation. These boundaries are

<table>
<thead>
<tr>
<th>$t_F$</th>
<th>$\psi_l$</th>
<th>$\psi_u$</th>
<th>$\theta_l$</th>
<th>$\theta_u$</th>
<th>$\dot{\psi}_l$</th>
<th>$\dot{\psi}_u$</th>
<th>$\dot{\theta}_l$</th>
<th>$\dot{\theta}_u$</th>
<th>$u_{G_l}$</th>
<th>$u_{G_u}$</th>
<th>$u_{H_l}$</th>
<th>$u_{H_u}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>$-\pi/2$</td>
<td>$\pi/2$</td>
<td>$-\pi/2$</td>
<td>$\pi/2$</td>
<td>-10</td>
<td>10</td>
<td>-8.7</td>
<td>8.7</td>
<td>-0.2</td>
<td>0.2</td>
<td>-0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 6.9: State and input constraints
The constraints introduced for $\psi$ and $\theta$ represent the limitations of the gimbal angle displacement. $\dot{\psi}$ and $\dot{\theta}$ boundaries correspond to the velocity of the DC motor with no load. This corresponds to the max velocity that is reached by the motors. Control boundaries $u_i$ were also obtained from the data sheet and correspond to the maximum load of each motor. No constraints to the wheel were introduced due to the lack of information about the wheel, however, by penalizing the wheel acceleration (cost function), it is expected a significant reduction of the wheel speed and torque.

The objective states for each trajectory are defined by a initial and final state constraints. These constraints are defined by the desired attitude and the safety interval for the angular velocity.

As mentioned in Chapter 4, the method of choice was the the trapezoidal direct collocation. This method is defined by a transcription process where both the dynamics and cost function are discretized using the trapezoidal rule. For this discretization, it is required to define the number of points that is used for this process. These points are equal equally distributed in time and are related through the collocation constraints, forming a grid. The number of points is an important parameter to be determined. In one way, the a low number of points results in less accuracy, while in another way too many points and the solution requires too much time to be computed. Note that the complexity of the problem increases exponentially with the number of grid points (for each grid point added, the number of variables to be defined is increased by the number of states). In this work, every point is separated by 0.1s from the next point resulting in 21 points for 2s trajectories and 51 for 5s trajectories. The impact of the number of points is discussed in [32].

A correct implementation of the trajectory optimization solver requires a fine tuning of several other parameters. In fact, as it was described, the trajectory optimization solver applied (OptimTraj [32]) makes use of Matlab’s fmincon() solver to compute the solutions of the optimization problem resultant from the transcription process. Apart from the optimization problem itself, the NLP receives a set of optional parameters such as the ‘ConstraintTolerance’ and the ‘StepTolerance’ whose fine tuning can influence the accuracy and time it takes for obtaining the results. The values of the hyperparameters used for the simulation are presented in Appendix D.

**Locomotion mode**

The implementation of the locomotion mode objectives is straight forward. After defining the dynamics, constraints and cost function, the trajectory objectives are inserted using boundary constraints. For locomotion, no partition of the trajectory is required, so both constraints follow the same structure. In table 6.10, the boundary conditions for the first locomotion maneuver are presented. For the second, the boundaries are presented in 6.11. The results for the first locomotion test are presented in figures (6.31-6.34) and for the second (6.35-6.38).

<table>
<thead>
<tr>
<th>$q_{x0}$</th>
<th>$q_{y0}$</th>
<th>$q_{z0}$</th>
<th>$q_{x_f}$</th>
<th>$q_{y_f}$</th>
<th>$q_{z_f}$</th>
<th>$q_{w_f}$</th>
<th>$\omega_0$</th>
<th>$\omega_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.342</td>
<td>0</td>
<td>0</td>
<td>0.940</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$\pm 0.02$</td>
</tr>
</tbody>
</table>

Table 6.10: Boundary constraints - Locomotion test 1
<table>
<thead>
<tr>
<th>$q_{x0}$</th>
<th>$q_{y0}$</th>
<th>$q_{z0}$</th>
<th>$q_{w0}$</th>
<th>$q_{xF}$</th>
<th>$q_{yF}$</th>
<th>$q_{zF}$</th>
<th>$\omega_{x0}$</th>
<th>$\omega_{y0}$</th>
<th>$\omega_{z0}$</th>
<th>$\omega_{xF}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.350</td>
<td>0.078</td>
<td>0.416</td>
<td>0.835</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.2</td>
<td>0.05</td>
<td>-0.1</td>
</tr>
</tbody>
</table>

Table 6.11: Boundary constraints - Locomotion test 2

Figure 6.31: Locomotion test 1: Trajectory optimization - Attitude and angular velocity. On the left image, the blue line represents the $\psi$ yaw angle, red line is the $\theta$ pitch angle and the yellow line is $\phi$ roll angle. This code of colors is used for the classic control results.

Figure 6.32: Locomotion test 1: Trajectory optimization - Gimbal Axes and Wheel velocity

**Observation mode**

In order to apply trajectory optimization to the observation maneuver, the 5 seconds observation trajectory is partitioned in three sub trajectories defined by an initial state and an objective state. The first sub trajectory lasts 2s, starts with the robot’s initial trajectory and finishes when the robot reaches the desired attitude for observation. The second sub trajectory corresponds to a 1 second attitude hold and is carried out by an attitude holder controller. When $t = 3s$, the last sub trajectory starts. It lasts for 2s and during this time the robot regains its original attitude (inside the safety range). The reader should note that only the first and the last sub trajectories are subjected to trajectory optimization. The second
Figure 6.33: Locomotion test 1: Trajectory optimization - Gimbal inputs

Figure 6.34: Locomotion test 1: Trajectory optimization - Wheel inputs

Figure 6.35: Locomotion test 2: Trajectory optimization - Attitude and angular velocity

is carried out by a previous defined attitude hold controller.

The initial sub trajectory objectives are defined by the initial and desired attitude states. The last sub
trajectory is defined by the state at $t = 3s$ and the safety conditions. In order to obtain the states at $t = 3s$, a simulation of the first two trajectories is performed, ending at $t = 3s$. At that point, the state is measured and the result is used for optimization.

$$
\begin{array}{cccccccccccccc}
q_{x_0} & q_{y_0} & q_{z_0} & q_{w_0} & q_{x_d} & q_{y_d} & q_{z_d} & q_{w_d} & q_{x_F} & q_{y_F} & q_{z_F} & q_{w_F} & \omega_0 & \omega_d & \omega_F \\
0 & 0 & 0 & 1 & 0.337 & 0.163 & 0.059 & 0.925 & 0 & 0 & 0 & 1 & 0 & 0 & \pm 0.02
\end{array}
$$

Table 6.12: Boundary constraints - Observation test 1

$$
\begin{array}{cccccccccccccc}
q_{x_0} & q_{y_0} & q_{z_0} & q_{w_0} & q_{x_d} & q_{y_d} & q_{z_d} & q_{w_d} & q_{x_F} & q_{y_F} & q_{z_F} & q_{w_F} & \omega_0 & \omega_d & \omega_F \\
0 & 0 & 0 & 1 & -0.340 & 0.439 & 0.118 & 0.823 & 0 & 0 & 0 & 1 & 0 & 0 & \pm 0.02
\end{array}
$$

Table 6.13: Boundary constraints - Observation test 2

The results of the first observation maneuver are presented in figures (6.39-6.42) and of the second
in figures (6.43-6.46).

![Figure 6.38: Locomotion test 2: Trajectory optimization - Wheel inputs](image1)

![Figure 6.39: Observation test 1: Trajectory optimization controller - Attitude and angular velocity](image2)

**Comments on the results**

Just like for the classic controller, the results of trajectory optimization accomplished the desired objectives. For both locomotion maneuvers, the robot achieves a safe attitude with almost 0 rad.s\(^{-1}\) of angular velocity, before reaches the ground. In addition, in both these maneuvers, but specially the first, it is possible to observe the variation of the \(\theta_{\text{pitch}}\) and \(\psi_{\text{yaw}}\) angles (Figure 6.31). This behaviour is not expected to be in a desired trajectory. Nevertheless, the robot reaches the safe attitude range.

From the first to the second locomotion maneuver, just like in the classical controller results, the effort that is implied by the actuators is significantly high. The reason for this extra effort is because it requires control over the three attitude axes. Also, the initial angular velocity requires the actuation to counter its resulting movement. The same increased effort happens between the two observation maneuvers, but
this time it is only due to the control over the tree axis, instead of just two, since both maneuvers start with angular velocity.

It is clear that for a maneuver that just requires the control over the $\psi_{\text{roll}}$ angle, the main actuation is on the outer gimbal. This can be concluded by observing 6.32 and comparing the velocity and amplitude.
of both gimbal angles. The actuation of over the inner gimbal is used mainly to position the wheel axis and maintain control over the precession. The same can be concluded for the classic control system.
The high values of angular acceleration, specially found in the observation trajectories are again present. In this case however, the requirements to the actuator are less demanding.

One important feature to mention is the account for the system constraints. In observation 2, the outer gimbal angle $\psi$ reaches a value close to the limit $\pi/2$ rad, or $90^\circ$. This detail can be seen in figure 6.44 before $t = 2s$. This feature also shows the limitations of the system for accomplishing this maneuver. In fact, tests were performed for more demanding maneuvers such as the one presented in [9]. When trying the maneuver described in this article (rotation of 179 degrees around the axis $[1 1 0]$ from the origin followed by a return to the initial orientation) the trajectory optimization solver was not able to reach convergence. The reason is mainly because the solution trajectory did not respected the state and input constraints.

Once again, the actuation of the wheel is again of extreme importance for accomplishing and maintaining the desired attitude. The reason is the same as for the classic controller. However, for the trajectory optimization maneuvers, the wheel speed is considerably smaller than the same for the previous controller. In addition, the curves of $\dot{\Omega}$ are considerably different. This results in a relaxed maneuver, where only the focus is to achieve the desired state. This approach leads to behaviours just like what was mentioned in the first paragraph. On the other hand, it is expected to lower the energy consumption associated with this maneuver.

Regarding the wheel, it is clear the high velocities reached both in the PD+dynamics inversion controller and the trajectory optimization controller. The reason is due to the low inertia of the wheel. It is expected that a wheel with higher inertia will not need to reach such high velocities. This feature should be considered in the redesign of the robot.

Finally, a short remark on implementation features regarding the convergence of the trajectory optimization solver. It was observed that for both locomotion maneuvers, the solver was able to converge in less that 160 iterations. The number of grid points was 51 (all grid points separated by 0.1s). The time the solver took was approximately 20 minutes for a 3.5GHz computer running Matlab. For both first sub trajectories of the observation maneuvers, the number of iterations was considerably less. Around 90 iterations, which are explained by a significantly less number of grid points - 21. For the third sub
trajectories, the number of iterations was considerably higher, reaching the 120 iterations. This is mainly due to the fact that the system departs from an initial state that is not only defined by the orientation and angular velocity. $\psi$, $\theta$ and the remaining states depart with a non zero value. In general, this method takes a long time to be computed, which leads to conclude that the direct collocation method is not desirable to be working in real-time, as was firstly suggested. Instead, trajectory optimization is used to compute the desired trajectory. The trajectory will then be followed by a controller.

### 6.3.5 Discussion

In order to evaluate the impact of this optimization, the energy spent for all maneuvers was computed. The expression corresponds to an integration of function $\dot{T}$, equation 6.2, over the total number of points that define the trajectory. The results were computed for a system with ideal joints and for a system with joint friction. The energy spent for an ideal joint model is found in table 6.14 for the optimized trajectory and 6.15 for the classic controller. These values are here presented to be compared to the result in [9]. For the model with friction the results are significantly higher and are presented following the same structure in tables 6.16 and 6.17.

The increased order of the results for the model with joint friction are mainly due to the extra effort required to maintain the wheel velocity at high values.

Based on the results for the ideal joints, it is clear the advantage of a trajectory planning performed by trajectory optimization techniques, based on the difference between the energy spent. In general, for simpler maneuvers it is clear the advantage on pre-planning the trajectory. In addition, it is possible to conclude that for maneuvers that do not require actuation over one axis, the difference is larger. This happens with locomotion 1 and observation 1. On the other hand, maneuvers that require direct actuation over the three axes, the difference is not as visible.

Comparing the ideal joints with the results in [9] it is clear that the values for the energy consumption during a maneuver are similar and have the same order of magnitude. More cannot be said as the values available in [9] are only for a specific maneuver.
6.3.6 Nonlinear Lyapunov controller

Using an open loop system to control an autonomous robot that is designed to explore remote environments is not an advisable practice. The main reason is that these controllers are not robust to disturbances nor model inaccuracies. This problem increases when the environment is unstructured such as the Moon’s surface. For this reason, the results of the open loop trajectory optimization controller are not enough by their own. Instead, their optimization results should be used by an closed loop system to perform the desired trajectories. This topic is tested in this section. For that purpose the nonlinear Lyapunov controller developed in [30] is used and adapted to this specific problem. The resulting control system is set to follow the trajectory developed in the observation 2 maneuver. The results are found in 6.47-6.50.

![Figure 6.47: Observation test 2: Lyapunov + Trajectory optimization - Attitude and angular velocity](image)

Comparing these results with the respective for the open loop system (6.43-6.46) it is possible to see that they are both similar in all states. In addition, the result for the energy consumed during the maneuver is equal to 80,08J (in ideal model). This value is close to the one from the open loop system.

The reason for choosing this controller instead of the (PD+Dynamic Inversion) is because the nonlinear controller does not require a matrix inversion, meaning that it is expected to be faster than the other. Additionally, in experiments, the inversion of the matrix output higher acceleration inputs than the nonlinear controller.

Few other tests (not here presented) show that the final controller has the ability to react to initial disturbances quite fast. This is not the case for the Trajectory optimization based controller (open loop). In fact, experiments show that this requires the exact information regarding the initial state to perform well. A mismatch in any of the states will result in lost of control. Table 6.18 presents the summary of
This result leads to conclude that a combination of the two concepts is possible. The robustness associated with a closed loop controller gathered with the efficiency of the pre planned trajectories present a appropriate solution for the attitude control problem of the Moonhopper.
<table>
<thead>
<tr>
<th></th>
<th>PD - Dynamic Inversion</th>
<th>Trajectory Optimization (Open loop)</th>
<th>Nonlinear Lyapunov (+ Traj Opt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objectives</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Energy (ideal)</td>
<td>80.44</td>
<td>75.42</td>
<td>80.08</td>
</tr>
<tr>
<td>Robustness to</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>Disturbances</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6.18: Summary of the comparison of the three controllers in Observation maneuver 2
Chapter 7

Conclusions

7.1 Achievements

As it was stated in the first chapter of this document, the objective of this thesis is to study the solution for an attitude controller based on the use of a DGVSCMG. The first step towards this objective was to develop a model of the Moonhopper prototype. For this objective, the dynamics of the system were separated in translation and rotation. For the translation motion, the model was developed following the dynamics of rigid body in a ballistic trajectory. Based on this model, estimates were performed in order to predict the behaviour of the prototype on Earth. These estimates were compared with experimental results performed in the robot using ISR’s MOCAP system. The differences between the estimates and the experiment were used to obtain a value of the efficiency $\eta$ of the take off mechanism and consequently, a prediction of the values of $h_{max}$ and $t_F$ on the Moon.

From these first results, two main conclusions were drawn: the first is related to the prototype itself and its design. By comparing the value of the efficiency of the Moonhopper with the same for the second generation of [4] (the robot in which this system was inspired), the conclusion was that the take off system is considerably inefficient, and requires redesign. The second conclusion was related to the expected time of flight. This value of approximately $5s$ was used in the simulations and controllers as a reference parameter.

For the rotation model, a state space model was developed to describe the behaviour of the system. This model was developed using the principle of conservation of angular momentum and was inspired by a model of a small satellite actuated by a DGVSCMG. However, the resulting state space model was developed accounting with an off centered wheel, a feature that is not common in many satellites.

The state space model was verified using a simulation of the robot in free space developed in Gazebo. Although the results do not match perfectly, the errors were considered to be small and reasons for the differences were presented. In addition, it was shown that the conservation of angular momentum is followed by the state space system, but it was not the case for the Gazebo simulation. The Moonhopper state space model was considered verified.

This first part of the project resulted in the accomplishment of the first objective. Although this was
designed for a robot on the moon, the application of this model is larger than it is presented. In fact, the approach can be applied to different planets and small bodies. Also, it is concluded that the attitude model can be used to describe the attitude dynamics of any body only actuated by a DGVSCMG.

From the resulting model three attitude controllers were developed. The first was developed based on a classic approach designed for the attitude control of the ISS. The second was an open loop controller commanded by trajectory optimization resulting inputs. The third was a nonlinear control approach for the attitude control of a spacecraft actuated by a DGVSCMG. The main objective of this second step was to develop an attitude controller. In addition, a study of the application of trajectory optimization techniques was performed. In order to evaluate the control strategies, these controllers were submitted to four trajectory tests. The results show that all controllers were able to accomplish the desired objectives. In addition, it was possible to conclude that in general, the input sets obtained from trajectory optimization were less demanding to the actuators and lead to a lower energy consumption.

Finally, the trajectories performed by the the open loop controller were used as input of the closed loop nonlinear control system. The positive results lead to conclude that a control strategy using trajectory optimization for trajectory planning and the close loop solution is a combination of robustnes and efficiency for the attitude control of an hopping robot. This way, the final objective of this thesis was accomplished: developing a controller for the DGVSCMG actuated Moonhopper. Again, it should be noted that these results and procedures are also larger than the Moonhopper project, being applicable to other systems just like the system model.

### 7.2 Future Work

As it was mentioned throughout this document, this work presents the first step of a much larger project whose objective is to have a fully functional hopping robot. This hopping robot is expected to be agile, robust and able to carry out specific tasks such as data gathering and collecting images. As it was concluded by the first experiments, the existing prototype is shown to be considerably inefficient. So a first task should be a redesign of the robot based on the literature review and references here presented.

Regarding trajectory optimization, it should be made clear that only one possible approach to this area was followed (direct collocation methods). In fact, it was shown that this strategy is applicable to attitude control and the results are positive. However, direct collocation methods are not able to output results in real time. A suggestion for future work is to study the application of this optimization strategies in real time. There are examples presented in [31] that accomplish this objective for complex systems such as the Astrobot.

Increasing the complexity of the control system, it should be important to connect the results here presented with mission planning. Using image captured by camera the robot should be able to determine a route from one point to another and use the locomotion mode to accomplish the trajectory step-by-step. Also, by detecting a target in space, the robot should be able to determine the desired attitude relative to it. From this data, the robot can perform an observation maneuver according to what was designed.
Bibliography


Appendix A

System Dynamics

A.1 Inertia Matrices and other quantities

Values are described according to SI units. Inertia matrices values are in kg.m² and mass values in kg.

\[
F[IL] = \begin{bmatrix}
0.0013 & 2.9174e-06 & -5.7147e-05 \\
2.9174e-06 & 9.8334e-04 & -2.9073e-06 \\
-5.7147e-05 & -2.9073e-06 & 9.0571e-04
\end{bmatrix}
\]

\[
F[IB] = \begin{bmatrix}
0.0017 & 9.7637e-05 & 7.0803e-05 \\
9.7637e-05 & 0.0165 & -2.9263e-05 \\
7.0803e-05 & -2.9263e-05 & 0.0028
\end{bmatrix}
\]

\[
F[IS] = \begin{bmatrix}
4.4590e-04 & 0 & 0 \\
0 & 4.4590e-04 & 0 \\
0 & 0 & 0.0011
\end{bmatrix}
\]

\[
G[IG] = \begin{bmatrix}
4.2974e-05 & 0 & 0 \\
0 & 4.3426e-05 & 0 \\
0 & 0 & 8.5739e-05
\end{bmatrix}
\]

\[
H[IH] = \begin{bmatrix}
1.1264e-05 & 0 & 0 \\
0 & 1.4680e-06 & 0 \\
0 & 0 & 1.1713e-05
\end{bmatrix}
\]

\[
H[IW] = \begin{bmatrix}
5.1682e-06 & 0 & 0 \\
0 & 5.1682e-06 & 0 \\
0 & 0 & 1.0334e-05
\end{bmatrix}
\]

\[
m_L = 0.261 \quad m_B = 0.82 \quad m_S = 0.20982 \quad m_G = 0.02207 \quad m_H = 0.020 \quad m_W = 0.00857
\]

\[
r_L = [0, 0, 0.05]' \quad r_B = [-0.002, 0.005, 0.25]' \quad r_S = [0, 0, 0.28]' \quad r_G = [0, 0, 0.345]' \quad r_H = [0, 0, 0.345]' \quad r_W = [0, 0, 0.345]' \quad r_W = [0, 0, -0.01]'
\]

A.2 Derivation of System Dynamics

In Chapter 3 only a soft explanation of the derivation of the attitude dynamics was provided. In fact, this derivation is quite exhaustive and requires special care with the manipulation of variables. The derivations steps are provided in this section together with the final expression for \( M \) and \( B \).

Since relations between the input 3.27 require an expression of \( H_i \) and these are functions of \( \dot{\omega} \), the
first step to determine an expression of $\dot{\hat{\vartheta}}$ and separate the terms of $\ddot{\hat{\vartheta}}, \ddot{\hat{\vartheta}}$ and $\dot{\Omega}$ from the rest.

From expression 3.26 it is possible to conclude that only the terms $\dot{G}_G, \dot{H}_H$ and $\dot{H}_W$ are functions of $\ddot{\hat{\vartheta}}, \dot{\hat{\vartheta}}$ and $\dot{\Omega}$. We separate these into $\dot{H}_{GcM}$ and $\dot{H}_{cW}$. 

$$\dot{H}_{GcM} = F_{[G]} \left( \ddot{\hat{\vartheta}} \right) = F_{[G]} \begin{bmatrix} 0 \\ \ddot{\hat{\vartheta}} \end{bmatrix}$$

$$\dot{H}_{cWB} = F_{[H]} \left( \omega \times \left( \ddot{\hat{\vartheta}} \right) \right) + \omega_{G/N} \times (F_{[G]} \omega_{G/N})$$

$$\dot{H}_{cWB} = F_{[H]} \left( \ddot{\hat{\vartheta}} + \ddot{\hat{\vartheta}}_2 \right) = F_{[H]} \begin{bmatrix} \ddot{\hat{\vartheta}} + \ddot{\hat{\vartheta}}_2 \end{bmatrix} + \omega_{H/N} \times (F_{[H]} \omega_{H/N})$$

$$\dot{H}_{W} = F_{[W]} \left( \ddot{\hat{\vartheta}}_1 + \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3 \right) + r_W \times \left( m_W \left( \ddot{\hat{\vartheta}}_1 + \ddot{\hat{\vartheta}}_2 \right) \times r_{W/H} \right)$$

$$\dot{H}_{W} = F_{[W]} - m_W \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3 + (\ddot{\hat{\vartheta}}_1 + \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3) + (\ddot{\hat{\vartheta}}_1 + \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3)$$

Then these quantities are introduced in equation 3.26 and organized in terms of $\ddot{\hat{\vartheta}}, \dot{\hat{\vartheta}}$ and $\dot{\Omega}$:

$$\ddot{\hat{\vartheta}}_M = -[I]^{-1}(F_{[G]} + F_{[H]} + F_{[W]} - m_W \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3)$$

$$\ddot{\hat{\vartheta}}_M = -[I]^{-1}(F_{[H]} + F_{[W]} - m_W \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3)$$

$$\ddot{\hat{\vartheta}}_M = -[I]^{-1}(F_{[W]} - m_W \ddot{\hat{\vartheta}}_2 + \Omega \hat{\vartheta}_3)$$

$$\ddot{\hat{\vartheta}}_B = -[I]^{-1}(\omega \times ([I]_{stas} \omega - \dot{H}_{GB} - \dot{H}_{HB} - \dot{H}_{WB})$$

Equation 3.26 is then rewritten as

$$\ddot{\hat{\vartheta}} = \begin{bmatrix} \dot{\hat{\vartheta}}_M(\cdot, 1) \\ \dot{\hat{\vartheta}}_M(\cdot, 2) \\ \dot{\hat{\vartheta}}_M(\cdot, 3) \end{bmatrix} + \dot{\hat{\vartheta}}_B \iff \ddot{\hat{\vartheta}} = \ddot{\hat{\vartheta}}_M + \dot{\hat{\vartheta}}_B$$

The next step is to implement the relations between angular momentum time derivatives and input torques 3.28. Starting with the computation of $\dot{H}_G$.
$$\dot{H}_{G_e} = F [I_G] \left( \dot{\omega} + \dot{\psi} \dot{f}_1 + \omega \times \left( \dot{\theta} \dot{f}_1 \right) \right) + \omega_{G/N} \times (F [I_G] \omega_{G/N})$$

$$= F [I_G] \dot{\omega} + F [I_G] \left( \dot{\psi} \dot{f}_1 \right) + F [I_G] \left( \omega \times \left( \dot{\theta} \dot{f}_1 \right) \right) + \omega_{G/N} \times (F [I_G] \omega_{G/N})$$

$$= F [I_G] \dot{\omega}_M \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \Omega \end{array} \right] + F [I_G] \dot{\omega}_B + F [I_G] \left( \dot{\psi} \dot{f}_1 \right) + F [I_G] \left( \omega \times \left( \dot{\theta} \dot{f}_1 \right) \right) + \omega_{G/N} \times (F [I_G] \omega_{G/N})$$

$$= \left[ \begin{array}{c} \{F[I_G]\dot{\omega}_M\dot{\phi} + F[I_G]\}(:,1) \\ \{F[I_G]\dot{\omega}_{M\theta}\}(:,2) \\ \{F[I_G]\dot{\omega}_{M\psi}\}(:,3) \end{array} \right] \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + F [I_G] \dot{\omega}_B$$

$$+ F [I_G] \left( \omega \times \left( \dot{\psi} \dot{f}_1 \right) \right) + \omega_{G/N} \times (F [I_G] \omega_{G/N}) = M_G \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + B_G$$

$$\dot{H}_{H_0} = F [I_H] \dot{\omega} + F [I_H] \left( \dot{\psi} \dot{f}_1 + \dot{\theta} \dot{g}_2 \right) + F [I_H] \left( \omega \times \left( \dot{\psi} \dot{f}_1 + \dot{\theta} \dot{g}_2 \right) + \left( \dot{\psi} \dot{f}_1 \right) \times \left( \dot{\theta} \dot{g}_2 \right) \right)$$

$$+ \omega_{H/N} \times (F [I_H] \omega_{H/N})$$

$$= F [I_H] \dot{\omega}_M \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + F [I_H] \dot{\omega}_B + F [I_H] \left( \dot{\psi} \dot{f}_1 + \dot{\theta} \dot{g}_2 \right)$$

$$+ F [I_H] \left( \omega \times \left( \dot{\psi} \dot{f}_1 + \dot{\theta} \dot{g}_2 \right) + \left( \dot{\psi} \dot{f}_1 \right) \times \left( \dot{\theta} \dot{g}_2 \right) \right) + \omega_{H/N} \times (F [I_H] \omega_{H/N})$$

$$= \left[ \begin{array}{c} \{F[I_H]\dot{\omega}_M\dot{\phi} + F[I_H]\}(:,1) \\ \{F[I_H]\dot{\omega}_{M\theta}\}(:,2) \\ \{F[I_H]\dot{\omega}_{M\psi}\}(:,3) \end{array} \right] \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + B_H = M_H \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + B_H$$

$$\dot{H}_{W_{off2}/H} = \frac{N_d}{dt} \left( r_{W/H} \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) \right)$$

$$= \frac{F_d}{dt} \left( r_{W/H} \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) \right) + \omega \times \left( r_{W/H} \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) \right)$$

$$= F_r \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) + r_{W/H} \times m_W \left( \left( \dot{\theta} \dot{g}_2 + \dot{\psi} \times \dot{\theta} \right) \times r_{W/H} \right) + \left( \dot{\theta} \dot{g}_2 \right) \times D_r^F$$

$$+ \omega \times \left( r_{W/H} \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) \right)$$

$$= (-m_W \dot{r}_{W/H} \dot{r}_{W/H}) \left[ FG \right] \left[ \begin{array}{c} 0 \\ \dot{\theta} \\ 0 \end{array} \right] + D_r^F \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) + r_{W/H} \times m_W \left( \left( \dot{\psi} \times \dot{\theta} \right) \times r_{W/H} \right) + \left( \dot{\theta} \dot{g}_2 \right) \times D_r^F$$

$$+ \omega \times \left( r_{W/H} \times m_W \left( \dot{\theta} \dot{g}_2 \times r_{W/H} \right) \right)$$

$$= \left[ \begin{array}{c} \{(-m_W \dot{r}_{W/H} \dot{r}_{W/H}) \left[ FG \right]\}(:,2) \\ 0 \end{array} \right] \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + B_{W_{off2}} = M_{W_{off2}} \left[ \begin{array}{c} \dot{\psi} \\ \dot{\theta} \\ \dot{\Omega} \end{array} \right] + B_{W_{off2}}$$
\[
H_{W_{offs}} = \frac{N_d}{dt} \left( r_{W/H} \times m_W \left( (\dot{\psi} \hat{f}_1) \times r_{W/H} \right) \right) \\
= D_F \times m_W \left( (\dot{\psi} \hat{f}_1) \times r_{W/H} \right) + r_{W/H} \times m_W \left( (\dot{\psi} \hat{f}_1) \times r_{W/H} + (\dot{\tilde{f}}_1) \times D_F \right) \\
+ \omega \times \left( r_{W/H} \times m_W \left( \theta \hat{g}_2 \times r_{W/H} \right) \right) \\
= (-m_W \tilde{r}_{W/H} \tilde{r}_{W/H}) \left( \begin{array}{c} \psi \\ \tilde{\theta} \\ \hat{\Omega} \end{array} \right) + B_{W_{offs}} = M_{W_{offs}} \left( \begin{array}{c} \psi \\ \tilde{\theta} \\ \hat{\Omega} \end{array} \right) + B_{W_{offs}} \\
\]

From this point the definition of M and B is straight forward. Following the relations in 3.27 M and B are defined as

\[
M = \left[ \begin{array}{c} [M_W + M_{W_{offs}} + M_{H} + M_G] \ (1,:) \\ [M_W + M_{W_{offs}} + M_H] \ (2,:) \\ [M_W] \ (3,:) \end{array} \right], \quad B = \left[ \begin{array}{c} [B_W + B_{W_{offs}} + B_{H} + B_G] \ (1,:) \\ [B_W + B_{W_{offs}} + B_H] \ (2,:) \\ [B_W] \ (3,:) \end{array} \right]
\]
A.3 Attitude Kinematics

In this section, attitude kinematics is presented in order to complete the definition of the attitude model. In attitude kinematics, the objective is to determine the temporal derivative of the rotation vector expressed in a specific representation. The most important result in the kinematics of each attitude representation is the expression of the rate of change of that representation in terms of the angular velocity \([44]\). In this work two were used: Euler Angles and quaternions.

The orientation of a rigid body with respect to an inertial coordinate system can be described by three successive transformations about body fixed axes. The three angles used for the successive transformation are the Euler angles. Any body fixed axis can be used for the first transformation. The second rotation must be performed by any of the two axes not taken for the first transformation. The final transformation is about any axes not employed by the second. In this context the sequence of rotation employed was Z-Y-X \(\psi_{\text{yaw}} - \theta_{\text{pitch}} - \phi_{\text{roll}}\). For this sequence, the kinematics equations are

\[
\begin{bmatrix}
\dot{\phi}_{\text{coll}} \\
\dot{\theta}_{\text{pitch}} \\
\dot{\psi}_{\text{yaw}}
\end{bmatrix} = \frac{1}{2} \begin{bmatrix}
1 & \tan(\theta_p) \cdot \sin(\phi_r) & \tan(\theta_p) \cdot \cos(\phi_r) \\
0 & \cos(\phi_r) & -\sin(\phi_r) \\
0 & \sin(\phi_r) / \cos(\theta_p) & \cos(\phi_r) / \cos(\theta_p)
\end{bmatrix} \omega
\]  

(A.2)

The quaternion representation is based on Euler’s rotational theorem which states that the relative orientation of two coordinate systems can be described by only one rotation about a fixed axis. A Quaternion is a 4-by-1 matrix which elements consists of a scalar part \(q_w\) and a vector part \([q_x, q_y, q_z]\). A quaternion representing a coordinate transformation from system A to system B is defined by

\[
q = \begin{bmatrix}
q_x \\
q_y \\
q_z \\
q_w
\end{bmatrix} = \begin{bmatrix}
||e|| \cdot \sin(\theta/2) \\
\cos(\theta/2)
\end{bmatrix}
\]  

(A.3)

where \(||e||\) is the normalized rotational axis and \(\theta\) is not the rotational angle but the transformation angle. According to \([44]\), the time derivative of the quaternion as function of the angular velocity vector is given by:

\[
\dot{q} = \frac{1}{2} \begin{bmatrix}
q_w & -q_z & q_y \\
q_z & q_w & -q_x \\
-q_y & q_x & q_w \\
-q_x & -q_y & q_z
\end{bmatrix} \omega
\]

(A.4)
Appendix B

Controllability, Singularities and Control Architecture

B.1 Controllability

\[ \xi_{11} = [g_1, f_{\mu}](R, \psi, \theta, \Omega) = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{11} \right] = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{11}((R, \psi, \theta, \Omega) + h g_1(R, \psi, \theta, \Omega)) \right] \]

\[ = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{11}(R, \psi, \theta, \Omega) + h (R S_{\text{stat}}^{-1} T \mu - H_{\text{cmg}}(\psi + h, \theta, \Omega)) \right] \]

\[ = - (R S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi(\psi, \theta, \Omega)), 0 \]  

(B.1)

\[ \xi_{12} = [g_1, \xi_{11}](R, \psi, \theta, \Omega) = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{12} \right] = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{12}((R, \psi, \theta, \Omega) + h \xi_{11}(R, \psi, \theta, \Omega)) \right] \]

\[ = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{12}(R, \psi, \theta, \Omega - h (R S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi(\psi, \theta, \Omega)), 0, 0, 0) \right] \]

\[ = - (R S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi(\psi, \theta, \Omega)), 0 \]  

(B.2)

\[ \xi_{13} = [\xi_{11}, \xi_{12}](R, \psi, \theta, \Omega) = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{13} \right] = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{13}((R, \psi, \theta, \Omega) + h \xi_{11}(R, \psi, \theta, \Omega)) - \dot{\xi}_{11}((R, \psi, \theta, \Omega) + h \xi_{12}(R, \psi, \theta, \Omega)) \right] \]

\[ = \frac{d}{dt}|_{h=0} \left[ \dot{\xi}_{13}((R, \psi, \theta, \Omega) - h (R S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi(\psi, \theta, \Omega)), 0, 0, 0) \right] \]

\[ = \frac{d}{dt}|_{h=0} \left[ -\{R - (R S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi(\psi, \theta, \Omega))\} S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi^2(\psi, \theta, \Omega) \right] \]

\[ = (R S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi(\psi, \theta, \Omega)) S_{\text{stat}}^{-1} H_{\text{cmg}}/\partial \psi^2(\psi, \theta, \Omega), 0, 0, 0) \]  

(B.3)
B.2 Close Loop - Attitude Hold

Using the definitions in Appendix A for \( \hat{H}_G \), \( \hat{H}_H \) and \( \hat{H}_W \), these results can be written as

\[
\begin{align*}
\hat{H}_{Ge} &= F[I_G] \left( \dot{\omega} + \dot{\psi} f_1 + \omega \times (\dot{\psi} f_1) \right) + \omega_{G/N} \times (F[I_G] \omega_{G/N}) \\
\hat{H}_{He} &= F[I_H] \left( \dot{\omega} + \dot{\psi} f_1 + \dot{\theta} g_2 + \omega \times (\dot{\psi} f_1 + \dot{\theta} g_2) + (\dot{\theta} g_2) \times (\dot{\theta} g_2) \right) + \omega_{H/N} \times (F[I_H] \omega_{H/N}) \\
\hat{H}_{We} &= F[I_W] \left( \dot{\omega} + \dot{\psi} f_1 + \dot{\theta} g_2 + \Omega h_3 + \omega \times (\dot{\psi} f_1 + \dot{\theta} g_2 + \Omega h_3) + (\dot{\theta} g_2) \times (\Omega h_3) \right) + \omega_{W/N} \times (F[I_W] \omega_{W/N}) \\
\end{align*}
\]

(B.4)

The components regarding the off centered position of the CMG are contained in the \( \dot{H}_{sat} \) vector. In addition, the offset position of the wheel with respect to the center of the gimbals is ignored \( r_{wz} \approx 0 \).

Using the expression A.1 for the \( \dot{\omega} \) the result is an expression in the form

\[
\hat{H}_{CMG} = \hat{H}_{Ge} + \hat{H}_{He} + \hat{H}_{We} = M_{HC_{CMG}} \begin{bmatrix}
\dot{\psi} \\
\dot{\theta}
\end{bmatrix} + B_{HC_{CMG}} \begin{bmatrix}
\dot{\psi} \\
\dot{\theta}
\end{bmatrix} = M_{HC_{CMG}}^{-1} \left( \hat{H}_{CMG} - B_{HC_{CMG}} \right) \quad (B.5)
\]

B.3 Nonlinear Controller

The following finalized Lyapunov control stability constraint is used in the Nonlinear control system.

\[
\begin{align*}
\dot{\Omega} \left\{ [I_W] \hat{h}_3 \right\} + \dot{\psi} \left\{ \frac{1}{2} [I_{GHW}] \left( \omega \times \hat{f}_1 \right) + \frac{1}{2} \hat{f}_1 \times ([I_{GHW}] \omega) + \omega \times ([I_{GHW}] \hat{f}_1) \right\} \\
+ \frac{1}{2} \hat{f}_1 \times ([I_{GHW}] \delta \omega) + \frac{1}{2} [I_{GHW}] \left( \delta \omega \times \hat{f}_1 \right) + \Omega \left( [I_W] \left( \hat{f}_1 \times \hat{h}_3 \right) + \hat{f}_1 \times ([I_W] \hat{h}_3) + \hat{h}_3 \times ([I_W] \hat{f}_1) \right) \right\} \\
+ \dot{\theta} \left\{ \frac{1}{2} [I_{HW}] \left( \omega \times \hat{g}_2 \right) + \frac{1}{2} \hat{g}_2 \times ([I_{HW}] \omega) + \omega \times ([I_{HW}] \hat{g}_2) + \frac{1}{2} \hat{g}_2 \times ([I_{HW}] \delta \omega) + \frac{1}{2} [I_{HW}] \left( \delta \omega \times \hat{g}_2 \right) \right\} \\
+ \Omega \left( [I_W] \left( \hat{g}_2 \times \hat{h}_3 \right) + \hat{h}_3 \times ([I_W] \hat{g}_2) \right) \\
+ \dot{\ast} \left\{ [I_{HW}] \left( \hat{f}_3 \times \hat{g}_2 \right) + \hat{f}_1 \times ([I_{HW}] \hat{g}_2) + \hat{g}_2 \times ([I_{HW}] \hat{f}_1) \right\} + \dot{\ast}^2 \left\{ \hat{f}_1 \times ([I_{GHW}] \hat{f}_1) \right\} \\
= K \sigma + [P] \delta \omega + L - \omega \times ([I] \omega) - \Omega \left\{ [I_W] \left( \omega \times \hat{h}_3 \right) + \frac{1}{2} \hat{h}_3 \times ([I_W] \omega) + \omega \times ([I_W] \hat{h}_3) \right\} \right. \\
+ \frac{1}{2} \hat{h}_3 \times ([I_W] \delta \omega) + \frac{1}{2} [I_W] \left( \delta \omega \times \hat{h}_3 \right) = L_r
\end{align*}
\]

The expression is written in term of the desired \( \dot{\psi}, \dot{\theta} \) and \( \dot{\Omega} \)

\[
\Omega_{des} a + \dot{\psi}_{des} b + \dot{\theta}_{des} c + \dot{\psi}_{des} \delta_{des} d + \dot{\psi}_{des}^2 e = L_r
\]

(B.6)
Using \( u = [\dot{\Omega}_{\text{des}}, \dot{\psi}_{\text{des}}, \dot{\theta}_{\text{des}}]^T \) the previous expression becomes \([R]u + u^T[S]u = L_r\), with

\[
[R] = \begin{bmatrix}
a & b & c \\
0 & 0 & 0 \\
0 & e & \frac{1}{2}d \\
0 & \frac{1}{2}d & 0 \\
\end{bmatrix}
\]

\[
[S] = \begin{bmatrix}
0 & 0 & 0 \\
0 & e & \frac{1}{2}d \\
0 & \frac{1}{2}d & 0 \\
\end{bmatrix}
\]

where matrices \([R]\) and \([S]\) are [3 x 1] and [3 x 3], respectively, but their entries are [3 x 1] vectors.

In order to solve this equation, a Newton-Raphson is applied to find the root of the relation

\[
f(u) = [R]u + u^T[S]u - L_r
\]

The result is value for \( u \). \( \dot{\Omega} \) is directly implemented, however, for \( \dot{\psi} \) and \( \dot{\theta} \) this is not the case: instead, these values are used to compute the input gimbal acceleration following a proportional controller:

\[
\ddot{\psi} = -\dot{\psi}(\dot{\psi} - \dot{\psi}_{\text{des}}) + \ddot{\psi}_{\text{des}} \quad \ddot{\theta} = -\dot{\theta}(\dot{\theta} - \dot{\theta}_{\text{des}}) + \ddot{\theta}_{\text{des}}
\]

where

\[
\ddot{\psi}_{\text{des}} = \frac{\dot{\psi}_{\text{des}}(t) - \dot{\psi}_{\text{des}}(t-h)}{h} \quad \ddot{\theta}_{\text{des}} = \frac{\dot{\theta}_{\text{des}}(t) - \dot{\theta}_{\text{des}}(t-h)}{h}
\]

this procedure is found in [30]. The values for the gains used in this work are presented bellow

\[
K = 0.5 \quad P = \text{diag}(\begin{bmatrix} 0.1 & 0.1 & 0.1 \end{bmatrix}) \\
K_{\dot{\psi}} = 0.2 \quad K_{\dot{\theta}} = 0.2
\]
Appendix C

Trajectory Optimization

Upon defining it, the trajectory optimization problem is transcribed into an optimization problem. According to the procedures described in Chapter 5, the resulting optimization problem is

\[
\min_{u_0, \ldots, u_{N_{grid}}} \sum_{k=0}^{N_{grid}-1} \frac{1}{2} \Delta_k (u_k^T u_k + u_{k+1}^T u_{k+1})
\]  
\tag{C.1}

\[
\begin{bmatrix}
  x_1(t_0) & x_2(t_0) & \cdots & x_{12}(t_0) & u_1(t_0) & u_2(t_0) & u_3(t_0) \\
  x_1(t_1) & \cdots & \cdots & x_{12}(t_1) & u_1(t_1) & u_2(t_0) & u_3(t_1) \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
  x_1(t_{N_{grid}-2}) & \cdots & x_{12}(t_{N_{grid}-2}) & u_1(t_{N_{grid}-2}) & u_2(t_{N_{grid}-2}) & u_3(t_{N_{grid}-2}) \\
  x_1(t_f) & \cdots & x_{12}(t_f) & u_1(t_f) & u_2(t_f) & u_3(t_f)
\end{bmatrix}
\]

subject to

Collocation conditions
\[
x_{5:12}(t_{k+1}) = x_{5:12}(t_k) + \frac{1}{2} (h_k (f(x(t_{k+1}), u(t_{k+1})), f(x(t_k), u(t_k)))) \quad \forall k = \{0, 1, \ldots, N_{grid} - 1\}
\]

\[
x_{1:4}(t_{k+1}) = Q x_{1:4}(t_k)
\]

Constraints

\[
-\psi_l \leq \psi(t) \leq \psi_u \quad -\theta_l \leq \theta(t) \leq \theta_u \quad \dot{\psi}_l \leq \dot{\psi}(t) \leq \dot{\psi}_u \quad \dot{\theta}_l \leq \dot{\theta}(t) \leq \dot{\theta}_u \quad \forall t
\]

Boundary constraints
\[
q(t_0) = q_0 \quad \omega(t_0) = \omega_0 \quad \psi(t_0) = \psi_0 \quad \theta(t_0) = \theta_0 \quad \Omega(t_0) = \Omega_0 \quad \dot{\psi}(t_0) = \dot{\psi}_0 \quad \dot{\theta}(t_0) = \dot{\theta}_0
\]

\[
q_{F_{l}} \leq q(t_f) \leq q_{F_{u}} \quad \omega_{F_{l}} \leq \omega(t_f) \leq \omega_{F_{u}}
\]

With \(Q\) comes from expression 5.6 as \(q(t_{k+1}) = Q q(t_k)\).
Appendix D

Results

D.1 Translaction

In this section, the results regarding the prediction of Moonhopper jump parameters are presented. Departing in a fully compressed state are presented, the robot is expected to achieve the conditions at take off

\[
F_{\text{max}} (N) \quad W (J) \quad v_0 (ms^{-1}) \quad h_{\text{max}} (m) \quad t_F (s)
\]

\[
413.91 \quad 34.60 \quad 7.14 \quad 2.59 \quad 1.45
\]

Table D.1: Ideal conditions at take off

During the decompression phase, the robot is submitted to forces that describe the following curve

Figure D.1: Reaction Force as function of leg compression. The leg compression is measured as the distance between the top and low gears. The two points marked correspond to the full compression state (red-left) and the take off state (yellow-right)

This curve is read from left to right as the force that is being applied by the springs at each point during the decompression. The value 0.16 corresponds to \(2\alpha\) and it is impossible to be reached as it would correspond to a state where both legs are fully stretched and parallel to each other. Another
way to see it, is that this curve presents the required force to fully compress the robot, departing from an equilibrium state (on the right) and traveling along the line to the left, the integral over that curve corresponds to the energy spent to fully compress the system.

D.2 Trajectory optimization implementation

The hyperparameters that are used in the NLP solver play an important role in the results. 'TolFun' is the termination tolerance on the first-order optimality (a positive scalar). First-order optimality is a measure of how close a point $x$ is to optimal. In fact, the first-order optimality measure must be zero at a minimum. Typically, if the first-order optimality measure is less than 'TolFun', solver iterations end.

'TolCon' corresponds to the maximum number of function evaluations allowed, and is a positive integer. Together with 'MaxIter' - maximum number of iterations allowed - they represent a stopping criteria used as auxiliary parameters to obtain these solutions.

Tolerance on the constraint violation, a positive scalar. It was used to ensure that all the constraints were satisfied. The low values was mainly due to the preoccupation regarding the collocation constraints

<table>
<thead>
<tr>
<th>Hyperparameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>'TolFun'</td>
<td>1e-8</td>
</tr>
<tr>
<td>'TolCon'</td>
<td>1e-11</td>
</tr>
<tr>
<td>'MaxFunEvals'</td>
<td>5e4.(nState+nControl)</td>
</tr>
<tr>
<td>'MaxIter'</td>
<td>400</td>
</tr>
</tbody>
</table>

Table D.2: Hyperparameters