



# Reaction Wheels: Enhancing Aerial Maneuvers for Legged Robots

Department of Industrial Engineering

Master's degree in Mechatronics Engineering

Candidate

Andrea Cumerlotti
ID number 214589

Thesis Advisors

Prof. Michele Focchi
Prof. Andrea Del Prete

External Advisors

Dr. Claudio Semini
Mr. Francesco Roscia

Academic Year 2021/2022

**Reaction Wheels: Enhancing Aerial Maneuvers for Legged Robots**
Master's thesis. Università degli Studi di Trento



Author's email: andreacumerlotti@gmail.com

*Dedicated to*
*all the people who supported me during this experience,*
*in particular, my family, my flatmates, and my friends.*

## Abstract

Legged locomotion is studied to realize robots that are able to traverse rough terrains. Different types of gait have been developed for quadrupeds robots, such as crawl, trot, and pace. They are distinguished, one from the other, by gait schedule and footstep location. It is not always possible to get around an obstacle by walking. Therefore, researchers have started to investigate more complex motions such as jumps. In this particular locomotion strategy, there is a phase in which all the feet break the contact with the ground. During this part, the translation of the robot's center of mass and the main body rotation around it are decoupled. The linear motion follows the ballistic trajectory. On the other hand, the rotation is constrained by angular momentum conservation.

In this thesis, I investigate the problem of controlling the orientation of the base of the robot during the aerial phase of a jump or of a fall. I design an orientation control system based on two reaction wheels tailored for Solo12, a quadrupedal open-source robot. The axes of rotation are designed to be incident, enabling the possibility to partition their contribution to the angular momentum for controlling the platform roll and pitch angles. I propose two methodologies to actuate the reaction wheels: a controller based on a PD law, and a bang-bang controller. The former counteracts the effects of external disturbances and model inaccuracies while tracking a given angular trajectory. The latter drives the base to a desired final orientation, pushing the motors to their actuation limit for a small amount of time. To demonstrate the effectiveness of the device and the control laws, I realize simulations of falls and jumps with different gravitational forces. In particular, I analyze three environments: outer space, Earth, and Moon.

## Acknowledgments

*I would like to show my deep appreciation to my advisors Prof. Michele Focchi and Prof. Andrea Del Prete, who helped me carry on my project. I wish to thank Dr. Claudio Semini for the opportunity that allowed me to conduct this thesis and to acknowledge the support provided by the staff in the Dynamic Legged System of the Istituto Italiano di Tecnologia, in particular Mr. Francesco Roscia and Dr. Matteo Villa.*

*Genova, 06 July 2022*
*Andrea Cumerlotti*

# List of Symbols

| Symbol | Meaning |
|---|---|
| $\mathbf{A}$ | Centroidal momentum matrix |
| $C$ | Center of mass |
| $\mathbf{C}$ | Matrix with the reaction wheel axes of rotation |
| $\mathbf{c}$ | Position of the center of mass |
| $\mathbf{e}$ | Base orientation error |
| $\mathbf{f}$ | Forces at the foots |
| $\mathbf{g}$ | Gravity vector |
| $\mathbf{h}$ | Non-linear effects |
| $h$ | Height of the reaction wheel |
| $\mathbf{I}$ | Tensor of inertia |
| $\mathbf{J}$ | Jacobian matrix |
| $K$ | Kinetic energy |
| $\mathbf{K}_d$ | Matrix with derivative gains |
| $\mathbf{K}_p$ | Matrix with proportional gains |
| $\mathbf{L}$ | Angular momentum |
| $L$ | Lagrangian |
| $\mathbf{M}$ | Generalized inertia matrix |
| $m$ | Mass |
| $n$ | Number of robot joints |
| $n_i$ | Number of foots on ground |
| $\mathbf{p}$ | Position of the foots |
| $\mathbf{Q}$ | Quaternion that represent the base orientation |
| $\mathbf{q}$ | Generalized coordinates |
| $\mathbf{q}_b$ | Position and orientation of the base |
| $\mathbf{q}_j$ | Joint variables |

| Symbol | Meaning |
|---|---|
| $r_1$ | Inner radius of the reaction wheel |
| $r_2$ | Outer radius of the reaction wheel |
| $\mathbf{S}$ | Selection matrix |
| $t$ | Time |
| $U$ | Potential energy |
| $\mathbf{u}$ | Input of the system |
| $\mathbf{x}$ | State of the system |
| $\alpha$ | Angle between the base $y$-axis and the rotation axis of the reaction wheels |
| $\gamma$ | Reaction wheel rotation |
| $\boldsymbol{\varepsilon}$ | Imaginary vector of a quaternion |
| $\eta$ | Scalar part of a quaternion |
| $\theta$ | Base pitch angle |
| $\rho$ | Density |
| $\boldsymbol{\tau}$ | Torques applied at the joints |
| $\boldsymbol{\tau}_f$ | Torques applied at the reaction wheels |
| $\tau_m$ | Torques applied at one motor |
| $\tau_{max}$ | Maximum torque available at the motors |
| $\boldsymbol{\phi}$ | Vector with roll-pitch-yaw angles |
| $\phi$ | Base yaw angle |
| $\psi$ | Base roll angle |
| $\boldsymbol{\omega}$ | Base angular velocities |
| $\omega_{fl}$ | Angular velocity of the left reaction wheel |
| $\omega_{fr}$ | Angular velocity of the fight reaction wheel |

# Contents

# Chapter 1

# Introduction

This dissertation deals with robots. Many different types of electromechanical systems are categorized in this big family, making it impossible to obtain a description that includes all. The most general definition is: a system able to perceive the environment and automatically act accordingly. Too many objects already present in people's daily life fit this interpretation, such as thermostats, dishwashers, and many others. Some roboticists do not like that these devices are included in the definition, while others are less restrictive and call them simple robots.
Despite the difficulty in giving a proper definition, it is easier to classify them. The two most important families are fixed-base and mobile. One of the main differences between these two categories is the first has a link (called base) constrained to be fixed, while the second is free to move under non-holonomic constraints[1] if present. Therefore robots of the first class can only operate in a working space limited by the kinematic structure, while the one in the second can freely move from one place to another [1]. Mobile robots are able to help or substitute humans in activities that require moving, like the exploration of hazardous environments or carrying payloads for long distances, which would not be possible with a fixed base. Different types of mobile robots have been developed depending on the environment where they should move. A widespread structure in mobile robots are the wheeled ones. They are already present in the life of people to help in their daily activities like Roomba [2], a cleaning robot. Even self-driving cars, such as the autonomous taxi Waymo One [3], belong to this category. They are even used in more complex scenarios like Mars exploration, with the rovers Perseverance [4] and Curiosity [5], or on battlefields, like PackBot 510 [6]. Wheels constrain these robots to move on almost flat terrain, making it impossible to go on rough ones. To increase mobility, researchers get inspired by humans and animals to design new architectures. Legged robots have been developed to overcome the limitations of the wheeled one. They are usually classified depending on the number of legs. Bipeds, like iCub [7], Asimo [8], HRP-2 [9], and HRP-2w [10], are mainly studied to develop a future robot assistant able to work in the daily environment of humans. The hydraulic actuated biped Atlas [11] and the electrical one Digit [12] show how the mobility of these platforms improves

[1] When you cannot integrate a constraint in velocity into position, the constraint is non-holonomic. Usually, it happens because there is a difference in the number of variables used to express the position and velocity field.

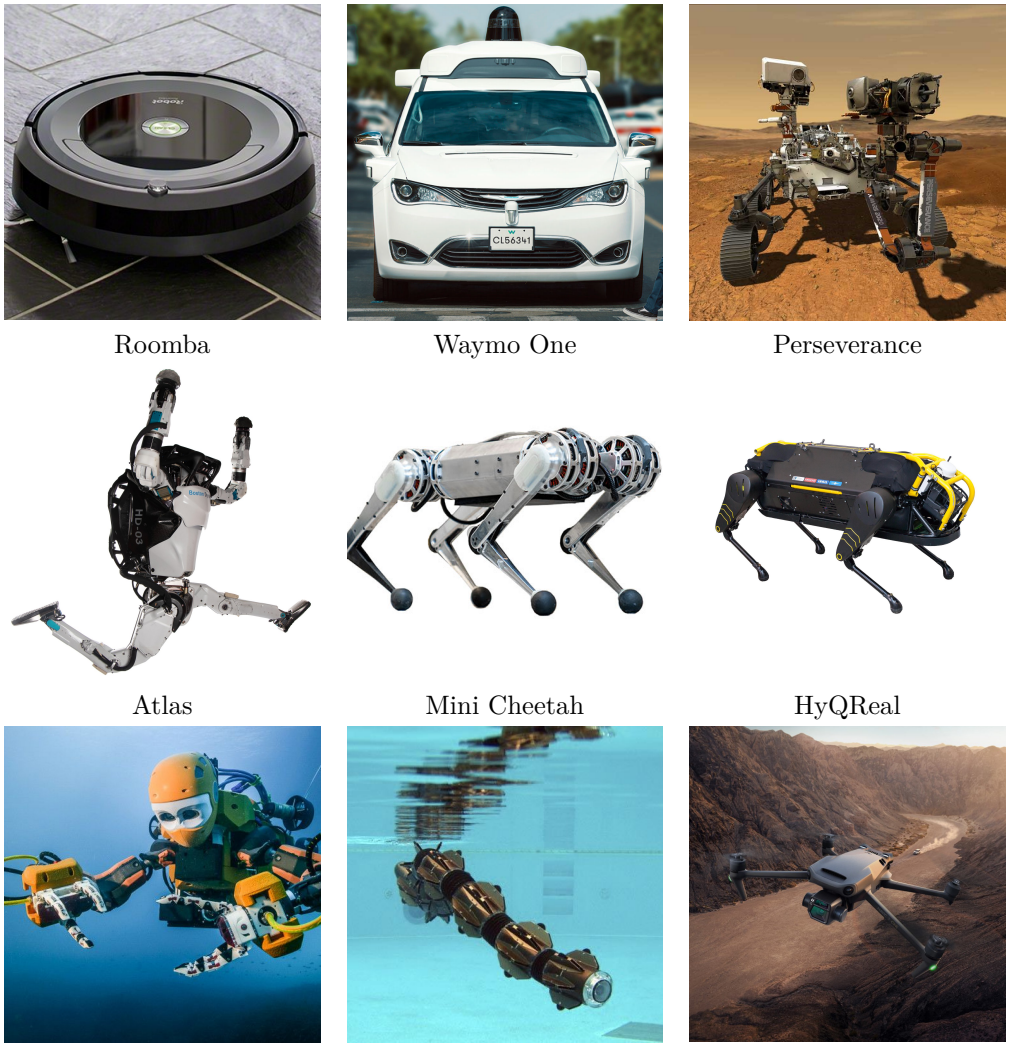



**Figure 1.1.** Some of the cited mobile robots are shown with the respective name. In the first row are present three wheeled robot, Roomba, Waymo One and Perseverance. Right above are shown three legged robot, the biped atlas, and the quadrupeds Mini Cheetah and HyQReal. Different morphology, the diver robot Ocean One, the amphibious ACM-R5H and the quadricopter Mavic 3 are displayed at last.[2]

with respect to the wheeled robots. Increasing the number of legs reduces the control effort necessary to avoid a fall and improves the total mass that the robot can carry (i.e., the payload), making legged morphology more suited for outdoor locomotion. Many different quadrupeds with disparate characteristics have been developed. Robots Spot [13], Mini Cheetah [14], and ANYmal [15] take advantage of electrical

[2]Images are taken from: Roomba `https://www.irobot.it/roomba/essenziali`, Waymo One `https://waymo.com/waymo-one/`, Perseverance `https://mars.nasa.gov/mars2020/`, Atlas `https://www.bostondynamics.com/atlas`, Mini Cheetah `https://robots.ieee.org/robots/minicheetah/`, HyQReal `https://echord.eu/hyqreal.html`, Ocean One `https://www.newsweek.com`, ACM-R5H `https://www.directindustry.it`, and Mavic 3 `https://www.swisscycles.com/`.

motors to obtain light and agile platforms. Instead, the robots HyQ [16] and HyQReal [17] exploit the force generated by hydraulic actuators to get a heavy robot capable of carrying high payloads. The number of legs can be further increased, like in CREX [18], a six-legged platform designed for crater exploration. In addition to getting inspired by other shapes, some robots are a combination of more categories. For example, CENTAURO [19] uses hybrid locomotion, both wheeled and legged, to move, and it is provided with two arms on the trunk to perform manipulation tasks. Mobile robots are not designed only to move on rough terrain. Ocean One [20] is a humanoid robot designed to explore oceans. ACM-R5H [21] is a snake robot capable of swimming and slithering. Mobile robots can be present also above our heads, flying. Their category is known as UAV (unmanned aerial vehicle). Some examples of drones can be the quadricopter Mavic 3 [22] or the fixed-wing eBee [23].

## 1.1 Motivation

Legged robots are designed to go on rough terrain. As a matter of fact, different types of gait, such as trot[24] or crawl[25] are used to move these robots. Thanks to the technological progress of the last years, the robots have become lighter and able to generate higher torques and forces at the joints, enabling the possibility of doing highly dynamic maneuvers. Sometimes it is not possible to get around an obstacle with the gaits mentioned above, and a more complex one, a jump, should be required. An aerial maneuver, such as a jump, can be divided into three different phases:

1. thrusting phase, the robot generates the time-varying profile of ground reaction forces necessary to break the foot contacts;
2. flying phase, no contact with the environment is present;
3. landing, the robot returns in contact with the ground with at least 2 (or more) legs.

The two transitions between the different parts are called lift-off (when all the contacts are broken) and touch-down (when the robot reestablish the contacts between feet and ground). During the flight phase (stage 2), the linear motion of the CoM is decoupled from the rotational one. The first corresponds to the ballistic trajectory, the motion of a particle (a projectile) starting with an initial velocity, subject to gravity only. The resulting path is a parabola, and the vertex is called the apex, the highest point reached by the CoM. Instead, the rotational motion is ruled by the conservation of the angular momentum explained in section 2.3.1. Errors in tracking the reaction forces generated during the trusting phase or external disturbances during the flight phase could cause a non-desired robot reorientation. A technique that properly stabilizes the base orientation is necessary to obtain a safe landing. An undesired rotation could cause feet misplacement, a re-bounce, or a fall, compromising the result of the whole aerial maneuver. In this thesis, I investigate a solution for this problem for Solo12 [26], an open-source quadruped robot. I modified the hardware to include an orientation control system. The main components of the device are two high inertia disks (usually hollow) rotating about

their symmetry axis. These disks are used to store rotational kinetic energy. The fact that this energy is used to reorient the base defines these components as *reaction wheels*, a sub-class of flywheels.

## 1.2 State of the Art

The orientation control is based on the concept of angular momentum and its conservation, explained in 2.3. The reorientation of the base of a legged robot can be obtained by changing the contribution of the other links (i.e., changing the joint configuration) to the total angular momentum of a legged robot, which is constant in the absence of contacts. This fact is widely used even in agile quadrupedal animals, like cats, which can rearrange their tail and trunk to correct the orientation during a fall [27]. It is possible to obtain the same result by creating repetitive circular motions with the feet[3] of a legged robot, like in [28] and [29]. Finding the joint motion that results in the correct reorientation maneuver for a robot is not an easy task, and it is computationally expensive due to the non-holonomy of the angular momentum [30]. Usually, it is addressed through numerical optimization. In the case of robots with limbs lighter than the trunk (e.g., the majority of quadrupeds), this operation requires fast motions. In [29] the authors overcome the problem by adding to Mini Cheetah special heavy boots and using a neural network to calculate online trajectory trained on many optimizations problems solved offline. However, this solution unnecessarily increases the inertia of the legs, which designers usually try to set as low as possible to ease the locomotion.

Many works inspired by animals use an additional link as a tail, like in [31] and [32]. This link rotates around an axis that does not pass through its center of mass (CoM). The distance of the axis of rotation from both the base and tail CoM's allows obtaining high inertia with a small link mass. However, this link hinges on the extremity of the trunk. The placement of this link makes the resulting robot asymmetric. Due to its limited range of motion, a tail can be used only for one jump, not for a repeated sequence [33].
Another option is to use a control moment gyroscope (CMG). It consists of a flywheel rotating at a constant angular speed inside an actuated gimbal. Tilting the axis of rotation of the flywheel generates a gyroscopic torque. It is widely used for the reorientation of spacecraft [34] and, less frequently, locomotion, both wheeled [35] and legged [36]. The CMG presents interesting capabilities, but its complexity (due to the presence of a pan-tilt unit in addition to the drive for the gyroscope) makes it impractical to mount it on a lightweight robot.

The last option discussed in this section is to use reaction wheels. Changing the angular speed of a rotating mass attached to the trunk generates a torque that can reorient and stabilize the system. This device is widely used in satellite orientation [37] but was sporadically investigated even in legged locomotion, both for bipeds [38] [39] and quadrupeds [40] [41]. A reaction wheel instead of a tail, or a generic joint of the robot, does not have a position limit, and since it rotates around its

[3]Moving the feet outwards increases the robot's inertia, so if a leg is extended during half of the motion and retracted in the other half, a net moment will result on the trunk (because the angular momentum is invariant).

center of mass, its angular momentum results holonomic [42]. To get a fast response, it is necessary to have an abrupt change in the reaction wheel's angular speed (i.e., angular acceleration). Using a brake avoids the employment of a motor able to deliver higher torques [43], keeping the system compact. The motor speeds up slowly the reaction wheel, and when necessary, the break stops it. Since the effect of the break is unidirectional, it is possible to generate a rotation of the base in the opposite direction of the reaction wheel angular velocity.
The flywheel can correct orientation errors due to disturbances (e.g., wind) during the flight and inaccuracies due to the angular momentum achieved at the lift-off (e.g., given by tracking issues and non-idealities). They enable the robot to land with a desired angular velocity (possibly zero) and orientation. In addition, they can enhance the landing phase by significantly reducing oscillations. The presence of this additional joint used only to control the orientation gives the possibility to relieve the effort of the legs. In more complex scenarios, like in a somersault, legs and orientation control system can work in parallel to achieve a rotation angle larger than the one achievable only with legs (e.g., due to torque limitation) [39].

## 1.3 Contribution

The contributions presented in this thesis are the following.

- **Mechanical design of the orientation control system.** I propose an orientation control system based on two reaction wheels for the quadruped robot Solo12. The designed structure allows for mounting the wheels above the trunk of Solo12 in different configurations. In one of them, the rotation axes of both the reaction wheels are parallel to the lateral axis of the base. On the other, the rotation axes are incident and lay in the plane generated by the $x$- and $y$-axis of the trunk, as shown in figure 2.2. In the first case, it is possible to control the orientation only in the lateral direction. In the second case, even the orientation on the roll direction is controllable. According to the actual design of the flywheels, it is possible to have a rotation of 30° on the lateral rotation during a flight time without contacts of 1 s.

- **Control methodologies.** A proportional and derivative (PD) controller has been used to control the reaction wheels. The error is calculated using two representations for the orientation, roll-pitch-yaw angles, and quaternion. The second case does not present singularities. This controller can reject external disturbances while tracking the desired orientation.
  The second strategy uses a Bang-Bang controller to obtain the desired rotation within a given time. It employs the bang-bang strategy: it applies the maximum torque for a short time and lets the robot rotate with the acquired angular velocity. This technique obtains large rotation angles shortly without saturating the motors' velocity. The time of switching is computed before the maneuver start. Then, during the rotation, it is recalculated again with the updated value of orientation, angular velocity, and inertia of the robot. The recomputation increases the accuracy of the final orientation. This controller is improved by considering a continuous trapezoidal shape for the torque. Applying the

opposite torque for the same amount of time right before the end of the maneuver, the angular velocity of the base goes to zero, and the robot stops rotating.

## 1.4 Outline of the Thesis

The remainder of this report is organized as follows. Chapter 2 introduces the background needed to design the orientation control system. In particular, the robotic platform is presented, together with traditional modeling techniques for legged robots. The chapter concludes with an overview of the concepts of angular momentum and its conservation. In chapter 3, the mechanical design of the orientation control system is described. Here, the sizing of reaction wheels that met the specification is detailed. Moreover, all the customized components are presented. These include the protective shell, the shaft, and a codewheel mounter for the encoder of each reaction wheel. A brief description of the off-the-shelf components ends the chapter. The above-mentioned control schemes are introduced in chapter 4 and validated with simulations in chapter 5. Lastly, a brief summary of the results of this work is presented together with the problems that are still open.

# Chapter 2

# Background

This chapter will provide an overview of the relevant elements for the design of the orientation system. First, both the hardware and software of the robot Solo12 are described. Then, I illustrate the dynamical models used to describe the robotic system. And in conclusion, I present the angular momentum and its conservation, according to which I framed the control law.

## 2.1 Robotic Platform

The robot used to demonstrate the effectiveness of flywheels in controlling orientation is Solo12. This quadruped is the result of the project *Open Dynamic Robot Initiative* which had to build a low-cost and lightweight platform as the objective. Since it is an open-source project, all the information about the hardware (mechanical drawings and electronic circuits) and the software are provided [26] and described in the following sections. The GitHub repository of the project [44] presents a detailed description of the robot with the procedure to mount it.

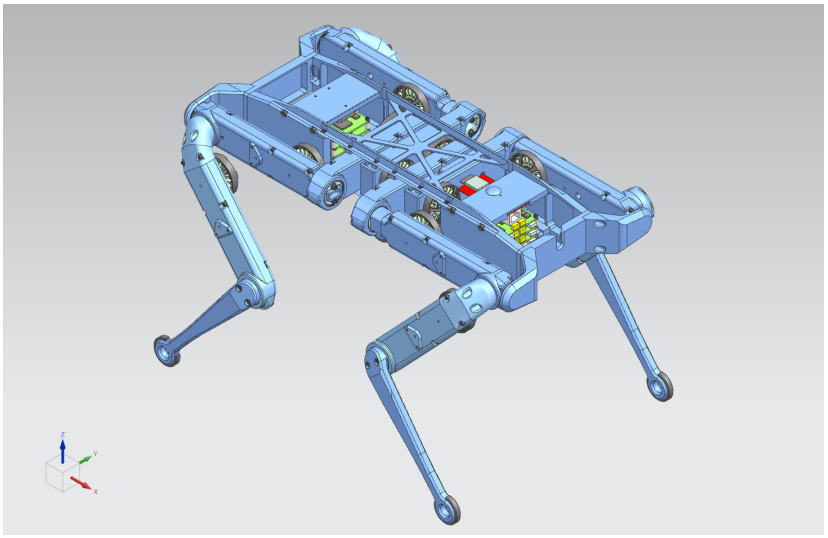


**Figure 2.1.** Solo12 3D model provided by *Open Dynamic Robot Initiative* project.

### 2.1.1 Hardware

The robot structure is realized in 3D-printed polymers to reduce the weight and costs. The platform consists of four identical legs and a trunk. Each leg consists of three bodies (hip mounting, upper leg, and lower leg) connected by three joints: Hip Abduction Adduction (HAA), Hip Flexion Extension (HFE), and Knee Flexion Extension (KFE). As suggested by the robot name, the total number of joints is 12. All the joints are actuated through an actuator module, which includes:

- brushless motor (T-Motor Antigravity MN4004 KV300)[1];
- high resolution incremental encoder (Broadcom AEDT-9810-Z00)[2];
- timing belts transmission (with a gear ratio of 9:1).

These modules are mounted inside the proximal link, in the vicinity of the joint. The motor boards used are miniaturized versions of the Texas Instruments Evaluation Boards. Each one can control two motors with field-oriented control (FOC). In addition, for both the controlled motors, the board provides the measurements of the current flowing through the motor coils and the angular position of the motor shaft measured by the encoders. Since the encoders are incremental, the absolute position of the joints is obtained with a homing procedure every time the robot is switched on. The inertial measurement unit (IMU) Lord Microstrain 3DM-CX5-25[3] is mounted on the trunk to provide information about the state of the robot trunk (i.e., base link). It includes a triaxial accelerometer and a gyroscope that provide respectively the acceleration and angular velocity of the base. From the data provided by the IMU and the encoders, it is possible to obtain an indirect measure of the position and orientation of the trunk. The motors boards and the IMU are connected with the master board via Serial Peripheral Interface (SPI) communication. The master board manages all the commands and data flow with an external computer attached via an Ethernet cable. An external power supply provides the voltage (28 V) and the current necessary for the robot to work. The six motor boards are split into two stacks: the first is mounded in the front part of the trunk, right under the IMU, and the second in the rear, right under the master board.

The kinematics and dynamics of the robot are described by setting the following convention: all the reference frames are aligned with the base frame when the robot is in the full stretched configuration. By definition, all the joints angle are zero in such a configuration. When the robot is switched on, the world reference frame is generated. The $z$-axis is parallel and in the opposite direction of the gravitational acceleration measured by the IMU. The other two axes lay in a plane perpendicular to the $z$-axis, placed in contact with the ground. In particular, the $x$-axis is in the forward direction of the robot base.

[1] More details about the motor can be found on the store website: `https://store.tmotor.com/goods.php?id=438`

[2] For additional information about the encoder, check out its datasheet: `https://docs.broadcom.com/doc/AEDT-981x-Three-Channels-Optical-Incremental-Encoder-Modules-DS`

[3] For additional information about the IMU, check out its datasheet: `https://www.microstrain.com/sites/default/files/3dm-cx5-25_datasheet_8400-0116_rev_f.pdf`

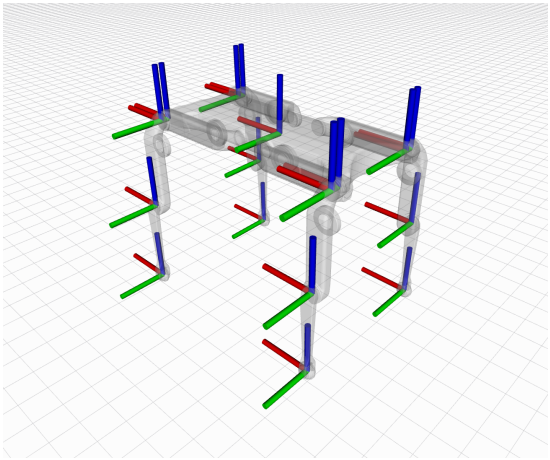


**Figure 2.2.** The figure above shows the robot is in zero-configuration with the joints reference frames. Red segments represents the x-axes of the frames, green is used for the y-axes and blue for the z-axes.

### 2.1.2 Software

Since an onboard computer is not present, an external one is connected to the robot via Ethernet or WiFi. The GitHub repository of the project [44] provides a software development kit (SDK) which includes a C++ class to communicate with the master board together with the Python bindings.

The controller structure, with the most important commands, is shown in figure 2.3.

```
if __name__ == '__main__':
  os.nice(-20)
  # Pinocchio Model
  robotPin = getRobotModel('solo')
  # Instantiate and initialize Robot
  robot = Solo12()
  robot.init()
  ...
  while not robot.is_timeout:
    robot.update_sensor_data()
    ...
    robot.q_des, robot.qd_des = JointReferenceGenerator()
    robot.tau_fb, robot.tau_ffwd = TorqueController()
    robot.set_motors_torques()
    robot.ros_publish()
    robot.wait_end_of_cycle(robot.rate)
  robot.stop()
```

**Figure 2.3.** Important commands used to run the real robot

The controller runs on a multitasking operating system that does not ensure real-time execution. The command `os.nice(-20)` sets the process *niceness* to the minimum value, which gives it the highest priority and more CPU time with the aim of emulating a real-time system. To execute this command is necessary to run the

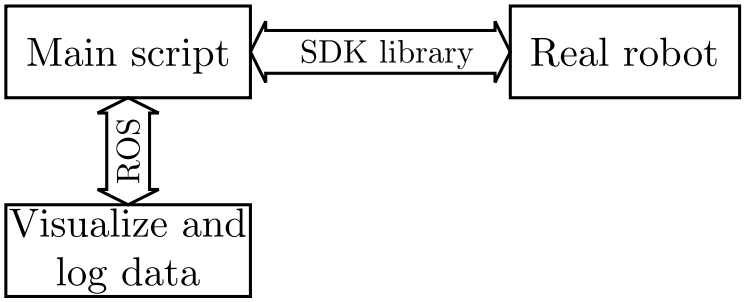



**Figure 2.4.** Schematic description of the communication during and experiment with the real robot.

script in super user mode (aka. sudo). The command `robotPin = getRobotModel('solo')` loads the Pinocchio model of the robot. Pinocchio [45] is an open-source software that computes the rigid body dynamics algorithms efficiently (look at subsection 2.2.1 for more information about this dynamical model). The robot controller class is instantiated and initialized with `robot = Solo12()` and `robot.init()` respectively. The second command allows the possibility to calibrate the encoders following a "homing" procedure:

- a feedforward torque leads all the joints to their known mechanical limits;
- once in this position is possible to associate the measure of the encoders with the actual angular value.

This procedure must be done every time the robot is switched on. The class Solo12 contains all the methods to send commands and receive data from the sensors and provides logging features. The methods `robot.update_sensor_data()` and `robot.set_motors_torques()` directly use the SDK library to communicate with the robot. The variables `robot.q_des`, `robot.qd_des`, `robot.tau_fb`, and `robot.tau_ffwd` must be set mandatory, otherwise if they are empty the command `robot.set_motors_torques()` send zero as their values. For logging the acquired data, `robot.ros_publish()` is called, which uses the robot operating system (ROS) [46]. ROS is not used to send the commands or to read the data to avoid further delays in communication. The last command `robot.wait_end_of_cycle(robot.rate)` ensures that the loop repeats with the desired control loop frequency `robot.rate` (500 Hz). Another script, based on the class `ExpController`, uses the published data to visualize the robot on RVIZ, make plots and store them in RosBags.

```
1 if __name__ == '__main__':
2   p = SimController('solo')
3   ...
4   while rospy.is_shutdown():
5     ...
6     p.send_command(q_des, qd_des, tau_ffwd)
```

**Figure 2.5.** Important commands used to run a simulation

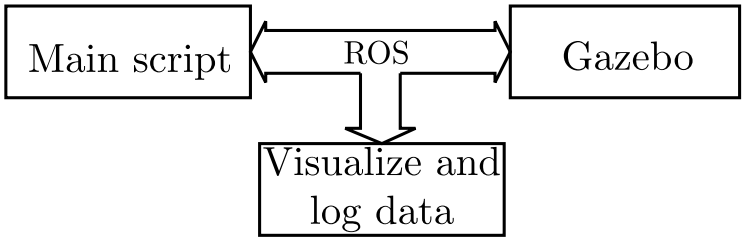



**Figure 2.6.** Schematic description of the communication during a simulation.

In a simulation, the code is slightly different from the one of the experiments. Everything is managed by the instance `p = SimController()`, which communicates with Gazebo [56], a simulator for the robot dynamics, interacting with the environment. This software uses ode as the physics engine to calculate the results of the simulations. In this case, the data are exchanged via ROS. The results published by the simulator are automatically updated through the methods `p._receive_jstate` and `p._receive_pose`. At the end of the loop is sufficient to use the function `p.send_command (q_des, qd_des, tau_ffwd)` to publish the desired states and feed-forward torques, log the data, and wait until the time step is elapsed (like in the experiments, control loop frequency is 500 Hz).

Both `ExpController` and `SimController` are classes that inherit from `Controller` in which are implemented all the common function between experiment and simulation [47].

## 2.2 Models for Floating Base Robots

Different types of mathematical models are used to represent the dynamic of legged robots. They are used to quantify a relationship between the control input $\mathbf{u}$ and the state $\mathbf{x}$ of the system. The models are usually an ordinary differential equation (ODE) in the form

$$\dot{\mathbf{x}}(t) = f\left(\mathbf{x}(t), \mathbf{u}(t)\right)$$

Assumptions are used to simplify the description of a complex system, like a robot. Generally, increasing the number of assumptions results in reduced complexity at the price of lower accuracy. The models used in this thesis are explained in the remainder of this section.

### 2.2.1 Rigid Body Dynamics

This model corresponds to the Lagrangian one. The only assumption of this model is that all the robot bodies are rigid and that their physical quantities, such as mass or inertia, do not change over time. Being a rigid body means: that given two arbitrary points of the body, their relative distance does not change under the action of external forces or moments. Therefore we suppose that the body does not deform during usage. With this assumption, it is possible to fully describe the system with the generalized coordinates $\mathbf{q} = [\mathbf{q}_b^T\ \mathbf{q}_j^T]^T$, which include the position and orientation of the base and joint variables. With a minimum of 6 coordinates (3 for the position and 3 for the orientation), it is possible to fully describe the state of

the base. Therefore, it is possible to use virtual links to see the base as a 6 degree of freedom joint. In the other links, the presence of a joint enforces some kinematic constraints on the link that allow only one degree of freedom of motion between two subsequent links. Therefore it is necessary for only one variable for each joint, which means $\mathbf{q}_j \in \mathbb{R}^n$, with $n$ the number of robot joints. The number of DoF of the whole robot is $6+n$. Considering that at the joints are applied the torques $\boldsymbol{\tau}$ the dynamics is [48]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^T\boldsymbol{\tau} + \mathbf{J}(\mathbf{q})^T\mathbf{f} \ , \tag{2.1}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the joint-space inertia matrix, $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{6+n}$ represents the non-linear effects (Centrifugal, Coriolis, gravity, and friction), $\mathbf{S} = [\mathbf{0}_{n\times 6}\, \mathbf{I}_{n\times n}]^T$ is the selection matrix that applies the actuation torque $\boldsymbol{\tau} \in \mathbb{R}^n$ only to the joints and $\mathbf{J}(\mathbf{q})$ is the Jacobian, whose transposed maps the contact forces at the foots $\mathbf{f} = [\mathbf{f}_1^T, ..., \mathbf{f}_{n_i}^T]$ to $6+n$ generalized forces. The selection matrix emphasizes that it is possible to split the model in 6 unactuated (2.2a) and $n$ actuated (2.2b) equations.

$$\mathbf{M}_u(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}_u(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{J}_u(\mathbf{q})^T\mathbf{f} \tag{2.2a}$$

$$\mathbf{M}_a(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}_a(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau} + \mathbf{J}_a(\mathbf{q})^T\mathbf{f} \tag{2.2b}$$

Splitting the dynamics into the actuated and the unactuated parts, it becomes clear that it is not possible to control directly the motion of the base through the applied torques $\boldsymbol{\tau}$ but only indirectly through the reaction forces $\mathbf{f}$.

### 2.2.2 Centroidal Dynamics

With a simple change of coordinates (expressing all the quantities with respect to the system CoM rather than at the base level)[49] equation 2.2a becomes:

$$\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{A}}(\mathbf{q})\dot{\mathbf{q}} = \begin{bmatrix} m\mathbf{g} + \sum_{i=1}^{n_i} \mathbf{f}_i \\ \sum_{i=1}^{n_i} \Big(\mathbf{p}_i - \mathbf{c}(\mathbf{q})\Big) \times \mathbf{f}_i \end{bmatrix} \tag{2.3}$$

where the left-hand side is the derivative of the momentum $\mathbf{A}(\mathbf{q})\dot{\mathbf{q}}$ written with respect to a frame anchored to the center of mass $\mathbf{c}$ (centroid) [48], and the right-hand side is the sum of the external forces and moments. The matrix $\mathbf{A}(\mathbf{q}) \in \mathbb{R}^{6\times(6+n)}$ is the centroidal momentum matrix. It projects the velocity of each body into the rate of change of momentum expressed in a frame attached to the CoM. The point $\mathbf{p}_i$ and the force $\mathbf{f}_i$ are respectively the position of the $i$-th contact and the external force applied to it. The number of feet in contacts is $n_i$.

This model is not used in this thesis, but it is the step between the rigid body model and the single rigid body model, explained in the following subsection.

### 2.2.3 Single Rigid Body Dynamics

Assuming that the momentum given by the joint velocity is negligible and that the inertia remains close to the nominal one, it is possible to remove the dependency of the joint coordinates from 2.3 [48]. The previous assumptions are reasonable if the

trunk mass is considerably larger than the limbs (which is the case of most quadruped robots) or when the joint legs' velocities are small. Under these assumptions, the well-known Newton Euler equations appear; that in robotics are also known as the single rigid body model:

$$m\ddot{\mathbf{c}} = m\mathbf{g} + \sum_{i=1}^{n_i} \mathbf{f}_i \tag{2.4a}$$

$$\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} = \sum_{i=1}^{n_i} (\mathbf{p}_i - \mathbf{c}) \times \mathbf{f}_i \ . \tag{2.4b}$$

where $\boldsymbol{\omega}$ is the angular velocity of the main body. The total mass of the robot is $m$, while $\mathbf{I} \in \mathbb{R}^{3\times3}$ is the tensor of inertia of the whole robot computed as if it was a rigid body with the joints in the nominal configuration. The model becomes simpler but less accurate, neglecting the joint coordinates from the computation of the CoM $\mathbf{c}$ and the tensor of inertia $\mathbf{I}$. Even with these simplifications, the model is still non-linear due to the cross product present in 2.4b.

To obtain a linear model (e.g., linear inverted pendulum model [50]) is necessary to make further assumptions that are too restrictive for the objective of this thesis.

## 2.3 Angular Momentum

The absolute angular momentum of a point mass $m$ with respect to point $O$ is

$$\mathbf{L}^O = \mathbf{p} \times \dot{\mathbf{p}}\, m \tag{2.5}$$

in which $\mathbf{p}$ is the position of the point mass with respect to $O$ and $\dot{\mathbf{p}}$ its velocity[4]. The definition can be extended to a rigid body integrating over all the body's mass.

$$\mathbf{L}^O = \int_m \mathbf{p} \times \dot{\mathbf{p}}\, dm \tag{2.6}$$

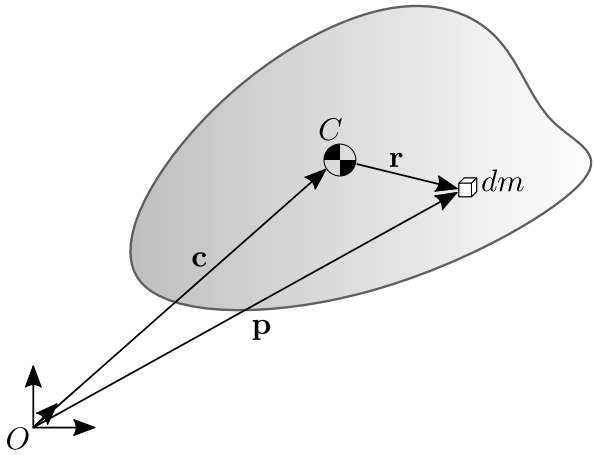



**Figure 2.7.** Representation of the vectors used to describe the center of mass and a generic infinitesimal mass of a rigid body.

[4]We assume that all the vectors used in this section are expressed with the respect of an inertial reference frame

If we consider that $\mathbf{p} = \mathbf{c} + \mathbf{r}$ and $\dot{\mathbf{p}} = \dot{\mathbf{c}} + \boldsymbol{\omega} \times \mathbf{r}$, the integral becomes

$$\begin{aligned}\mathbf{L}^O &= \int_m (\mathbf{c} + \mathbf{r}) \times (\dot{\mathbf{c}} + \boldsymbol{\omega} \times \mathbf{r})\, dm \\ &= \mathbf{c} \times \dot{\mathbf{c}} \int_m dm + \mathbf{c} \times \left( \boldsymbol{\omega} \times \int_m \mathbf{r}\, dm \right) + \int_m \mathbf{r}\, dm \times \dot{\mathbf{c}} + \int_m \mathbf{r} \times (\boldsymbol{\omega} \times \mathbf{r})\, dm \ .\end{aligned} \tag{2.7}$$

In the latter, $\boldsymbol{\omega}$, $\mathbf{c}$, and $\dot{\mathbf{c}}$ do not depend on the mass. Thus they can be taken out from the integral sign. In the first term, $\int_m dm$ corresponds to the total mass of the rigid body. From the definition of center of mass, it is straightforward to obtain $\int_m \mathbf{r}\, dm = 0$:

$$\begin{aligned}\mathbf{c} &= \frac{1}{m} \int_m \mathbf{p}\, dm \\ &= \frac{1}{m} \int_m (\mathbf{c} + \mathbf{r})\, dm \\ &= \mathbf{c} + \frac{1}{m} \int_m \mathbf{r}\, dm \quad \Rightarrow \quad \int_m \mathbf{r}\, dm = 0\end{aligned}$$

The fourth summand of equation 2.7 correspond to the angular momentum if the pole of rotation pass through the body CoM $\mathbf{I}_c\, \boldsymbol{\omega}$ [51]. The matrix $\mathbf{I}_c$ is the inertia tensor of the body expressed with respect to the CoM. The cross-product between vectors can be seen as matrix multiplication. To do so is necessary to transform a vector $\mathbf{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^T$ in to its skew symmetric matrix with the operand $[\mathbf{v}]_\times$ that perform the operation

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} .$$

$$\begin{aligned}\int_m \mathbf{r} \times (\boldsymbol{\omega} \times \mathbf{r})\ dm &= \int_m \mathbf{r} \times (-\mathbf{r} \times \boldsymbol{\omega})\ dm \\ &= \int_m -[\mathbf{r}]_\times [\mathbf{r}]_\times\ dm\ \boldsymbol{\omega} \\ &= \mathbf{I}_c\, \boldsymbol{\omega}\end{aligned}$$

Taking these simplifications into account, the angular momentum of a rigid body in equation 2.7 becomes:

$$\mathbf{L}^O = \mathbf{c} \times \dot{\mathbf{c}} m + \mathbf{I}_c \boldsymbol{\omega} \ .$$

In the case of a multi-body system (e.g., a robot), the angular momentum is the sum of the contributions of each rigid body. Writing the angular momentum with respect to the CoM of the multi-body system $C$, its expression is more comprehensive, like in [30]. If the system is composed of $N$ bodies is possible to number them from 1 to $N$ in order to write

$$\mathbf{L}^C = \sum_{i=1}^{N} (\mathbf{c}_i - \mathbf{c}) \times \dot{\mathbf{c}}_i m_i + \mathbf{I}_{c_i} \boldsymbol{\omega}_i \ .$$

The vector $\mathbf{c}_i$ points to the center of mass of the $i$-th body $C_i$ while $\mathbf{c}$ points to the center of mass of the whole system, as shown in figure 2.8.

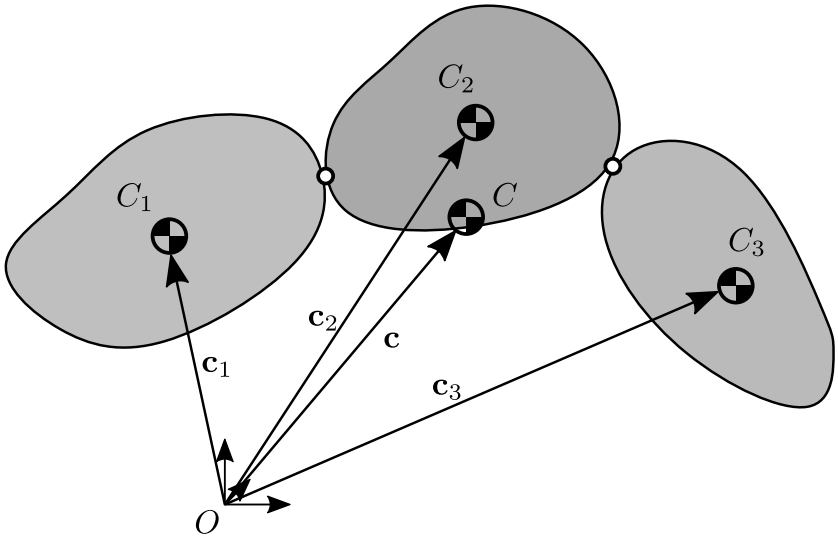



**Figure 2.8.** Representation of the vectors used to describe the CoMs of a multi-body system.

### 2.3.1 Euler's Equation and Conservation of the Total Angular Momentum

Euler's equation rules the rotational motion. This equation is explained in the theorem of the angular momentum present in [52]:

**Theorem 1** (Angular momentum theorem)**.** *For an arbitrary system, the absolute time derivative (i.e., the time derivative in an inertial reference base) of the absolute angular momentum with respect to a reference point O fixed in inertial space equals the resultant torque with respect to the same reference point:*

$$\dot{\mathbf{L}} = \sum \mathbf{M}$$

When the external moment applied to the system is zero, Euler's equation simplifies in

$$\dot{\mathbf{L}} = 0 \;\Leftrightarrow \mathbf{L} = \text{const. for any } t \tag{2.8}$$

which corresponds to the conservation of angular momentum. Considering a mobile robot, this condition happens when the system is not in contact with the ground or other objects, for example, during a fall or the flight phase of a jump. In the case of a legged robot, it is possible to change the angular velocity of the base, changing the joints velocities, as a result of the *nonholonomy* of the angular momentum [30]. If the angular momentum of the $i$-th body changes, the one of the other body must modify accordingly to maintain the total sum constant. As explained in section 1.2, many works use this property for robot reorientation.

# Chapter 3

# Reaction Wheel System Design

This chapter describes the design of the orientation control system based on reaction wheels. The new device should allow controlling both the roll and the pitch simultaneously. To accomplish this task, the reaction wheels' contribution to the total angular momentum should lie in both $x$-axis and $y$-axis of the base frame. To fulfill this design objective, I present the following two main ideas:

1. employ four reaction wheels provided with brakes, two of them aligned with the roll axis, the other aligned with the pitch axis;

2. use two reaction wheels with incident axes.

In the first case, it is possible to take advantage of the braking force. Generally, the torque generated by the brakes is higher than the one exerted by an electric motor. Since the moment on the reaction wheel is proportional to the one on the base, this approach can produce a higher angular acceleration on the robot base with respect to the case with only the motors. Since the effect of this component is only in the opposite direction of the angular velocity, it is necessary to use two wheels sped up in the opposite way for each axis of rotation of the robot base. The negative aspect of this approach is the high number of components necessary to realize it. As a matter of fact are necessary four motors, four brakes, the electronic board necessary to control all the parts (if we use the same board of the other joint, two are necessary to run only the motors). In the second case, the only source of torque is the motors. To control the orientation on both the roll and the pitch, it is sufficient that the wheels' axes of rotation are incident and parallel to the XY plane of the base reference frame.

For this purpose, I design two symmetrical modules, mountable on the base in different configurations. The design of the wheels considers only the correction of the pitch angle. It is supposed to have both the reaction wheels parallel with the axis aligned with the $y$-axis of the base. The wheels are sized to achieve a correction angle of at least 30° in 1 s. This requirement is given considering a fine correction of the orientation during the flight phase to compensate for errors coming from a not-perfect tracking of the ground reaction forces coming from an offline optimization, during the thrusting phase.

## 3.1 Off-the-shelf Components

The reaction wheel module design starts from the actuator module used to control each joint. We choose to maintain compatibility with the other components present in the robot. We employ the same motor, encoder, motor board, and bearings as the other actuator modules since their technical specifications meet the working conditions of the device.

The brushless motor present on the other joints is the T-Motor Antigravity 4004, 300KV. This actuator is commonly used for drone applications, but its characteristic allow to employ it in other fields. It can generate a maximum torque of 0.3 Nm, but for safety reasons, we decide to limit it to 0.225 Nm. Its lightness (53 g) satisfies the requirement of not increasing to much the total robot mass of the robot.
The encoder Broadcom AEDT-9810-Z00 used on the other motors guarantees a measure of the velocity up to 12000 RPM making it suitable for the case. As a matter of fact the motor's maximum speed is 5000 RPM. One motor board is sufficient to control both the motors and receive data from the encoders. On the master board, it is already present a free port at which to connect the additional motor board. With only a few modifications to the SDK library, it is possible to send and receive data from the orientation control system.

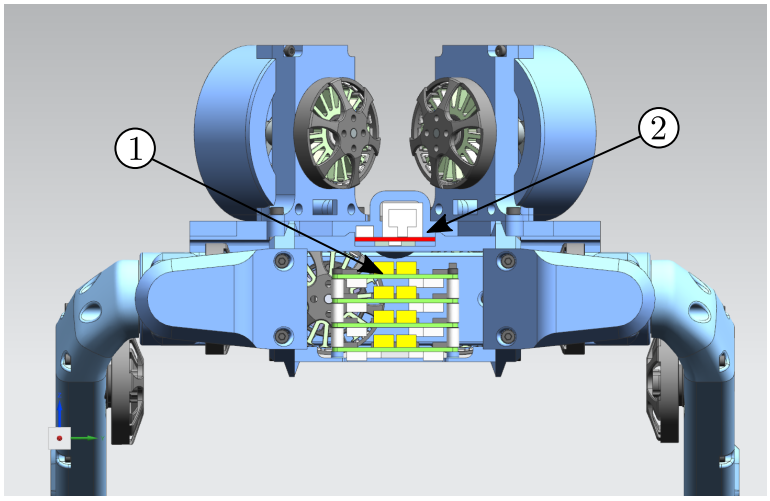



**Figure 3.1.** The additional motor board ① is added on top of the micro driver stack located in the back of the robot, under the master board ②. The back cover of the trunk is not shown to see inside of it.

| Component | Parameter | Value |
|---|---|---|
| T-Motor Antigravity 4004, 300KV | Mass | 53 g |
| | Maximum torque | 0.3 Nm |
| | Maximum speed | 5000 RPM |
| Broadcom AEDT-9810-Z00 | Resolution | 5000 |
| | Maximum velocity | 12000 RPM |

**Table 3.1.** Most important data taken from the datasheet of motor and encoder.

If the master board is mounted upside-down, facing the outside of the robot, there is a free space on top of the micro-driver stack, as shown in figure 3.1. It has been decided to mount it under the master board and not under the IMU to avoid heating this last one. To prop up the rotating shaft on the other side of the motor it is used the same bearing used on the motor, the EZO bearing MR84 VA[1]. Thanks to its high static and dynamic radial capacity it can support the shaft even with the additional load of the flywheel. To assemble each reaction wheel system are employed screws.

| Component | Quantity |
|---|---|
| T-Motor Antigravity 4004, 300KV | 2 |
| Broadcom AEDT-9810-Z00 | 2 |
| Motor board | 1 |
| EZO bearing MR84 VA | 2 |
| $M3$ threaded grains | 12 |
| $M3$ hexagonal socket head screw | 6 |
| $M2.5$ hexagonal socket head screw | 8 |
| $M3$ flat head screw | 8 |
| $M3$ Philips flat head screw | 4 |

**Table 3.2.** Bill of materials

## 3.2 Inertia Selection

In order to find the inertia necessary to comply with the requirements of the orientation control system, some analyses are performed with the "Elroy's Beanie" shown in figure 3.2. This model consists of two bodies connected with a revolute joint on their Center of Mass (CoM). One of the bodies represents the robot (inertia $I_r$), simplified as a single rigid body, and the other one the flywheel (inertia $I_{fw}$). I perform the analysis for only one flywheel and only then extend the results to two bodies.

[1] More details can be font on the producer web page: `https://catalog.ezo-usa.com/viewitems/metric-series-bearings/open-metric-ball-bearings`

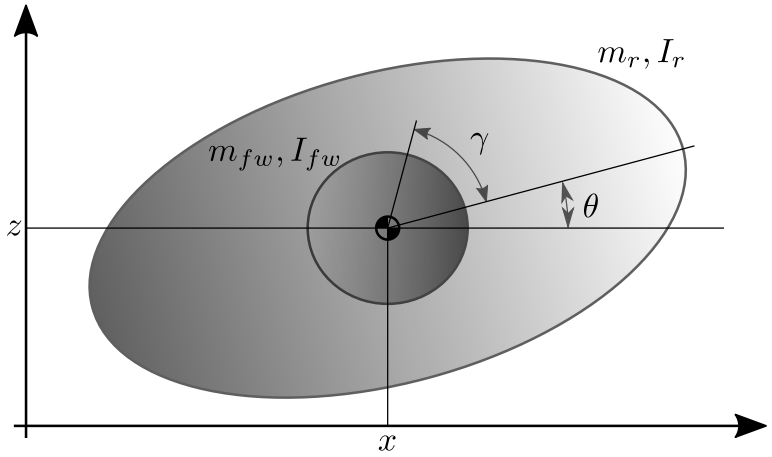



**Figure 3.2.** Schematic representation of the Elroy's Beanie model used for the preliminary analysis of the pitch motion.

The angular momentum $L$ of this system can be written as

$$L = (I_r + I_{fw})\,\dot{\theta} + I_{fw}\dot{\gamma} \tag{3.1}$$

in which $I_r$ is the $I_{yy}$ component of the centroidal inertia matrix of the robot and $I_{fw}$ is composed by both the reaction wheels inertia of the along their axis of rotation, $\dot{\theta}$ and $\dot{\gamma}$ correspond respectively to the pitch-rate and angular speed of the wheels. I employed the only the $I_{yy}$ component because, it gives the information about the pitch rotation. The other elements of the tensor of inertia that affect the pitch, such as $I_{yx}$ and $I_{yz}$, are not considered since they are negligible with the respect to $I_{yy}$. Using the $I_{xx}$ component of the centroidal inertia matrix instead of the $I_{yy}$ allows us to perform the the same analysis as below for the roll rotation. In this case, the variable $\theta$ should be substitute by $\phi$, the roll angle. Under the condition of conservation of the angular momentum 2.3.1 it is possible to estimate of the lower bound for $I_{fw}$ to get a desired angular velocity of the main body $\dot{\theta}_{des}$. Assuming that at the lift-off the reaction wheels are not spinning ($\dot{\gamma}_{lo} = 0$), the angular momentum is $L_{lo} = (I_r + I_{fw})\,\dot{\theta}_{lo}$, in which $\dot{\theta}_{lo}$ is the base pitch rate at lift-off. This value does not change for the whole duration of the flight. The minimum inertia necessary to reach $\dot{\theta}_{des}$ is obtained when the flywheels rotate at their maximum speed. For higher inertia it is sufficient a smaller angular velocity:

$$I_{fw} = I_r \frac{\dot{\theta}_{lo} - \dot{\theta}_{des}}{\dot{\theta}_{des} + \dot{\gamma}_{max} - \dot{\theta}_{lo}}\ . \tag{3.2}$$

The variable $\dot{\gamma}_{max}$ is the maximum motor speed that correspond to 5000 RPM ( 523.6 rad/s). In figure 3.3 the results for different initial and desired angular velocities are reported. It is evident that high inertia is required when the inertial and the desired angular velocity are opposite. Usually, this condition is avoided thanks to a good thrusting phase.

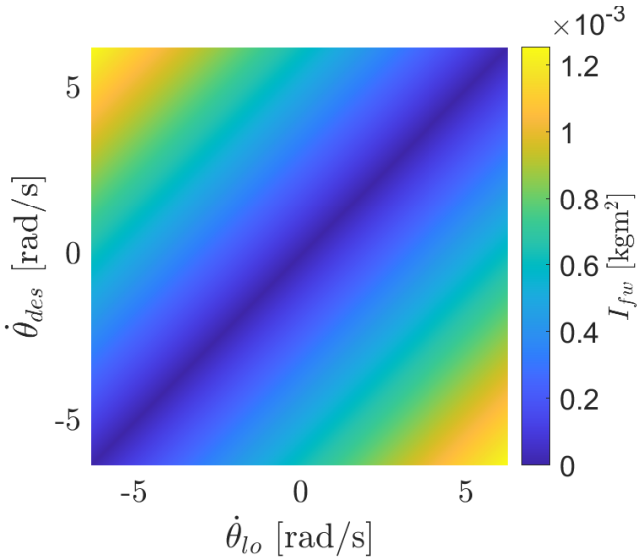



**Figure 3.3.** Minimum inertia necessary to obtain the angular velocity $\dot{\theta}_{des}$ starting with different lift-off $\dot{\theta}_{lo}$ conditions.

The kinematic model employed before has not considered the transient to reach the desired angular speed. For this reason, the dynamical model of the "Elroy's Beanie" mechanism is derived. The dynamics of this system is obtained with the Lagrangian method. To calculate the Lagrangian $L$ it is necessary to compute the kinetic energy $K$ and the potential energy $U$. In this quantities must be included also the translation part, therefore are used the masses of the bodies, $m_r$ and $m_{fw}$, and the position of the CoM, $x$ and $z$.

$$K = \frac{1}{2}\left(m_r + m_{fw}\right)\left(\dot{x}^2 + \dot{z}^2\right) + \frac{1}{2}I_r\dot{\theta}^2 + \frac{1}{2}I_{fw}\left(\dot{\theta} + \dot{\gamma}\right)^2$$

$$U = \left(m_r + m_{fw}\right)gz$$

$$\begin{aligned} L &= K - U \\ &= \frac{1}{2}\left(m_r + m_{fw}\right)\left(\dot{x}^2 + \dot{z}^2\right) + \frac{1}{2}\left(I_r + I_{fw}\right)\dot{\theta}^2 + \frac{1}{2}I_{fw}\dot{\gamma}^2 + I_{fw}\dot{\theta}\dot{\gamma} - \left(m_r + m_{fw}\right)gz \end{aligned} \tag{3.3}$$

$$\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}}\right) = \boldsymbol{\tau} \tag{3.4}$$

From the Lagrangian $L$ is possible to derive the equation of motion using 3.4, in which $\mathbf{q} = \begin{bmatrix} x & z & \theta & \gamma \end{bmatrix}^T$ are the generalized coordinates and $\boldsymbol{\tau}$ are the generalized forces applied to the bodies. In this particular case, it is analysed the flight phase in which are not applied external forces or torques, apart from the flywheels' ones. Then we can set $\boldsymbol{\tau} = \begin{bmatrix} 0 & 0 & 0 & \tau_m \end{bmatrix}^T$, in which $\tau_m$ is the direct-drive torque applied by

the motor that control the flywheel. The resulting equation of motion are

$$\begin{bmatrix} m_r + m_{fw} & 0 & 0 & 0 \\ 0 & m_r + m_{fw} & 0 & 0 \\ 0 & 0 & I_r + I_{fw} & I_{fw} \\ 0 & 0 & I_{fw} & I_{fw} \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \\ \ddot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ (m_r + m_{fw})\, g \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \tau \end{bmatrix}. \tag{3.5}$$

To obtain the values of $\mathbf{q}$, the differential equations are integrated using Runge-Kutta 4th order method. From 3.5, we can see that the linear and the angular dynamic are decoupled. The linear part, which corresponds to the well-known motion of a projectile, is used to calculate the time of flight. I suppose that the robot lands on a flat surface at height zero. The landing is estimated when the CoM reach the height of 0.23 m (distance between CoM and ground in homing configuration) with a negative velocity. This information is needed to understand the maximum time in which the reorientation maneuver must be concluded. The linear dynamics can be neglected, imposing a fixed time constraint based on the specification of the duration of the flight phase to 1 s.

The inertia of the reaction wheels can be chosen minimizing the work done by the motor to perform a reorientation maneuver. In particular, the task analyzed was a rotation of the main body of 30° ($\simeq 0.5$ rad) in the time $t_f = 1$ s. A fifth-order polynomial generates the desired trajectory of the pitch of the robot that performs the rotation starting and finishing with zero angular velocity and acceleration. In order to track the trajectory, we assume that we use a proportional and derivative (PD) controller. The gains $k_p$ and $k_d$ are chosen to obtain a small settling time avoiding overshoots.

$$\tau_m(t) = k_p \left(\theta_{des}(t) - \theta(t)\right) + k_d \left(\dot{\theta}_{des}(t) - \dot{\theta}(t)\right)$$

To select the inertia we intend to perform an optimization where we minimize the integral of the motor power absolute value, during the reorientation task. The inertia $I_r$ affects the dynamics of the system and then modifies the evolution of $\tau_m(t)$ and $\dot{\gamma}(t)$ during the task.

$$\min_{I_r} \int_0^{t_f} \left|\tau_m(t)\dot{\gamma}(t)\right| dt$$

In figure 3.4 are shown the values of the work done by the motors,for correcting both roll and pitch of Solo12. The dissipated energy is monotonically decreasing in the flywheel's inertia. From the results, it is evident that there is not a local minimum and the optimal value for the inertia $I_{fw}$ is the greatest as possible.

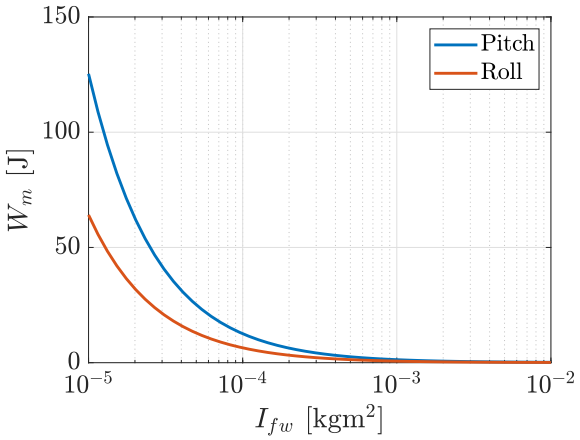



**Figure 3.4.** Work done by the motor $W_m$ using different values of inertia $I_{fw}$ for the reaction wheel during a reorientation manoeuvrer of 30° in 1$s$. The $x$-axis is in logarithmic scale to show a wide spectrum of inertias. The results of the roll (in red) and pitch (in blue) are difference due to the different value of $I_r$, indeed the components of the centroidal inertia of the robot $I_{xx}$ and $I_{yy}$ are very different.

With the same analysis, I investigated whether the inclusion of a gearbox between the motor and the reaction wheel would bring some benefits. The dynamical model is modified including the gearbox, with a gear reduction ratio $n$. In this case, the motor angular velocity and the one of the reaction wheel are no more equal. I continue to use $\gamma$ to define the motor angle, while the one of the flywheel is $\gamma/n$. The torque applied at the reaction wheel is the torque of the motor magnified by a a factor $n$.

$$\begin{bmatrix} m_r + m_{fw} & 0 & 0 & 0 \\ 0 & m_r + m_{fw} & 0 & 0 \\ 0 & 0 & I_r + I_{fw} & I_{fw}/n \\ 0 & 0 & I_{fw} & I_{fw}/n \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \\ \ddot{\gamma} \end{bmatrix} + \begin{bmatrix} 0 \\ (m_r + m_{fw})\, g \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ n\tau_m \end{bmatrix} \tag{3.6}$$

The problem of minimization is modified including the gear reduction ratio as a decision variable.

$$\min_{I_r, n} \int_0^{t_f} |\tau_m(t)\dot{\gamma}(t)|\, dt \tag{3.7}$$

The results, in figure 3.5, point out that the presence of a gearbox with a reduction ratio as great as possible results in a minimization of the energy used to perform the reorientation maneuvers. The simulation of the task points out that adding the gearbox reduce the tracking error, but decrease of a factor $1/n$ the maximum speed of the motor. I decide to maintain the value of $n = 1$ (without gearbox), to avoid the saturation of the motor velocity during a common task and because using a greater value do not comport a relevant decrease in energy.

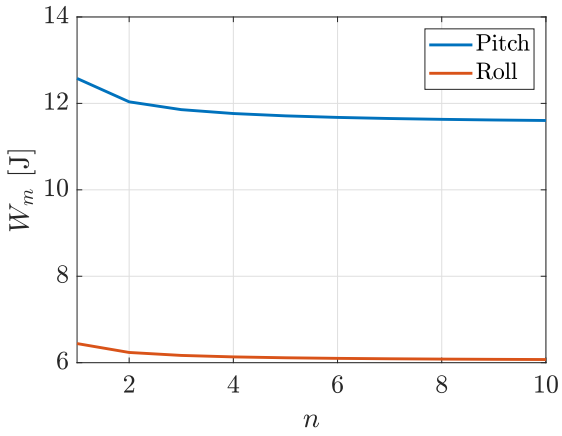



**Figure 3.5.** Work done by the motor $W_m$ using different values of reduction ratio $n$ for the reaction wheel during a reorientation manoeuvrer of 30° in 1$s$. The $x$-axis is in logarithmic scale to show a wide spectrum of inertias. The results of the roll (in red) and pitch (in blue) are difference due to the different value of $I_r$, indeed the components of the centroidal inertia of the robot $I_{xx}$ and $I_{yy}$ are very different.

## 3.3 Mechanical Design

Different studies, like [53], search for an optimal shape for the flywheel that maximizes the kinetic energy of this device. The objective is to maximize the inertia while minimizing the mass to keep the overall weight of the robot contained. This technique results in a non squared cross section, hard to obtain with the conventional manufacturing processes, such as turning and milling. Usually this shapes are obtained with additive manufacturing or melting procedure, that become economically advantageous only for large scale volumes. To maintain the production costs low, I decide to choose an hollow cylinder as a shape of the reaction wheel, that should be achievable with a milling machining. Such a shape is known to increase the momentum of inertia because is aiming to locate the mass far away from the rotation axis, instead of a full cylinder. The property of its mass and inertia depend only on the density of the material $\rho$, outer radius $r_2$, inner radius $r_1$, and height $h$. They are shown in figure 3.6. The mass can be calculated as

$$m = \pi \rho h \left( r_2^2 - r_1^2 \right) \ . \tag{3.8}$$

Since axis shown in figure 3.6 corresponds to the principal axis of inertia, then the only non-zero terms of the tensor are the ones on the diagonal. Moreover, due to the symmetry of the object, $I_{xx}$ and $I_{yy}$ have the same value.

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

[2]The axes are defined principal axes of inertia if the tensor of inertia in that frame results diagonal.

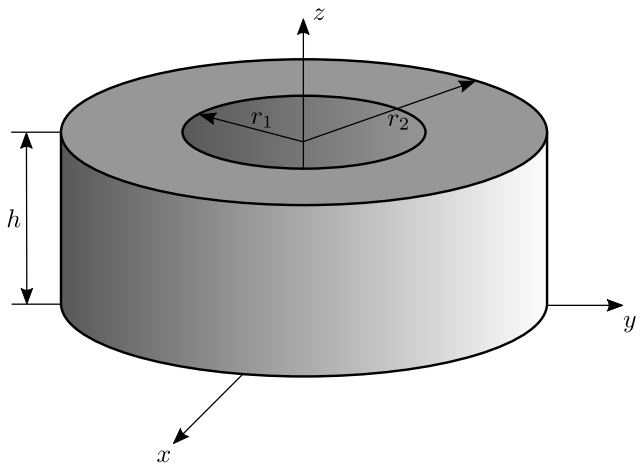



**Figure 3.6.** Shape of the reaction wheel, an hollow cylinder. The reference system is placed to have the $z$-axis on the main principal axes of inertia tensor[2]of the flywheel.

$$\begin{aligned} I_{zz} &= \frac{1}{2} m \left( r_2^2 + r_1^2 \right) \\ &= \frac{1}{2} \pi \rho h \left( r_2^4 - r_1^4 \right) \end{aligned}$$

$$\begin{aligned} I_{xx} = I_{yy} &= \frac{1}{12} m \left( 3 \left( r_2^2 + r_1^2 \right) + 4h^2 \right) \\ &= \frac{1}{12} \pi \rho h \left( 3 \left( r_2^4 - r_1^4 \right) + h^2 \left( r_2^2 - r_1^2 \right) \right) \end{aligned}$$

From the results obtained in the preliminary analysis in section 3.2, I decided to choose the parameters that maximize the inertia $I_{zz}$. I add a constraint on the mass to avoid the resulting wheels being too heavy to perform a jump. In particular, it is used a maximum value of 0.1 kg. An upper bound is applied to the outer radius and the height to obtain a space-saving solution. The larger the outer radius will result into a larger inertia with the same mass. Since the reaction wheels are placed above the trunk, using a larger wheel would make the supporting structure weaker. Especially during a lateral fall, larger wheels result in higher oscillations and more stress on the screws that connect the structure to the robot. The upper bounds of 3 cm for the outer radius and 1 cm for the height, correspond to a good trade-off between getting a space-saving solution and obtaining a value of inertia that meet the specification. The density depends only on the material used. In this project, I used stainless steel. Thanks to its high density, it allows obtaining great inertia in a small volume. The other materials available in the mechanical workshop have smaller density or too higher costs for the production.

The maximum value of inertia is obtained putting the mass far away from the rotation axis. Therefore I chose the outer radius and the height as high as possible (for both I set the respective upper bound). The inner radius is chosen to meet the

| Parameter | Value |
|---|---|
| $r_1$ | 2.2 cm |
| $r_2$ | 3.0 cm |
| $h$ | 1.0 cm |
| $m$ | 0.1 kg |
| $I_{zz}$ | $7.1 \cdot 10^{-5}$ kgm$^2$ |
| $I_{xx}, I_{yy}$ | $3.6 \cdot 10^{-5}$ kgm$^2$ |

**Table 3.3.** Size and dynamic parameters of a single reaction wheel.

constraint on the mass. From equation 3.8, the inner radius can be expressed as:

$$r_1 = \sqrt{r_2^2 - \frac{m}{\pi \rho h}} \tag{3.11}$$

If the result of $r_1$ is an imaginary number means that the full cylinder has mass a lower value than the maximum acceptable. In this case the value of $r_1$ should be set to 0 m. The resulting dimensions and dynamic parameters are shown in table 3.3.

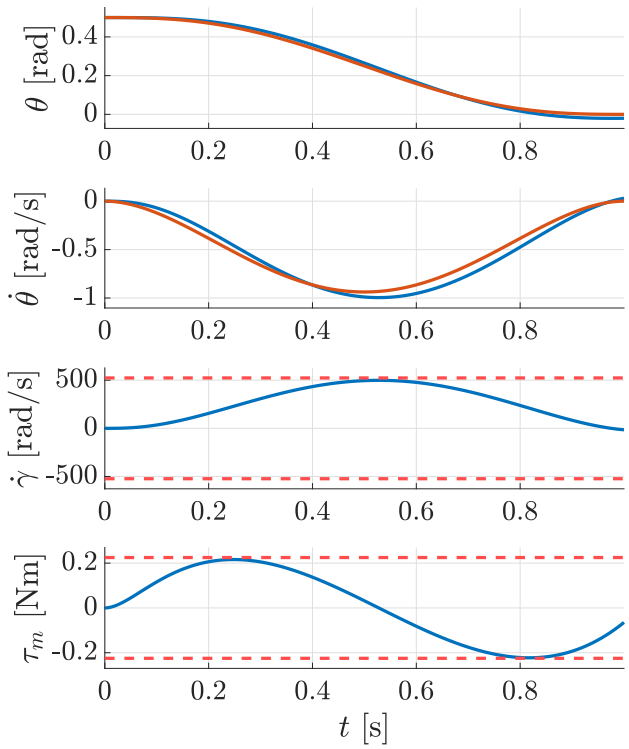
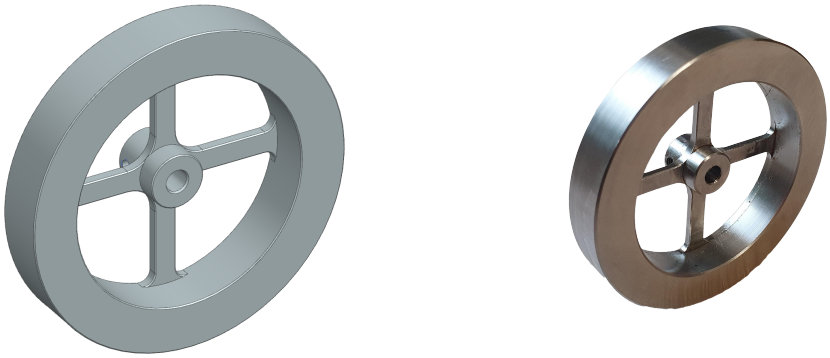



**Figure 3.7.** Results of the simulation performing a reorientation task of 30° in 1 s obtained from the reaction wheel described in table 3.3. The pitch $\theta$ and the pitch rate $\dot{\theta}$ of the main body, and the angular velocity $\dot{\gamma}$ and torque $\tau_m$ are reported in blue. The reference trajectory for the main body is represented in orange, and the bound of the motor in red dashed.

The performances of the wheels obtained are evaluated using the dynamical model presented in section 3.2. Notice that in Elroy's Beanie model, a single reaction wheel is present, thus one must set $I_{fw} = 2I_{zz}$ and $m_{fw} = 2m$. The simulation is performed using the same task and controller. The results are reported in figure 3.7. The two reaction wheels with the dimensions reported in table 3.3 are able to perform the task without saturating the motors velocity and torque.
Starting from the dimension shown in table 3.3, a detailed 3D model of the reaction wheel has been realized in NX software tool [54], an advanced high-end CAD software. In this model, the spokes and all the other details needed to mount the wheel to the shaft of the motor are included. To fasten the wheel to the shaft it has been decided to use two M3 threaded grains. This is not the best option to connect two rotating objects but allows an easier mounting and dismounting procedure. With this type of attachment, the tolerances of hole is set to $G7$, in this way a clearance fit with the shaft is generated. The final design of the flywheel is shown in figure 3.8, while in figure A.3 is reported the technical drawings. The CAD software used to realize the model allows the evaluation of dynamical properties. The mass and inertia of the final model (which include the spokes) are calculated and reported in table 3.4.



(a) (b)

**Figure 3.8.** Reaction wheel design: 3D model realized on NX (a), and picture of the manufactured piece (b).

| Parameter | Value |
|---|---|
| $m$ | 0.116 kg |
| $I_{zz}$ | $7.4 \cdot 10^{-5}$ kgm$^2$ |
| $I_{xx}, I_{yy}$ | $4.0 \cdot 10^{-5}$ kgm$^2$ |

**Table 3.4.** Dynamic parameters of one reaction wheel calculated from the 3D model realized in NX

## 3.4 Custom Components Design

In addition to the reaction wheels, the orientation control system includes some other custom components, such as the shaft, the codewheel mounter, and the shell

(a) (b)

**Figure 3.9.** Custom shaft design: 3D model realized on NX (a), and picture of the manufactured piece (b).

that is the structure containing the whole flywheel assembly. They are realized both with machining procedures and additive manufacturing techniques.

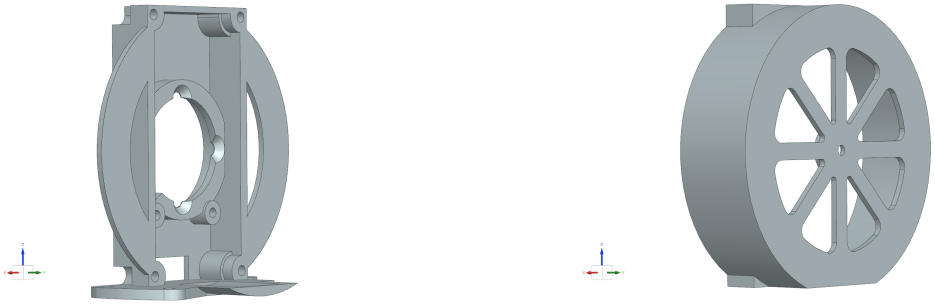
*Shaft.* Due to the presence of the reaction wheel, the shaft provided with the motor must be modified. In figure A.1 is reported the technical drawing of the component with the dimension and tolerances. I choose the diameter of 4 mm and the tolerance $h9$ as the shaft used in the other joints. With these values, it fits in the motor bearings without additional adjustments. On the shaft some slot are present, that allow to connect motor, flywheels and the codewheel mounter. A larger diameter of 6 mm is used to realize a mechanical stop, to have a better placement for the wheel and the codewheel mounter. In figure 3.9 are reported the picture of the model and the real component.

*Codewheel Mounter.* The codewheel is an plastic ring used form the optical encoder to detect shaft rotations. The codewheels used on the robot are provided without a mounter necessary to mount them on the shaft. The Open Dynamic Robot Initiative propose the usage of a 3D printed component glued to the gear wheel used in the transmission. Since the gear wheel is not present in the flywheel transmission, I designed a new mounter that ensures the axial constraint of the components. The technical drawings of this component are reported in figure A.2. Even this component is fastened with two headless screw to the shaft with threaded grains, instead of being glued like the ones in the leg's motors. In order to fit on the shaft the tollerance of the inner hole is set to $G7$. The hole has been designed to have the same tolerance as the flywheel's hole. Both the shaft and the codewheel mounter are realized in stainless steel with conventional manufacturing process, such as lathing and milling. Figure 3.10 reports the model and the manufactured piece of the codewheel mounter.

*Shell.* This shell has been designed to avoid the reaction wheel colliding with other objects, like elements of the surrounding environment or other joints of the robot. Additive manufacturing has been used to realize a case able to contain all the components of the orientation control system. It is used a 3D printer based on selective laser sintering (SLS), a laser melts the polymer powder deposited layer after layer by a counter-rotating leveling roller. I decided to use 3D printed components

(a) (b)

**Figure 3.10.** Codewheel mounter design: 3D model realized on NX (a), and picture of the manufactured piece (b).



(a) (b)

**Figure 3.11.** First design of the shell. The two parts (a) and (b) are screwed together when the other components are assembled.

to obtain a lightweight and cheap case, easy to substitute in case of breaks. Inside the faces of the shell are realized some holes. They allow airflow when the flywheels spin, avoiding the generation of turbulence inside the shell and having the effect of cooling down the motor coils. An additional hole has been realized to pass the wires that connect the motor and the encoder to the motor board. In figure 3.11 is reported the 3D model realized for the shell.

To verify the strength of the shell, we employed two dynamic finite element model (FEM) simulations. This simulation returns the local value of the von Mieses stress[3] for each node of the model. This value has to be compared with the yield stress of the material, in this case, ABS. These analyses are used to point out the critical points of the structure. The simulation are performed only on critical impacts, that are the worst conditions in which the orientation control system will operate. The scenarios analysed are two:

1. a fall from a height of $40cm$ in which the robot lands horizontally directly on the belly, without using the legs to soften the impact;

2. a fall from a height of $40cm$ in which the robot lands on the side.

[3]The von Mieses stress is used to predict the yield of a material under multi-axial load using the uni-axial yield stress

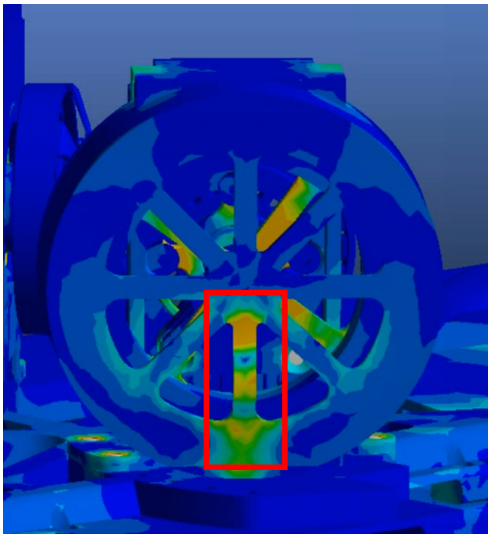
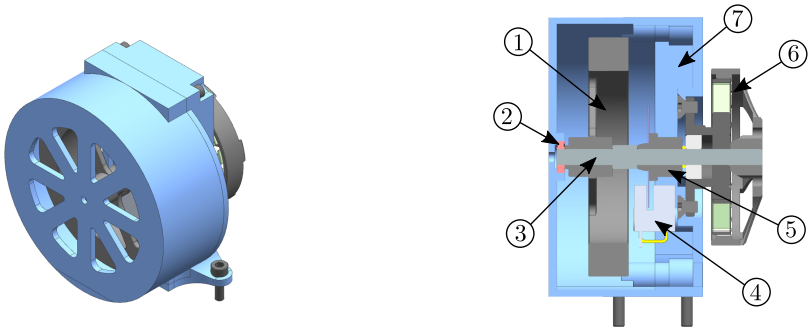


**Figure 3.12.** Snapshot of one FEM simulation result. During the simulation one of the shell spoke (the one highlighted with the red rectangle) is not able to withstand the impact. The range of colors from blue to red are used to indicate the local value of the von Mieses stress. The orange correspond to a value critical for the ABS, but not for steel.

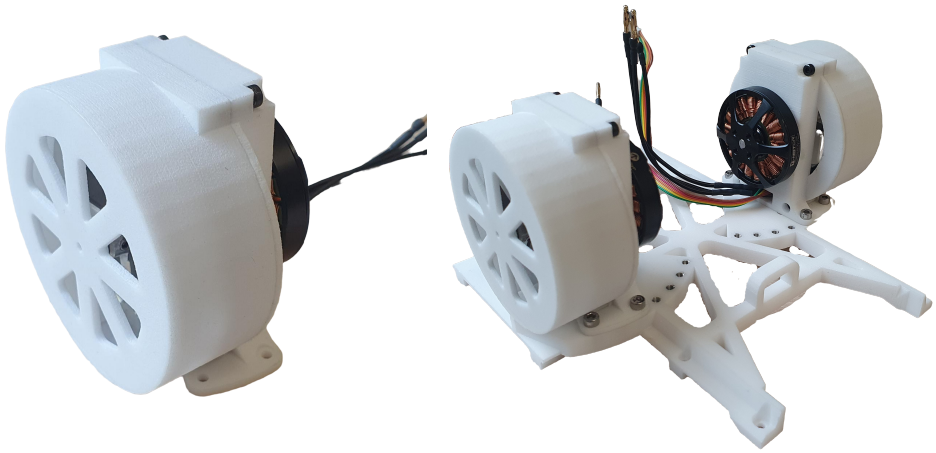
The twenty milliseconds after the impact are simulated on Ansys [55], an engineering simulation software. In particular, in the first scenario, the spokes of the shell are not able to withstand the impact, as shown in figure 3.12. To solve the problem the thickness and width of the spokes have been increased, respectively to 6 mm and to 2 mm. The second scenario points out oscillations of the structure. To reduce them, some high radius fitting between the placement of the screws and the structure has been added. The final result of the design can be seen in figure 3.13. In figure A.4 is present the technical drawing with the most important dimensions.

To assemble the two modules on the trunk of the robot, the body structure has




(a) (b)

**Figure 3.13.** The final design of one orientation control module is represented in (a) an isometric view and (b) a section that shows the assembly of all the components. The components are: ① reaction wheel, ② bearings, ③ custom shaft, ④ encoder, ⑤ codewheel with the mounter, ⑥ motor, and ⑦ shell.



(a) (b)

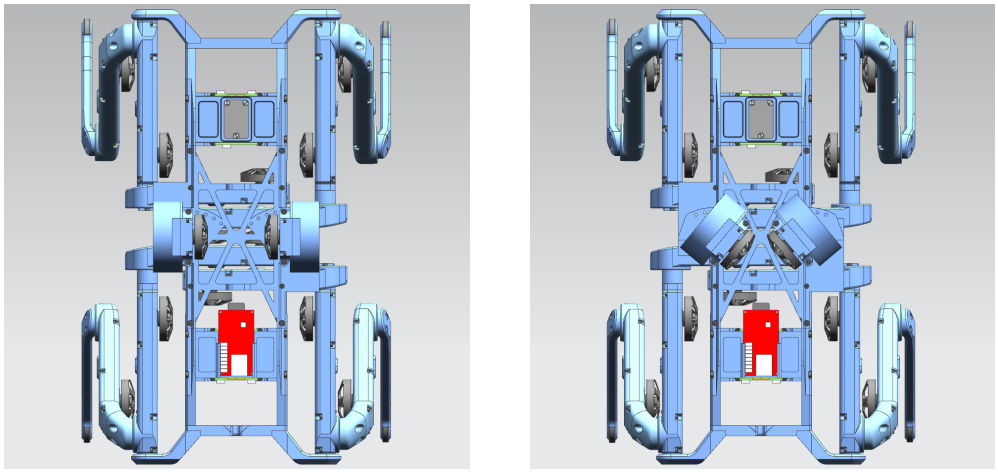
**Figure 3.14.** Orientation control system mounted: (a) a single shell with the components inside, and (b) complete system mounted

been redesigned. The old structure has been enlarged to sustain the modules and new holes are added to allow the anchoring of the shells. These holes are placed to allow the rotation of the shell from 0° up to 40° with steps of 10°. In figure 3.15 are reported the two extremes configuration in which the reaction wheels can be mounted. The technical drawing can be found in figure A.5.

Starting from the centroidal inertia of the robot in homing configuration the optimal angle, at which position the reaction wheels should be mounted, is estimated. In this way, the angular momentum given by the flywheel is partitioned proportionally to the roll and the pitch. More details about that are reported in the chapter 4.

$$\alpha_{opt} = \tan^{-1}\left(\frac{I_{xx}}{I_{yy}}\right) \simeq 40^{\circ} \tag{3.12}$$

Different configurations for assembling the reaction wheel allows verifying the correctness of the theoretical optimal angle. Depending on the maneuver performed, it may not be necessary to have the angular momentum split in that way and maybe more contribution on the pitch should be given.



(a) (b)

**Figure 3.15.** Two possible configurations in which the reaction wheels can be mounted: the axes of rotation are mounted parallel to the $y$-axis of the robot (a) or with an angle of 40° with respect to the $y$-axis of the robot (b).

The system assembled on the robot result as in figure 3.16.

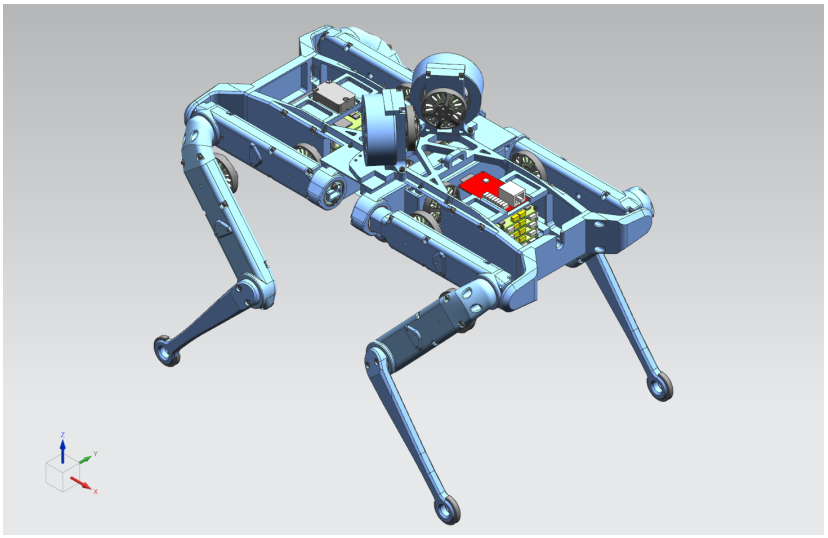


**Figure 3.16.** Orientation control system mounted on the trunk of Solo12.

# Chapter 4

# Control Schemes

The control laws proposed in this thesis are obtained leveraging the reaction wheel contribution to the total angular momentum of the robot in the base frame $\mathbf{L}_f$. For better clarity, we express it in the base reference frame:

$$ {}_b\mathbf{L}_f = \mathbf{L}_{fl} + \mathbf{L}_{fr} = I_f\boldsymbol{\omega}_{fl} + I_f\boldsymbol{\omega}_{fr} = \begin{bmatrix} I_f\left(\omega_{fl} + \omega_{fr}\right)\sin(\alpha) \\ I_f\left(\omega_{fl} - \omega_{fr}\right)\cos(\alpha) \\ 0 \end{bmatrix} \tag{4.1} $$

with $\boldsymbol{\omega}_{fl} = \omega_{fl}\begin{bmatrix} sin(\alpha) & cos(\alpha) & 0 \end{bmatrix}^T$ and $\boldsymbol{\omega}_{fr} = \omega_{fr}\begin{bmatrix} sin(\alpha) & -cos(\alpha) & 0 \end{bmatrix}^T$. Above, $I_f$ is the $I_{zz}$ component of the tensor of inertia of the wheels (the values are reported in table 3.4), $\alpha$ is the angle at which the reaction wheels are mounted, and the scalars $\omega_{fl}$ and $\omega_{fr}$ are the angular speeds of the left and right reaction wheels, respectively. It is possible to notice from figure 4.2 that the sum of the angular momentum vectors of the flywheels affects the roll, the $x$-axis rotation. While the

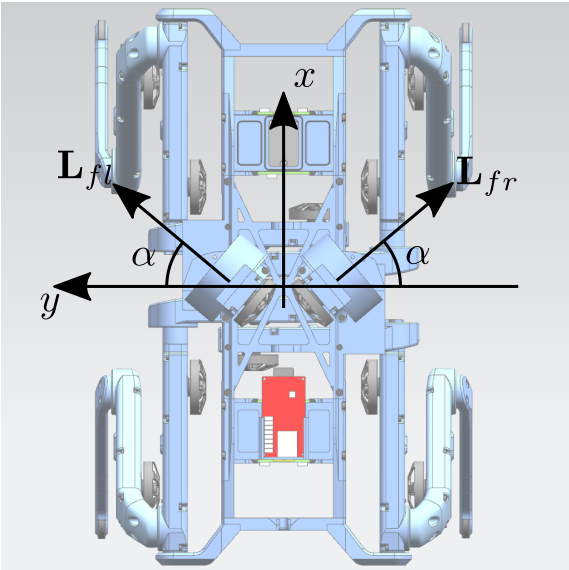



**Figure 4.1.** Representation of the angular momentum vectors of the reaction wheels, seen from the top of the robot

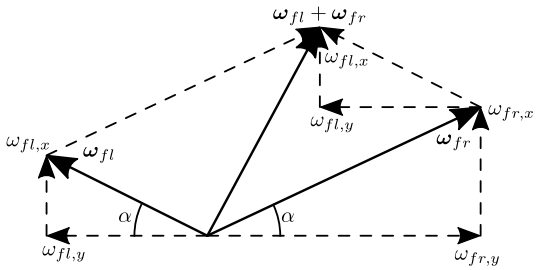



**Figure 4.2.** The sum of the two reaction wheels angular velocities, can be obtain by the sum of the $x$-components and the difference of the $y$- one.

difference affects the pitch, the $y$-axis rotation. None of the contribution affects the yaw rotation, indeed this rotation in not actuated by the flywheels. A disturbance on this rotation can not be directly compensated, but, thanks to the non-holonomity of the angular momentum constraint can be corrected following a pattern of rotation in a time horizon. Observe that, if the reaction wheels have the axis of rotation in common and parallel to the y-axis, that correspond to $\alpha = 0$, only the pitch is controlled. Equation 4.1 can be rewritten as a matrix product:

$$\mathbf{L}_f = I_f \begin{bmatrix} \sin(\alpha) & \sin(\alpha) \\ \cos(\alpha) & -\cos(\alpha) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_{fl} \\ \omega_{fr} \end{bmatrix} = I_f \mathbf{C}\mathbf{u} \tag{4.2}$$

In which $\mathbf{u}$ is our control input. The control schemes developed in this thesis use the matrix $\mathbf{C}$ to map the angular velocity and the torques of the reaction wheels into the base frame. Indeed, the column of this matrix represents the axes of rotation of the two wheels in the base reference frame. We can observe that if we map Lf in an inertial frame, the third row will not be any more zeros. This means the flywheel action for non-flat orientations will influence the yaw direction of the robot.

## 4.1 PD Controller

A proportional and derivative (PD) controller is a simple control strategy that allows to steer the orientation of the robot, to track a desired reference, without overshoot. Its block diagram is shown in figure 4.3. Defining the reference values for orientation and angular velocity, the PD action will try to track them Since orientations in the 3-dimensional space are elements of $SO(3)$, one can either express with Euler angles $\boldsymbol{\phi}^{des} \in \mathbb{R}^3$ or with quaternions $\mathbf{Q}^{des} \in \mathbb{H}$. The angular velocity is expressed as $\boldsymbol{\omega}^{des} \in \mathbb{R}^3$. Additionally, the control action can compensate for external disturbances. Due to the under-actuation, only the external moment along the base $x-$ and $y$-axes are affected by the reaction wheels.

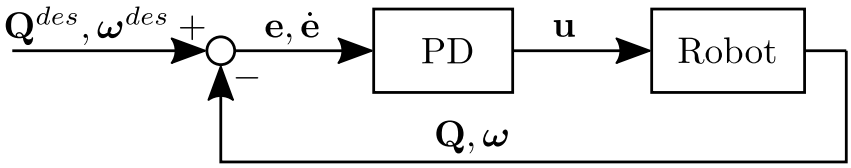



**Figure 4.3.** Block diagram of the PD control scheme. In the scheme, quaternions are used to represent the actual and desired orientation, but they can be substituted with the roll-pitch-yaw angles $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{des}$. The subtraction is an operation that must be intended to be performed in the quaternion algebra or in the rotation group to obtain the orientation error [51].

To define the control law we first define the angular momentum as the sum of the contribution given by the reaction wheel $\mathbf{L}_f$ and the contribution given from the rest of the robot $\mathbf{L}_b$:

$$\mathbf{L}_f + \mathbf{L}_b = const.$$

During the absence of contact with the environment the total angular momentum is conserved and is possible to rewrite it as in equation 2.8:

$$\mathbf{I}_f \dot{\boldsymbol{\omega}}_f + \mathbf{I}_b \dot{\boldsymbol{\omega}}_b + \boldsymbol{\omega}_b \times (\mathbf{I}_b \boldsymbol{\omega}_b) = 0$$

Form this expression we can define $\boldsymbol{\tau}_{fb}$, that is the moment on the base caused by the acceleration of the flywheels:

$$\begin{aligned} \boldsymbol{\tau}_{fb} &= \mathbf{I}_f \dot{\boldsymbol{\omega}}_f \\ &= -\mathbf{I}_b \dot{\boldsymbol{\omega}}_b - \boldsymbol{\omega}_b \times (\mathbf{I}_b \boldsymbol{\omega}_b) \end{aligned}$$

We can use the PD action $\mathbf{K}_p \mathbf{e} + \mathbf{K}_d \dot{\mathbf{e}}$ to generate a reference acceleration for the base $\dot{\boldsymbol{\omega}}_b$.
The matrices $\mathbf{K}_p \in \mathbb{R}^{3\times3}$ and $\mathbf{K}_d \in \mathbb{R}^{3\times3}$ are diagonal positive-definite gains matrices for the error in attitude and angular velocity. The proportional error $\mathbf{e} \in SO(3)$ needs the algebra of the special rotational group to be calculated. Depending on the representation used to define the orientation, different expressions exist to describe the error. The derivative error can be calculated using $\dot{\mathbf{e}} = \boldsymbol{\omega}^{des} - \boldsymbol{\omega}$, in which $\boldsymbol{\omega}^{des}$ and $\boldsymbol{\omega}$ are, respectively, the desired and actual angular velocity of the base:

$$\boldsymbol{\tau}_{fb} = -\mathbf{I}_b \left(\mathbf{K}_p \mathbf{e} + \mathbf{K}_d \dot{\mathbf{e}}\right) - \boldsymbol{\omega}_b \times (\mathbf{I}_b \boldsymbol{\omega}_b)$$

Projecting $\boldsymbol{\tau}_{fb}$ onto the flywheel axes with $\mathbf{C}^T$ we obtain the control action $\mathbf{u}$

$$\mathbf{u} = \mathbf{C}^T \boldsymbol{\tau}_{fb} \tag{4.3}$$

### 4.1.1 Orientation Error with Roll-Pitch-Yaw Angles

In the first case the error is calculated with

$$\mathbf{e} = \boldsymbol{\phi}^{des} - \boldsymbol{\phi} \tag{4.4}$$

in which $\boldsymbol{\phi}^{des} \in \mathbb{R}^3$ and $\boldsymbol{\phi} \in \mathbb{R}^3$ are respectively the column vectors of the desired and actual roll $\psi$, pitch $\theta$, and yaw $\phi$ angles. This representation of the error is

intuitive, but the singularities present in the calculation of $\boldsymbol{\phi}$ can lead to errors if the pitch reaches the values of $\pm\pi/2$ (in the case of the yaw-pitch-roll Euler sequence which is the one commonly used in robotics). Since the objective of this thesis is to enhance the aerial capability of Solo12, constraining the motion away from the singularities would be too restrictive.

### 4.1.2 Orientation Error with Quaternion Angles

Describing the orientation with quaternions avoids the issue of representing the singularities. A quaternion $\mathbf{Q}$ is composed by a scalar part and by an imaginary vector part

$$\mathbf{Q} = \begin{bmatrix} \eta \in \mathbb{R} \\ \boldsymbol{\varepsilon} \in \mathbb{R}^3 \end{bmatrix}$$

To calculate the error between two quaternions are necessary the following operations:

- the inverse of a quaternion

$$\mathbf{Q}^{-1} = \begin{bmatrix} \eta \\ -\boldsymbol{\varepsilon} \end{bmatrix};$$

- the multiplication between two quaternions

$$\mathbf{Q}_3 = \mathbf{Q}_1 \otimes \mathbf{Q}_2 = \begin{bmatrix} \eta_1\eta_2 + \boldsymbol{\varepsilon}_1^T\boldsymbol{\varepsilon}_2 \\ \eta_1\boldsymbol{\varepsilon}_2 + \eta_2\boldsymbol{\varepsilon}_1 + \boldsymbol{\varepsilon}_1 \times \boldsymbol{\varepsilon}_2 \end{bmatrix}.$$

The error between two of them is defined as

$$\Delta\mathbf{Q} = \mathbf{Q}^{-1} \otimes \mathbf{Q}^{des} = \begin{bmatrix} \Delta\eta \\ \Delta\boldsymbol{\varepsilon} \end{bmatrix} \tag{4.5}$$

, in which $\mathbf{Q}^{des}$ and $\mathbf{Q}$ are respectively the desired orientation and the actual one. The vector part $\Delta\boldsymbol{\varepsilon}$ of the resulting quaternion $\Delta\mathbf{Q}$ corresponds to the orientation error expressed in the base frame. The error obtained with this method is in the base frame, therefore there is no need of rotation.

## 4.2 Bang-Bang Controller

The second control technique developed to actuate the orientation control system is a bang-bang controller. This consist in accelerating the reaction wheels with the maximum torque for a small amount of time and then let them rotating at constant speed. The robot base will turn in the opposite direction, due to conservation of angular momentum. This strategy can be used on predefined jumps in which the robot lands on a surface with a different inclination with respect to the one of the thrusting. This control scheme is developed to leave the legs to control only the CoM trajectory while the reaction wheels obtain the final desired orientation. The calculation of the duration of the acceleration requires:

- the dynamical model of the robot;

- the time at which the reorientation maneuver finishes;
- the rotation to be performed during the jump.

With the "Elroy's Beanie" model, described in section 3.2, it is possible to analyze the case in which only the pitch or the roll are changed. This linear model allows the analytical integration of the equation of motion. Below are shown the angular acceleration of the pitch base $\ddot{\theta}(t)$ and of the reaction wheels $\ddot{\gamma}(t)$.

$$\begin{aligned} \begin{bmatrix} \ddot{\theta}(t) \\ \ddot{\gamma}(t) \end{bmatrix} &= \begin{bmatrix} I_0 + I_1 & I_1 \\ I_1 & I_1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \tau(t) \end{bmatrix} \\ &= \begin{bmatrix} -\tau/I_0 \\ \tau\left(I_0 + I_1\right)/I_0 I_1 \end{bmatrix} \end{aligned} \tag{4.6}$$

From the analytical expression of $\theta$ (obtained integrating two times $\ddot{\theta}(t)$), one can estimate the duration of the impulse necessary to generate the desired motion. Notice that in this model, the moment $\tau$ does not correspond directly to the torque of a single motor, but it is the overall contribution along the desired axes. The torques of the two motors are transformed in $\tau$ similarly to the previous control schemes. In this case, $\tau$ corresponds only to the $y$-component of the motor moment.
Let apply the maximum achievable torque during the interval of time of length $\Delta T$.

$$\tau(t) = \begin{cases} \tau_{max} & \text{if } 0 < t \leq \Delta T \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

From now on, $\tau_{max} > 0$ refers to the maximum torque applied by the orientation control system to the desired axis of the robot. The discontinuities in 0 and $\Delta T$ do not allow to obtain the same behavior on the mechanical system, which filters the response depending on its dynamics. Assuming that $\theta(0) = 0$, and $\dot{\theta}(0) = 0$ the analytic solution results as follow. Different initial conditions can be included in $\theta(t_f)$, as will be shown later in this section.

$$\ddot{\theta}(t) = \begin{cases} -\dfrac{\tau_{max}}{I_0} & \text{if } t \leq \Delta T \\ 0 & \text{otherwise} \end{cases}$$

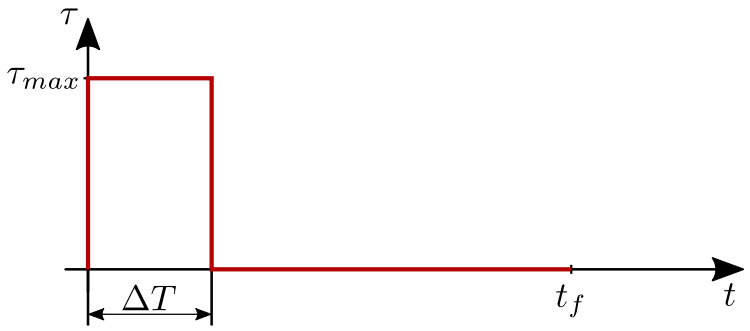



**Figure 4.4.** Bang-bang torque profile presented in equation 4.7.

$$\dot{\theta}(t) = \begin{cases} -\dfrac{\tau_{max}}{I_0} t & \text{if } t \leq \Delta T \\ -\dfrac{\tau_{max}}{I_0} \Delta T & \text{otherwise} \end{cases} \tag{4.8}$$

$$\theta(t) = \begin{cases} -\dfrac{\tau_{max}}{2I_0} t^2 & \text{if } t \leq \Delta T \\ \dfrac{\tau_{max}}{I_0} \Delta T \left( \dfrac{\Delta T}{2} - t \right) & \text{otherwise} \end{cases} \tag{4.9}$$

An estimation of $\Delta T$ can be done imposing the desired orientation $\theta_f$ at time $t_f$ in equation 4.9.

$$\Delta T = t_f \pm \sqrt{t_f^2 + 2\frac{I_0}{\tau_{max}}\theta_f} \tag{4.10}$$

From the solutions, only the one with the minus is accepted. The other solution it is discarded since the pulse duration $\Delta T$ is longer than the flight time $t_f$. Since the rotation is in the opposite direction of the torque, the term $2\frac{I_0}{\tau_{max}}\theta_f$ is always negative. We obtain a solution only if the argument of the square root is positive.

$$t_f^2 + 2\frac{I_0}{\tau_{max}}\theta_f > 0 \tag{4.11}$$

If this condition is not respected the argument of the square root is negative, which denotes that does not exist a value of $\Delta T$ that gives the desired final orientation. This means that the flight time $t_f$ is not sufficient to obtain the desired final orientations with the designed flywheels' inertia. In this case, $\Delta T = t_f$ obtain the closest result to the desired one.
Since the torque profile in (4.7) has two instantaneous changes (at $t = 0$ and $t = \Delta T$) that are impossible to be realized by a physical system, the trapezoidal shape is investigated. The ramp duration $t_r$ is chosen in order to obtain a feasible profile for the dynamics of the motor with the flywheel.

$$\tau(t) = \begin{cases} \tau_{max}\dfrac{t}{t_r} & \text{if } 0 < t \leq t_r \\ \tau_{max} & \text{if } t_r < t \leq \Delta T + t_r \\ \tau_{max}\dfrac{\Delta T + 2t_r - t}{t_r} & \text{if } \Delta T + t_r < t \leq \Delta T + 2t_r \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

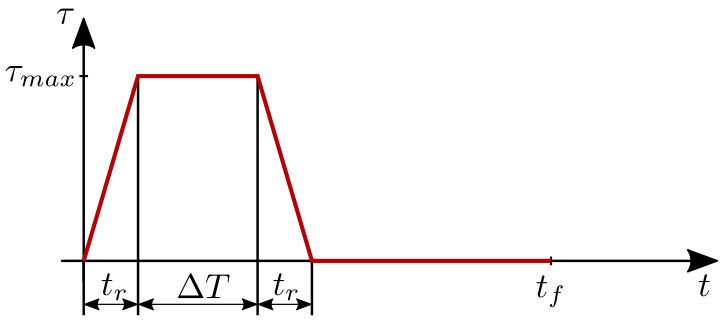



**Figure 4.5.** Bang-bang torque profile presented in equation 4.12.

With this torque profile, the value of $\Delta T$ becomes

$$\Delta T = t_f - \frac{3t_r}{2} - \sqrt{t_f^2 - t_f t_r + \frac{t_r^2}{4} + 2\frac{I_0}{\tau_{max}}\theta_f} \tag{4.13}$$

If the inertia of the robot has a relevant variation with respect to its nominal value, the predefined shape of the torque may lead the system orientation to an undesired final value. Therefore, the movements of the legs generate a wrong estimation of $\Delta T$. To overcome the problem, $\Delta T$ is continuously recomputed with the new updated value of the current position, velocity, and centroidal inertia updated at the actual configuration. Changing the initial conditions in equation 4.6 and starting from the middle of the pulse modify the expression of $\Delta T$. In particular, using the torque profile in (4.7) $\Delta T$ becomes

$$\Delta T = t_f - \sqrt{t_f^2 - 2\frac{I_0}{\tau_{max}}\left(\dot{\theta}_0 t_f + \theta_0 - \theta_f\right)}$$

whilst if (4.12) is used, $\Delta T$ transforms into

$$\Delta T = t_f - \frac{t_r}{2} - \sqrt{t_f^2 - \frac{t_r^2}{12} - 2\frac{I_0}{\tau_{max}}\left(\dot{\theta}_0 t_f + \theta_0 - \theta_f\right)} \tag{4.14}$$

in the case it is used the trapezoidal profile.

The angle $\theta_0$ and the angular velocity $\dot{\theta}_0$ are the initial conditions and $t_f$ is the remaining time until the landing. If we substitute $t_r = 0$ (that means an instantaneous change of torque) in the solution (4.14) we obtain the same solution of the discontinuous case. Adding this modification, the controller results are more accurate. With the profile of toques used until now, we choose the pulse duration to obtain the desired final orientation $\theta_f$, but it is not possible to set a condition on the final angular velocity $\dot{\theta}_f$. It is preferable to obtain a zero final angular velocity in order to have less destabilizing effect, and reduce the impact at the landing. To solve the problem, we shape a second torque profile in the opposite direction of the first one in order to decelerate to zero the angular velocity right before the landing. In this case,for sake of simplicity, we discuss only control inputs with discontinuities.

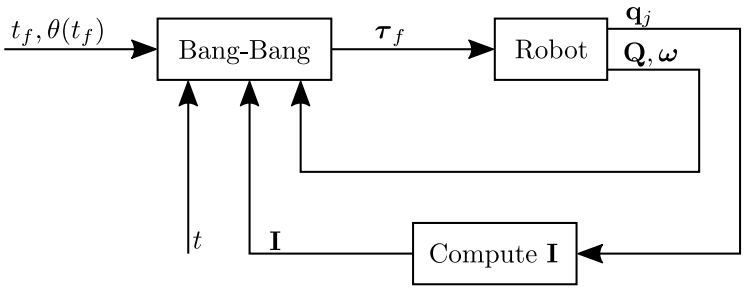



**Figure 4.6.** Bloch diagram of the bang-bang like controllers. To recompute the right time at which put zero to the torque the controller needs, the actual orientation $\mathbf{Q}$ and angular velocity $\boldsymbol{\omega}$, actual the tensor of the centroidal inertia $\mathbf{I}$ (which needs the current joint configuration $\mathbf{q}_j$ to be computed).

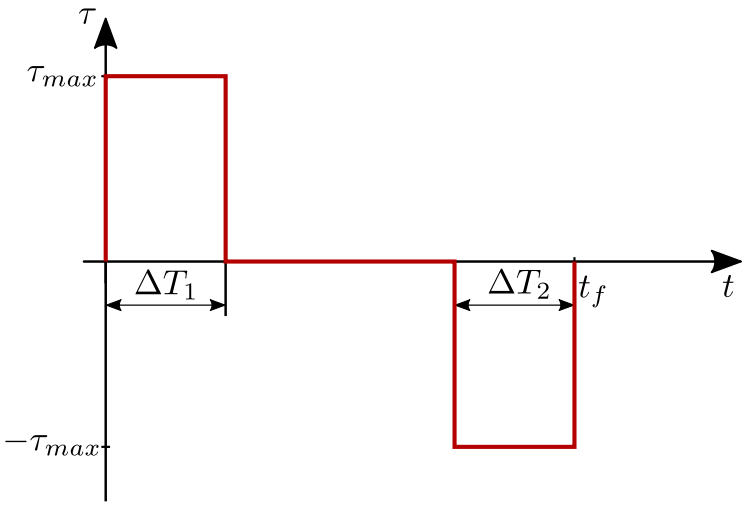



**Figure 4.7.** Bang-bang torque profile presented in equation 4.15.

Defining the two peaks duration, $\Delta T_1$ and $\Delta T_2$, and the total time of flight $t_f$, the torque applied is

$$\tau = \begin{cases} \tau_{max} & \text{if } 0 < t \leq \Delta T_1 \\ -\tau_{max} & \text{if } t_f - \Delta T_2 < t \leq t_f \\ 0 & \text{otherwise} \end{cases} \tag{4.15}$$

The two pulse duration are calculated imposing the final desired values, $\theta(t_f) = \theta_f$ and $\dot{\theta}(t_f) = 0$ in equations 4.9 and 4.8.

$$\begin{cases} \Delta T_1 = \dfrac{t_f}{2} - \sqrt{\dfrac{t_f^2}{4} + \dfrac{I_0}{\tau_{max}}\theta_f} \\ \Delta T_2 = \Delta T_1 \end{cases}$$

The timings are calculated also for generic initial conditions, $\theta_0$ and $\dot{\theta}_0$.

$$\begin{cases} \Delta T_1 = \dfrac{t_f}{2} + \dfrac{I_0}{2\tau_{max}}\dot{\theta}_0 - \sqrt{\dfrac{t_f^2}{4} + \left(\dfrac{I_0\dot{\theta}_0}{2\tau_{max}}\right)^2 - \dfrac{I_0}{\tau_{max}}\left(\dfrac{\dot{\theta}_0 t_f}{2} + \theta_0 - \theta_f\right)} \\ \Delta T_2 = \Delta T_1 - \dfrac{I_0}{\tau_{max}}\dot{\theta}_0 \end{cases} \tag{4.16}$$

Like in the other cases this new expression is used to recompute the timings with an update value for the centroidal inertia $I$ at the current joint configuration $\mathbf{q}_j$, the current position $\theta$, and velocity $\dot{\theta}$.

# Chapter 5

# Validation

In this chapter, I will show results of simulations to confirm the capabilities of the mechanical design and the controllers.

## 5.1 Simulations

The software used to perform the simulations are Gazebo and ROS. The first performs the integration of the dynamics of the robot in an physical environment to obtain the results, in particular it integrates the rigid body model of the robot. The second manages the communication between the simulator and the controller. All the parameter used by the simulator, such as sampling rate, physics engine, and more, are stored in a world file. The reaction wheels modeled as rigid bodies, with the respective values of mass and tensor of inertia, and included with 2 additional rotational joints inside the Unified Robot Description Format (URDF) [57]. A URDF is an XML format for representing a robot model. In this file are defined the joints type present on the robot with their properties, such as joint limit, actuation limit, and friction. Here are also included all the links with their: mass, tensor of inertia, and meshes for collisions and also visualization. The graphical representation of the robot is shown in figure 5.1. This way, it is possible to obtain a realistic

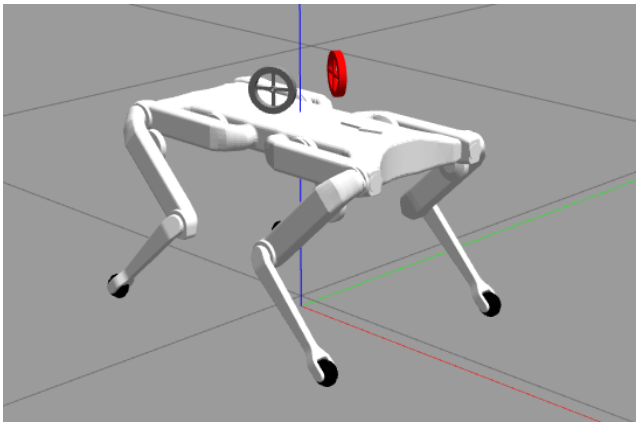


**Figure 5.1.** Model of the robot with the reaction wheels in Gazebo simulator.

simulation of the system. This simulator allows changing the physics properties of the environment, such as the gravity vector. Exploiting this feature allows us to analyze the behavior in scenarios different with respect to the Earth, like Moon (reduced gravity) or space (without gravity). The gravity does not affect directly the reorientation task but decreasing it results in a longer flight phase, which last for an infinite time in case it is zero.

The results of the simulations are reported in the accompanying videos[1].

### 5.1.1 Simulations with Zero Gravity

The gravity is set to 0 to simulate a space-like environment. The robot floats for an infinite time, allowing to study the convergence of the reorientation task. At the beginning of the simulation, the robot is placed at the height of 0.8 m with the base parallel to the ground, and zero initial angular velocity. I chose the distance from the ground to ensure none of the links collide with the ground during the rotation. It stays with the joint blocked in homing configuration for one second, and then the reorientation maneuver starts.

*PD controller.* The first simulation shows that the value of inertia used in the design allows rotating the base of $30° \simeq 0,5236$ rad around the base $y$-axis in 1 s. The reaction wheels are mounted parallel to the pitch axis and they are controlled by the PD controller. The reorientation task starts after one second, when the reference for the pitch is set to 0.5 rad[2]. In figure 5.2 are reported the data of the base orientation, while figure 5.3 shows the plots of the reaction wheels effort and angular velocity. The gains used are $K_p = 40.0$ and $K_d = 2.0$. The simulation results show the device is able to meet the specification required.

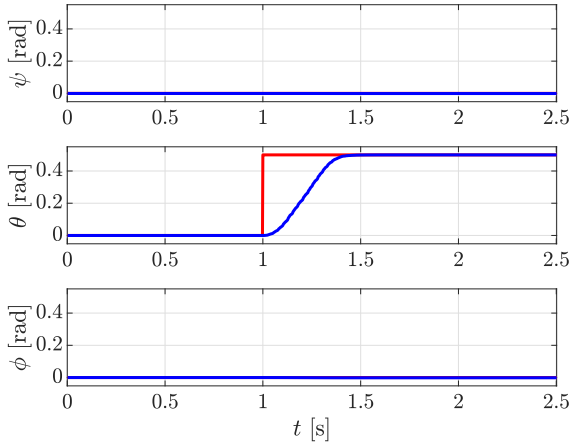



**Figure 5.2.** Base orientation, represented in roll, pitch and yaw angles (respectively $\psi$, $\theta$ $\phi$), during a rotation on the pitch using the PD controller. The red lines represent the desired orientation and blue is used for the actual.

[1]The videos of the simulation are stored in the shared folder: `https://www.dropbox.com/sh/rgffnu1s2orhv5j/AACy7g-q13r4j_zDRUViqjdWa?dl=0`

[2]The reference in roll-pitch-yaw is transformed in the respective quaternion and then passed to the controller.

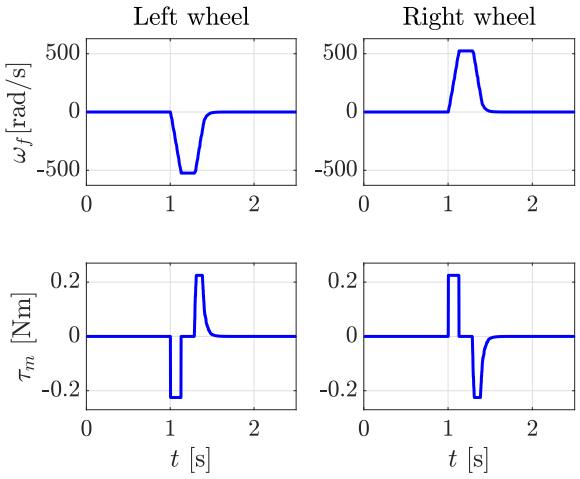



**Figure 5.3.** Reaction wheels velocity $\dot{q}_{fw}$ and effort $\tau$ during the reorientation maneuver shown in figure 5.2.

Placing the wheels with incident axes, it is possible to control even the roll. The simulation was realized with $\alpha = 40°$ and the same controller as before. The reference values for the base orientation where $-0.5$ rad for the pitch and 0.2 rad for the roll. The simultaneous rotation about the $x$-axis and $y$-axis generates disturbances on the yaw angle. Due to the under-actuation of the orientation, it is not possible to nullify this error. In figure 5.4 are reported the results.

*Bang-Bang controller* I tested the bang-bang controller with its variants. As for the PD controller, we check if using this controller is possible to meet the specification. Therefore the final desired value for the pitch is set to 0.5 rad to be reached in 1 s. Since the gravity is zero, an there are no external forces acting on the robot and the controller is open loop, it continues to rotate with the final angular velocity. Figure 5.5 shows the results of this simulation.

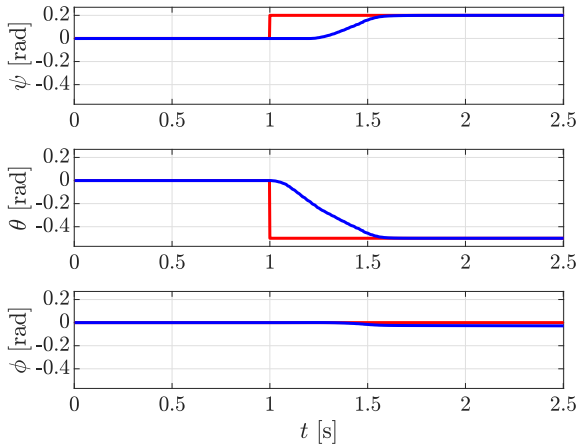



**Figure 5.4.** Base orientation, represented in roll, pitch, and yaw, during a rotation on the pitch and the roll using the PD controller. Red lines represent the desired orientation and blue is used for the actual.

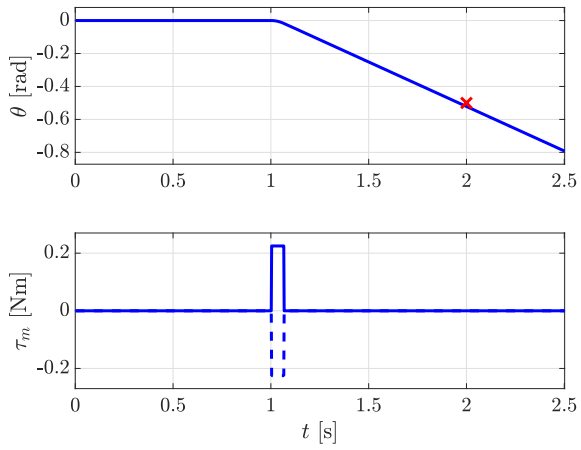



**Figure 5.5.** Robot pitch $\theta$ orientation and torque $\tau$ at the reaction wheels using the bang-bang controller. The red x shows the desired orientation at the end of the task. In the torque graph the solid line represents the torque applied to the left wheel while the dashed one to the right wheel.

The other two variations of this controller are tested using the same final orientation $\theta_f$ and time of flight $t_f$, where I set $\theta_f = -0.5$ rad and $t_f = 1$ s. In figure 5.6 are shown the results for bang-bang with the trapezoidal profile. It is possible to notice how using two ramps of length $t_r$ entails a minimum base rotation. The torque profile should become a triangle ($\Delta T = 0$) with a height smaller than $\tau_m ax$ to achieve a rotation smaller than this angle. Both the simulations with the bang-bang controller point out that at the end of the reorientation maneuver, the angular velocity of the robot is not zero. The bidirectional bang-bang can be used to nullify the base angular velocity at the end of the task. From the results in figure 5.7, we can see

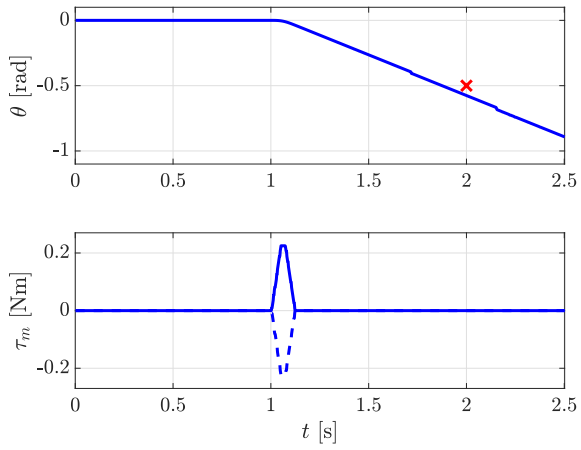



**Figure 5.6.** Robot pitch $\theta$ orientation and torque $\tau$ at the reaction wheels using the bang-cost-bang controller with trapezoidal profile. The red x shows the desired orientation at the end of the task. In the torque graph the solid line represents the torque applied to the left wheel while the dashed one to the right wheel.

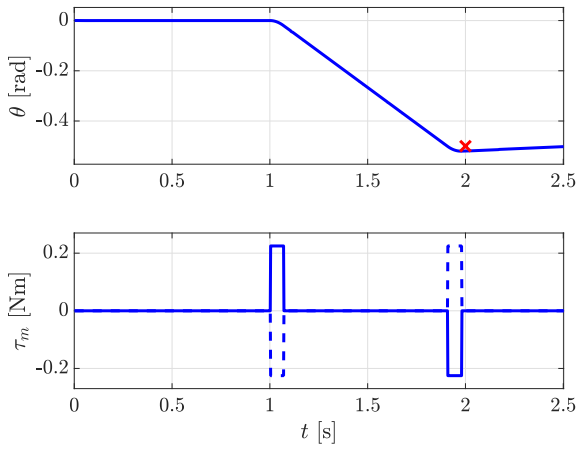



**Figure 5.7.** Robot pitch $\theta$ orientation and torque $\tau$ at the reaction wheels using the bang-cost-bang controller. The second pulse annihilates the angular velocity of the base maintaining the constant the final orientation.

that the final velocity is not zero. This is due to the usage of an open-loop controller, which can not compensate for modeling inaccuracies.

In the last simulation, whose results are show in figure 5.8 and 5.9, we show the robot starting with a non-zero angular velocity. The convergence is still guaranteed but requires more time than the other cases. In contrast to the other simulations, the reaction wheels rotate at constant angular speed after that the reorientation has been completed. As a matter of fact, the angular momentum is not zero when the simulation starts, and it must been constant for all the time. To perform the simulation I use the PD controller. The robot starts with a pitch angle of $-15°$ ($\simeq 0.26$ rad) and a pitch rate of $-0.2$ rad/s.

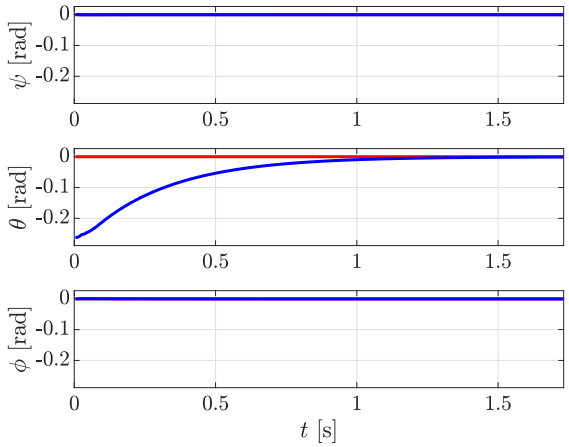



**Figure 5.8.** Orientation of the robot base, represented in roll, pitch and yaw angles (respectively $\psi$, $\theta$, and $\phi$), during a reorientation maneuver. The robot base starts with a pitch angle of $-15°$ ($\simeq 0.26$ rad) and an initial pitch rate of $-0.2$ rad/s. Red lines represent the desired orientation and blue is used for the actual.

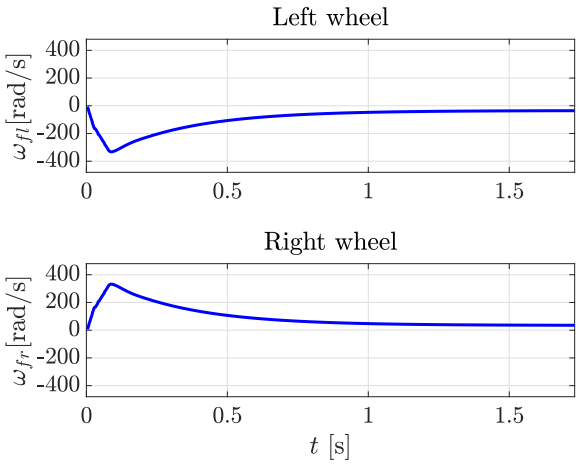



**Figure 5.9.** Angular velocity of the reaction wheels to perform the reorientation in figure 5.8. The angular rotation of the reaction wheels converges to an absolute value of 35 rad/s.

### 5.1.2 Simulations with Earth Gravity

To validate the effects of the reaction wheels on a more realistic scenario, I add the gravitational acceleration to the simulation environment. This subsection shows the simulations in which the gravity is set to 9.81 m/s$^2$, the Earth one. The first case analyzed is a falling. The robot is placed in mid-air with a given (non-horizontal) orientation and zero initial angular velocity. Due to the gravity, the robot starts falling, and the orientation control system begins the reorientation maneuver to get the base of the robot to be parallel to the ground.

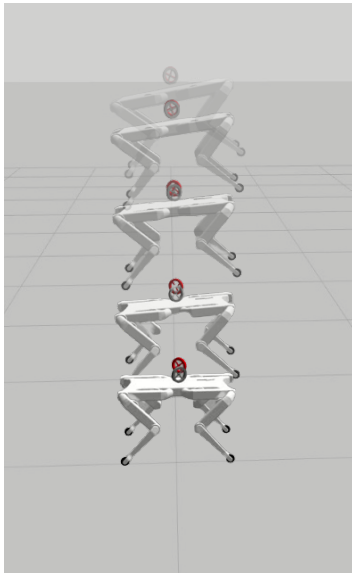


**Figure 5.10.** Snapshot the reorientation maneuver during a falling of 1.5 m. The robot starts with an initial orientation of $-15°$, and during the fall try to bring this value to zero.

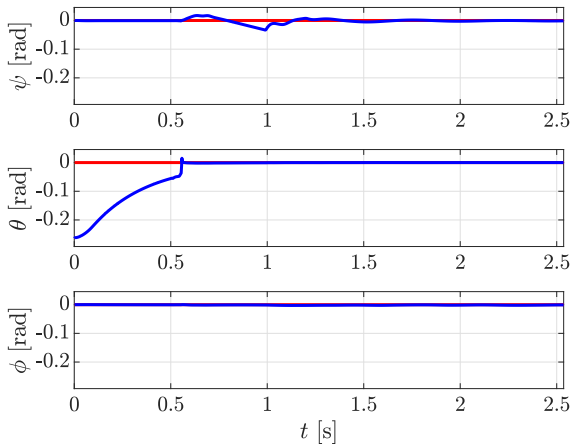



**Figure 5.11.** Orientation of the robot base in roll pitch yaw angles ($\psi$, $\theta$, and $\phi$ respectively) during the reorientation maneuver in a fall from 1.5 m. The robot starts with an initial pitch orientation of $-15°$, and during the fall tries to bring this value to zero. Red lines represent the desired orientation and blue is used for the actual.

The aerial motion lasts slightly more than 0.5 s during which the robot perform a pitch rotation from $-15°$ to $-2.3°$. The final orientation is considered acceptable due to its small value. During the simulation, a PD controller at joint level keeps the legs in homing configuration. This strategy does not obtain a good landing and generates large oscillations both in the base position and orientation. Since the reaction wheels are oriented with $\alpha = 0°$, the orientation system compensates only for the pitch. The legs attenuate the other oscillations stabilizing the final pose. Figure 5.10 shows some snapshot of the robot during the maneuver and in figure 5.11, is plotted the orientation of the base.

Another maneuver that can be done on the Earth is a jump. To realize this highly dynamic movement, we use Crocoddy [58], an optimal control library based on differential dynamic programming (DDP) algorithms. This algorithm calculates references for joint angular position, velocity and torque in order to minimize a given cost function. Setting a proper cost function is possible to realize a jump in which the parameters, such as maximum height, forward and lateral landing position, and duration of thrusting, flying, and landing phases can be set. The flywheel joints are part of the optimization and the optimization will provide refernces also for them But in these simulations, I use this technique to obtain the reference trajectory only for the legs' joints while controlling the reaction wheels with the already mentioned methods.

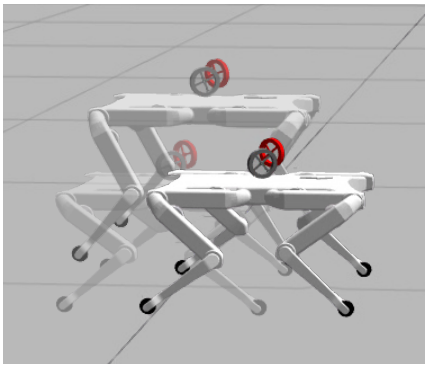


**Figure 5.12.** Snapshot of a forward jump maneuver in earth gravity.

Since jumps on earth do not last long, the orientation error generated from a wrong tracking of the ground reaction forces is not big. Instead, at the landing, there are oscillations of the base, unless a proper landing controller strategy is implemented. The device developed in this thesis is able to dump the rotational oscillations along the $x$- and $y$-axis, dramatically improving the stability at landing. In the simulation the robot perform a jump of 30 cm in the forward direction reaching a maximum height of 40 cm. The thrusting phase lasts 0.2 s while the flight time lasts 0.3 s. After the lift-off the PD controller actuate the reaction wheels to maintain the orientation of the base parallel to the ground. The gains are set to $K_p = 40.0$ and $K_d = 2.0$. Figure 5.13 shows the orientation of the base performing the same jump with all the possible mounting positions of the reaction wheels. Positioning the axes of rotation parallel to the base $y$-axis ($\alpha = 0$) reduces the pitch oscillation, while in the other cases, it is possible to lower both the roll and the pitch ones. The yaw rotation is similar in all the cases and it is not possible to counteract it with the designed controllers.

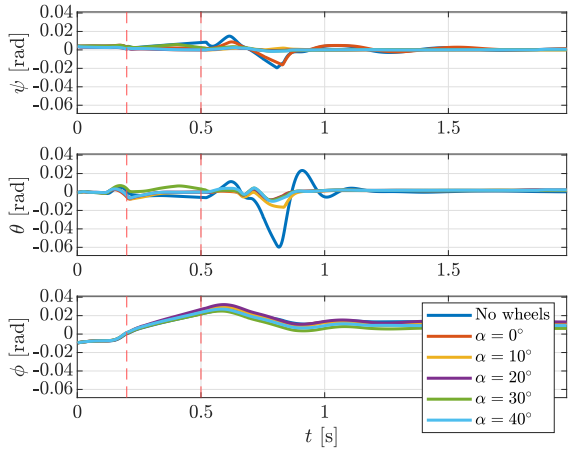



**Figure 5.13.** Orientation of the robot base in roll pitch yaw angles ($\psi$, $\theta$, and $\phi$ respectively) during a jump with earth gravity. The orientation control system corrects the orientation during the flight phase and dumps the oscillations generated during the landing. The red dashed lines indicates the lift-off and the landing.

### 5.1.3 Simulations with Moon Gravity

Finally, the gravitational force is set to the one of the Moon (1.62 m/s$^2$). With respect to the earth's gravity, it is possible to obtain a higher jump with a longer flight phase while still staying inside actuator limits. In particular, I analyze the effectiveness of disturbances rejection of the orientation control system. The only controller able to reject disturbances is th PD controller. The gains are set to $K_p = 40.0$ and $K_d = 2.0$. During the aerial motion, an external moment of $M_{ext} = \begin{bmatrix} 0.3 & 0.3 & 0 \end{bmatrix}^T$ Nm, expressed in the world frame, is applied to the base for a total time interval of 0.05 s. The test is performed with different orientations of the reaction wheels. In the cases without reaction wheels and with $\alpha = 0°$, the orientation control system can not compensate properly for the disturbance, causing a fall on the robot back. In the simulation with $\alpha = 10°$, the developed system can not fully counteract the effect of the external moment on the roll direction, but the orientation error remains small enough to be nullified at the landing. Figure 5.14 shows these three cases. As shown in figure 5.15, for higher values of $\alpha$, the disturbances on the roll and the pitch are nullified by the effect of the reaction wheels. With these simulations we validate the ability of the reaction wheel system to annihilate the effect of small external moments applied to the base.

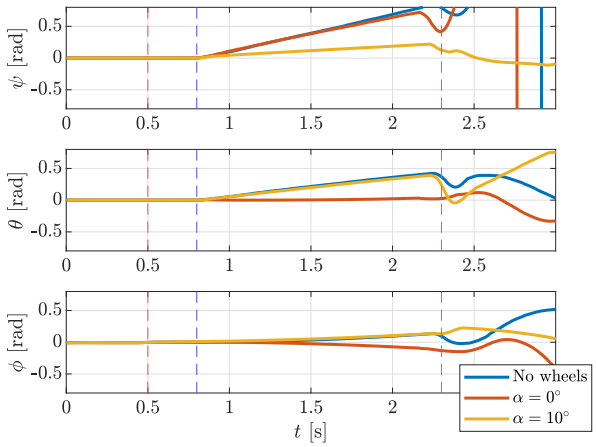



**Figure 5.14.** Orientation of the robot in roll pitch yaw angles ($\psi$, $\theta$, and $\phi$ respectively) while performing a jump on the moon. At 0.8 s an external moment to the base is applied, simulating an external disturbance. In the simulation shown in this picture, the orientation control system can fully compensate for pitch disturbances but not for the roll one. The red dashed lines indicates the lift-off and the landing. The blue dashed line indicate the time at which the external moment is applied to the base.

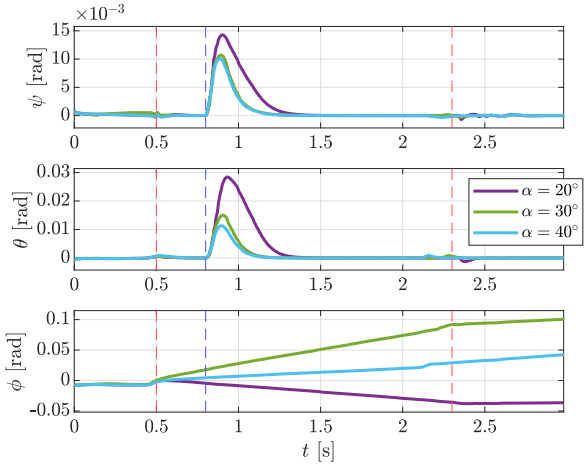



**Figure 5.15.** Orientation of the robot in roll pitch yaw angles ($\psi$, $\theta$, and $\phi$ respectively) while performing a jump on the moon. At 0.8 s the simulator applies an external moment to the base simulating an external disturbance. With the current values of $\alpha$, the orientation control is able to compensate for the external moment, and the robot returns to the desired orientation. The red dashed lines indicates the lift-off and the landing. The blue dashed line indicate the time at which the external moment is applied to the base.

# Chapter 6

# Conclusions and Future Works

I developed from scratch an orientation control system based on reaction wheels for Solo12, an open-source quadruped robot. First, describing the robot dynamics with the single rigid body model, I illustrated a procedure to define the kinematic and dynamic parameters of the two wheels requiring to obtain a rotation on the pitch angle of 30° in 1 s. Then, I designed the components necessary to build the device. These are: the shaft to connect each wheel to its motor, the mounter to assemble the codewheel to the motor shaft, the protective shell to contain all the system, and a new body structure to fasten the new components to the remainder of the robot. Next, I developed two methods to control the base orientation. One actuates the reaction wheels using a proportional and derivative (PD) law on the orientation error. The other derives from a bang-bang strategy, in which the motors deliver the maximum available torque for a pre-computed time interval. Some variants of the second methodology allow for having a continuous torque profile or for obtaining a final zero angular velocity of the base. The thesis report concludes with some simulations to validate the effectiveness of the orientation control system. These have been carried out in case of zero, Earth's, and Moon's gravity. The motions analyzed are falls and jumps. Simulations show the ability of the designed device to spin the robot from an initial non-zero orientation to the horizontal orientation, and to reject disturbances during the flight phase. Hardware has been mounted, and some preliminary tests have been performed on basic functionalities.

In the future, I want to validate the results by performing some experiment. No additional considerations are required to replicate the simulations with Earth gravity on the physical hardware, while the others expose some technical complexity. The scenario with zero gravity might be imitated by hanging the robot with a rope, which cancels the translation due to the weight force. If one can connect the rope to the robot CoM, the gravity does not generate any moment on the base.
Below I report some additional ideas to improve this project.
The encoder and the relative codewheel can be removed from the design of the device. The motor board should be substituted by an electronic speed controller (ESC), which governs the motor speed without using any external sensor, thus reducing friction. This technique uses the back-electromotive force (back-EMF) to determine the rotor angular velocity instead of employing the measures coming from the encoder. Removing these components, the orientation control system will result

lighter and more compact.
The procedure used to define the sizes of the reaction wheels for Solo12 is valid also for other quadrupeds and even for robots with different morphology, such as jet packs or underwater vehicles.
The control strategies developed in this thesis can be improved using non-linear model predictive control (NMPC). This technique takes into consideration the future samples of the orientation reference. I expect this feature allows for stabilizing the yaw to the desired value, even if it is not locally controllable, enhancing the non-holonomy property of the angular momentum. Another control strategy can be inherited from quadricopter controllers: during steady operations, they are kept horizontal in mid-air, but they must lean first if a translation is required. This comes from the fact that the control input (thrust) can only be applied in the direction normal to the base. Similarly, the designed reaction wheels can only generate torques on the robot's frontal and sagittal axes while we want to control rotations about the vertical direction. A strategy that quadricopter employs to achieve the maneuver is dynamic feedback linearization. I suppose it may enhance the capability of following yaw trajectories, as the NMPC.

# Appendix A

# Technical Drawings

In this appendix the technical drawing used to realized the custom components used in the project are reported: the shaft to connect a wheel to its motor, the mounter to assemble the codewheel to the motor shaft, the shell in which all the system is installed, and a new body structure to fasten the new component. All the linear dimensions reported on the tables are in millimeters. Figures A.1, A.2, and A.3 show the pieces realized in steel. The figures A.4, and A.5 show the components realized with SLS additive manufacturing. The path of the laser is generated automatically by the printer software from the 3D model of the object. For this reasons on the technical drawing of the latter shown on this thesis is not necessary to put all the dimensions.

## A.1 Custom shaft

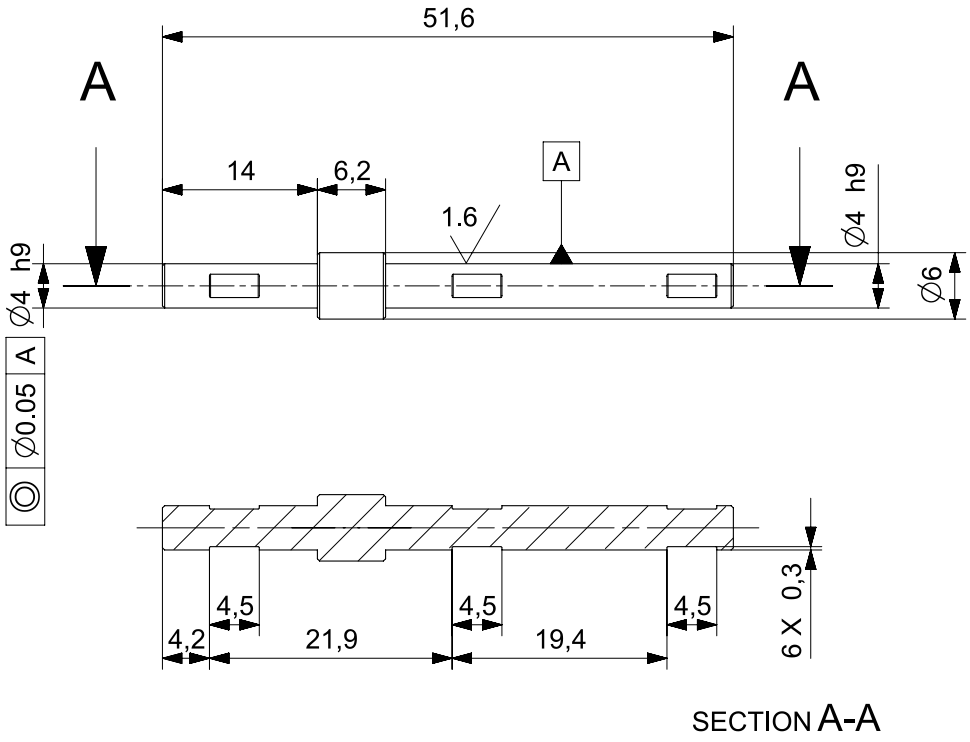



**Figure A.1**

## A.2 Codewheel mounter

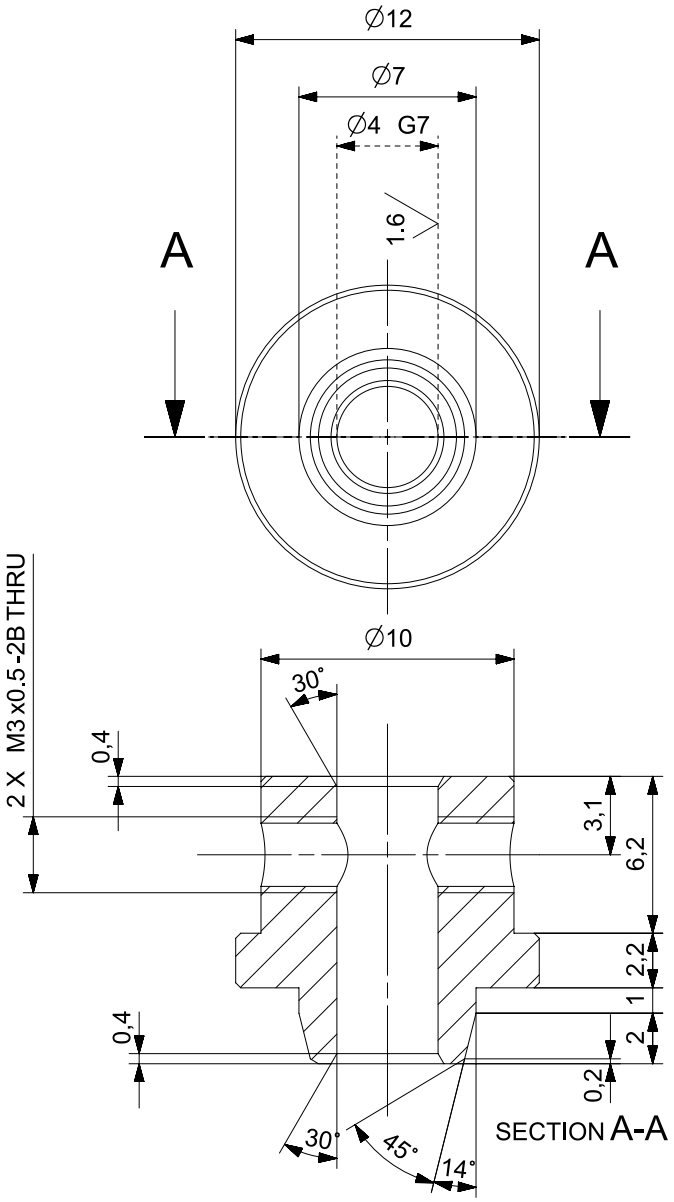



**Figure A.2**

## A.3 Reaction wheel

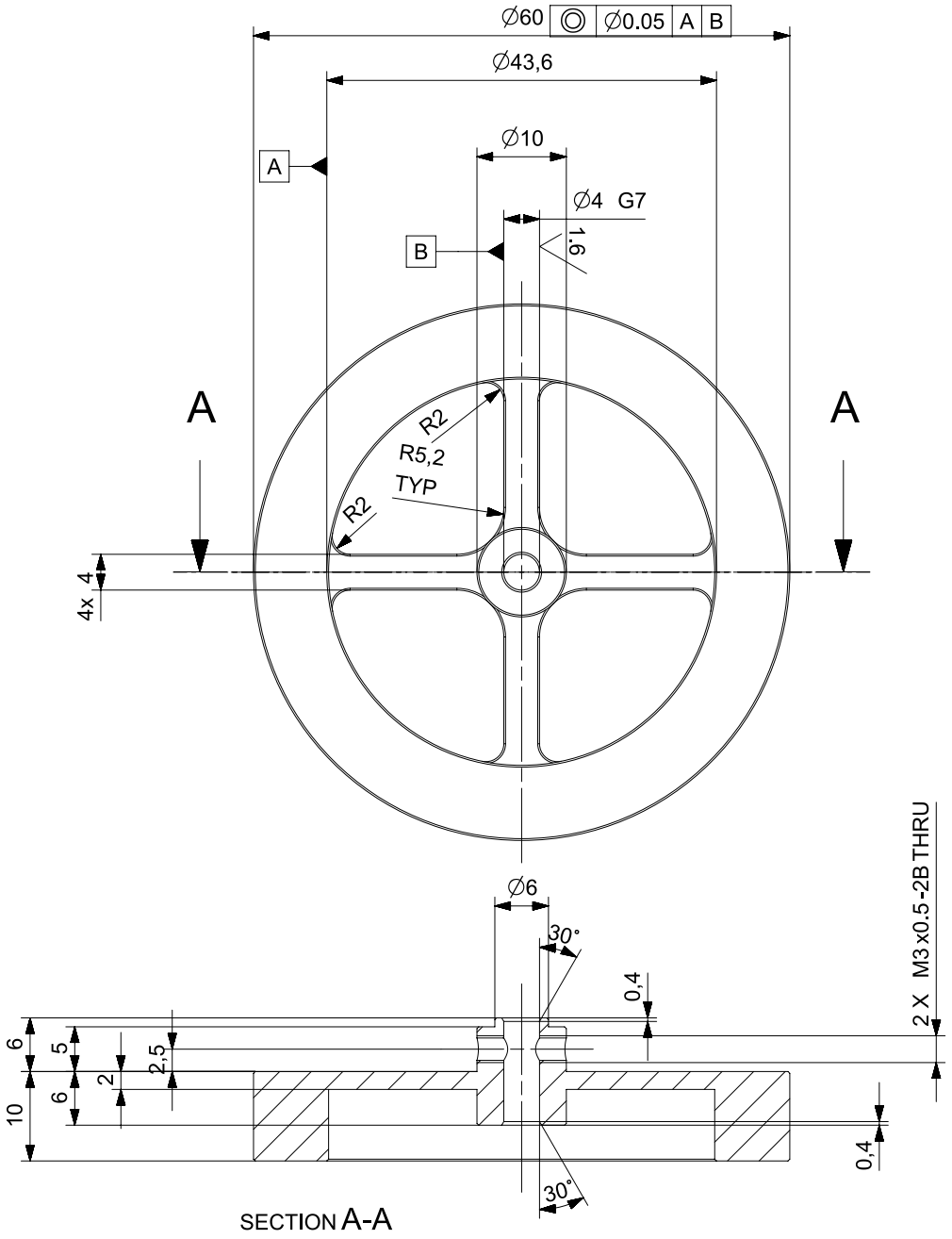



**Figure A.3**

## A.4 Shell

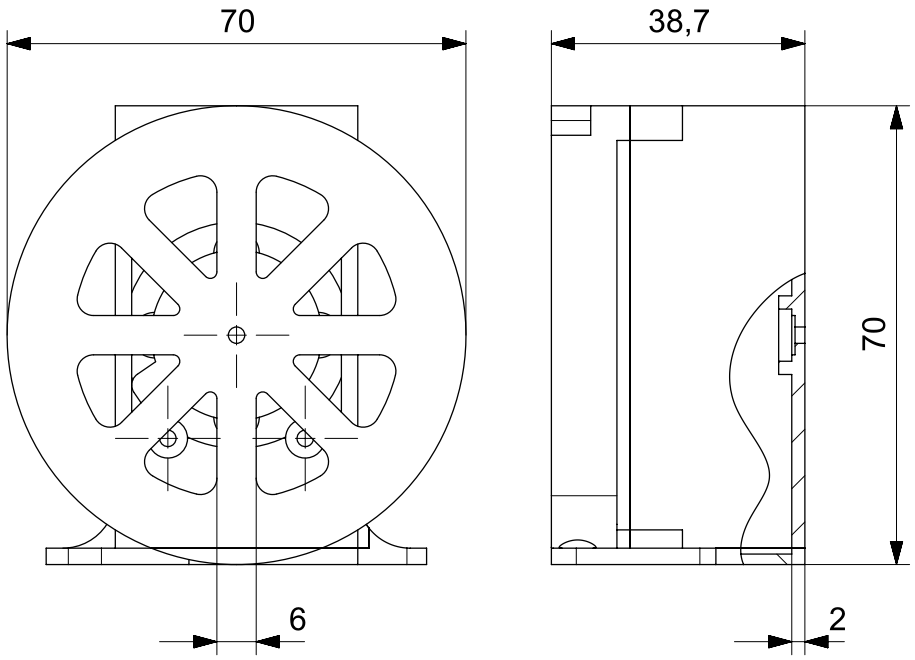



**Figure A.4**

## A.5 Body structure

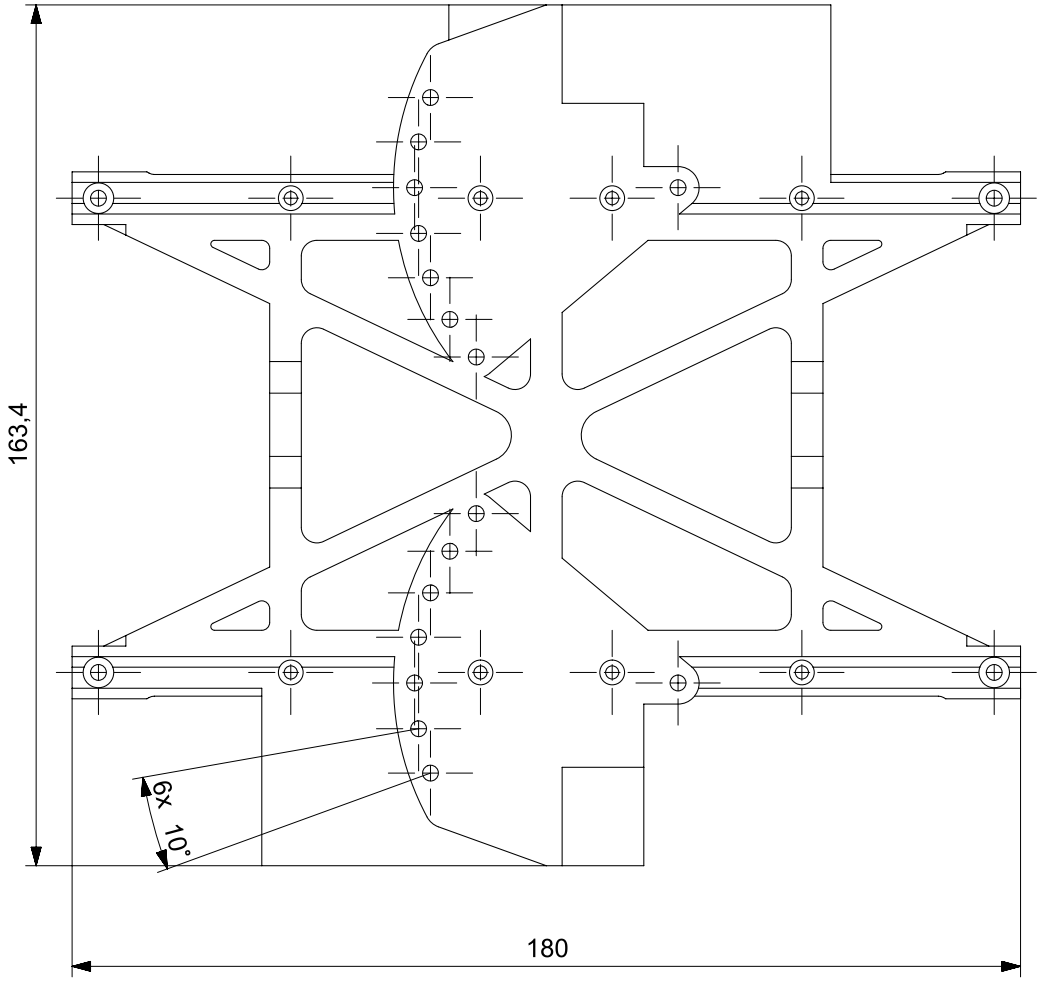



**Figure A.5**

# Bibliography


[1] S. G. Tzafestas, "Mobile robot control and navigation: A global overview," *Journal of Intelligent & Robotic Systems*, vol. 91, no. 1, pp. 35–58, 2018.

[2] iRobot, *Roomba*. [Online]. Available: `https://www.irobot.com/`.

[3] Waymo, *Waymo One*. [Online]. Available: `https://waymo.com/waymo-one/`.

[4] Mars Science Laboratory, *Perseverance*. [Online]. Available: `https://mars.nasa.gov/mars2020/spacecraft/rover/`.

[5] J. P. Grotzinger, J. Crisp, A. R. Vasavada, R. C. Anderson, C. J. Baker, R. Barry, D. F. Blake, P. Conrad, K. S. Edgett, B. Ferdowski, *et al.*, "Mars science laboratory mission and science investigation," *Space science reviews*, vol. 170, no. 1, pp. 5–56, 2012.

[6] Teledyne FLIR, *PackBot 510*. [Online]. Available: `https://www.flir.it/products/packbot/`.

[7] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori, "The icub humanoid robot: An open platform for research in embodied cognition," in *Proceedings of the 8th workshop on performance metrics for intelligent systems*, 2008, pp. 50–56.

[8] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent asimo: System overview and integration," in *IEEE/RSJ international conference on intelligent robots and systems*, IEEE, vol. 3, 2002, pp. 2478–2483.

[9] T. Isozumi, K. Akachi, M. Hirata, K. Kaneko, S. Kajita, and H. Hirukawa, "Development of humanoid robot "hrp-2"," *Journal of the Robotics Society of Japan*, vol. 22, no. 8, pp. 1004–1012, 2004.

[10] T. Inamura, K. Okada, S. Tokutsu, N. Hatao, M. Inaba, and H. Inoue, "Hrp-2w: A humanoid platform for research on support behavior in daily life environments," *Robotics and Autonomous Systems*, vol. 57, no. 2, pp. 145–154, 2009.

[11] Boston Dynamics, *ATLAS*. [Online]. Available: `https://www.bostondynamics.com/atlas`.

[12] Agility Robotics, *Digit*. [Online]. Available: `https://agilityrobotics.com/robots`.

[13] Boston Dynamics, *Spot*. [Online]. Available: `https://www.bostondynamics.com/products/spot`.

[14] S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim, "Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 3307–3312.

[15] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE, 2016, pp. 38–44.

[16] C. Semini, N. G. Tsagarakis, B. Vanderborght, Y. Yang, and D. G. Caldwell, "Hyq-hydraulically actuated quadruped robot: Hopping leg prototype," in *2008 2nd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*, IEEE, 2008, pp. 593–599.

[17] C. Semini, V. Barasuol, M. Focchi, C. Boelens, M. Emara, S. Casella, O. Villarreal, R. Orsolino, G. Fink, S. Fahmi, *et al.*, "Brief introduction to the quadruped robot hyqreal," *Istituto di Robotica e Macchine Intelligenti (I-RIM)*, 2019.

[18] T. M. Roehr, F. Cordes, and F. Kirchner, "Reconfigurable integrated multirobot exploration system (rimres): Heterogeneous modular reconfigurable robots for space exploration," *Journal of Field Robotics*, vol. 31, no. 1, pp. 3–34, 2014.

[19] N. Kashiri, L. Baccelliere, L. Muratore, A. Laurenzi, Z. Ren, E. M. Hoffman, M. Kamedula, G. F. Rigano, J. Malzahn, S. Cordasco, *et al.*, "Centauro: A hybrid locomotion and high power resilient manipulation platform," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1595–1602, 2019.

[20] O. Khatib, X. Yeh, G. Brantner, B. Soe, B. Kim, S. Ganguly, H. Stuart, S. Wang, M. Cutkosky, A. Edsinger, *et al.*, "Ocean one: A robotic avatar for oceanic discovery," *IEEE Robotics & Automation Magazine*, vol. 23, no. 4, pp. 20–29, 2016.

[21] Tokyo Institute of Technology and HiBot, *ACM-R5H*. [Online]. Available: `http://www.hibot.co.jp`.

[22] DJI, *Mavic 3*. [Online]. Available: `https://www.dji.com/it/mavic-3`.

[23] senseFly, *eBee*. [Online]. Available: `https://www.sensefly.com/`.

[24] V. Barasuol, J. Buchli, C. Semini, M. Frigerio, E. R. De Pieri, and D. G. Caldwell, "A reactive controller framework for quadrupedal locomotion on challenging terrain," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 2554–2561.

[25] M. Focchi, R. Orsolino, M. Camurri, V. Barasuol, C. Mastalli, D. G. Caldwell, and C. Semini, "Heuristic planning for rough terrain locomotion in presence of external disturbances and variable perception quality," in *Advances in Robotics Research: From Lab to Market*, Springer, 2020, pp. 165–209.

[26] F. Grimminger, A. Meduri, M. Khadiv, J. Viereck, M. Wüthrich, M. Naveau, V. Berenz, S. Heim, F. Widmaier, T. Flayols, *et al.*, "An open torque-controlled modular robot architecture for legged locomotion research," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3650–3657, 2020.

[27] T. Kane and M. Scher, "A dynamical explanation of the falling cat phenomenon," *International journal of solids and structures*, vol. 5, no. 7, pp. 663–670, 1969.

[28] E. M. Hoffman and A. Paolillo, "Exploiting visual servoing and centroidal momentum for whole-body motion control of humanoid robots in absence of contacts and gravity," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 2979–2985.

[29] V. Kurtz, H. Li, P. M. Wensing, and H. Lin, "Mini cheetah, the falling cat: A case study in machine learning and trajectory optimization for robot acrobatics," *arXiv preprint arXiv:2109.04424*, 2021.

[30] P.-B. Wieber, R. Tedrake, and S. Kuindersma, "Modeling and control of legged systems," in *Springer Handbook of Robotics, 2nd Ed*, B. Siciliano and O. Khatib, Eds. Springer, 2016.

[31] X. Chu, C. H. D. Lo, C. Ma, and K. W. S. Au, "Null-space-avoidance-based orientation control framework for underactuated, tail-inspired robotic systems in flight phase," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3916–3923, 2019.

[32] G. Wenger, A. De, and D. E. Koditschek, "Frontal plane stabilization and hopping with a 2dof tail," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 567–573.

[33] A. M. Johnson, T. Libby, E. Chang-Siu, M. Tomizuka, R. J. Full, and D. E. Koditschek, "Tail assisted dynamic self righting," in *Adaptive mobile robotics*, World Scientific, 2012, pp. 611–620.

[34] H. Yoon and P. Tsiotras, "Spacecraft adaptive attitude and power tracking with variable speed control moment gyroscopes," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 6, pp. 1081–1090, 2002.

[35] H. B. Brown and Y. Xu, "A single-wheel, gyroscopically stabilized robot," in *Proceedings of ieee international conference on robotics and automation*, IEEE, vol. 4, 1996, pp. 3658–3663.

[36] N. Mikhalkov, A. Prutskiy, S. Sechenev, D. Kazakov, A. Simulin, D. Sokolov, and I. Ryadchikov, "Gyrubot: Nonanthropomorphic stabilization for a biped," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 4976–4982.

[37] E. Oland and R. Schlanbusch, "Reaction wheel design for cubesats," in *2009 4th International Conference on Recent Advances in Space Technologies*, IEEE, 2009, pp. 778–783.

[38] T. L. Brown and J. P. Schmiedeler, "Reaction wheel actuation for improving planar biped walking efficiency," *IEEE Transactions on Robotics*, vol. 32, pp. 1290–1297, 5 2016, ISSN: 15523098. DOI: 10.1109/TRO.2016.2593484.

[39] X. Xiong and A. D. Ames, "Sequential motion planning for bipedal somersault via flywheel slip and momentum transmission with task space control," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2020, pp. 3510–3517.

[40] H. Kolvenbach, E. Hampp, P. Barton, R. Zenkl, and M. Hutter, "Towards jumping locomotion for quadruped robots on the moon," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 5459–5466.

[41] V. Vasilopoulos, K. Machairas, and E. Papadopoulos, "Quadruped pronking on compliant terrains using a reaction wheel," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2016, pp. 3590–3595.

[42] K. Machairas and E. Papadopoulos, "On quadruped attitude dynamics and control using reaction wheels and tails," in *2015 European Control Conference (ECC)*, IEEE, 2015, pp. 753–758.

[43] M. Gajamohan, M. Merz, I. Thommen, and R. D'Andrea, "The cubli: A cube that can jump up and balance," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 3722–3727.

[44] Open-Dynamic-Robot-Initiative, *Open robot actuator hardware*. [Online]. Available: `https://github.com/open-dynamic-robot-initiative/open_robot_actuator_hardware`.

[45] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE International Symposium on System Integrations (SII)*, 2019.

[46] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: An open-source robot operating system," in *ICRA workshop on open source software*, Kobe, Japan, vol. 3, 2009, p. 5.

[47] M. Focchi, *Locosim*. [Online]. Available: `https://github.com/mfocchi/locosim`.

[48] A. W. Winkler, "Optimization-based motion planning for legged robots," PhD thesis, ETH Zurich, 2018. DOI: `10.3929/ethz-b-000272432`.

[49] S. Traversaro, D. Pucci, and F. Nori, "A unified view of the equations of motion used for control design of humanoid robots," *On line*, 2017.

[50] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, IEEE, vol. 1, 2001, pp. 239–246.

[51] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Contro*. Springer, 2009.

[52] J. Wittenburg, *Dynamics of Multibody Systems*, Second Edition. Springer, Berlin, Heidelberg, 2008, ISBN: 978-3-540-73913-5.

[53] P. Singh and H. Chaudhary, "Optimal design of the flywheel using nature inspired optimization algorithms," *Open Agriculture*, vol. 3, no. 1, pp. 490–499, 2018.

[54] Siemens, *NX*. [Online]. Available: `https://www.plm.automation.siemens.com/global/it/products/nx/`.

[55] *Ansys*. [Online]. Available: `https://www.ansys.com/`.

[56] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, IEEE, vol. 3, 2004, pp. 2149–2154.

[57] Open Robotics, *The URDF pacakge for robot models.* [Online]. Available: `http://wiki.ros.org/urdf`.

[58] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, "Crocoddyl: An efficient and versatile framework for multi-contact optimal control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 2536–2542.