NTNU
Kunnskap for en bedre verden

Department of Engineering Cybernetics

TTK4550 - Specialization Project

Design and Control of a Spring-actuated Jumping Quadruped in Earth Gravity

Author:
Johannes Ihle
Daniel Rosmæl Skauge

Supervisor:
Prof. Dr. Kostas Alexis

Co-supervisor:
Jørgen Anker Olsen

Date: 18th December 2024

Abstract

This specialization project presents the concept design of a small, lightweight jumping quadruped robot optimized for jumps in low gravity environments, which uses torsional springs mounted parallel to knee motors for energy storage and release. This work is motivated by the exploration of Martian lava tubes and other challenging extraterrestrial terrains where traditional wheeled rovers face mobility limitations. The robot is designed to be a platform for developing and testing Deep Reinforcement Learning (DRL) policies for jumping, aerial stabilization, and landing. The robot is designed to be lightweight and low-cost to reduce the risk and cost of damage during testing. Through Simulink Simscape simulations we demonstrate that pure motor actuation is insufficient for achieving significant jump heights, and that that parallel torsional springs significantly increase the jump height. Grid search optimization reveals that equal-length leg segments maximize jump height, with maximum jump height of 1.18m in mars gravity and 0.5m in Earth gravity. The project delivers a detailed CAD model of the spring-actuated leg design, establishing the foundation for hardware implementation and control development in the subsequent master's thesis. While several challenges remain, including the lack of feedback control during jumps, the design shows promise for enabling agile locomotion in Martian lava tubes and other challenging extraterrestrial terrains.

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Description
DRL	Deep Reinforcement Learning
BLDC	Brushless DC
DOF	Degrees of Freedom
CAD	Computer-Aided Design
CNC	Computer Numerical Control
MCU	Microcontroller Unit
MSE	Mean Squared Error
PWM	Pulse Width Modulation
I2C	Inter-Integrated Circuit
ADC	Analog-to-Digital Converter

1 Introduction

1.1 Motivation

Traditional wheeled rovers have successfully explored Mars, with six missions to date [**mars˙rovers˙x6**]. However, specialized robots like Axel [**Axel**] and Reachbot [**ReachBot**] are needed for more challenging terrain. One key target for exploration is Martian and Lunar lava tubes [**lavatubes**] - hollow caverns formed by ancient lava flows. These tubes interest scientists because they could shelter future missions from radiation and micrometeorites, maintain stable temperatures, and contain subsurface water ice [**lavatubes**].

Exploring lava tubes requires navigating rough terrain, sharp rocks, and steep slopes - challenges for wheeled rovers. Wheeled robots are also limited to ground movement and cannot take advantage of the lower gravity on Mars, the Moon, and asteroids. In contrast, jumping quadrupeds can utilize low gravity to potentially jump several meters high [**OLYMPUS2**], allowing them to cross obstacles that would stop wheeled rovers.

While quadruped robots have advanced significantly, they still struggle to jump effectively in Earth gravity. Testing hardware and control systems for low-gravity jumping is difficult since we cannot easily simulate low gravity on Earth. The high-speed impacts during jumping and landing also risk damaging expensive hardware. These challenges motivate our main goal: designing a small, lightweight, low-cost jumping quadruped robot. The low weight and cost reduce damage risk during testing and make the platform more accessible to researchers. We focus on achieving long jumps while maintaining the versatility of quadruped robots, such as walking on rough terrain, adjusting body pose, and carrying scientific equipment.

1.2 Scope

This report covers the design phase of a jumping quadruped robot as part of the TTK4550 Engineering Cybernetics Specialization Project at NTNU. While this project focuses on design, the work will continue in a master's thesis that includes building, testing, and developing control algorithms for the robot.

The project scope includes:

- Creating a MATLAB/Simulink simulation to evaluate design choices
- Selecting an actuation method, either motors only, or a combination of motors and springs
- Choosing specific hardware components
- Developing a CAD model for one leg that:
  - Fits the chosen motors and springs
  - Can be manufactured using NTNU's 3D printing and CNC facilities
  - Withstands jumping forces and impacts

1.3 Related Work

Several researchers have studied robotic jumping for Earth and low-gravity environments. NTNU's Autonomous Robots Lab developed the Olympus robot [**OLYMPUS1**] [**OLYMPUS2**], which uses a spring-assisted 5-bar linkage leg for jumping. The robot weighs TODO kg and can jump TODO meters in Earth gravity, with testing done in simulated low gravity.

EPFL's 600g RAVEN robot (Robotic Avian-inspired Vehicle for multiple ENvironments) [**RAVEN**] uses bird-inspired legs with two degrees of freedom. Similarly to our approach, it uses geared BLDC motors to wind up embedded torsional springs for jumping. RAVEN can jump TODO cm in Earth gravity while also walking and hopping like a bird.

The 15g Grillo robot [**GRILLO**] demonstrates high-speed jumping, reaching takeoff velocities of 1.5 m/s (30 body lengths per second).

2 Theory

2.1 BLDC Motor Model

The brushless DC motor (BLDC) is a type of synchronous electric motor that is driven by direct current (DC) electricity [**PMSM˙BOOK**]. Like many other motors, a BLDC motor's ability to generate torque generally decreases as the motor's speed increases [**Microchip˙BLDC**]. This relationship is often referred to as a motor's torque-speed curve. Figure **??** shows an example of a typical torque-speed curve for a BLDC motor [**Microchip˙BLDC**]. As can be seen, the curve is characterized by four parameters: the peak torque, the maximum speed, the rated speed, and the rated torque. As can be seen from the figure, the rated torque is the highest torque that the motor can deliver for an extended period of time, while the rated speed is the highest speed at which the motor can deliver the rated torque. The motor can typically be run at a higher speed than the rated speed, but at these speeds the delivered torque will start dropping. For brief periods of time, for example from standstill followed by acceleration, the motor can deliver a higher torque than the rated torque, so long as it adheres to the torque speed curve.

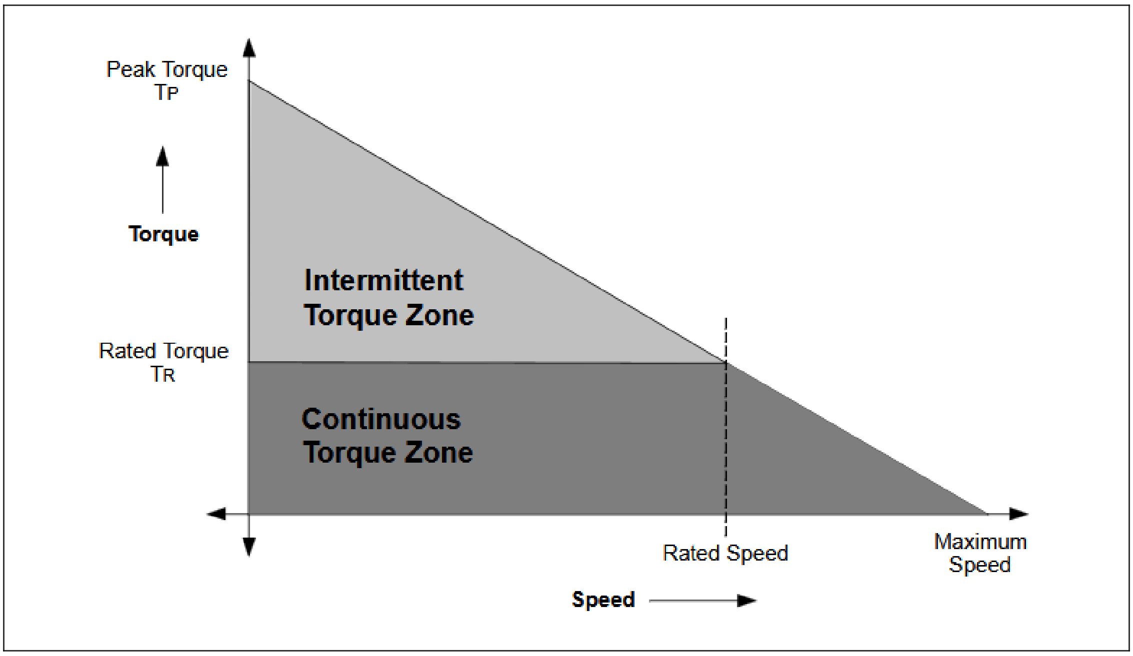


Figure 1: Torque-speed characteristics of a Brushless DC (BLDC) motor. TODO: Replace with an image that isn't stolen, and that we are allowed to use.

2.2 Gear Transmission Friction Model

Although electric motors for robotics are available for a wide power range, they are often too high speed and low power to do any useful work. For this reason, it is often necessary to use a gear transmission to increase the torque and reduce the speed of the motor.

In the presence of a geared transmission, assuming no power loss, the output torque and velocity of a geared motor are given by equation **??** and equation **??**, respectively, where N is the gear ratio, τ_{in} is the input torque, τ_{out} is the output torque, w_{in} is the input velocity, and w_{out} is the output velocity [**modern˙robotics˙book**].

$$w_{out} = \frac{w_{in}}{N} \tag{1}$$

$$\tau_{out} = N\tau_{in} \tag{2}$$

In reality, however, there is always some power loss in the transmission. A common way to model this is to use friction model consisting of a viscous friction term and a Coulomb friction term [**modern˙robotics˙book**]. The viscous friction term is proportional to the velocity of the transmission, and the Coulomb friction term is a constant friction torque that must be overcome before the transmission starts moving. The total friction torque is the sum of these two terms, as seen in equation **??**. It is also possible to drop one or the other of these terms, depending on the application [**modern˙robotics˙book**].

$$\tau_{friction} = b_{viscous}\dot{\theta} + b_{coulomb}\text{sign}(\dot{\theta}) \tag{3}$$

In addition to friction, heavily gearing motors can lead to a very high apparent rotor inertia. If one looks at equation **??**, it is clear that the apparent rotor inertia is proportional to the square of the gear ratio [**modern˙robotics˙book**]. This can lead to a very high apparent rotor inertia, which can often be problematic to robotic applications. This is especially the case for cases with contact forces, as the high apparent rotor inertia can lead to very stiff and damaging collisions [**proprioceptive**].

$$K = \frac{1}{2}I_{rotor}(G\dot{\theta})^2 = \frac{1}{2}I_{rotor}G^2(\dot{\theta})^2 = \frac{1}{2}I_{apparent}(\dot{\theta})^2 \tag{4}$$

2.3 Spring Modeling

Springs are mechanical devices that store and release energy when subjected to displacement. There are two main types of springs: extension springs and torsion springs.

Extension springs are designed to operate with a tension load, meaning they extend as the load is applied. The force exerted by an extension spring is proportional to the displacement from its equilibrium position, following Hooke's Law, which is given by:

$$F = -kx \tag{5}$$

where F is the force exerted by the spring, k is the spring constant, and x is the displacement from the equilibrium position. The potential energy stored in an extension spring is given by:

$$U = \frac{1}{2}kx^2 \tag{6}$$

Torsion springs, on the other hand, are designed to operate with a rotational or twisting load. They exert a torque that is proportional to the angular displacement from their equilibrium position. The torque generated by a torsion spring is given by:

$$\tau = -k\theta \tag{7}$$

where τ is the torque, k is the torsion spring constant, and θ is the angular displacement. The potential energy stored in a torsion spring is given by:

$$U = \frac{1}{2}k\theta^2 \tag{8}$$

Both types of springs are widely used in various mechanical systems to provide force or torque, absorb shock, and store energy.

2.4 Kinematics

2.4.1 Robot Kinematics

Consider a robotic link arm existing in $\mathbb{R}^2$ consisting of n links, each with a length l_i and a joint angle θ_i. The position of the end-effector is given by the vector $\mathbf{x} = [x, y]^T$, where x and y are the coordinates of the end-effector in the global coordinate system. Using simple trigonometry, the position of the end-effector can be expressed as a function of the joint angles and link lengths as seen in equation **??**. Axes and joint angles corresponding to the expression in equation **??** can be seen in figure **??**.

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} l_i \cos\left(\sum_{j=1}^{i} \theta_j\right) \\ \sum_{i=1}^{n} l_i \sin\left(\sum_{j=1}^{i} \theta_j\right) \end{bmatrix} \tag{9}$$

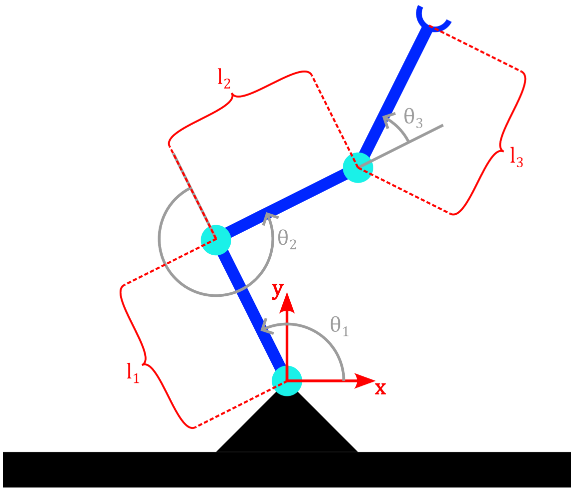


Figure 2: Illustration of a 3 link robotic link arm in $\mathbb{R}^2$ with n links.

2.4.2 Jacobian Matrix

As described in section **??**, the position of the end-effector can be expressed as a function of the joint angles and link lengths. In robotics, it is often useful to express the relationship between infinitesimal changes in the joint angles and the resulting change in the end-effector position. As can be seen in equation **??**, infinitesimal changes in variables δy and δx can be described by means of the partial derivative [**modsim**]. If this is compared to the definition of the jacobian in equation **??**, it is clear that the jacobian matrix $\mathbf{J}$ can be used to map infinitesimal changes in joint angles to changes in the end-effector position, as illustrated in equation **??**. Using the chain rule on **??** one can also arrive at equation **??**, which maps joint velocities to end-effector velocities by means of the jacobian.

$$\delta y = \frac{\partial y}{\partial x} \delta x \tag{10}$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \cdots & \frac{\partial x}{\partial q_n} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \cdots & \frac{\partial y}{\partial q_n} \end{bmatrix} \tag{11}$$

$$\delta \mathbf{x} = \mathbf{J} \delta \mathbf{q} \tag{12}$$

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}} \tag{13}$$

2.4.3 Force/Torque Mapping

Consider a general robotic manipulator, such as the one illustrated in figure **??**, but with an arbitrary amount, n, of joints and links. Assume the joint angles are given by q_i and the joint torques by τ_i. Then, for this manipulator to bear an arbitrary end effector force F, the torques at the joints must be given by equation **??** [**ASADA˙LECTURE˙NOTES**].

This result can be derived via the principle of virtual work. Consider virtual displacements of the individual joint angles δq_i and the end-effector position $\delta\mathbf{x}$. Virtual displacements are infinitesimal displacements of a mechanical system that need only be consistent with the geometric constraints of the system, ie. not the full laws of motion [**ASADA˙LECTURE˙NOTES**].

If the torques $\tau = [\tau_1, \tau_2, ..., \tau_n]$ act on the joints, and the endpoint force $-\mathbf{F}$ acts on the end effector, then the virtual work done by the forces and moments is given by equation **??** [**ASADA˙LECTURE˙NOTES**].

$$\delta\mathbf{W} = \tau^{\mathbf{T}}\delta\mathbf{q} - \mathbf{F}^{\mathbf{T}}\delta\mathbf{x} \tag{14}$$

According to the principal of virtual work, if the system is in equilibrium, then, for virtual displacements satisfying geometric constraints. the virtual work must vanish. If, in equation **??**, $\delta\mathbf{W}$ is set to zero and $\delta\mathbf{x}$ is replaced with $\mathbf{J}\delta\mathbf{q}$ in accordance with equation **??**, the resulting equation is equation **??** [**ASADA˙LECTURE˙NOTES**].

$$\tau = \mathbf{J}^{\mathbf{T}}\mathbf{F} \tag{15}$$

2.4.4 Inverse Kinematics

Inverse kinematics solve for joint angles that achieve a desired end-effector pose. For a planar two-link manipulator with link lengths L_1 and L_2, given a desired end-effector position (x, y), the joint angles θ_1 and θ_2 can be found analytically.

The forward kinematics equations are decribed by the 2-link case of equation **??**.

$$x = L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) \tag{16}$$

$$y = L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \tag{17}$$

From the law of cosines, θ_2 is:

$$\theta_2 = \pm \arccos\left(\frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2}\right) \tag{18}$$

The angle θ_1 is then:

$$\theta_1 = \arctan 2(y, x) - \arctan 2(L_2 \sin(\theta_2), L_1 + L_2 \cos(\theta_2)) \tag{19}$$

The $\pm$ in the equation for θ_2 indicates that two solutions exist for most end-effector positions—one with the second joint angle positive and one negative. When $L_1 = L_2$, a singularity occurs at $(0, 0)$ where infinite solutions exist. At the workspace boundary where $x^2 + y^2 = (L_1 + L_2)^2$, only one solution exists.

2.5 Contact Friction

TODO

2.6 Numerical Solvers

Although a detailed review of numerical solvers and the associated theory is beyond the scope of this theory section, it is worth mentioning that numerical simulation of contact dynamics is particularly challenging. This is due to the discontinuities and high-frequency events that occur during contact. These dynamics often require the use of stiff solvers to accurately capture the rapid changes in forces and velocities [**stiff˙contact˙ODE˙1**][**stiff˙contact˙ODE˙2**]. Stiff solvers are designed to handle problems with widely varying timescales, ensuring stability and accuracy in the simulation of contact events. Without the use of stiff solvers, simulations can become unstable or fail to converge, leading to inaccurate results. Examples of stiff solvers include the well known ode15s and ode23s solvers in MATLAB, which are specifically designed to handle stiff ordinary differential equations [**MATLAB˙ODE**].

2.7 Linear Least Squares Regression

Linear least squares regression is a method for finding the best-fitting line through a set of points by minimizing the sum of squared residuals. Given a set of observations (x_i, y_i) and a linear model $y = X\beta$, where X is the matrix of input variables and β contains the model parameters, the residual r_i for each observation is:

$$r_i = y_i - X_i\beta \tag{20}$$

The sum of squared residuals S is then:

$$S = \sum_{i=1}^{n} r_i^2 = (y - X\beta)^T(y - X\beta) \tag{21}$$

To minimize S, we take its derivative with respect to β and set it to zero:

$$\frac{\partial S}{\partial \beta} = -2X^T(y - X\beta) = 0 \tag{22}$$

Solving for β yields the normal equations:

$$X^T X\beta = X^T y \tag{23}$$

The solution is therefore:

$$\beta = (X^T X)^{-1} X^T y \tag{24}$$

This solution minimizes the sum of squared residuals and provides the optimal parameters β in the least squares sense.

2.8 Mean Squared Error

Mean Squared Error (MSE) measures the average squared difference between predicted values and actual values. For a set of n predictions $\hat{y}_i$ and corresponding true values y_i, MSE is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{25}$$

MSE penalizes larger errors more heavily due to the squared term, making it useful for evaluating model fit quality. A lower MSE indicates better model performance.

2.9 Moving Average Filter

A moving average filter smooths data by replacing each point with the average of a window of neighboring points. For a window size w, the filtered value y_i at index i is:

$$y_i = \frac{1}{w}\sum_{j=i-\lfloor w/2\rfloor}^{i+\lfloor w/2\rfloor} x_j \tag{26}$$

where x_j are the input values. The filter reduces high-frequency noise while preserving lower-frequency trends in the data. Larger window sizes provide more smoothing but can attenuate rapid changes in the signal.

2.10 Centered Finite Differences

Centered finite differences approximate derivatives using symmetric sampling around each point. For time series data with constant time step Δt, the first and second derivatives at index i are approximated as:

$$\dot{x}_i = \frac{x_{i+1} - x_{i-1}}{2\Delta t} \tag{27}$$

$$\ddot{x}_i = \frac{x_{i+1} - 2x_i + x_{i-1}}{(\Delta t)^2} \tag{28}$$

2.11 Inertia

The moment of inertia, I, quantifies the rotational inertia of an object, indicating how much torque is required for a desired angular acceleration.

2.11.1 Parallel Axis Theorem

The Parallel Axis Theorem allows the calculation of the moment of inertia of a body about any axis, given its moment of inertia about a parallel axis through its center of mass. It is mathematically expressed as:

$$I = I_{\text{cm}} + md^2 \tag{29}$$

where:

- I is the moment of inertia about the chosen axis,
- I_{cm} is the moment of inertia about the center of mass axis,
- m is the mass of the object,
- d is the distance between the two parallel axes.

2.11.2 Thin Rod

For a thin rod of length l and mass m, rotating about an axis perpendicular to the rod and passing through its end, the moment of inertia is derived using the Parallel Axis Theorem. The moment of inertia about the center of mass is:

$$I_{\text{cm, rod}} = \frac{1}{12} m l^2 \tag{30}$$

Applying the Parallel Axis Theorem with $d = \frac{l}{2}$:

$$I_{\text{rod}} = I_{\text{cm, rod}} + m \left(\frac{l}{2}\right)^2 = \frac{1}{12} m l^2 + \frac{1}{4} m l^2 = \frac{1}{3} m l^2 \tag{31}$$

2.11.3 Disk

For a solid disk of radius r and mass m, the moment of inertia about its center is:

$$I_{\text{cm, disk}} = \frac{1}{2} m r^2 \tag{32}$$

Using the Parallel Axis Theorem to find the moment of inertia about an axis parallel and a distance d away:

$$I_{\text{disk}} = I_{\text{cm, disk}} + m d^2 = \frac{1}{2} m r^2 + m d^2 \tag{33}$$

3 Modeling and Simulation

For the purpose of doing design verification and optimization, a simplified model of the robot was created. The model was created in Simscape, a physical modeling toolbox integrated with MATLAB/Simulink.

3.1 Simscape

Simscape is a simulation tool that allows you to rapidly create models of physical systems within Mathworks' MATLAB/Simulink environment. With Simscape, physical systems are built by interconnecting blocks representing physical components, such as rigid bodies, joints and springs in a block diagram. The blocks are parameterized by physical properties, such as mass, inertia, and damping. Simscape automatically generates the equations of motion for the system, which can be solved numerically to simulate the system's behavior. Like you can do with Simulink without Simscape, you can also add ordinary Simulink blocks, including Matlab Function blocks, to the model. Simscape is also compatible with Simulink's multiple numerical solvers, such as ode15s, ode45, and ode23s.

An example of a typical SimScape block diagram can be found in figure **??**. A visualization of the corresponding model can be seen in figure **??**. As one can see, each element in a block diagram is typically either a rigid body, or joints connecting the various rigid bodies. Since a given body has multiple possible locations that a joint could be connected to, as well as axes it can act on, blocks can export different frames, with different origins and orientations, depending on the desired position and orientation of the joint. For example, for a block representing the robotic equivalent of a thigh, natural output frames would be the ones with origins at the top and bottom of the thigh, with a select axis aligned with the desired knee or hip axis of rotation. TODO: We want

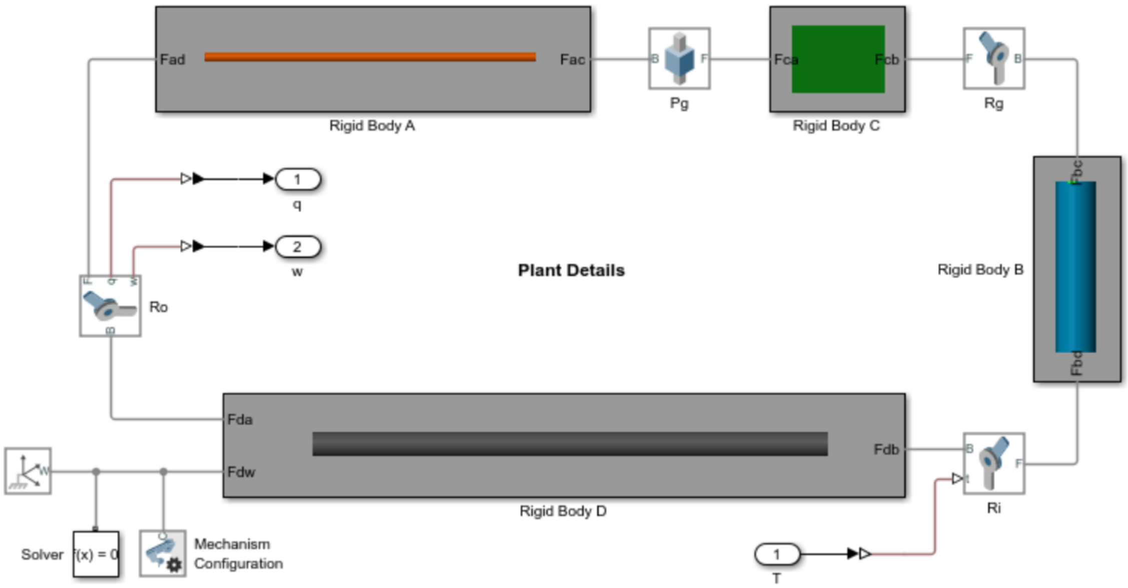


Figure 3: A typical Simscape block diagram.

to use something other than the tutorial here, but our robot model is too split into submodels of submodels, so it loses the intuition you get from a "flatter" model. Will fix later.

3.2 Rigid Body Components of the Robot Model

Since the purpose of the Simscape model is not to facilitate the development of a complicated full degree of freedom feedback controller, nor to optimize every small detail of the design, a simplified model was selected. This model consists of a main body with four legs, each of which with two degrees of freedom. A visualization of the model, as well as an overview of the body's naming conventions can be found in figure **??**. An overview of the body's angle conventions can be found in figure **??**. Note the absence of a hip abduction/adduction joint. This is because the model's main purpose is to verify the design for jumping in the sagittal (forward-backward and upwards-downwards) plane, and the hip abduction/adduction joint is not necessary for this purpose.

Regarding the naming conventions presented in figure **??**, note especially the naming of the different legs corresponding to location on the body, namely RH (Right Hind), RF (Right Front), LH (Left Hind), and LF (Left Front). Note also the naming of the joints hip (HIP) and knee (KNEE). If you see the angle conventions in figure **??**, you can see that the angles of these joints correspond to the angles θ_1 and θ_2 respectively. Note that an orientation of zero degrees for the hip joint corresponds to the leg pointing straight downwards, and an orientation of zero degrees for the knee joint corresponds to the shank pointing in the same direction as the thigh. Note how each joint has its own coordinate system, with rotation being defined positive around the joints z axis. This is required for all Simscape Revolute joints. Note how the joints' z axes correspond to body y axis. This is the case for all the robots legs, both the right and the left side legs.

As can be seen in both figure **??** and figure **??**, in addition to the main body and legs colored in grey, the robot model also contains large purple blocks. These blocks represent motor masses, and their mass can be adjusted to represent different motors. In the current figures, the hip motors are A06CLS V2 motors, and the knee motors are A80BHP-H motors.

3.3 Rigid Body Masses and Inertias

In Simscape, the mass and inertia properties of a rigid body can be specified by the user or automatically calculated based on the body's geometry and material properties. Hybrid solu-

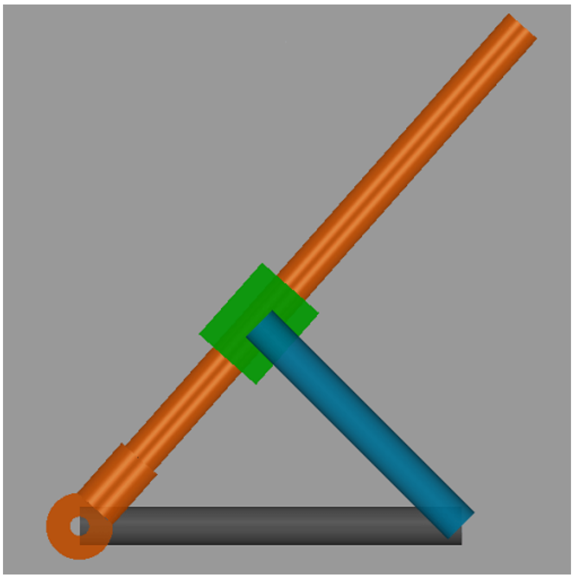


Figure 4: A visualization of the model in figure ??.

Component	Dimensions (cm)	Density (kg/m^3)	Mass (g)
Thigh	L1 x 0.5 x 0.25	2700	Calculated
Shank	L2 x 0.5 x 0.25	2700	Calculated
Hip Motor (A06CLS V2)	2 x 0.8 x 1.78	Calculated	7g
Knee Motor (A20BHM)	2.3 x 1.2 x 2.75	Calculated	20g
Knee Motor (A80BHP-H)	4.0 x 2.0 x 3.75	Calculated	79g
Thigh Spine	2 x 0.5 x 0.25	2700	Calculated
Shank Spine	1 x 0.5 x 0.25	2700	Calculated

Table 2: Dimensions and density of the rigid bodies in the robot model. L1 and L2 have no set size, but are the variables we intend to optimize over.

tions are also possible, where the user specifies some properties and Simscape calculates the rest [**simscape˙tutorial**].

In the case of this model, a summary of the origin of the mass and inertia properties of the rigid bodies can be found in table **??**. Exceptions are the properties of the main body and the paws, which will be specified in more detail in the two next paragraphs. For the parts whose mass and inertia are calculated by the geometry, a summary can be found in table **??**.

The mass properties of the main body are based on the hardware components used by the Eurepus robot constructed by Maurer and El Agroudi [**finn˙tarek˙master**]. Since early in the design process a rough estimate was needed for the robot body mass, and it seemed likely that the electronic solution, apart from the motors, would be similar to the Eurepus robot, an approximate mass of the main body was calculated based on the mass of some of the Eurepus robot's electronics plus an approximate amount of Nylon body material, four motors, and a random chosen microcontroller (MCU) mass. The formula used for the approximate of the main body mass can be found in equations **??** to **??**. The masses corresponding to variables in equations **??** to **??** can be found in table **??**. The motor mass chosen in table **??** corresponds to the mass of the AGF-RC A20BHM motor. This motor was chosen because, although the current plan is to use the A06CLS V2 motor, which is lighter, we would rather overestimate the mass of the main body than underestimate it, and a change of motor is one of the more likely changes to the robot design, as well as one of the changes that would increase the mass of the robot the most. TODO: Add method used for the inertia as well.

The geometrical, mass and inertia properties of the paws differ based on which of two scenarios we intend to simulate. The first scenario is the normal jumping scenario, in which the paw mass

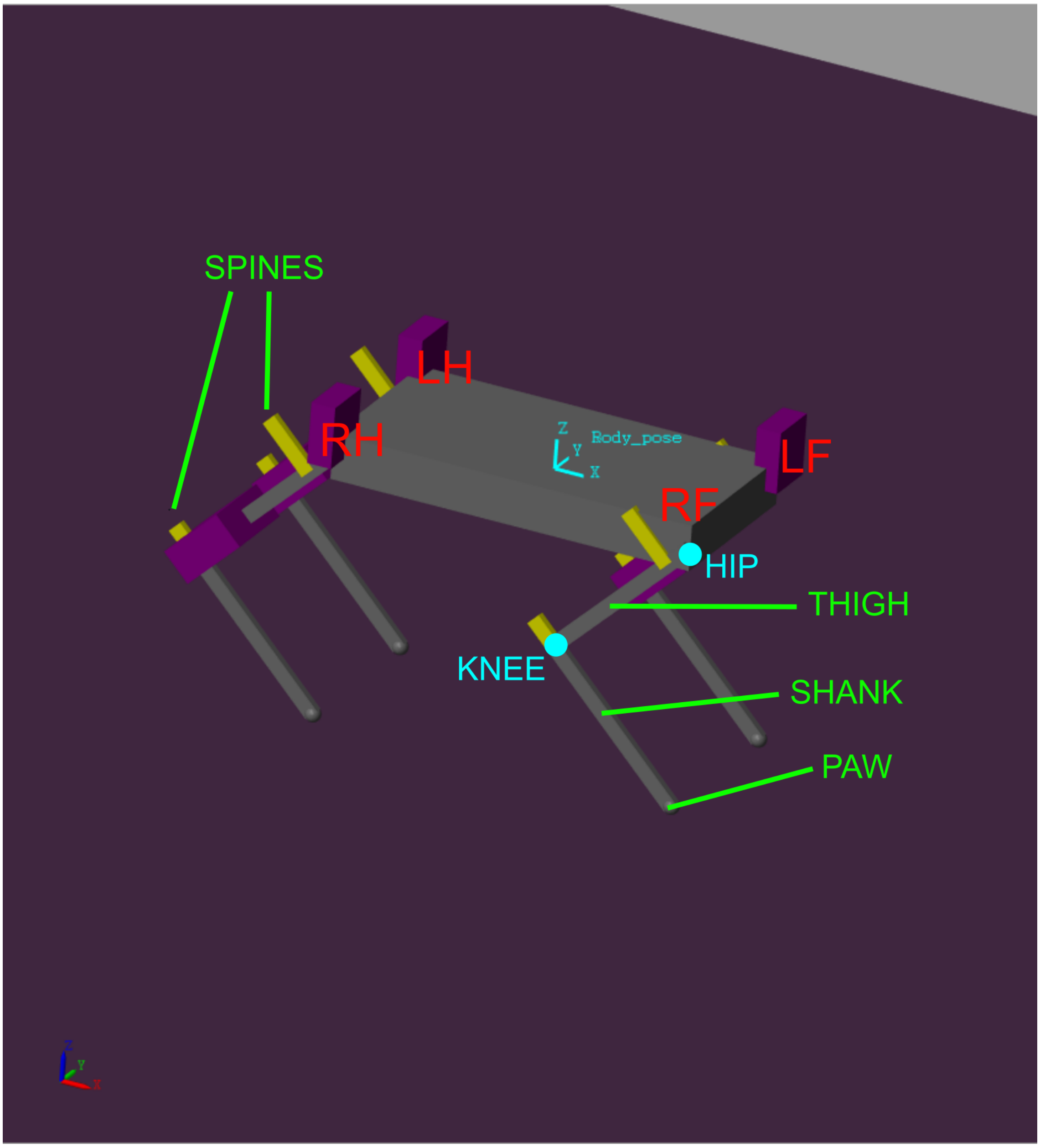


Figure 5: Naming conventions for the parts of the robot.

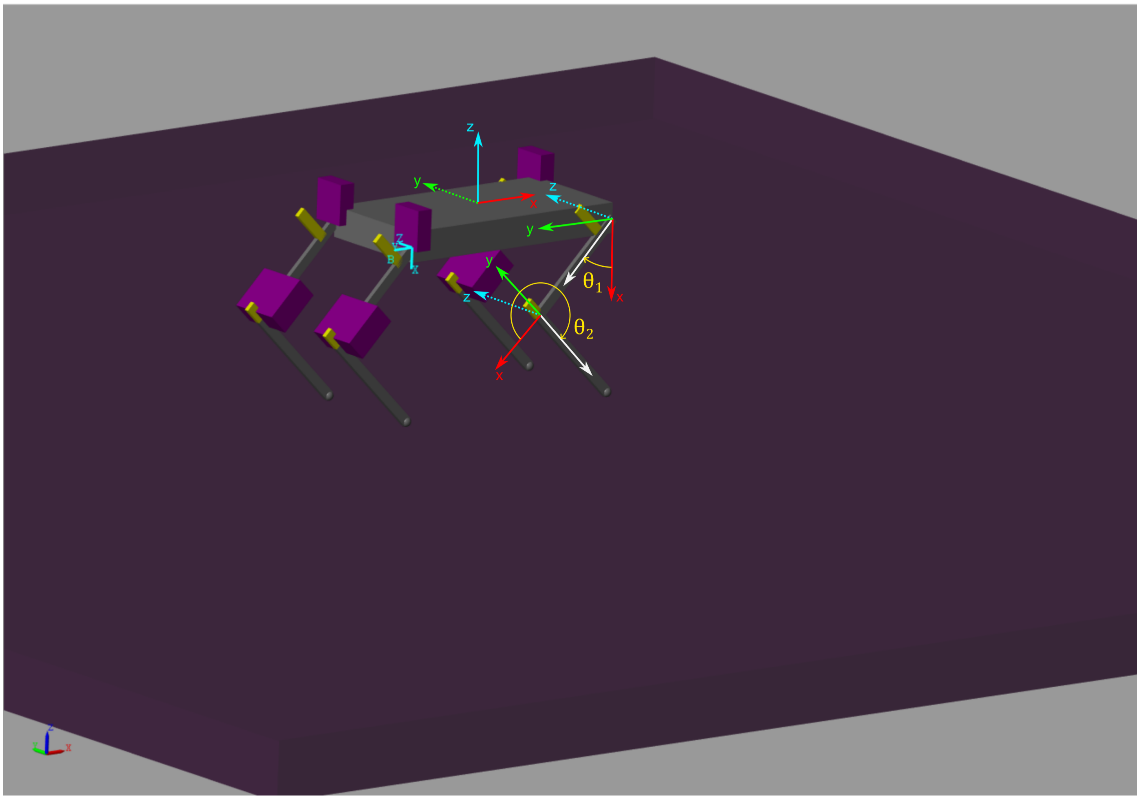


Figure 6: Angle conventions for the robot body.

is simply set to 1000 kg/m^3, though choosing the actual density of rubber would maybe be more appropriate. The dimensions of the paw are in this scenario simply chosen so that the diameter coincides exactly with max(shank width, shank height). The second scenario is the one described in section **??**, where the paw mass and volume is increased to estimate hip flexor/extensor motor requirements.

$$m_{plate} = \rho_{nylon} \cdot V_{plate} = \rho_{nylon} \cdot l_{plate} \cdot w_{plate} \cdot h_{plate} \tag{34}$$

$$m_{\text{eurepus\_electronics}} = m_{\text{battery}} + m_{\text{I2C}} + 12 \cdot m_{\text{ADC}} + m_{\text{PWM\_driver}} \tag{35}$$

$$m_{\text{main\_body}} = m_{\text{plate}} + m_{\text{eurepus\_electronics}} + m_{MCU} + 4 \cdot m_{\text{motor}} \tag{36}$$

Component	**Mass**	**Density (kg/m^3)**	**Inertia**	**Geometry**
Main Body	See section **??**	See section **??**	See section **??**	Rectangular Prism
Thigh	From geometry	2700 (Aluminium 6061)	From geometry	Rectangular Prism
Shank	From Geometry	2700 (Aluminium 6061)	From Geometry	Rectangular Prism
Law	From Geometry	2700 (Aluminium 6061)	From Geometry	Rectangular Prism
Hip Motor	Actual motor mass	From Geometry	From Geometry	Rectangular Prism
Knee Motor	Actual motor mass	From Geometry	From Geometry	Rectangular Prism
Thigh Spine	From Geometry	2700 (Aluminium 6061)	From Geometry	Rectangular Prism
Shank Spine	From Geometry	2700 (Aluminium 6061)	From Geometry	Rectangular Prism
Paw	See section **??**	See section **??**	See section **??**	Sphere

Table 3: Mass and inertia properties of the rigid bodies in the robot model. A list of the Eurepus robot's electronics can be found in [**finn˙tarek˙master**].

Variable	Description	Value
ρ_{nylon}	Density of Nylon	1520 kg/m^3
l_{plate}	Length of the plate	10 cm
w_{plate}	Width of the plate	6 cm
h_{plate}	Height of the plate	1.67 cm
m_{battery}	Mass of the battery	27 g
m_{motor}	Mass of one motor	20 g
m_{I2C}	Mass of the Inter-Integrated Circuit (I2C) module	5.1 g
m_{ADC}	Mass of one Analog-to-Digital Converter (ADC) module	2.4 g
$m_{\text{PWM\_driver}}$	Mass of the Pulse Width Modulation (PWM) driver	8.5 g
m_{MCU}	Approximate mass of some microcontroller	30 g
$m_{\text{main\_body}}$	Resultant mass of the main body	332 g

Table 4: Masses and dimensions used in the main body mass calculation.

3.4 Elastic Components: Springs

In addition to the model's many rigid bodies, we also implemented two different forms of spring based passive actuation, namely:

- **A torsional spring** acting in parallel with the knee joint, as illustrated in figure **??**. This spring is at zero extension when the knee joint is at zero degrees, and applies a torque that is proportional to the knee joint angle, as covered in section **??**.
- **An extension spring** acting in parallel with the knee joint, attached to the shank and thigh spine, as illustrated in figure **??**. The force generated by the extension spring is proportional to its displacement, as covered in section **??**. The spring is intended to be unloaded when the knee joint is at zero degrees.

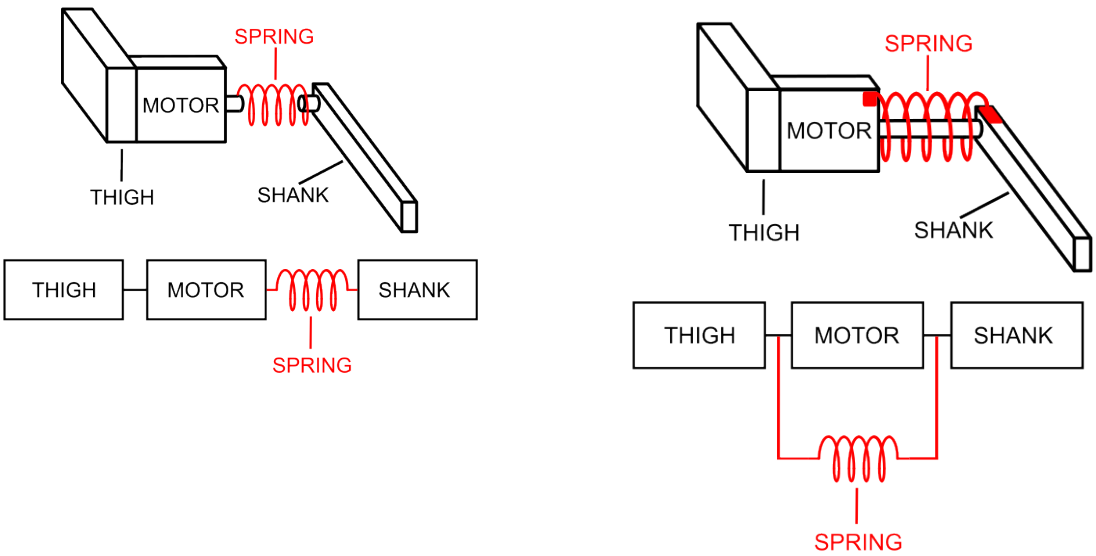


Figure 7: A figure displaying the conceptual difference between a parallel spring (right), as used in this work, and a series elastic spring (left) as used in the ANYmal robot [**hutter˙anymal˙2016**].

The torsion spring was implemented in the model using Simscape's prismatic joint option to add spring stiffness. The extension spring, on the other hand, was connected in parallel to the joint using Simscape's natural block diagram functionality, as seen in figure **??**.

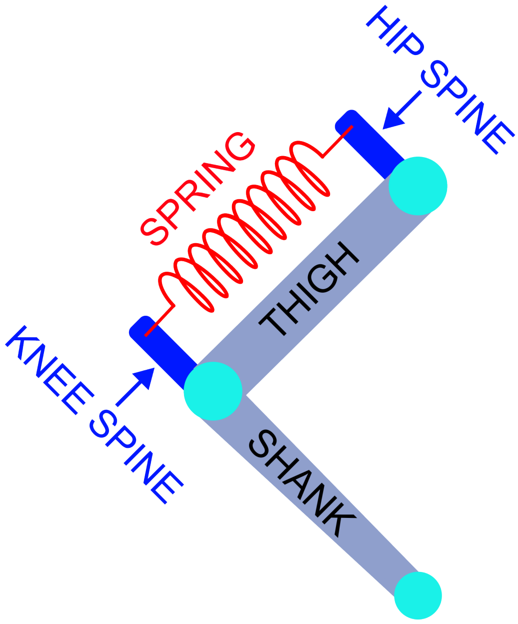


Figure 8: A figure displaying the concept of a parallel extension spring, note that the spring generates torque directly in parallel with the knee joint just like for the parallel torsional spring seen in figure ??. Mathematically, the main difference is the non-linear relationship between knee angle and spring displacement, giving a non-linear torque-angle relationship.

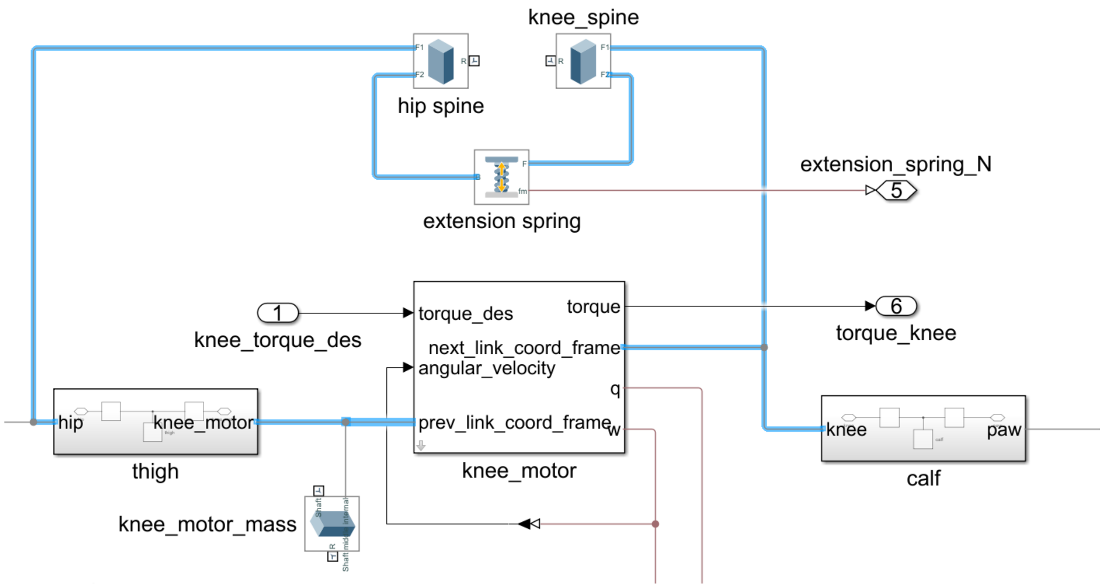


Figure 9: Extension spring implementation in Simscape. As can be seen, the spring is connected between the appropriate points (output frames) of the thigh and shank spine, and in parallel to the knee motor model, which contains the knee joint.

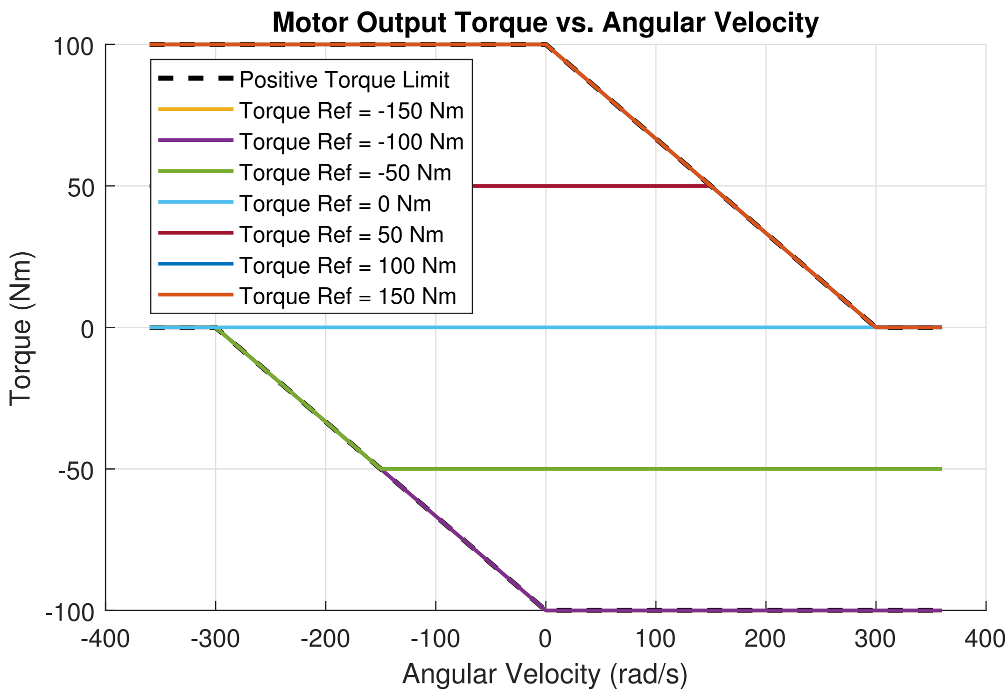


Figure 10: Torque-speed characteristics for a given motor using our implemented motor model.

3.5 Motor Modeling

To be able to more accurately judge the jumping capability of different motors, a motor model in the form of a torque speed curve was implemented. The Brushless DC (BLDC) torque-speed curve presented in figure ?? is characterized by four parameters, namely the stall torque, the operating torque, the rated speed, and the maximum speed. Since motor suppliers contacted (primarily agf-rc and T-motor) were unable to provide most of the desired parameters, we chose a motor model with only two parameters, namely stall torque and maximum speed. The torque-speed model thus became a simple model where torque decreases linearly from stall torque, at speeds smaller than or equal to zero, to zero at speeds greater than or equal to maximum speed. The model is identical for negative velocities and torques, but with opposite signs. An example of the relation between desired torques and achieved torques for a given motor max speed can be seen in figure ??.

3.6 Solver selection

As described in section ??, the potential energy of a loaded spring can be easily calculated. Similarly, the potential energy due to gravity of a robot at the peak of a jump can also be determined. For the reasons discussed in section ?? we chose a stiff numerical solver, initially, we chose the ode15s solver. However, it was eventually observed that simulations using ode15s occasionally resulted in jump trajectories where the gravitational potential energy at the robot's peak height exceeded the combined potential energy of the four fully loaded springs by a factor of 2, even for jumps with only passive (spring) actuation. To address this inaccuracy, we experimented with different numerical solvers and ultimately selected the ode23s solver. This solver provided accurate simulations without artificially generating excess energy, and it performed efficiently.

4 Robot Design

4.1 The Single Vertical Manipulator Jump Model

In the absence of air resistance, which is negligible for a robot like this during a jumping maneuver like this, the factor that determines jumping distance is body velocity at takeoff. To keep things simple, in the following will be given an example where the sole goal is to maximize vertical jump height. Further, only a one leg robot with a 2 Degrees of Freedom (DOF) leg with equal link lengths is considered.

For a robot such as the one described above, the position of the paw relative to the body can be described by the standard 2 link manipulator equation as seen in equation **??**, in accordance with the theory in section **??**. For this simple robot, it is assumed that the optimal jump is one where the body center of mass and paw position move in opposite directions, with velocities strictly along the up/down y axis. This example is illustrated in figure **??**. As is obvious from the figure and from the kinematics, for such a jump $\theta_1 = -\frac{(\theta_2+\pi)}{2}$, and $\dot{\theta}_2 = -2\dot{\theta}_1$. If this is combined with the jacobian for the end effector, given in equation **??**, one can plot the vertical velocity of the paw as a function of the knee angle θ_2, this is shown in figure **??**. This is done using the jacobian as in equation **??**.

As can be seen in figure **??**, the joint velocity of the leg much more readily translates to body velocity when the knee is crouched. Without giving a detailed derivation, the intuition meant to be gained from figure **??** is that how quickly the leg joints are accelerated can be just as important as the speed it is accelerated to. This has an important interpretation for the choice of motors, in the sense that, if a given motor is unable to accelerate the leg joints quickly enough, there is little sense in looking for a faster motor, unless it is also stronger.

$$\begin{aligned} x_{\text{end}} &= L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) \\ y_{\text{end}} &= L_1 \sin(\theta_1) + L_2 \sin(\theta_1 + \theta_2) \end{aligned} \tag{37}$$

$$J = \begin{bmatrix} -L_1 \sin(\theta_1) - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

4.2 Actuation Method Selection: Motors Only

Initially, experiments were done utilizing motors alone. Due to the need for a light and strong motor, initially a series of AGF-RC motors were looked at, due to their high torque to weight ratio. As covered in section **??**, a motor model consisting of a stall torque and a max velocity, with a linear decrease in max torque between these two, was chosen. For the AGF-RC motors, their provided stall torque and operating speed were used for the motor model parameters. Attempts were made to get more information about the motors' performance characteristics, but the supplier, AGF-RC, was unable to provide more detailed information. While extensive testing was done with various motors, the jumping performance was so far from acceptable that the idea was abandoned, and the full scope of the experiments are considered outside the scope of this report.

More details about the results of the motor only jumps can be found in section **??**, but a brief overview of the reasoning involved will be given here. The most relevant of the motors chosen for these experiments was the A80BHP-H motor by AGF-RC, which was chosen because it had both the highest stall torque and operating speed of all the AGF-RC motors. Info about the A80BHP-H motor can be found in appendix **??**. Attempts were made to identify motors in the same weight range with similar stall torques and operating speeds, but none were found. The experiments using the A80BHP-H were therefore, to a certain degree, considered an "optimistic estimate". Since no satisfactory results were achieved with the A80BHP-H motor, as discussed in section **??**, it was decided to explore alternatives to motor-only actuation.

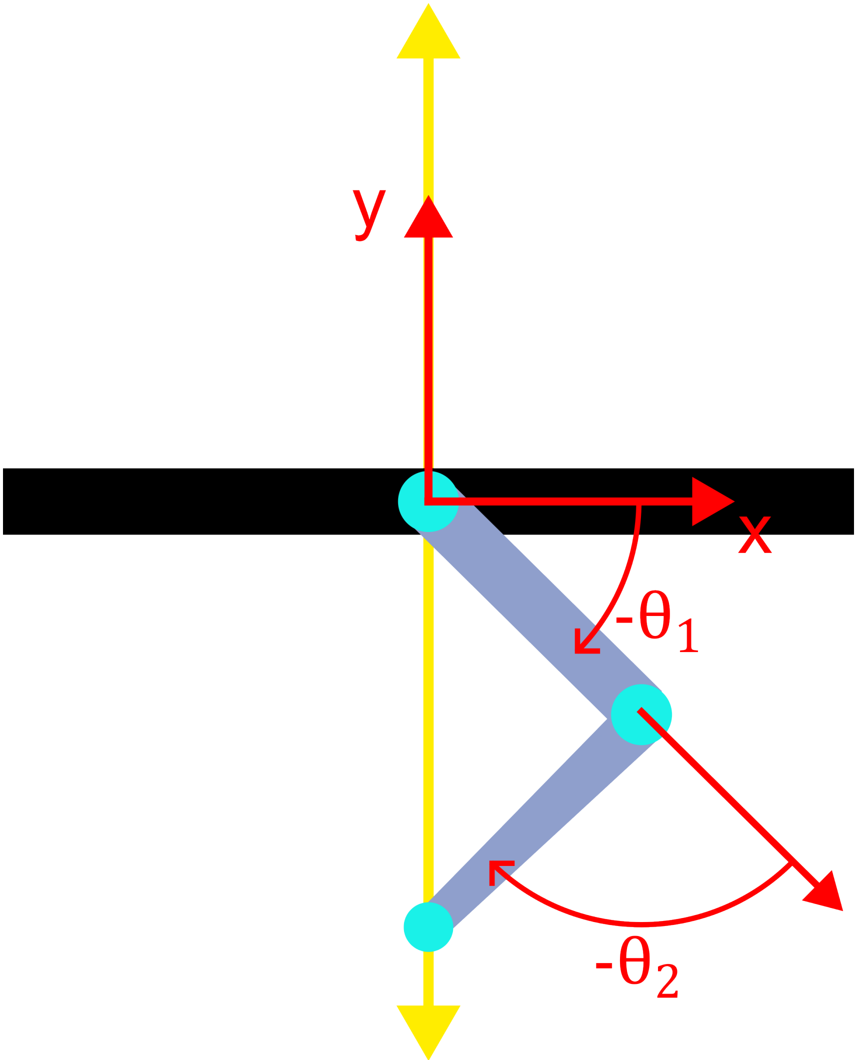


Figure 11: The manipulator corresponding to a vertical one leg jump.

The tests with the A80BHP-H motor were done by providing the knee motors with reference torques equal to their max torque, with the motor model limiting the resultant torque to more realistic values. Meanwhile, the hip joints were actuated according to the control law seen in equation **??** to limit slipping. This control law is based on equation **??**, as derived in section **??**, combined with the theory on friction presented in section **??**. Equation **??** is derived with an assumption of zero velocity/equilibria (TODO), an assumption that is obviously not valid for this dynamic jumping scenario. Despite this, very little slipping is observed in practice.

$$N = \text{Normal force} \tag{38}$$
$$\mu = \text{friction coefficient} = 0.8 \tag{39}$$
$$\tau_{\text{friction cone limit}} = J^T \begin{bmatrix} N\mu \\ 0 \end{bmatrix} \tag{40}$$
$$\max(|\tau_{knee}|) = \tau_{\text{friction cone limit}} \tag{41}$$

Using this control law, jumping for variously dimensioned robots were attempted. As covered in section **??**, none of them were satisfactory. Tests were eventually terminated, as no leg and body link length configuration was capable of jumping satisfactorily. By satisfactorily, it is meant that the robot center of mass cleared the ground by a distance more than twice that of the leg length, ie., if the legs are 20cm long, the center of mass should clear the ground by at least 40cm.

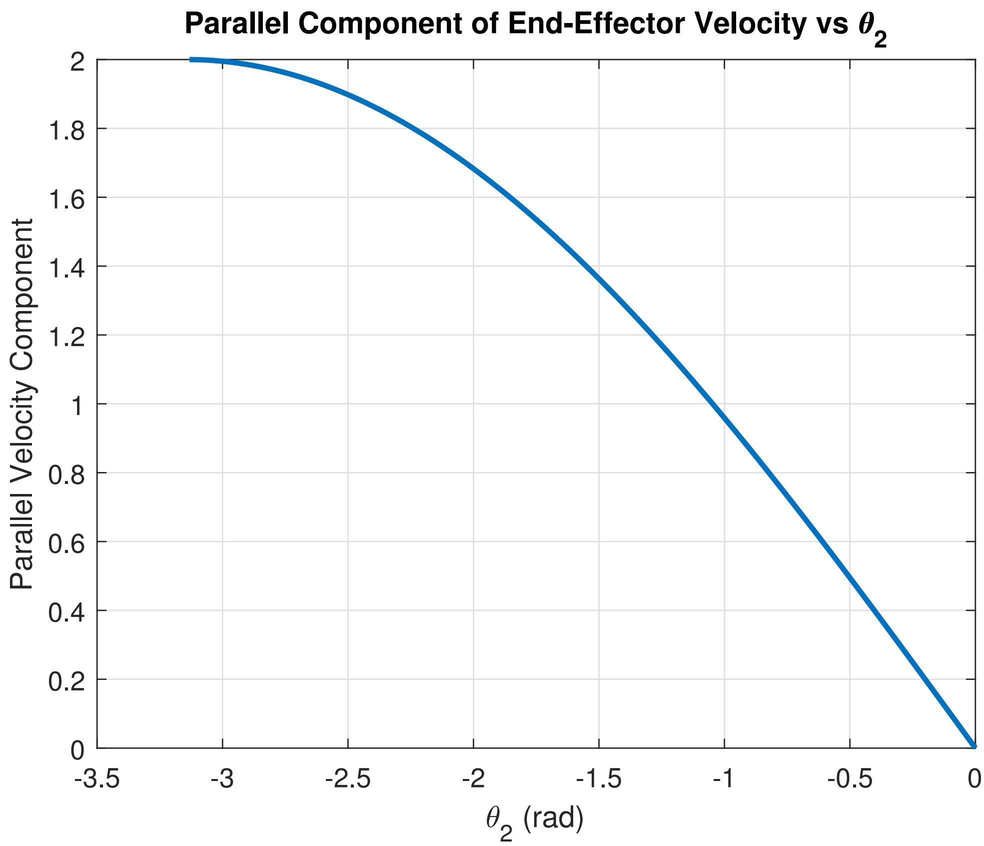


Figure 12: Vertical Paw velocity as a function of knee angle.

For comparisons between this motor-only jumping and later spring+motor jumping, it is worth mentioning that the spring-motor jumping is done not with the A80BHP-H motors, but with weaker and slower motors, as well as with more realistic motor friction models. The reason the A80BHP-H motors were not used for the spring-motor jumping simulations, is that the motor supplier's website (AGF-RC) stated an operating travel range of 90 degrees. This was assumed to be correct, as other AGF-RC motors with stated operating ranges of 180 degrees have had actual mechanical limits of 220 degrees, so such a limitation was not considered out of the ordinary. For that reason, the motors initially purchased were not the A80BHP-H motors, and they are therefore not used for the rest of the experiments in this report. Later consultations with the supplier's website revealed conflicting information regarding the operating range of the A80BHP-H motor, and direct contact with the supplier revealed that the motor is in fact not limited to 90 degrees of travel. For this reason, potential use of the A80BHP-H motor for the spring-motor jumping is discussed in the future work section, section **??**.

4.3 Actuation Method Selection: Motors and Torsional Spring

The next actuation method considered was to use a combination of motors and torsional springs, as this is the design we ended up choosing, a CAD model of this design can be seen in figure **??**. With this design, the knee motors are used to compress the torsional springs, once the knee joint reaches a desired angle, ie. has stored the desired amount of energy, the motors are turned off, and the springs accelerate the legs' joints quickly enough for the robot to take off. Results from simulations are presented in section **??**, but in short, jumping performance is superior to that of the motor-only jumps. For the spring-motor jumps, a friction model as derived in section **??** was used. Again, note the much weaker motor used in

4.4 Actuation Method Selection: Motors and Extension Spring

In addition to the torsional spring design, attempts were made to design a leg utilizing an extension spring. The resulting design is shown in figure **??**. Although experiments akin to the ones done for torsional springs were done with comparable results, the extension spring design was ultimately abandoned. The reason for this was pure geometrical constraints, which are discussed in section **??**. Because the extension spring design was abandoned, the details surrounding the extension spring simulations and simulation results are considered outside the scope of this report.

4.5 Hip Motor Strength Requirements

While the main purpose of the designed robot is to be proficient at jumping, the inclusion of torsional knee springs as a major part of the actuation method, and the subsequent lessened importance of the hip motors for jumping, complicated the choice of hip motors. For a conservative bound on on the torque and speed required for in air attitude stabilization [**finn˙tarek˙master**] uses a heuristic of three 90-degree back-and-forth lateral swings per second as a benchmark. Although the purpose of our robot is not to do in-air attitude stabilization, this heuristic was partially utilized for verification that the hip motors were strong enough. More specifically, the robot body was fixed in space, and the robot's paws were set to be 1cm diameter spheres of iron (ie. with a density of $7800\frac{kg}{cm^3}$, for a total mass of ≈32 g). The hip motor was then commanded to follow a trajectory according to the heuristic specified in [**finn˙tarek˙master**]. The results of this simulation are presented in section **??**. While the paw masses used in [**finn˙tarek˙master**] were 80g, the Eurepus robot's main body mass was also closer to 800g, compared to our body mass of approximately 300g. Our experimental paw masses of 32g were therefore considered appropriate, considering attitude stabilization is not strictly speaking a goal for the robot.

5 Motor Friction Estimation

This section details the estimation of parameters for the motor friction models of the hip and knee motors. The motors serve as pivot points for pendulums consisting of aluminum rods attached to the motor shafts with ballast masses. The pendulums are released from a horizontal position, and their angular velocity and acceleration are measured. These measurements facilitate the estimation of motor friction coefficients using linear regression. The experimental setup for both motors is depicted in Figure **??**.

We adopt a linear friction model as described in Section **??**, incorporating both viscous and Coulomb friction, to accurately fit the observed data.

5.1 Pendulum Modeling

The pendulum used in the motor friction tests consists of an aluminum rod of length $l_{\text{arm}} = 0.21$ meters and a ballast mass $m_{\text{ballast}} = 0.301$ kg with radius $r_{\text{ballast}} = 0.03$ meters attached at a distance $d_{\text{ballast}} = 0.20$ meters from the pivot. The total mass of the arm is $m_{\text{arm}} = 0.034$ kg. The pendulum is modeled as a rigid body rotating about the motor shaft with a total moment of inertia I composed of the arm inertia I_{arm} and ballast inertia I_{ballast}. The arm is approximated as a thin rod, and the ballast as a disk, both with uniform density. Their inertias are calculated using the parallel axis theorem:

$$I_{\text{arm}} = \frac{1}{12} m_{\text{arm}} l_{\text{arm}}^2 + m_{\text{arm}} \left(\frac{l_{\text{arm}}}{2} \right)^2$$

$$I_{\text{ballast}} = \frac{1}{2} m_{\text{ballast}} r_{\text{ballast}}^2 + m_{\text{ballast}} d_{\text{ballast}}^2$$

$$I = I_{\text{arm}} + I_{\text{ballast}}$$

The torque due to gravity consists of contributions from both the arm and ballast mass:

$$\tau_{\text{gravity,arm}} = m_{\text{arm}} \frac{l_{\text{arm}}}{2} g \sin(\theta)$$

$$\tau_{\text{gravity,ballast}} = m_{\text{ballast}} d_{\text{ballast}} g \sin(\theta)$$

$$\tau_{\text{gravity}} = \tau_{\text{gravity,arm}} + \tau_{\text{gravity,ballast}}$$

The equations of motion, accounting for both viscous and Coulomb friction, are:

$$I\ddot{\theta} + b_v\dot{\theta} + b_c \operatorname{sign}(\dot{\theta}) + \tau_{\text{gravity}} = 0$$

where:

- θ is the angular displacement (positive counterclockwise, zero at vertical down position)
- $\dot{\theta}$ and $\ddot{\theta}$ are the angular velocity and acceleration, respectively
- b_v is the viscous damping coefficient
- b_c is the Coulomb damping coefficient
- $g = 9.81\,\text{m/s}^2$ is the acceleration due to gravity

5.2 Data Acquisition

Pendulum angles θ were sampled by manually annotating video frames at 60 Hz using the Tracker program [**tracker**].

The angular velocity $\dot{\theta}$ and acceleration $\ddot{\theta}$ are computed using centered finite differences:

$$\dot{\theta}_i = \frac{\theta_{i+1} - \theta_{i-1}}{2\Delta t}$$

$$\ddot{\theta}_i = \frac{\theta_{i+1} - 2\theta_i + \theta_{i-1}}{(\Delta t)^2}$$

where Δt is the time step between measurements.

The angular position θ was smoothed using a moving average filter with a window size of 5 before computing $\dot{\theta}$. Both $\dot{\theta}$ and $\ddot{\theta}$ were smoothed using a moving average filter with a window size of 3.

5.3 Linear Regression Derivation

Rearranging the equations of motion for linear regression:

$$I\ddot{\theta} + \tau_{\text{gravity}} = -b_v\dot{\theta} - b_c \operatorname{sign}(\dot{\theta})$$

This can be expressed in matrix form:

Figure 13: Experimental setup for motor friction estimation. An aluminum rod with ballast mass attached acts as a pendulum, with the motor shaft as the pivot point.

$$Y = X\beta$$

where:

- $Y = -I\ddot{\theta} - \tau_{\text{gravity}}$,
- $X = \begin{bmatrix} \dot{\theta} & \text{sign}(\dot{\theta}) \end{bmatrix}$,
- $\beta = \begin{bmatrix} b_v \\ b_c \end{bmatrix}$.

The linear least squares solution for β is:

$$\beta = (X^T X)^{-1} X^T Y$$

This yields both the viscous damping coefficient b_v and the Coulomb damping coefficient b_c.

To valiate the model we forward simulate the pendulum motion using the estimated parameters with a runge-kutta solver and compare the results with the measured data. This and the results are shown in section **??**.

6 Link Length Optimization

This section details the optimization of the robot's link lengths L1 and L2 to maximize jumping performance. The optimization ensures the robot can achieve sufficient jumping performance to clearly demonstrate future DRL control policies. A grid search approach using the Simscape robot model from section **??** systematically explores different link length configurations.

6.1 Impact of link lengths

Due to the lack of feedback control during the jumping maneuver, the jumping performance is determined fully by the link lengths and initial pose. By initial pose we mean the angles theta1 and theta2 of the knee and hip joints as the knee motor turns off and lets the loaded knee spring actuate the leg. Link lengths constrain the possible initial poses for any given jumping angle, while the initial pose determines spring compression through the knee angle theta2 and thus available potential spring energy. Additionally, link lengths affect the center of mass trajectory during jumps, influencing how gravitational and ground contact forces impact the robot's movement.

6.2 Problem simplification

First, we define the jump angle θ_J as the angle of the velocity vector of the center of mass of the robot body at the moment the robot leaves the ground. While the robot should be able to jump both vertically and at angles sufficient to overcome obstacles, optimizing for angled jumps presents challenges. We want to directly compare the jumping performance of different link lengths, but it is not obvious how to place the initial pose of the robot to achieve a θ_J across different link lengths.

To simplify the optimization, only the vertical jumping performance was considered, so that different link lengths can be compared directly. Vertical jumps are achieved across link lengths by flipping the front legs such that the legs are symmetric about the vertical axis, as shown in figure **??**. This flipping transforms the asymmetric leg configuration in figure **??** where the front legs point forward into a symmetric configuration where both legs point backward. In this configuration, the movement of the legs during the jump is symmetric and the robot center of mass remains in the horizontal center of the robot, such that any horizontal component of the jump is canceled out. In practice, the robot will use the asymmetric configuration. The symmetric configuration simply provides an approximation for the jumping performance of a given link length configuration.

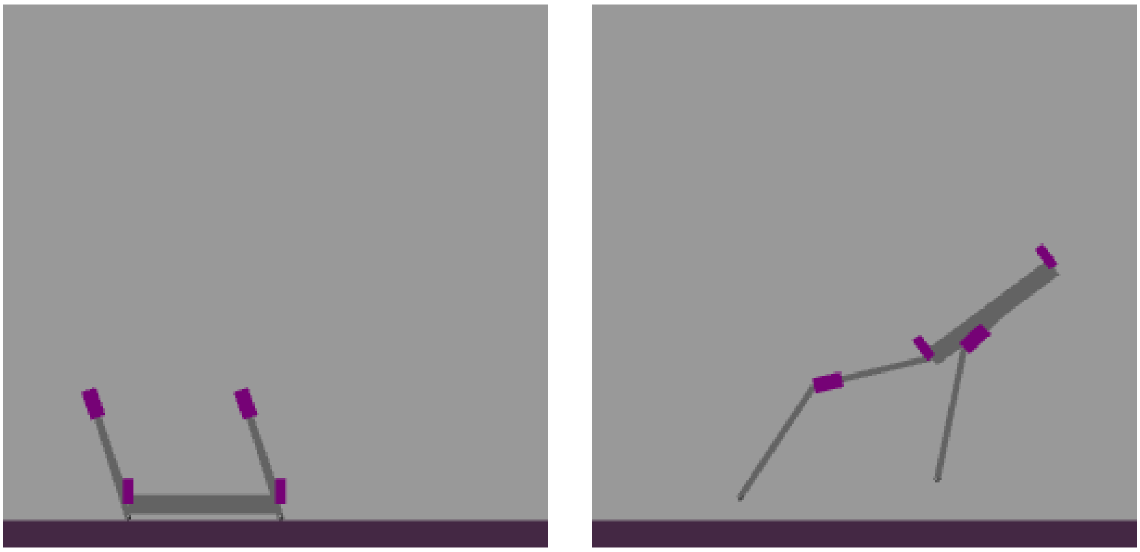
The metric used to evaluate the jumping performance is the maximum height reached by the center of the robot torso, minus the maximum standing height reached by the robot torso when the legs are fully extended vertically downwards and the paws are in contact with the ground.

The asymmetric leg configuration can also achieve vertical jumps by adjusting the angle between the hip-to-paw vector and vertical, as shown in figure **??**. However, finding or calculating the optimal offset for arbitrary link lengths is complex. We therefore simplify by placing paws directly beneath hips, except when $L2 > L1$. While this produces less realistic jump heights, it greatly simplifies the optimization.

For $L2 > L1$ configurations, placing paws directly under hips results in near-vertical legs that slip during jumps. Slipping does not occur to the same extent for asymmetric configurations with the same link lengths where the hip-to-paw angle is adjusted to give a vertical jump. In order to make the symmetric configuration a better approximation of the asymmetric case, we wish to avoid the slipping behavior. We do so by adjusting the hip-to-paw angle such that paws are slightly inward towards the body by a constant offset of 0.3 radians. This increases the vertical component of the paw force and thus the friction force (figure **??**). This exact offset was chosen because experimentation showed that this makes asymmetric configuration jumps generally more vertical across a range of link lengths. This adjusted paw placement better approximates the practical asymmetric case that requires similar adjustment for vertical jumps, although the exact angle offset will vary depending on the link lengths. To handle the increased opposing horizontal forces between paws in the symmetric configuration, we double the friction coefficients from 1.0/0.8 to 2.0/1.6 (static/kinetic), which further reduces slipping.

6.3 Initial Pose Calculation

For each set of link lengths, an initial pose must be calculated that satisfies several constraints:



(a) Initial pose with unadjusted paws

(b) Resulting angled jump

Figure 14: Unadjusted paw placement results in angled jump

- The paws must maintain ground contact
- Knee angle theta2 must be maximized to store maximum spring potential energy
- Knees cannot penetrate the ground
- Both knees bend outward rather than inward

The pose calculation considers three cases based on link length ratios:

1. $L_1 = L_2$ (equal lengths)
2. $L_2 < L_1$ (longer lower link)
3. $L_1 > L_2$ (longer upper link)

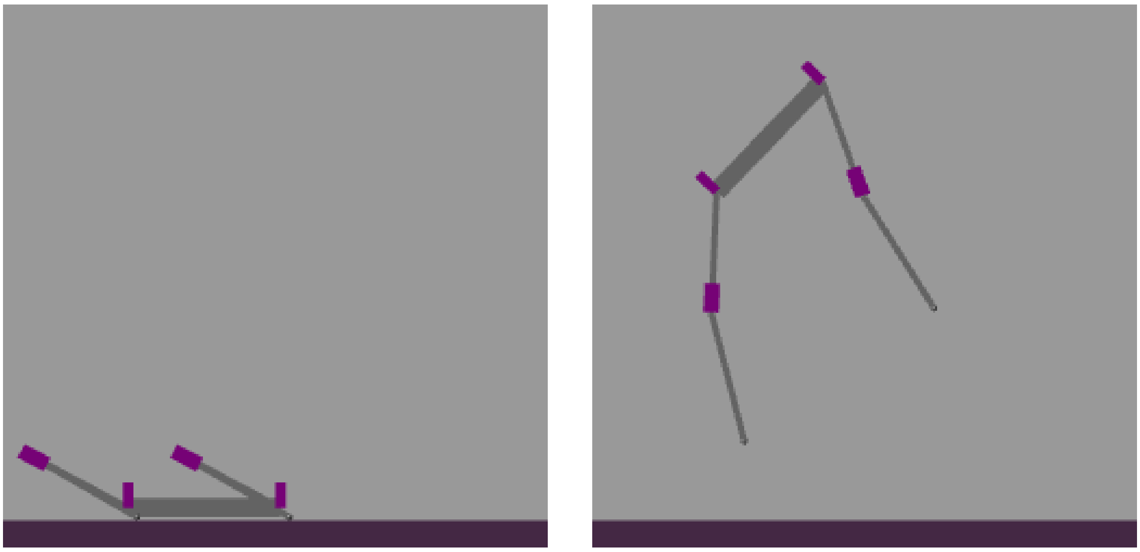
For all cases, we calculate the distance d from hip joint to paw. For cases $L1 = L2$ and $L1 > L2$, where paw is placed directly under hip, d is vertical from the hip. For case $L2 > L1$, where paw is adjusted inwards towards the body, d is along the adjusted hip-to-paw vector.

d is minimized to maximize $theta2$ and thus spring potential energy while satisfying the pose constraints. Inverse kinematics then determines $theta1$ and $theta2$, selecting the solution where knees bend outward. d is calculated as follows:

- For $L_1 = L_2$: $d = \epsilon$, where $\epsilon = 1$mm ensures unique inverse kinematics solutions
- For $L_2 > L_1$: $d = L_2 - L_1 + \epsilon$, where $\epsilon = 1$mm ensures the paw position is reachable given the numerical precision of the inverse kinematics solver
- For $L_1 > L_2$: $d = \sqrt{L_1^2 - L_2^2}$, derived when L_2 is horizontal (maximizing spring load) and forms a right triangle with L_1 and the hip-to-paw vector

6.4 Grid Search

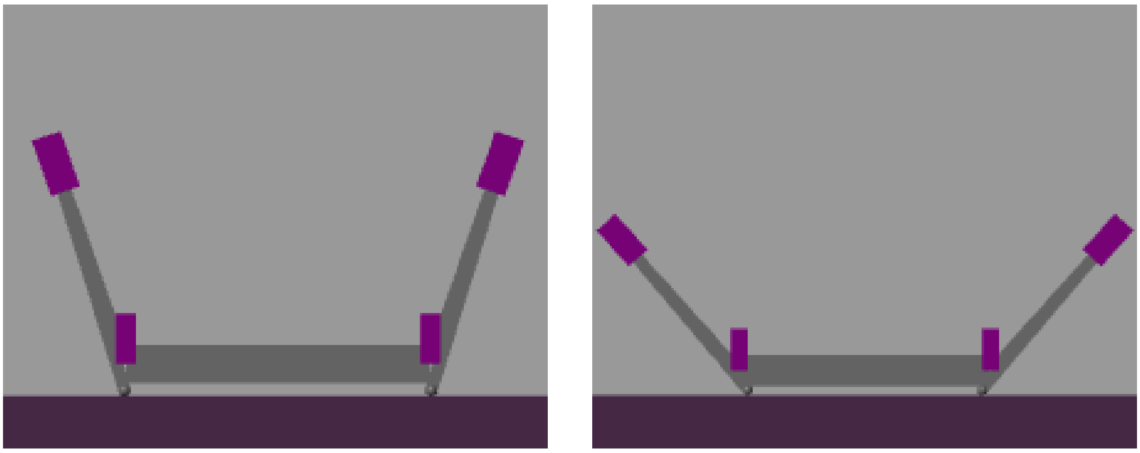
The grid search varies two parameters: L2/L1 ratio, and total leg length L1+L2, focusing around $L2/L1 \approx 1$ where preliminary tests indicated generally better performance. SimScape automatically updates the robot mass for each parameter set. Tests were run in both Earth (9.81 m/s^2) and Mars (3.72 m/s^2) gravity, with results in section **??**.



(a) Initial pose with adjusted paws

(b) Resulting vertical jump

Figure 15: Adjusted paw placement enables vertical jump



(a) Paws directly under hips leading to near-vertical legs

(b) Paws translated inward to make legs more horizontal, which reduces slipping

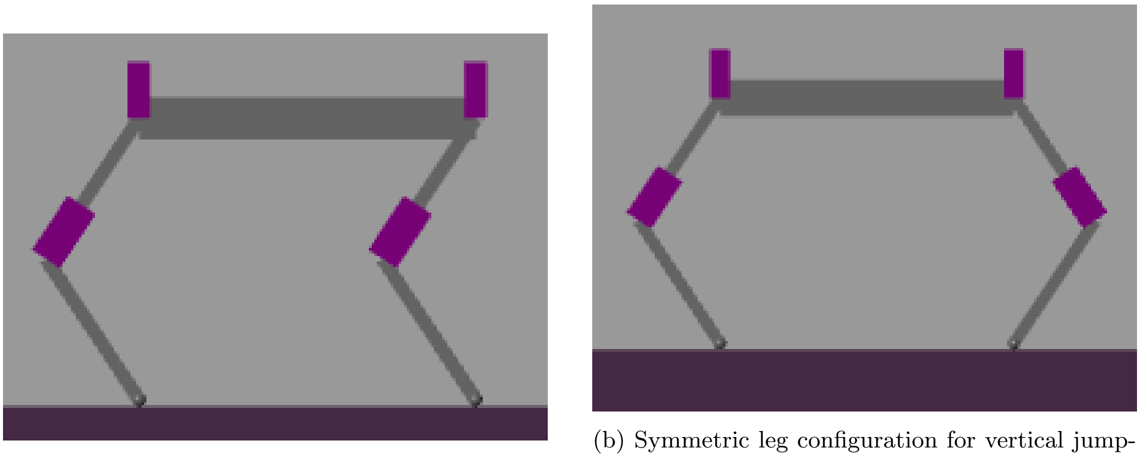
Figure 16: Paw placement adjustment for $L2 > L1$ configurations

7 Robot Hardware

7.1 Torsional Spring Leg

Figure ?? shows the CAD model for the torsional spring leg design, including both knee and hip extension/flexion motor, as well as the hip adduction/abduction motor. The CAD model was created using the CAD software Solidworks. An annotated exploded view of the leg can be seen in figure ??, while an annotated exploded view for the hip motor housing can be seen in figure ??.

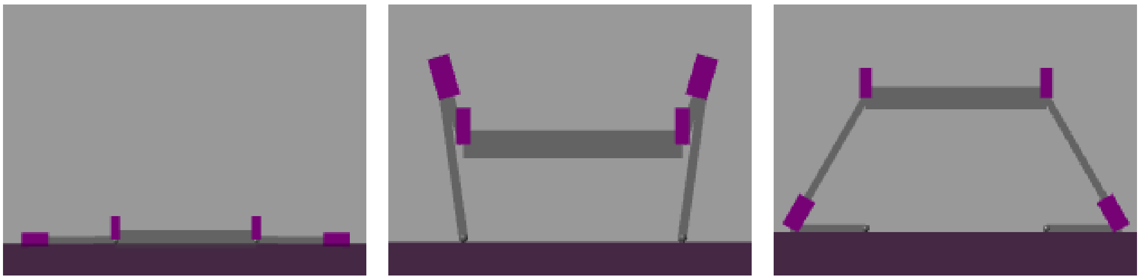
Figure ?? shows the components that are currently planned to manufacture in inhouse. The axle that will be threaded and screwed directly into the motor shaft, and lead directly into a ball bearing, is emphasized in red. The leg has been designed to make it easily manufacturable in aluminum. Aluminum was chosen as a potential material due to its high strength-to-weight ratio, and the fact that it is easy to machine. Although many easily 3D-printable plastics are generally lighter, they are not as strong as aluminum, in fact aluminum has a much higher weight to yield ratio than 3D printable plastics, and thus, barring complications resulting from manufacture, a leg



(a) Normal asymmetric leg configuration

(b) Symmetric leg configuration for vertical jumping

Figure 17: Comparison of normal and symmetric leg configurations



(a) Equal lengths (L1=10cm, L2=10cm)

(b) Longer L2 (L1=15cm, L2=10cm)

(c) Longer L1 (L1=15cm, L2=10cm)

Figure 18: Initial poses for different link length ratios

made in aluminum should be lighter than one made of plastic.

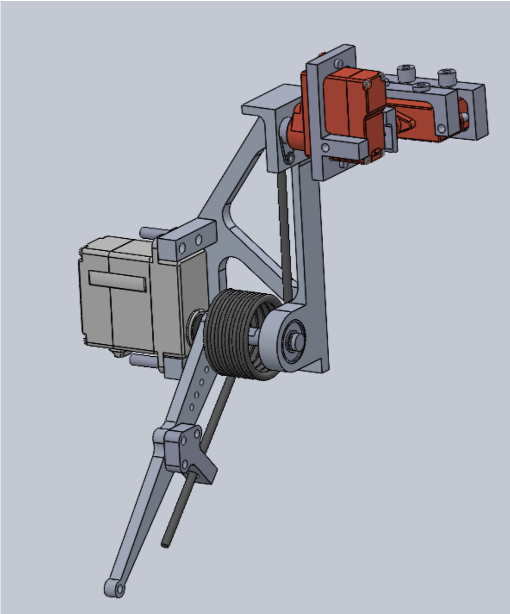
A 3D printed version of the leg thigh together with the purchased spring can be seen in figure **??**. While the full leg has not been assembled, initial physical load tests seem to indicate that the thigh is able to withstand the load of the spring. This could indicate that the leg will be 3D printed rather than made in aluminum. The axle that will be screwed into the motor shaft, however, will still be made in aluminum.

7.2 Extension Spring Leg design

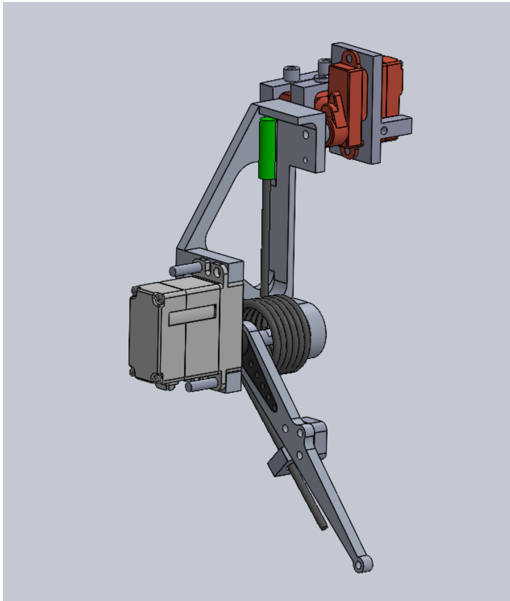
The extension spring leg design is shown in figure **??**. As can be seen in the figure, the current design is such that the extension spring will collide with the robot shank as the knee angle approaches fully coiled (ie. -180 degrees). Despite efforts, no solutions were found for this problem, and this design direction was therefore abandoned. Among the suggested solutions was moving the shank-end attachment point of the spring inwards towards the body, so the extension spring could lie in parallel next to the shank when the leg is fully coiled. This would however introduce a significant moment arm acting directly on the motor shaft, and this design was therefore abandoned in favor of the torsional spring design.

7.3 Motor Selection

As discussed in section **??**, the choice fell on AGF-RC motors due to their high torque to weight ratio, as well as our inability to find similarly fast motors of similar strength.



(a) Notice the bearing holding the knee-joint shaft in place. This is important to reduce the load the motor shaft suffers in directions other than the load direction.



(b) As can be seen, there is a green plastic (PLA) holster where the spring is in contact with the leg, this is to reduce friction.

Figure 19: An overview of the leg CAD model with a torsional spring.

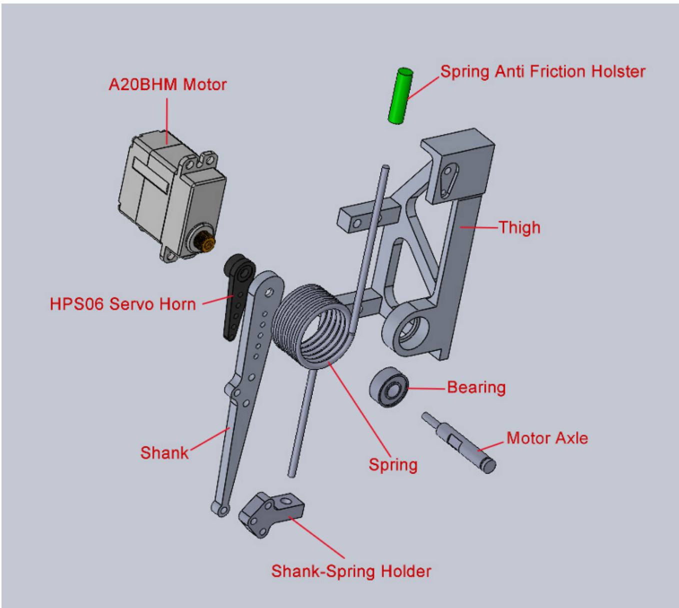


Figure 20: Annotated exploded view of the CAD leg design.

Our specific choice of motors can be found in table ??. Info about the specific motors can be found in appendices ?? to ??.

Corresponding Joint	Motor Name
Knee flexion/extension	A20BHM
Hip flexion/extension	A06CLS V2
Hip adduction/abduction	A06CLS V2

Table 5: Selected Motors

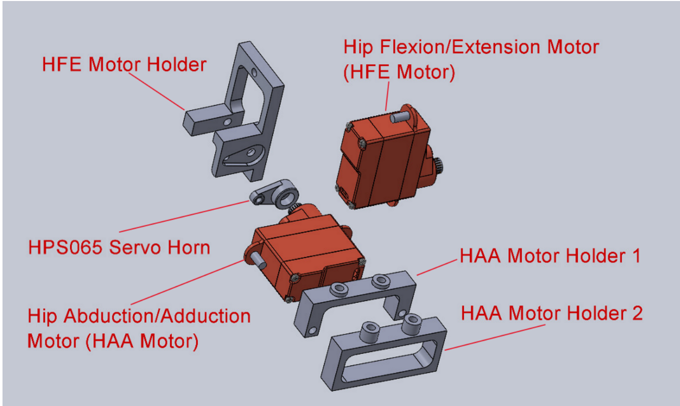


Figure 21: Exploded view of the motor housings for the hip joint.

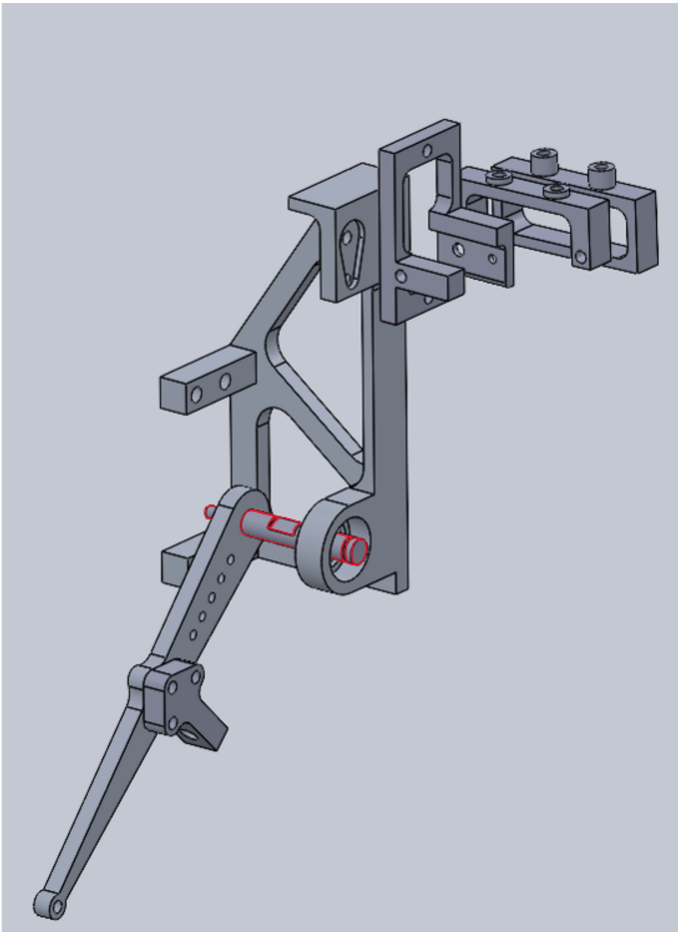


Figure 22: An overview of the CAD model of the leg design, but showing only the components that will be manufactured inhouse. Axle that will be threaded and screwed directly into the motor shaft, and lead directly into a ball bearing, is emphasized in red.

The reason these motors in particular were chosen is the fact that no motors were found in a similar weight class that could provide the same torque and speed. For example, if considering AGF-RC motors, the next motor, strength-wise, after the A20BHM motor, is the A35CHM motor, info about which can be found in appendix **??**. Despite being significantly heavier, the A35CHM motor is only marginally stronger than the A20BHM motor. The reasoning behind the choice of the A06CLS V2 is detailed in section **??**.

It is worth mentioning that the mass of the thigh shown in figure **??** is considerably higher than

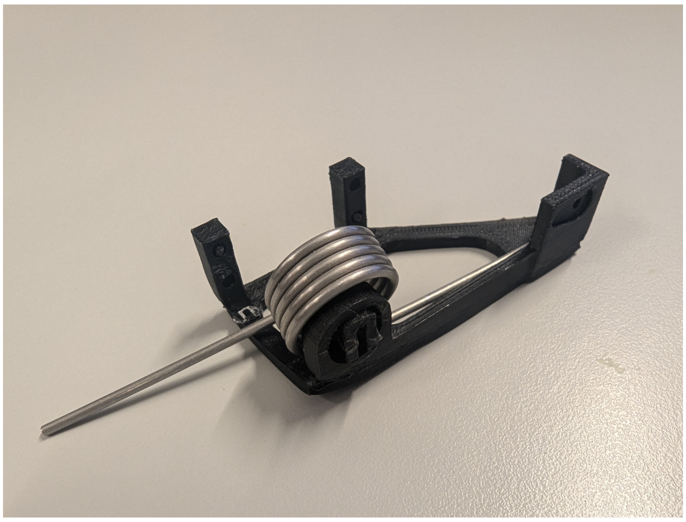


Figure 23: A 3D printed version of the leg thigh with the torsional spring.

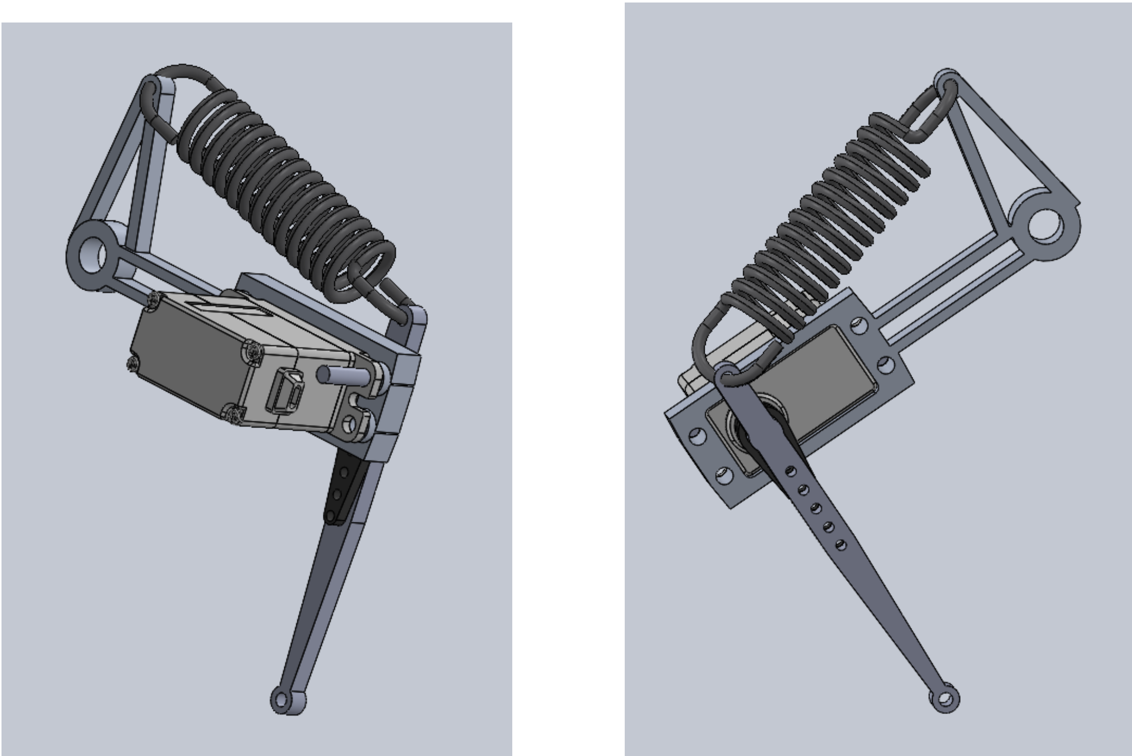


Figure 24: Comparison of extension spring configurations: outside (left) and inside (right) the leg.

a similarly long thigh in the simulation, as described in section ??. The main difference is that the thigh in the simulations is a long and thin rectangular prism, whereas the actual leg design is currently quite wide and bulky, so as to accommodate the motor and spring while maintaining structural integrity. This is a discrepancy between the simulations and the actual design, however, since the leg masses are so small compared to the mass of the motors and the main body, the discrepancy is not considered significant. For reference, the mass of the thigh for various material densities compared to the leg design used in simulation is shown in table ??.

7.4 Other Components, Spring, etc: TODO

Bearing, spring, paw must also be mentioned. Servo horns also.

Material Density	Mass of thigh in simulation	Mass of actual thigh design
1200 kg/m^3 (Tough PLA, see appendix ??)	N/A	6.45
2700 kg/m^3 (Aluminum 6061)	2.26 g	14.27 g

Table 6: Mass of thigh for various material densities compared to the leg design used in simulation. For the real thigh, mass properties were calculated using Solidworks "Mass Properties" tool.

8 Results

8.1 Motor Friction Estimation

Figures **??** and **??** compare the measured pendulum angles against simulations using two friction models: one with only viscous friction and one with both viscous and Coulomb friction. Table **??** presents the estimated friction parameters and Mean Squared Error (MSE) between simulated and measured angles for both motors.

The combined viscous-Coulomb model achieves better accuracy, particularly in the later phases of motion where angular velocities are lower. This aligns with expectations, as Coulomb friction becomes more dominant relative to viscous friction at lower speeds. The MSE values confirm this improved performance, as the MSE with Coulomb is about 1/3 of the MSE without Coulomb for both motors

Parameter	Hip Motor	Knee Motor
Viscous friction coefficient (without Coulomb) [$\mathrm{Nm \cdot s/rad}$]	0.003552	0.011612
Viscous friction coefficient (with Coulomb) [$\mathrm{Nm \cdot s/rad}$]	0.001703	0.006124
Coulomb friction [Nm]	0.010963	0.024127
MSE without Coulomb	0.003203	0.005989
MSE with Coulomb	0.000953	0.001967

Table 7: Friction model parameters and Mean Squared Error (MSE) for both motors, comparing models with and without Coulomb friction.

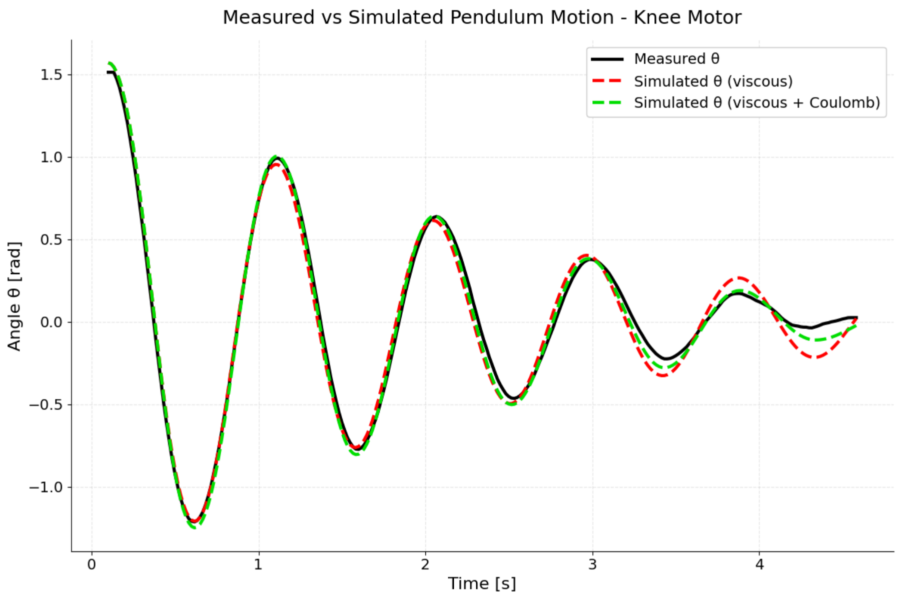


Figure 25: Linear regression fit of the pendulum data for the knee motor. Derived theta dot dot is the double derivative of the pendulum angle.

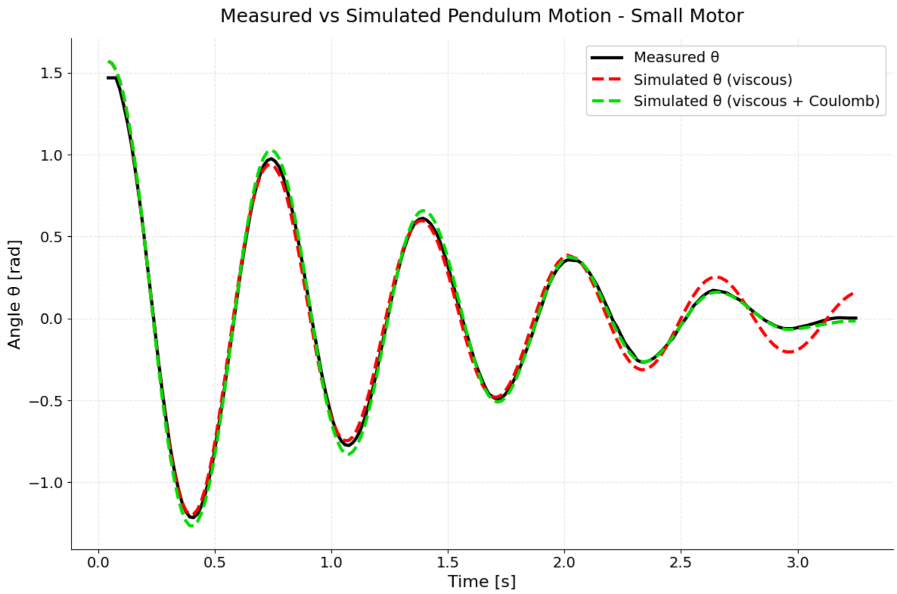


Figure 26: Linear regression fit of the pendulum data for the hip motor.

8.2 Link Length Optimization

Grid search results for Earth and Mars gravity are shown in figures **??** and **??**. The search explored link length ratios $\frac{L_1}{L_2}$ from 0.8 to 1.6 and total lengths $L_1 + L_2$ from 10 cm to 36 cm, in increments of 0.1 and 1 cm respectively.

For both gravitational environments, a link length ratio of 1.0 yields optimal performance across all total lengths. The optimal total length in Mars gravity (28 cm) exceeds Earth gravity (20 cm) by 8 cm, achieving a 102.4 cm greater maximum jump height.

Jump performance degrades rapidly for ratios below 1.0, but declines more gradually above 1.0. The effective jump height is less steep in the total length dimension around the optimal point, allowing flexibility in length selection with minimal performance impact. The performance landscape is non-convex, containing a secondary peak centered at $\frac{L_1}{L_2} \approx$ 1.3-1.4 and total length 24 cm for Earth gravity, and at $\frac{L_1}{L_2} \approx 1.5$ and total length 35 cm for Mars gravity.

Table **??** summarizes the optimal parameters and their performance.

Gravity	Jump Height (cm)	Ratio	Total Length (cm)	L1 (cm)	L2 (cm)
Earth	45.39	1.0	20	10	10
Mars	149.79	1.0	28	14	14

Table 8: Best performing link length configurations and their corresponding jump heights for Earth and Mars gravity.

8.3 Hip Motor Dimensioning Test

As can be seen in figure **??**, the hip motors follow the angle reference well, achieving three back and forth swings of 90 degrees over a one second period. The torque output of the motors during this maneuver can be seen in figure **??**. In other words, the hip motors satisfy the heuristic as presented in [**finn˙tarek˙master**].

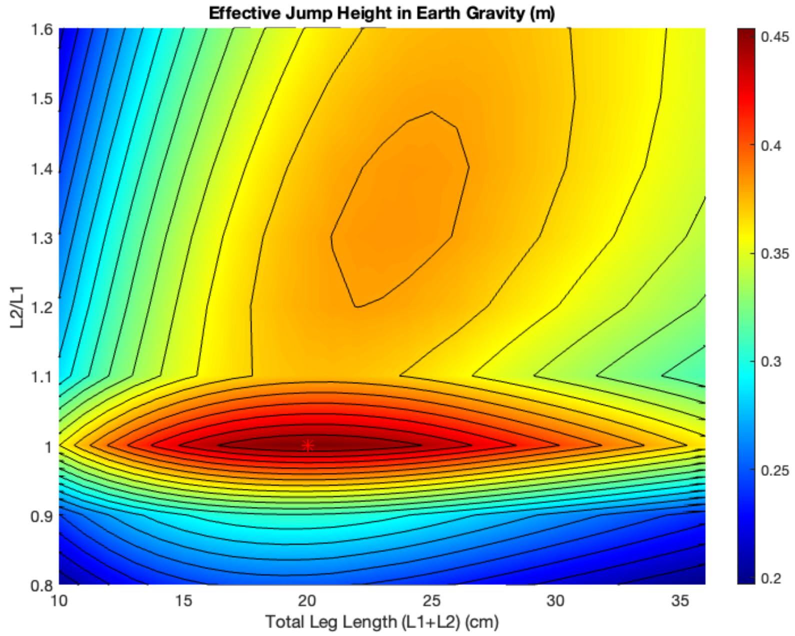


Figure 27: Grid search results showing jump height performance across different link length configurations under Earth gravity.

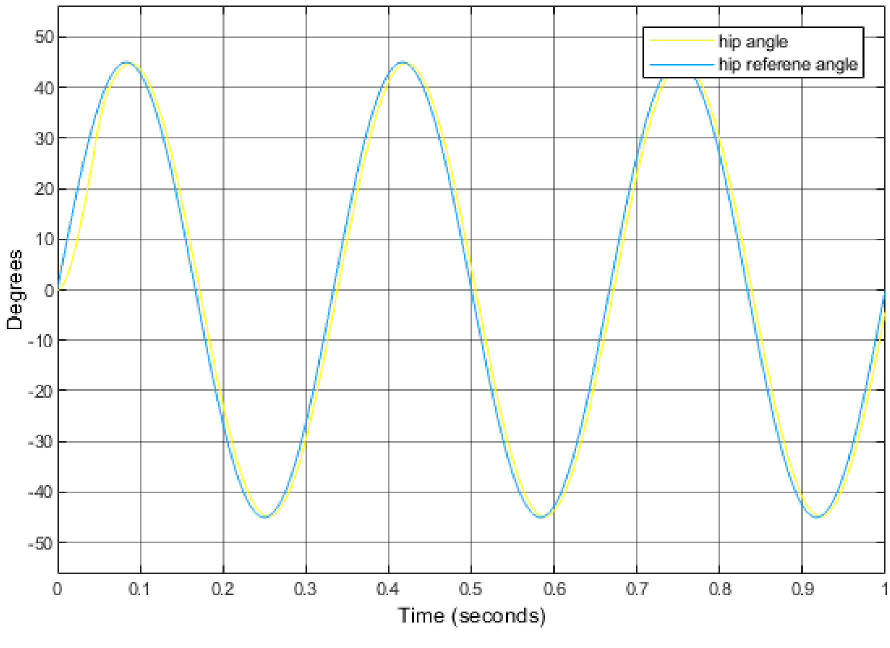


Figure 29: Commanded and actual hip joint angle achieved during the hip motor strength test simulation.

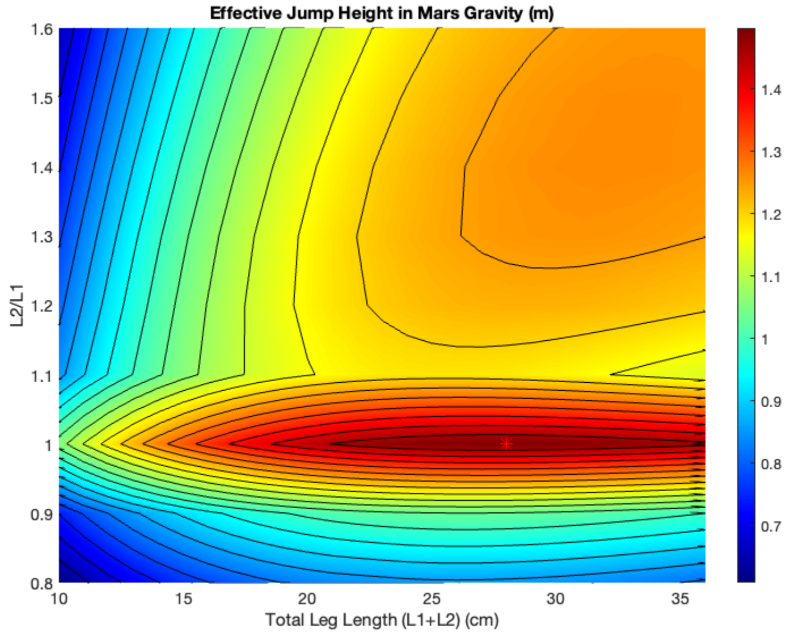


Figure 28: Grid search results showing jump height performance across different link length configurations under Mars gravity.

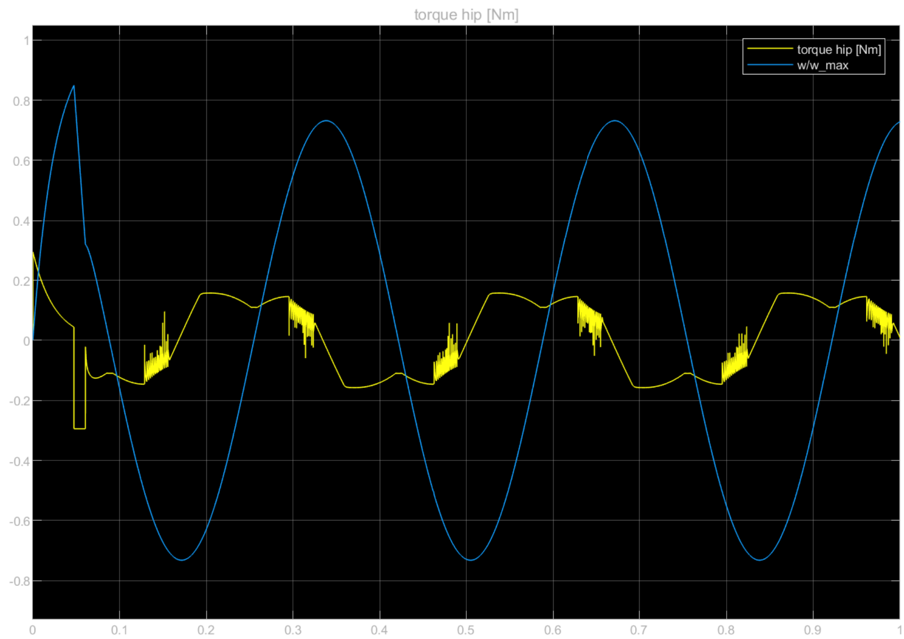


Figure 30: Torque output of the hip motors during the hip motor strength test simulation.

8.4 Motor Only Jumping Results

As described in section ??, motor-only simulations were done using the assumed characteristics of the A80BHP-H motor. Because the results were so far from acceptable, only the results of one experiment are presented here, but simulations with shorter leg and body lengths were performed. Figure ?? shows the knee joint actuation torque, knee joint velocity and knee joint angle during takeoff for the A80BHP-H motor. The robot is not able to reach a high enough speed to achieve a good jump, and the peak body center of mass height reached is 70cm, despite a total leg length of 40cm.

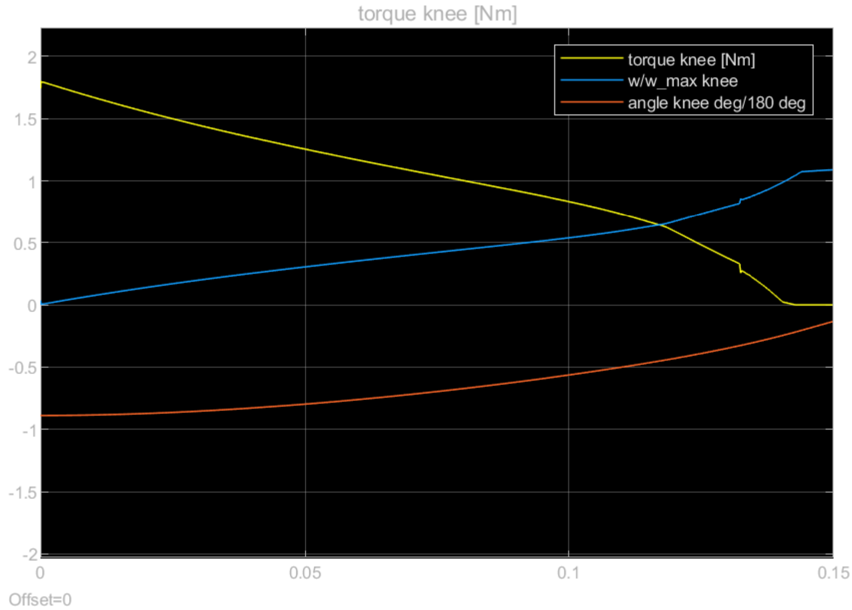


Figure 31: Knee speed until takeoff with A80BHP-H motor.

9 Discussion

9.1 Model Limitations and Uncertainties

9.1.1 Symmetric Jump Model Limitations

The symmetric jumping configuration approximates asymmetric jumping performance. For asymmetric legs, achieving vertical jumps requires careful paw placement relative to hip joints, varying with link lengths. While symmetric models place paws directly under hips for L1=L2 and L1¿L2, this produces angled rather than vertical jumps in asymmetric cases.

Experiments show vertical jumps are possible with asymmetric legs by adjusting hip-to-paw vector angle. However, the symmetric model only partially captures this through a constant angle offset when $L2 > L1$, failing to produce vertical jumps across all link lengths.

This introduces systematic biases:

- When $L_1 \neq L_2$: Underestimates jump height since asymmetric vertical jumps require initial poses that angle L2 and L1 more horizontally, better converting angular to vertical velocity with less slipping.
- When $L_1 = L_2$: Overestimates jump height since achieving vertical jumps requires L2 to be less horizontal, where they are fully horizontal in the symmetric case **??**.

These biases explain the sharp performance peak at L2 = L1 in the grid search results (figure **??**) for both Earth and Mars gravity.

9.1.2 Unmodeled Dynamics

While simulation results are promising, the lack of hardware testing introduces uncertainties about actual jumping performance. The simulated optimal link lengths may differ from real-world optima due to several unmodelled dynamics and approximations. There are unmodelled frictions like those between the springs and the legs. There are mass distribution differences from approximating legs and torso as uniform density rectangular prisms, and there are likely differences in the real and simulated masses of the robot.

9.1.3 Motor Loading Uncertainty

A critical uncertainty is the knee motors' ability to maintain stall torque while loading springs. As discussed in section **??**, prolonged stall torque risks motor overheating. The time required to fully load springs at stall torque remains unknown, making it unclear whether motors can safely achieve maximum spring compression. Our 12% safety margin for stall torque falls below the recommended 20% **??** TODO, increasing risk that motors cannot load springs maximally.

9.2 Design Limitations

The optimal link lengths found for both Earth and Mars are too long for the current robot design. To avoid leg collision with bent knees, the longer lengths would require either elongating the robot body, or translating the hip joints outward. Body elongation would increase overall weight, while outward hip translation raises collision risks during aerial stabilization. Additionally, longer legs increase the inertia that motors must overcome during aerial maneuvers, potentially slowing airial stabilization response.

Given these constraints, we opted for shorter link lengths without sacrificing significant performance. A potential fix would be to incorporate body length into the optimization process. Additionally, activating hip motors during jumps may yield shorter optimal link lengths.

9.2.1 Landing Challenges

The knee springs' equilibrium position at $\theta_2 = 0$ (straight leg) complicates landing. Achieving crouched landing poses requires knee motors to work against spring force, increasing response time. Further testing is needed to quantify motor response times across knee angles.

9.2.2 Hip Abduction/Adduction Design

As seen in figure **??**, significant z-axis distance between shank end and hip abduction/adduction motor axis creates a significant and undesired motor loading. As the specialization project has primarily ignored abduction/adduction dynamics, adjusting the shank design to avoid this issue is considered an avenue of future work.

9.3 Post-takeoff rotational velocity

The common knee bending direction shifts the center of mass backward, causing the rear legs to support more weight while the front legs experience less resistance to acceleration. This imbalance generates backward rotation at takeoff, making aerial stabilization more challenging. Two potential solutions exist: increasing the spring stiffness or motor power of the rear legs to compensate for the higher loading, or activating hip motors during jumps with greater torque applied to the back legs.

Potential solutions include increasing rear leg spring stiffness/motor power or applying greater back leg hip motor torque during jumps.

9.3.1 Limited Jump Control

Current design lacks motor actuation during jumps, preventing feedback control from compensating for parameter variations like spring stiffness. This could widen the sim-to-real gap. Adding hip motor actuation during jumps would enable feedback control - a focus for future masters thesis work.

10 Future work

The upcoming masters thesis will investigate adding hip motor actuation during jumps to enhance performance and enable feedback control. Then we will use our optimized design to construct the physical robot and create an accurate model in Nvidia Isaac Sim. This model will be used to train reinforcement learning controllers for jumping, aerial stabilization, and landing. Domain randomization techniques will help minimize the sim-to-real gap, and curriculum learning will be explored to make the learning process more tractable. Finally, we will evaluate the trained policies on the physical robot in practical jumping scenarios.

We will perform tests to evaluate the real-life jumping performance of the simulation-derived design and see if knee motors are able to provide stall torque for long enough to load springs maximally.

Due to the oversights addressed in section **??**, the promising A80BHP-H motor was overlooked when selecting motors. This motor not only has almost twice the stall torque of the current knee motor, it also has more than twice the operating velocity. Future work will include investigating the jumping capabilities of a robot combining this motor with torsional springs.

As mentioned in section **??**, adjusting the shank design to avoid a significant load on the hip abduction/adduction motor during jumping is something that will be addressed in future work.

Although the current robot incorporates springs in the actuation method, it does not incorporate series-elastic springs the way they are often used in legged robots, ie. to reduce the impact forces on the motors [**proprioceptive**]. As the robot is intended to jump, future work will include adding elastic elements to the legs to reduce the impact forces on the motors.

11 Conclusion

This project developed a small, lightweight and low-cost jumping quadruped that combines parallel torsional springs with BLDC motors. It is intended to be used as a platform for training DRL policies for jumping, arial stabilization, and landing.

Simscape simulations showed motor-only actuation produced insufficient jump heights, leading to the addition of torsional springs. Grid search optimization revealed that equal-length links maximize jump height, though the optimal lengths were too long for a compact robot design.

Motor friction estimation through pendulum tests improved simulation accuracy, but hardware testing remains incomplete. We have not tested if motors can maintain stall torque during spring loading without overheating. Unmodeled dynamics like spring-leg interface friction and mass distribution differences will affect real performance. The symmetric jumping model differs from realistic asymmetric leg configurations, and simplified paw placement affects jump trajectory predictions.

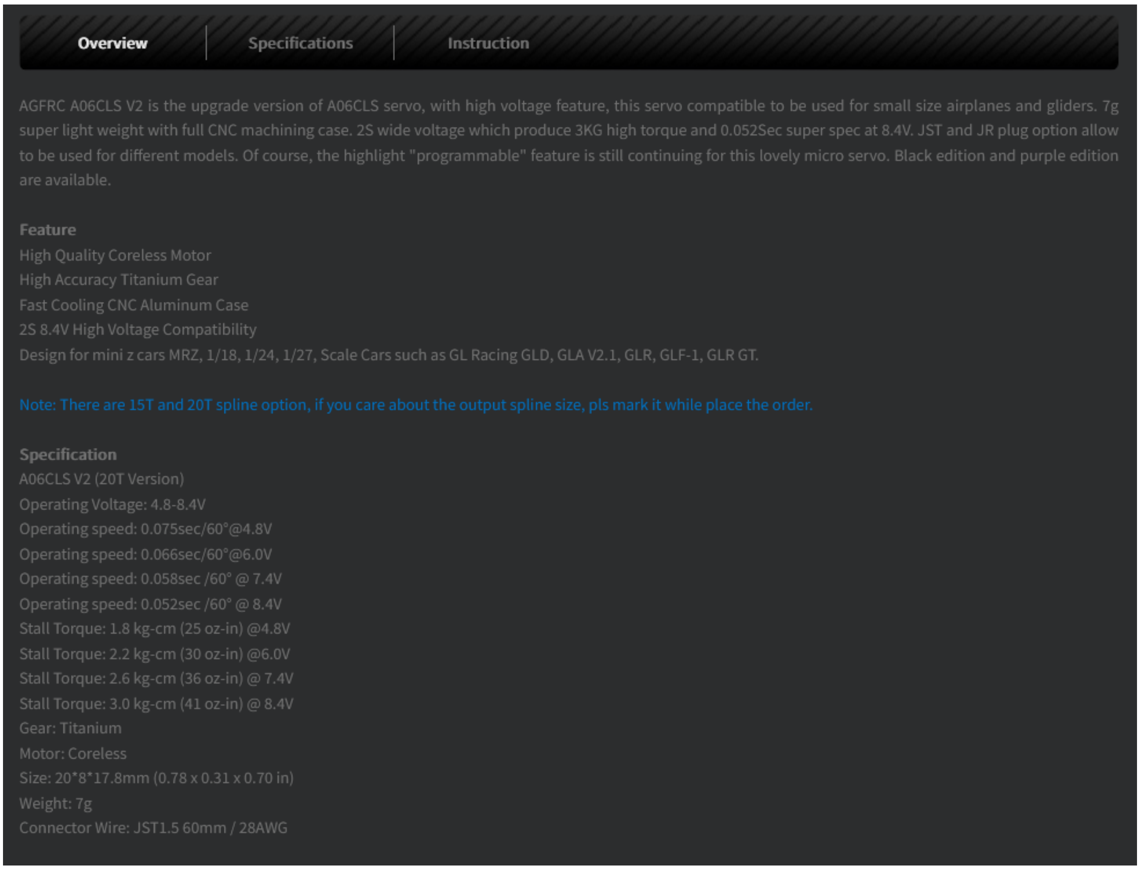
The current design has several limitations. Without motor actuation during jumps, potential energy remains untapped. Lack of feedback control during jumps prevents adjustment for unmodeled dynamics differences between simulation and real hardware. Spring equilibrium at straight leg slows landing response. Hip abduction/adduction motor placement creates undesired loading, and the common knee bending direction generates backward rotation at takeoff.

Despite these limitations, simulations indicate the design can achieve sufficient jump heights for demonstrating DRL controllers in both Earth and Mars gravity, setting the foundation for future hardware validation and control development in the upcoming master's thesis.

Appendix

A A06CLS V2 Website Information

This information is taken from the website: https://www.agfrc.com/index.php?id=2666



Overview | Specifications | Instruction

AGFRC A06CLS V2 is the upgrade version of A06CLS servo, with high voltage feature, this servo compatible to be used for small size airplanes and gliders. 7g super light weight with full CNC machining case. 2S wide voltage which produce 3KG high torque and 0.052Sec super spec at 8.4V. JST and JR plug option allow to be used for different models. Of course, the highlight "programmable" feature is still continuing for this lovely micro servo. Black edition and purple edition are available.

Feature
High Quality Coreless Motor
High Accuracy Titanium Gear
Fast Cooling CNC Aluminum Case
2S 8.4V High Voltage Compatibility
Design for mini z cars MRZ, 1/18, 1/24, 1/27, Scale Cars such as GL Racing GLD, GLA V2.1, GLR, GLF-1, GLR GT.

Note: There are 15T and 20T spline option, if you care about the output spline size, pls mark it while place the order.

Specification
A06CLS V2 (20T Version)
Operating Voltage: 4.8-8.4V
Operating speed: 0.075sec/60°@4.8V
Operating speed: 0.066sec/60°@6.0V
Operating speed: 0.058sec /60° @ 7.4V
Operating speed: 0.052sec /60° @ 8.4V
Stall Torque: 1.8 kg-cm (25 oz-in) @4.8V
Stall Torque: 2.2 kg-cm (30 oz-in) @6.0V
Stall Torque: 2.6 kg-cm (36 oz-in) @ 7.4V
Stall Torque: 3.0 kg-cm (41 oz-in) @ 8.4V
Gear: Titanium
Motor: Coreless
Size: 20*8*17.8mm (0.78 x 0.31 x 0.70 in)
Weight: 7g
Connector Wire: JST1.5 60mm / 28AWG

Figure 32: A06CLS V2 Motor Information (Curt)

B A20BHM Website Information

This information is taken from the website:
https://www.agf-rc.com/agfrc-a20bhm-21g-high-speed-0068sec-114kg-programmable-digital-brushless-48-84v-strength-steel-gear-micro-wing-servo-18010-for-airplane-aircraft-p4231199.html

Description | Item specifics | Reviews (0)

AGFRC Aluminum Case HV High Torque Brushless Micro Digital Servo (A20BHM)

AGFrc A20BHM, is the 8.4V high voltage compatible performance digital mini servo, also the first launch brushless motor 12mm thickness RC servo. It is made in a robust CNC aluminum case with high accuracy metal gears, it particularly ideal for RC cars and other application.

Feature

High Performance Digital Wing servo
High Precision Metal Gears
High Quality Brushless Motor
Double Ball Bearings
Full Aluminum Case

Specification

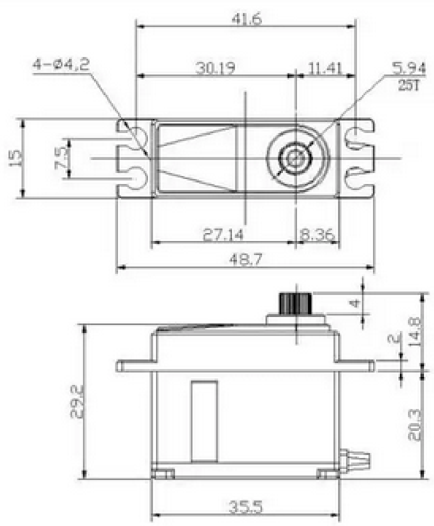
A20BHM
Operating Voltage: 4.8-8.4 V
Operating speed: 0.092sec/60°@ 6.0V
Operating speed: 0.075sec/60°@ 7.4V
Stall Torque: 8.5 kg-cm (118 oz-in) @ 6.0V
Stall Torque: 10.2 kg-cm (142oz-in) @ 7.4V
Gear: Strength Steel + Copper
Motor: Bls
Size: 23*12*27.5mm
Weight: 21g
Connector Wire: TYU 180 mm / 26AWG

C A35CHM Motor Information

PRODUCT SPECIFICATION

A35CHM

Control System	Pulse width modulation control
Refresh Rate	333Hz
Neutral Position	1520uS
Signal Mode	Digital
Dead band	2 uSec
Operating Voltage	4.8V ~ 8.4V
Operating Temperature	-15C°~ +70C°
Bearing	Dual Ball Bearing
Mechanical Limit Angle	220°
Size	35.5*15*29.2mm
Net Weight	41g
Wire	JR 180mm /22AGW
Operating Travel	180°±10°
Signal Range	500 to 2500 uSec



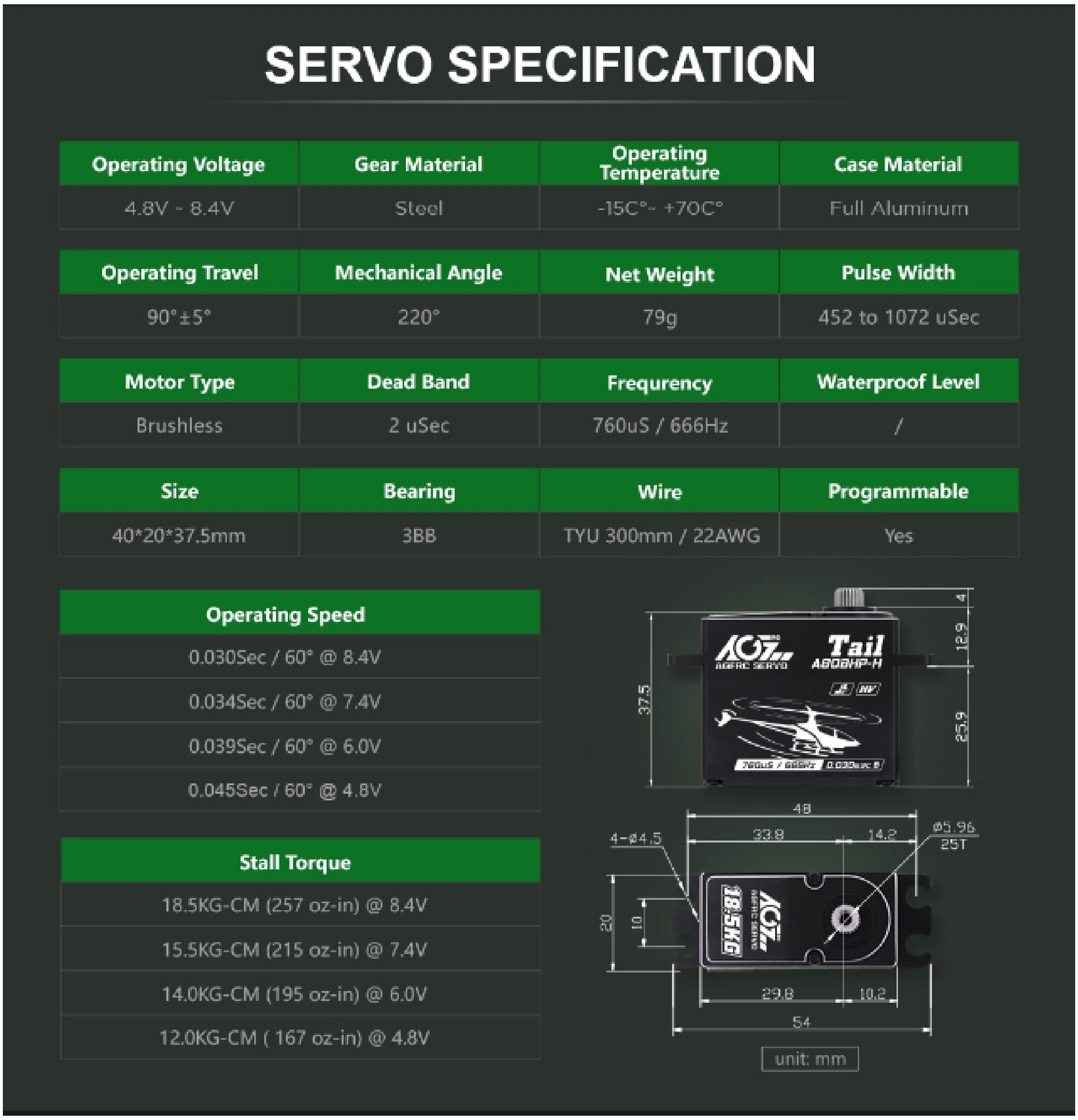
Stall Torque	9.5 kg-cm (132 oz-in) @ 4.8V 10.5 kg-cm (146 oz-in) @ 6.0V 11.5 kg-cm (160 oz-in) @ 7.4V 12.5 kg-cm (174 oz-in) @ 8.4V	**Unload Current**	230mA @ 4.8V 300mA @ 6.0V 380mA @ 7.4V 420mA @ 8.4V
Operating Speed	0.145sec/ 60°@ 4.0V 0.125sec/60° @ 6.0V 0.095sec/60° @ 7.4V 0.085sec/60° @ 8.4V	**Loading Current**	2000mA @ 4.8V 2400mA @ 6.0V 2600mA @ 7.4V 2800mA @ 8.4V

Direction	☑ CCW	☐ CW		
Waterproof Level	☐ IP65	☐ IP67		
Angle Sensor	☑Potentiometer	☐ Magnet Angle Sensor		
Motor Type	☐Brushless	☑Coreless	☐DC	
Motor Drive	☑FET Drive	☐IC Drive	☐Transistor Drive	
Programmable	☑ Yes	☐ No		
Gear Material	☐Strength Steel	☑Titanium	☐Copper	☐Plastic
Horn Gear Spline	☑25T-φ5.92mm	☐25T-φ4.94mm	☐Other______	
Case Material	☑AL6061T6	☐AL+Plastic	☐Plastic	
Bearing Material	☑Metal	☐Plastic		
Horn Accessories	☐AL6061T6	☑Plastic		
Wire Color	Negative: ☑ Black	☐Brown		
	Positive: ☐ Black	☑Red		
	Signal: ☐Grey	☑White	☐Orange	

Figure 33: A35CHM Motor Information

D A80BHP-H Motor Information

Please note that this brochure gives an operating travel of 90 degrees, this was later confirmed to be false, the actual operating travel is 180 degrees.



SERVO SPECIFICATION

Operating Voltage	Gear Material	Operating Temperature	Case Material
4.8V ~ 8.4V	Steel	-15C°~ +70C°	Full Aluminum

Operating Travel	Mechanical Angle	Net Weight	Pulse Width
90°±5°	220°	79g	452 to 1072 uSec

Motor Type	Dead Band	Frequrency	Waterproof Level
Brushless	2 uSec	760uS / 666Hz	/

Size	Bearing	Wire	Programmable
40*20*37.5mm	3BB	TYU 300mm / 22AWG	Yes

Operating Speed
0.030Sec / 60° @ 8.4V
0.034Sec / 60° @ 7.4V
0.039Sec / 60° @ 6.0V
0.045Sec / 60° @ 4.8V

Stall Torque
18.5KG-CM (257 oz-in) @ 8.4V
15.5KG-CM (215 oz-in) @ 7.4V
14.0KG-CM (195 oz-in) @ 6.0V
12.0KG-CM (167 oz-in) @ 4.8V

Figure 34: A80BHP-H Motor Information

E Ultimaker Tough PLA Technical Data Sheet

Ultimaker

Ultimaker Tough PLA
Technical data sheet

General overview

Chemical composition	See Tough PLA safety data sheet, section 3
Description	Ultimaker Tough PLA is a technical PLA filament with toughness comparable to Ultimaker ABS. Ideal for reliably printing technical models at large sizes, our Tough PLA offers the same safe and easy use as regular PLA
Key features	With an impact strength similar as and higher stiffness compared to Ultimaker ABS, Tough PLA is less brittle than regular PLA and gives a more matte surface finish quality. Heat resistance is similar to standard PLA filaments, so printed parts should not be exposed to temperature above 58 °C. More reliable than ABS for larger prints, with no delamination or warping. Ultimaker Tough PLA is compatible with Ultimaker support materials (PVA and Breakaway), giving full geometric freedom when designing parts
Applications	Functional prototyping, tooling, manufacturing aids
Non-suitable for	Food contact and in vivo applications. Long term outdoor usage or applications where the printed part is exposed to temperatures higher than 58 °C.

Filament specifications

	Method (standard)	Value
Diameter	–	2.85 ± 0.05 mm
Max roundness deviation	–	0.05 mm
Net filament weight	–	750 g
Filament length	–	~96 m

Color information

Color	Color code
Black	RAL 9017
White	RAL 9003
Green	RAL 6038
Red	RAL 3018
Gray	RAL 7000
Yellow	RAL 1018
Blue	RAL 5019

Figure 35: UM220509 Tough PLA TDS