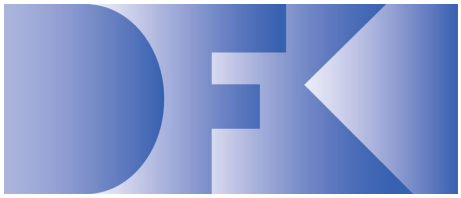
Hopping Leg Energy Shaping Control

Hauke Isermann

July the 31st, 2021



CONTENTS

CHAPTER

ONE

CARTESIAN STIFFNESS CONTROL

The low level control of the hopping leg has been realized using PD control. Here, three functions have been implemented after Di Carlo, 2020: A simple joint space PD controller with the control law:

$$\boldsymbol{\tau} = \boldsymbol{\tau_{ff}} + \boldsymbol{K_{p,j}}(\boldsymbol{q_{des}} - \boldsymbol{q}) + \boldsymbol{K_{d,j}}(\dot{\boldsymbol{q}}_{\boldsymbol{des}} - \dot{\boldsymbol{q}}), \tag{1.1}$$

Cartesian stiffness control, with the control law:

$$\boldsymbol{\tau} = \boldsymbol{J^T}(\boldsymbol{f_{ff}} + \boldsymbol{K_{p,c}}(\boldsymbol{p_{des}} - \boldsymbol{p}) + \boldsymbol{K_{d,c}}(\dot{\boldsymbol{p}}_{\boldsymbol{des}} - \dot{\boldsymbol{p}})), \tag{1.2}$$

and combined stiffness control, which combines the two controllers from above:

$$\boldsymbol{\tau} = \boldsymbol{\tau_{ff}} + \boldsymbol{K_{p,j}}(\boldsymbol{q_{des}} - \boldsymbol{q}) + \boldsymbol{K_{d,j}}(\dot{\boldsymbol{q}}_{\boldsymbol{des}} - \dot{\boldsymbol{q}}) + \boldsymbol{J^T}(\boldsymbol{f_{ff}} + \boldsymbol{K_{p,c}}(\boldsymbol{p_{des}} - \boldsymbol{p}) + \boldsymbol{K_{d,c}}(\dot{\boldsymbol{p}}_{\boldsymbol{des}} - \dot{\boldsymbol{p}})). \tag{1.3}$$

$\boldsymbol{\tau}$ denotes here the motor torque vector, $\boldsymbol{\tau_{ff}}$ the commanded feed forward torques, $\boldsymbol{K_{p,j}}$ and $\boldsymbol{K_{d,j}}$ the joint space gain matrices, $\boldsymbol{K_{p,c}}$ and $\boldsymbol{K_{d,c}}$ the Cartesian space gain matrices, $\boldsymbol{J}$ the hybrid Jacobian at the end effector and $\boldsymbol{f_{ff}}$ the feed forward force in Cartesian space. The current and desired joint positions are given with $\boldsymbol{q}$ and $\boldsymbol{q_{des}}$ in joint space, and the actual and desired end effector position vectors are written as $\boldsymbol{p}$ and $\boldsymbol{p_{des}}$.

1.1 Parameter tuning

The mentioned low level controllers require quite a few parameters to be tuned. Here, we see some differences between the tuning of the real system and tuning the simulation, therefore they have to be tuned individually. For the real system, the joint level control is likely to be done by the motor internal PD controller, as it is more stable. Therefore, it is recommended to just use the Cartesian stiffness control and tune the joint space gains separately in the internal motor controller. In simulation, there might be no internal motor controller. Therefore, the combined stiffness can be used.

When tuning the gains, one has to consider that higher joint level gains reduce the influence of the Cartesian stiffness and vice-versa. To stabilize the joint positions and counteract vibrations, larger

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

joint space gains are more likely to give good results. For effectively applying some Cartesian feed forward forces instead, or to apply different stiffnesses in x and y direction, low joint level gains are required, since they can neutralize the feed forward forces otherwise.

CHAPTER

TWO

HOPPING CONTROL

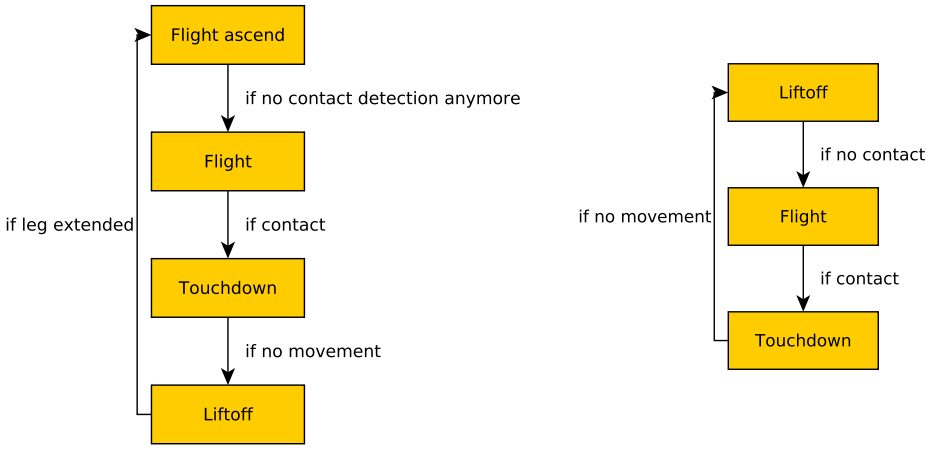
Hopping can be simplified with the Spring Loaded Inverted Pendulum (SLIP) model, in which a mass is mounted on a massless spring Tedrake, 2021. Due to this spring, the energy in the system stays constant. By adding or removing energy into the system, e.g., by applying some force to the spring, the jumping height can be controlled.

Since our hopping leg is designed to have lightweight legs, it is similar to a SLIP model. The main difference is, that our hopping leg has no inbuilt elasticity. Therefore, it has to apply the desired jumping energy during every lift-off. To achieve this, two controllers have been implemented on top of a state machine. The state machine divides the jumping cycle into 3 (or 4) phases with different goals: In the flight phase, the leg prepares its position for the touchdown. In the real system, this flight phase is divided into two phases, flight ascend and flight, since the contact detection does not work reliable. During the flight phase, the leg is stiff, since it has to reach a specific position. The flight phase ends with the first contact of the leg with the ground. During the touchdown, the movement of the leg is damped, since its energy cannot be used due to the absence of elastic elements. When the downwards movement is fully stopped, the lift-off phase starts. The conditions for the phase transitions are written in Table 2.1 more precisely. The two implemented control approaches both use this lift-off phase, to apply the desired energy for the next jump. In Figure 2.1 you can see the phase diagrams of the state machine used in the real system and in simulation.

Table 2.1: Phase transition conditions for the state machine in the real system and in simulation.

Phase to enter	**Real system**	**Simulation**
Touchdown	$(\lvert\tau_1\rvert \geq 10$ or $\lvert\tau_2\rvert \geq 10)$ and $x \leq x_{des} - 0.005$	contact
Lift-off	$\lvert\dot{\boldsymbol{p}}\rvert < 0.1$ and $x \leq x_{des}$	$\dot{x} \geq 0$
Flight ascend	$x \geq x_{des} + 0.16$	–
Flight	not $(\lvert\tau_1\rvert \geq 10$ or $\lvert\tau_2\rvert \geq 10)$	no contact

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de



(a) Real system (b) Simulation

Figure 2.1: Phase diagrams of the state machine for the real system (a) and the simulation (b).

2.1 Gain scheduling control

The gain scheduling control uses Cartesian stiffness control to mirror the behaviour of the spring in the SLIP model. For this, a high stiffness is necessary to reapply sufficient energy into the system. To be able to lift-off and to counteract dissipation, the desired leg length during the lift-off phase can be increased and/or a feed forward force can be applied.

2.2 Energy shaping control

The advantage of energy shaping control is, that we can control the jumping height by applying energy to the system until the desired energy E_{des}, which is needed to jump at a desired height h_{des}, is reached. With the simplification, that we assume all mass m in the shoulder of the jumping leg the desired energy can be calculated with:

$$E_{des} = mgh_{des}, \tag{2.1}$$

where g is the gravitational acceleration. Since the hopping is one dimensional in the current setup, only the kinetic energy in this direction (x-direction, see Figure 2.2) will be used for further calculations.

The used approach is non-continuous energy shaping, where we just observe the reached energy, which is:

$$E = mgh_{reached} \tag{2.2}$$

For reaching this energy, we had to apply a feed forward force F_{ff} during the lift-off phase. The needed force F_e is almost constant and can be estimated as

$$F_e = \frac{mg(h_{des} - h_0)}{\Delta h_{lift-off}}, \tag{2.3}$$

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

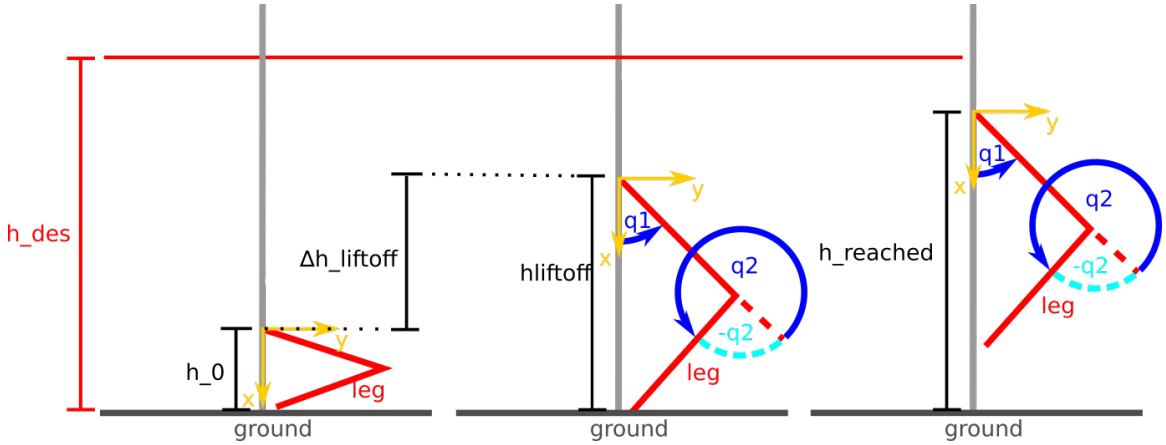


Figure 2.2: Schematic diagram of the hopping leg and the used parameters.

where h_0 is the base position at the start of the lift-off phase and $\Delta h_{lift-off}$ is the distance, over which the leg has ground contact, and hence can apply force.

Since the applied feed forward force is proportional to the reached energy, we can write:

$$E \propto kF_e \tag{2.4}$$

and hence:

$$\dot{E} \propto \dot{k}F_e \tag{2.5}$$

Thus, we can add energy to the system by increasing k and remove energy by decreasing it. To control k the following update rule can be used:

$$k_{i+1} = k_i(\frac{E_{des}}{E})^2 \tag{2.6}$$

CHAPTER

THREE

STATE ESTIMATION

For the energy shaping control, a height feedback is necessary. Therefore, a simple state estimation has been implemented. Since during the flight phase no additional forces can be applied to the system, we can expect the acceleration to be g. As we know the lift-off position and velocity, we can calculate the current height of the base h with:

$$h = \frac{1}{2}g(t^2 - t_0^2) - gt_0(t - t_0) + v_0(t - t_0) + h_0, \tag{3.1}$$

where t is the current time and t_0 is the time at lift-off. h_0 and v_0 denote the height and velocity of the base at time t_0.

This simple state estimator, which is only used during flight phase, since with ground contact forward kinematics are more accurate, gives us quite good results. Nevertheless, we still have to admit that it ignores friction and dynamic effects of the leg movements.

CHAPTER

FOUR

INCREASING THE JUMPING HEIGHT

During our trials on the real system, we found that we are limited to around 0.5 m jumping height when we just apply feedforward forces in x-direction (see Figure 2.2). We can find the reason for this behaviour in the way, how the torques are calculated from the feed forward forces, as Equation 4.1 always leads to zero torques in the shoulder joint, when the end effector position as well as the feed forward force is zero in y-direction.

$$\boldsymbol{\tau} = \boldsymbol{J}^{\boldsymbol{T}} \boldsymbol{f}_{\boldsymbol{ff}} \tag{4.1}$$

Thus, just one motor applies torques during the lift-off. To use the full potential of both motors, we have to apply some feedforward forces within the friction cone in y-direction (see Figure 2.2) too. Using heuristics, we found a feed forward force of 15 N in y-direction to be beneficial for the jumping height. In further research, the optimal feed forward force distribution can be found using optimal control.

CHAPTER

FIVE

BACKFLIP CONTROL IN SIMULATION

For the energy shaping control, a support to do flips has been implemented too. For these backflips, a trajectory has been added to the original desired hip joint angle q_1 during flight phase. Additionally, the joint positions had to be corrected by adding $n \cdot 2\pi$, where n is the number of forward flips minus the number of backflips. This added values are updated directly before the flying phase, where a flip will happen. Hence, the trajectory of the flip is cancelling out this difference in the beginning of the flip. For a backflip, the new joint position $q_{1,t}$ is defined with:

$$q_{1,t} = q_1 - 2\pi + (-\pi \cos(\frac{\pi t}{\Delta t_{i-1} - t_s}) + \pi), \tag{5.1}$$

where t is the time since the flip started, Δt_{i-1} is the duration of the last flight phase and t_s is a safety time, which was set to 0.1 s. For a forward flip, the expression was accordingly:

$$q_{1,t} = q_1 + 2\pi - (-\pi \cos(\frac{\pi t}{\Delta t_{i-1} - t_s}) + \pi). \tag{5.2}$$

Additionally, flips with a change of the knee direction have been implemented. For a backflip, the expression for the hip joint position was:

$$q_{1,t} = -q_1 - 2\pi + (-(|q1| + \pi) \cos(\frac{\pi t}{\Delta t_{i-1} - t_s}) + \pi), \tag{5.3}$$

and for a forward flip:

$$q_{1,t} = -q_1 - 2\pi - (-(|q1| + \pi) \cos(\frac{\pi t}{\Delta t_{i-1} - t_s}) + \pi). \tag{5.4}$$

At the same time, the position of the knee joint q_2 has been updated for both flip directions with:

$$q_{2,t} = -q_2 \cos(\frac{\pi t}{\Delta t_{i-1} - t_s}) \tag{5.5}$$

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

To use these backflips, a list has been used to define arbitrary sequences of normal jumps and forward- and backflips, were always the next element is given to a prepare function, which adds or subtracts 2π to the current adjustment term and saves the new knee direction for inverse kinematics. Afterwards, the jumping mode is performed during flight phase.

5.1 Backflip in the real system

In the real system, a backflip has been performed as well. This has been done much easier, by just commanding the current hip joint position plus 2π during the flight phase. Afterwards, the script stopped. In future, a similar approach to the simulation would be interesting.

CHAPTER

SIX

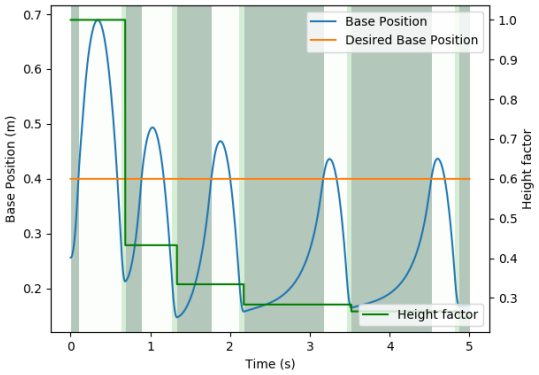
RESULTS

6.1 Hopping control in simulation

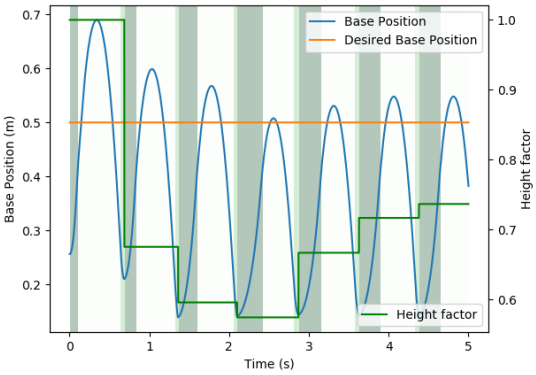
The simulation was able to get close to the desired height values of 0.4 m, 0.5 m and 0.6 m as plotted in Figure 6.1. But since the state recognition in simulation is worse than in the real system, the desired height cannot always be reached correctly. But in previous tests, using not the estimated but the real state, the simulation was able to maintain desired height quite stable. In Figure 6.2, 6.3 and 6.4 exemplary jumping sequences of the 3 desired heights are plotted.

If we look at the joint torques (Figure 6.5), we see, that for all three desired heights, we haven't reached the torque limits during the lift-off phase (dark green background). Hence, there is still more potential for higher jumps, even without applying feed forward forces in y-direction as proposed in chapter 4.

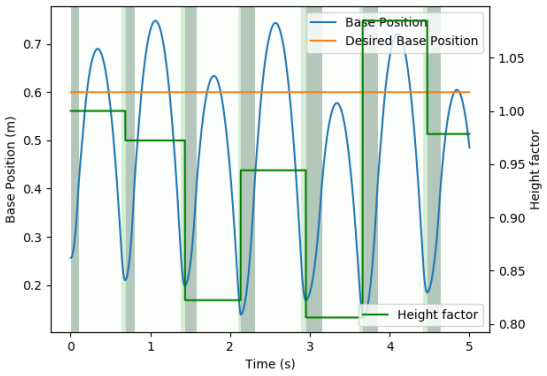
Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de



(a) Desired jumping height: 0.4 m



(b) Desired jumping height: 0.5 m



(c) Desired jumping height: 0.6 m

Figure 6.1: Jumping heights with energy shaping control and a desired height of 0.4 m (a), 0.5 m (b) and 0.6 m (c) (yellow line) in simulation. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

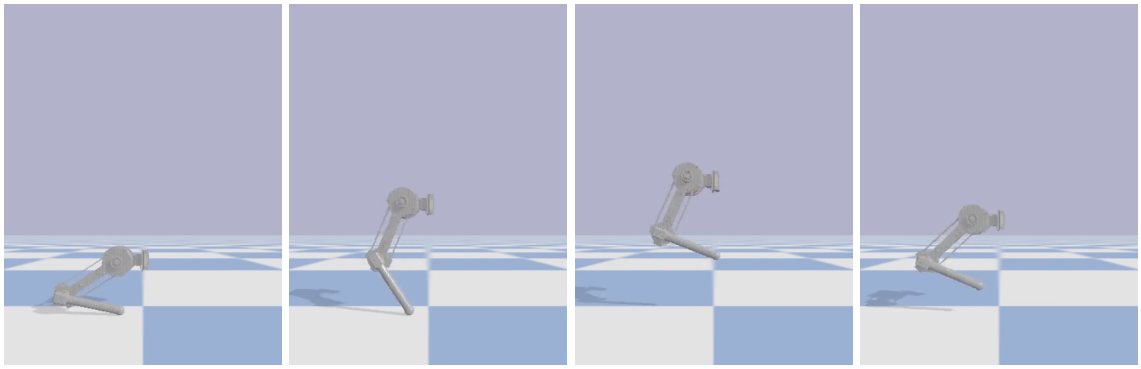


Figure 6.2: Simulated jumping sequence for 0.4 m desired height

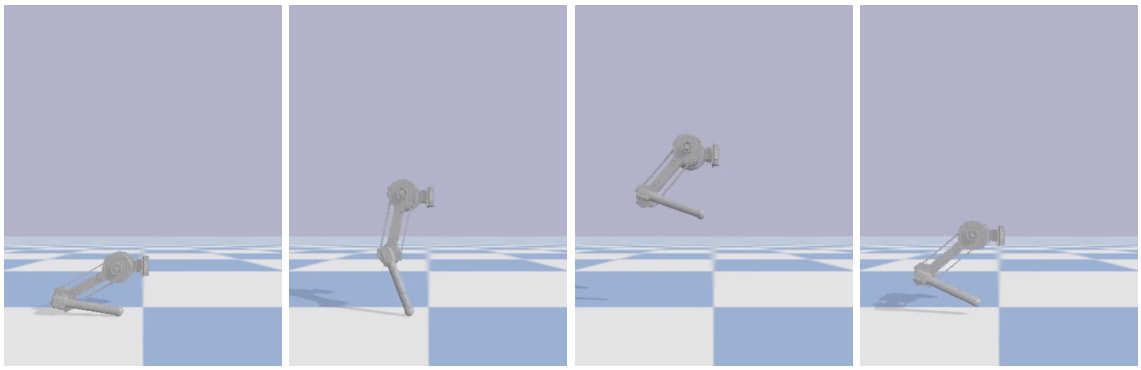


Figure 6.3: Simulated jumping sequence for 0.5 m desired height

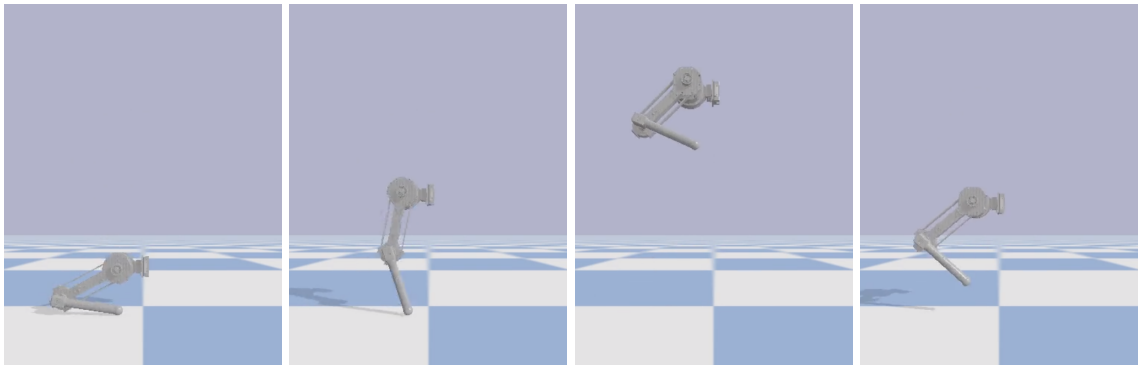
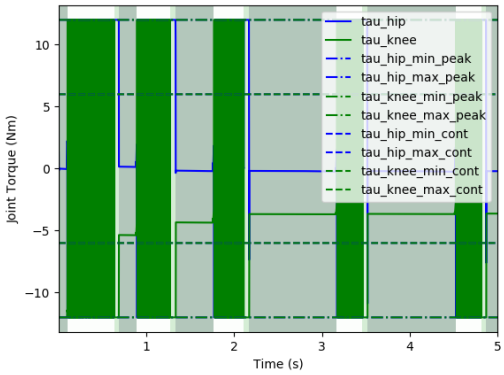
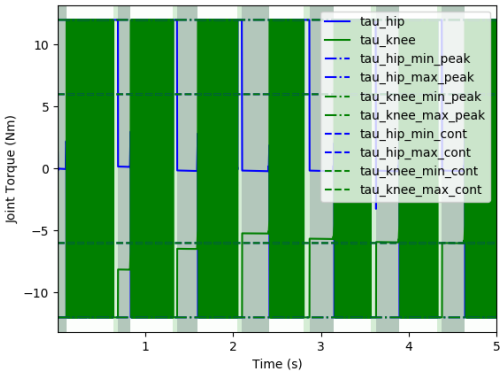


Figure 6.4: Simulated jumping sequence for 0.6 m desired height

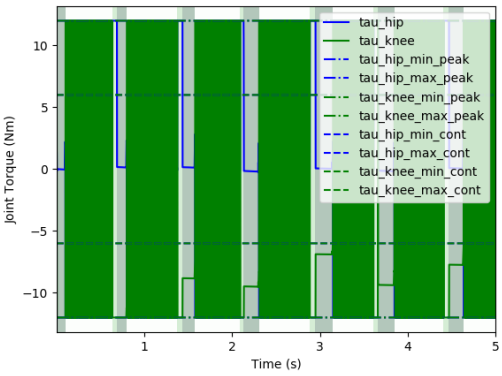
Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de



(a) Desired jumping height: 0.4 m



(b) Desired jumping height: 0.5 m



(c) Desired jumping height: 0.6 m

Figure 6.5: Torque of the energy shaping control simulation with a desired height of 0.4 m (a), 0.5 m (b) and 0.6 m (c). The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

6.2 Hopping control in the real system

In the real system we observed, that the energy shaping control was able to reach preset jumping heights of 0.4 m (see Figure 6.6), 0.5 m (see Figure 6.7) and 0.6 m (see Figure 6.8) according to the estimated state. By observing the torques during these jumps (Figure 6.9, 6.10 and 6.11) we can see, that during the lift-off phase (dark green background) the applied torques increase for higher jumping heights. At 0.6 meters, the straight part of the desired elbow torques indicates, that the torques are clipped due to the motor torque limits. Hence, the elbow cannot apply higher torques, and we reached the limit of the energy shaping controller in combination with the commanded feed forward force of 15 N in y-direction (see chapter 4).

If we compare the state estimation with the video sequences, we took during the jumping trials (see Figure 6.12, 6.13 and 6.14) we can see, that the jumping height was a bit overestimated, especially for the desired height of 0.6 m we actually just reached about 0.58 m (see Figure 6.14).

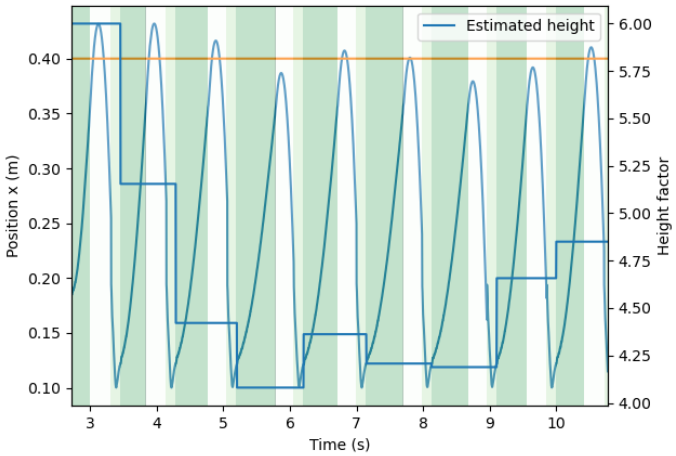


Figure 6.6: Jumping heights with energy shaping control and a desired height of 0.4 m (yellow line). The height of the base is printed with the sin like blue line, and the height factor k_i is printed with the stabbed blue line on the secondary y-axis. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

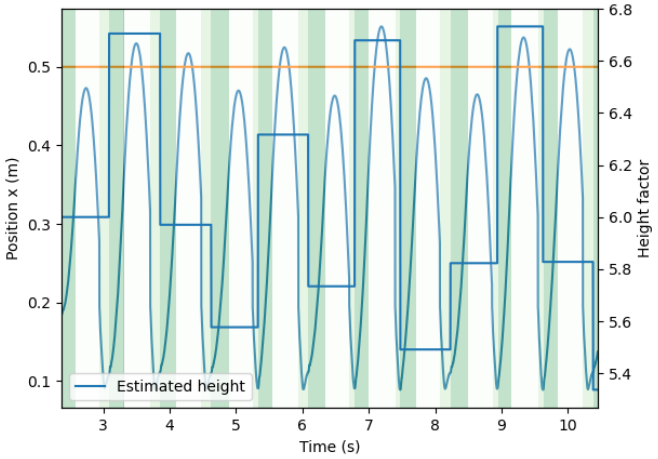


Figure 6.7: Jumping heights with energy shaping control and a desired height of 0.5 m (yellow line). The height of the base is printed with the sin like blue line, and the height factor k_i is printed with the stabbed blue line on the secondary y-axis. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

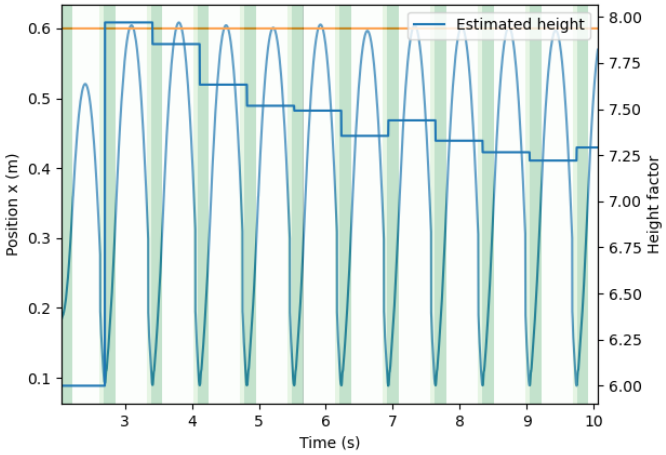


Figure 6.8: Jumping heights with energy shaping control and a desired height of 0.6 m (yellow line). The height of the base is printed with the sin like blue line, and the height factor k_i is printed with the stabbed blue line on the secondary y-axis. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

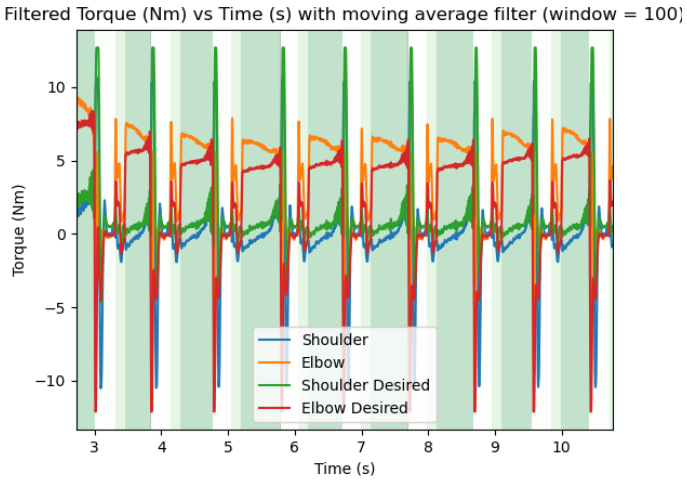


Figure 6.9: Filtered torque of the energy shaping control with a desired height of 0.4 m. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

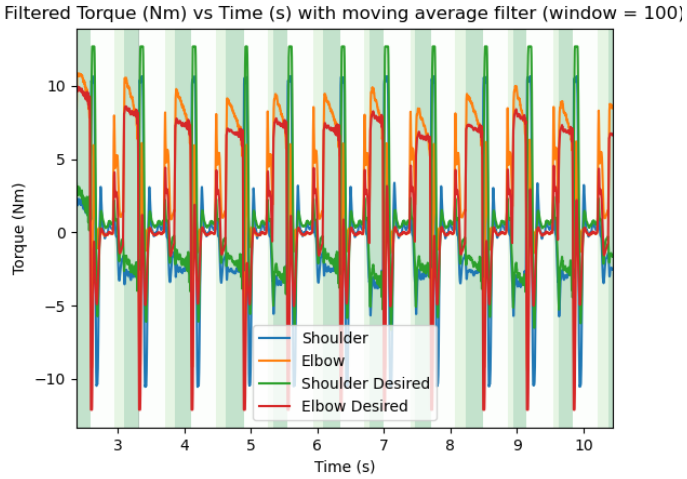


Figure 6.10: Filtered torque of the energy shaping control with a desired height of 0.5 m. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

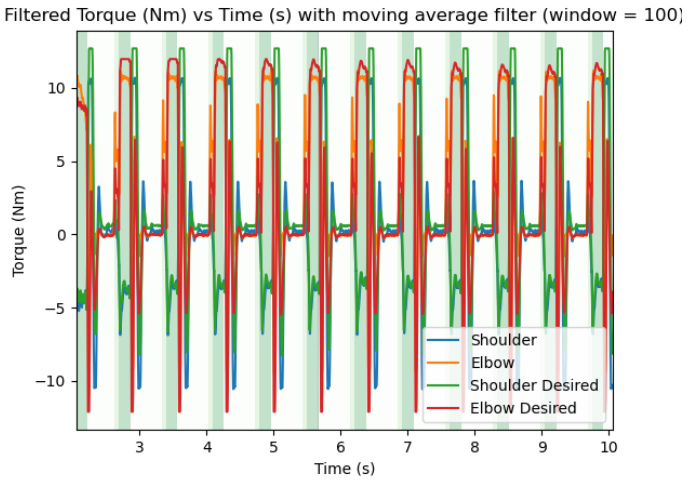


Figure 6.11: Filtered torque of the energy shaping control with a desired height of 0.6 m. The background colour indicates the current phase of the state machine. The flight phase is white, the touchdown phase is light green and the lift-off phase has a dark green background colour.

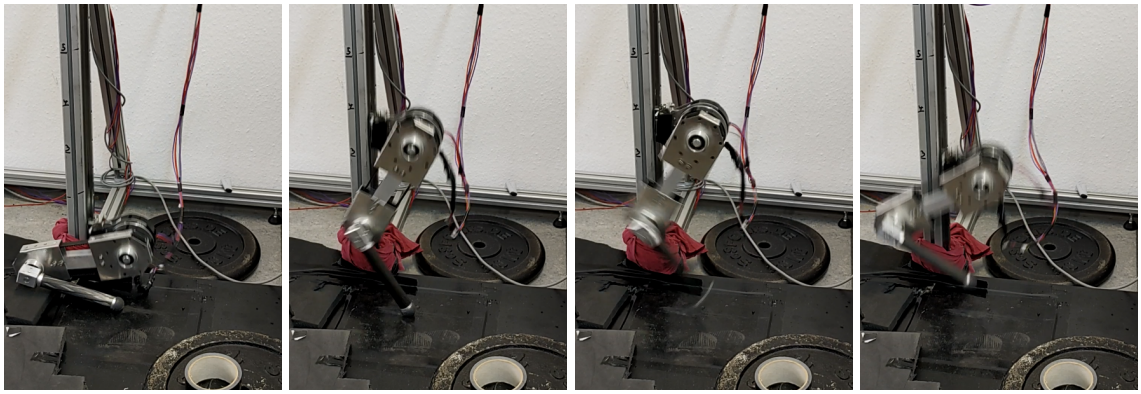


Figure 6.12: Jumping sequence for 0.4 m desired height

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

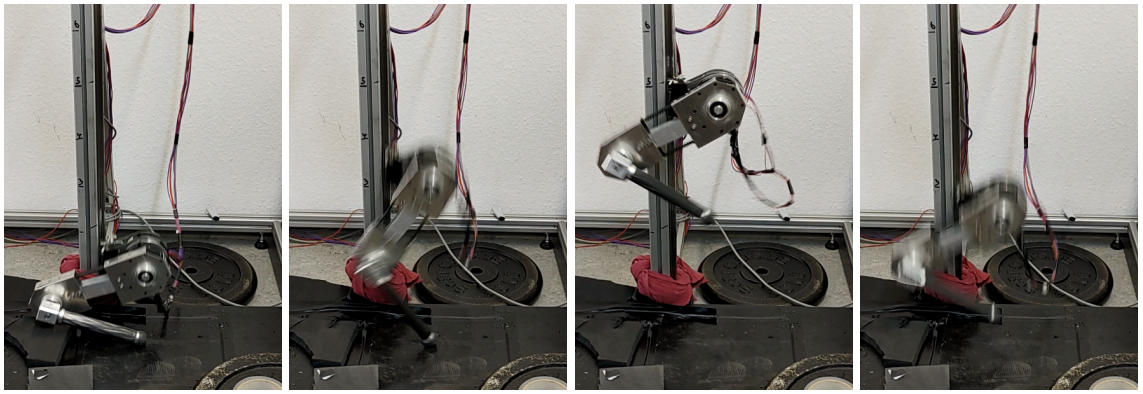


Figure 6.13: Jumping sequence for 0.5 m desired height

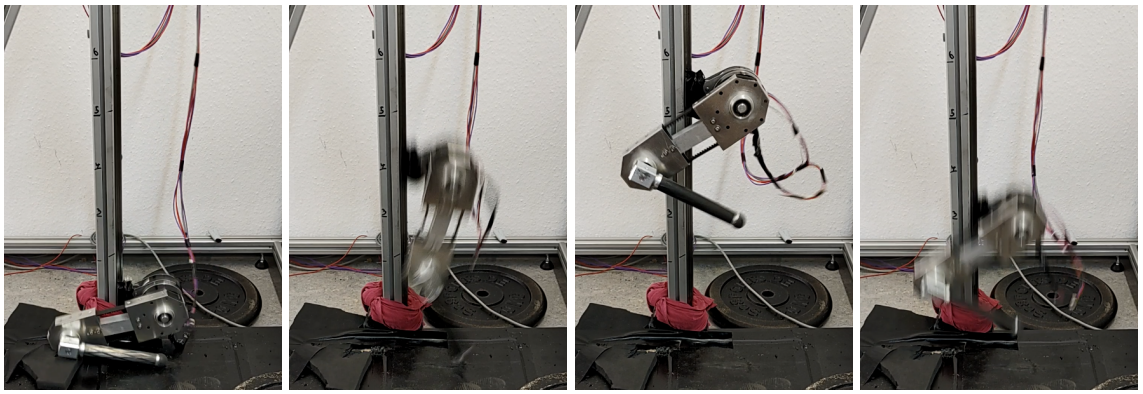


Figure 6.14: Jumping sequence for 0.6 m desired height

Robotics Innovation Center
Robert-Hooke-Str. 1
28359 Bremen
Deutschland
robotik@dfki.de

6.3 Backflip control

In simulation, energy shaping control has been implemented inclusive backflip support. This backflip support adds a trajectory to the current desired state and can follow arbitrary sequences of backflips, forward flips and normal jumps. In our simulations, we found that this backflip control works quite stable. In Figure 6.15 you can see the sequence of an exemplary backflip in simulation.

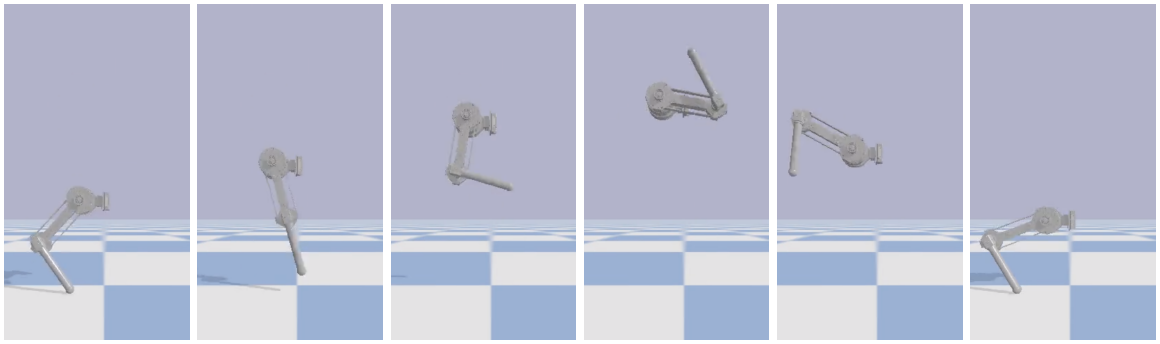


Figure 6.15: Simulated sequence of a backflip.

In the real system, we performed a backflip as well. Here we faced more issues due to the wiring, since the wires are likely to twist around the screws in the hopping leg. Hence, we lost the connection during our backflip. Nevertheless, we managed to do a full backflip. An image sequence of this is printed in Figure 6.16. In future, it would be nice to try a sequence of forward- and backflips, as we have already implemented in simulation.

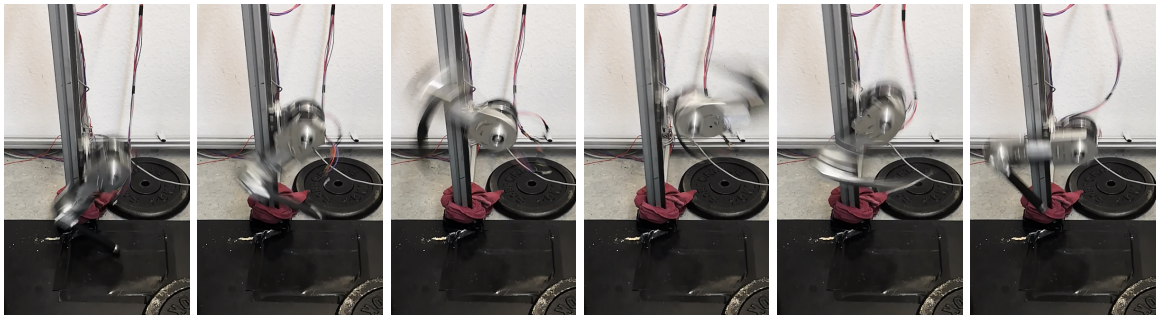


Figure 6.16: Sequence of a backflip on the real system.

BIBLIOGRAPHY

Di Carlo, J. (2020). *Software and control design for the mit cheetah quadruped robots.*

Tedrake, R. (2021). Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation (course notes for mit 6.832). Downloaded on 28.07.2021 from http://underactuated.mit.edu/.