CTU

CZECH TECHNICAL
UNIVERSITY
IN PRAGUE

Faculty of Electrical Engineering
Department of Cybernetics

Bachelor’s thesis

High-Fidelity Modeling of Hexapod Walking Robot Locomotion

Nguyen Minh Thao

May 2019
Supervisor: Doc. Ing. Jan Faigl, Ph.D.
Supervisor specialist: Ing. Petr Čížek

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Nguyenová Minh Thao**	Personal ID number:	**466018**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Control Engineering**		
Study program:	**Cybernetics and Robotics**		

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

High-Fidelity Modeling of Hexapod Walking Robot Locomotion

Bachelor’s thesis title in Czech:

Model pohybu šestinohého kráčejícího robotu

Guidelines:

1. Familiarize yourself with the hexapod walking robot platform of the Computational Robotics Laboratory and its locomotion control [1].
2. Design a prototype of a new hexapod robot platform based on the experience with the existing platform and given specifications. The robot has six legs each with three joints in yaw-pitch-pitch configuration actuated by the Dynamixel XL-320 servo motors. The frame of the robot is 3D printed. The robot has a symmetric trunk that can carry a control board, e.g., Orange PI.
3. Model the newly developed hexapod robot in the realistic robotic simulator V-REP [2].
4. Implement the communication protocol to control the individual servo motors.
5. Experimentally verify the functionality of the robot using locomotion control, e.g., [3], with the real walking robot and evaluate simulation results and real world deployment.

Bibliography / sources:

[1] J. Mrva, J. Faigl: Tactile sensing with servo drives feedback only for blind hexapod walking robot, Robot Motion and Control (RoMoCo), 2015, pp. 240-245.
[2] M. Freese, S. Singh, F. Ozaki, N. Matsuhira: Virtual robot experimentation platform V-REP: a versatile 3d robot simulator, Simulation, Modeling, and Programming for Autonomous Robots, 2010, pp. 51-62.
[3] R. J. Szadkowski, P. Čížek, J. Faigl: Learning Central Pattern Generator Network with Back-Propagation Algorithm, Conference on Information Technologies - Applications and Theory, Computational Intelligence and Data Mining (CIDM) Workshop Proceedings, 2018, pp. 116-123.

Name and workplace of bachelor’s thesis supervisor:

doc. Ing. Jan Faigl, Ph.D., Artificial Intelligence Center, FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Ing. Petr Čížek, Artificial Intelligence Center, FEE

Date of bachelor’s thesis assignment: **24.01.2019** Deadline for bachelor thesis submission: **24.05.2019**

Assignment valid until: **20.09.2020**

doc. Ing. Jan Faigl, Ph.D. Supervisor’s signature	prof. Ing. Michael Šebek, DrSc. Head of department’s signature	prof. Ing. Pavel Ripka, CSc. Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

_________ . ______________   ______________________________

Date of assignment receipt   Student’s signature

Declaration

I declare that the presented work was developed independently and that I have listed all sources of the information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, May 25, 2019

. . . .
Minh Thao Nguyenová

Acknowledgement

I wish to thank doc. Ing. Jan Faigl, Ph.D., for supervising this thesis, his remarks and advises. Moreover, I would like to thank Ing. Petr Čížek for all the knowledge, guidance and patience with me. I also want to thank Bc. Jan Bayer and Bc, Jiří Kubík, whose mad electronics skills and 3D printing wizardry helped me bring Marvin to life. Most of all, my gratitude goes to my family for their intensive support during my studies, never letting me procrastinate as much as I wanted to, and for feeding me all these years, even though it still has not turned out to be a reasonable investment.

Abstrakt

Věrohodnost simulace chování robotických prostředků je důležitým faktorem v návrhu řídících algoritmů i samotném řízení autonomních robotických prostředků, protože pomáhá omezit rizika spojená s testováním algoritmu v podmínkách reálného experimentu. Tato práce se zabývá návrhem nového šestinohého robotu a metodikou realistické simulace jeho chování pro účely ladění algoritmu pohybu a sběru proprioceptivních dat. V práci je představen návrh nového šestinohého kráčejícího robotu, postaveného z běžně dostupných komponent a speciálně navržených 3D tištěných součástí. Dále je navržena metodika pro modelování dynamického chování šestinohých robotů v realistickém robotickém simulátoru. Tato metodika je poté aplikována pro modelování nově navrženého robotu a jeho předchůdce PhantomX robotu v simulátoru Virtual Robot Experimentation Platform (V-REP). Výsledný model je poté aplikován pro nalezení parametrů chůze řízené pomocí chaotického oscilátoru optimalizující její rychlost za omezení daných podmínek, např. maximálního momentu servopohonu a požadavku na bezkolizní pohyb jednotlivých nohou.

Klíčová slova: Kráčející robot, Modelování robotu, Dynamická simulace

Abstract

High-fidelity simulation of mobile robotic platforms is a crucial tool to enable verification of control strategies and algorithms prior their deployment on the real robots. In this thesis, a design of Marvin, a lightweight experimental six-legged walking platform built of off-the-shelf components and custom designed 3D-printed parts, is introduced. Furthermore, a methodology for high-fidelity modeling of six-legged robots in a realistic robotic simulator is proposed, that is used to construct the model of the Marvin and PhantomX robots in the Virtual Experimentation Platform (V-REP). Models of both the PhantomX and Marvin are then utilized to adjust parameters of a Central Pattern Generator (CPG) dynamic locomotion controller to improve the robot locomotion in terms of the robot forward velocity and stability, with respect to the robot's physical limits including the maximum torques and self collision-free execution.

Keywords: Legged robot, Robot modeling, Dynamics simulation

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Mobile robotics is a rapidly evolving subfield of robotics and information engineering with many real-world applications. Unlike industrial robots, mobile robots can move around and are not fixed to a certain location, which considerably extends their possible usage in the field. We recognize three types of land mobile robots - wheeled robots, walking robots, and tracked robots. Legged robots, often inspired by insects, have aroused great interest among the robotics community in the last few decades. An example of three legged robots used in Computational Robotics Laboratory is shown in Figure 1. While wheeled and tracked robots are usually able to carry heavier loads, legged robots are superior to them in terms of terrain negotiation. Walking robots are generally able to traverse larger variety of rough terrains and overcome dimensionally-proportionate obstacles in a non-detrimental way. Nevertheless, this ability is at the cost of higher morphology complexity, regarding the number of degrees of freedom (DoF). For comparison, in case of a car-like wheeled robot, there are two controllable DoF, while our six-legged robot with three actuators per leg has 18 controllable DoF in total. It is problematic to model and control such systems using analytical methods. Therefore, different approaches in motion control are required. One of the options is using dynamics simulation systems, often referred to as *physics engines* to develop control strategies of walking robots.

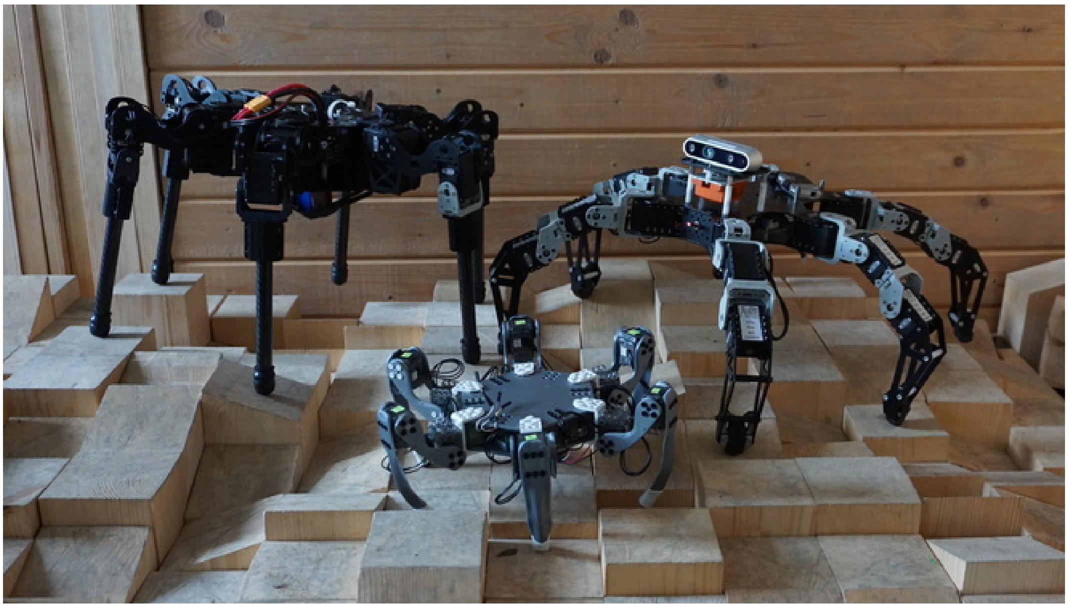


Figure 1: Walking robots of Computational Robotics Laboratory - Marvin (front), HAntR (left), and PhantomX (right).

With the development of robotics and its application in the real world, the role of simulations has become increasingly important. They have become essential tools for design, motion planning, or developing control strategies of robotic systems. A high-fidelity model enables us to verify the feasibility of actions prior to their execution by the real robot so as to prevent mission failure resulting in fatal damage to the robot. It can also be used to run multiple experiments at the same time, which can considerably speed up the research. Large datasets can be created for other purposes, such as training neural networks [1], motion planning [2], localization, etc., which is usually very demanding in terms of time and power consumption or needed space and equipment when performing experiments on real robots. Additionally, the simulation provides ground truth data, often missing when performing experiments on the real robot due to inaccuracy or absence of adequate sensors. Such data is crucial for algorithm true efficiency evaluation when developing motion planning techniques and advanced locomotion strategies [3]. This thesis aims to create a high-fidelity model of a walking six-legged robot in a realistic robotic simulator, namely Virtual Experimentation Robotic Platform [4] (V-REP). Our approach is firstly deployed on an established PhantomX hexapod, originally created by Trossen Robotics. The PhantomX model can then be used to develop locomotion strategies [5] or to generate large datasets for the development of localization strategies and terrain mapping algorithms. Furthermore, it can be used in education [6], as the simulator is cross-platform, easy to use, and open-source, to introduce students to robotics using a safe and inexpensive method.

Next, the design of a new six-legged walking robot named Marvin is proposed. The main purpose of this robot should correspond to a remote mobile sensor with terrain negotiation ability to collect data or serve the purpose of a remote networking mote in hardly accessible areas. Therefore, it is supposed to be lightweight with smaller dimensions than PhantomX to ensure traversability even in tight spaces. Further, the robot body is centrally symmetric. This attribute allows the robot to be truly omnidirectional, unlike PhantomX, whose rectangular shape implies the use of crab walking or turn [7]. The overall cost does not outreach 400 $, which is also a positive aspect, considering possible loss or damage of the robot in hazardous environments. The designed robot has three joints per leg, 18 joints in total, actuated by Dynamixel XL-320 servomotors. The structural parts of the robot were designed using a Computer Aided Design system (CAD), specifically Fusion 360, 3D printed and assembled with the servomotors to build the first prototype, which is then modeled using the methodology deployed on the PhantomX robot, which allows us to verify both the methodology versatility and performance of the designed robot.

The rest of this thesis is organized as follows. Chapter 2 specifies the task we are approaching with minor background information. A brief overview of works, which are most related to ours, is presented in Chapter 3. Chapter 1 describes the simulation platform used for modeling of the robots, which are both described in Chapter 5 with special focus on the characterisation of the designed Marvin robot, as its documentation is not yet accessible elsewhere. In Chapter 6 we explain selected verification scenarios used in our methodology for modeling of legged robots. The overall results of the modeling process are shown in Chapter 7. Chapter 8 sums up and evaluates our goals along with the results achieved. A discussion of the results and possible future work extensions are suggested as well.

Chapter 2
Problem Statement

The addressed problem is to develop a model of a walking six-legged robot in a realistic robotic simulator. Fidelity of the created model lies in similarity of data obtained in simulation with those measured on the real robot. We evaluate this similarity simply as the Mean Square Error of the simulated signals. This thesis focuses on proprioceptive data, which comprises information about robot's own state, particularly joint position and torque, in the case of PhantomX and a newly constructed Marvin robot.

Although both Dynamixel AX-12A and XL-320 actuators provide the possibility of reading the present load, this value is based on an internal output value and not directly measured by any torque sensor. It is computed from the instantaneous position error using a linear relation [8], [9]. Additionally, this value is updated every 130 ms, which is not frequent enough for observing the motor torque during dynamic actions. Figure 2 shows the comparison between reading the position and the load of the servo. Both data belong to the same servo and were obtained during the same motion, but it can be observed, that the updates of the motor load are less frequent, which makes the reading of load values unusable especially during dynamic actions.

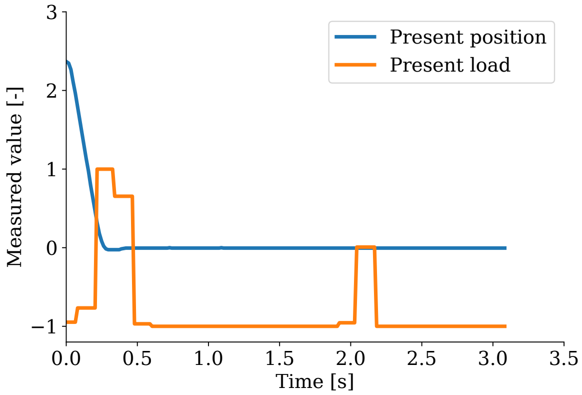


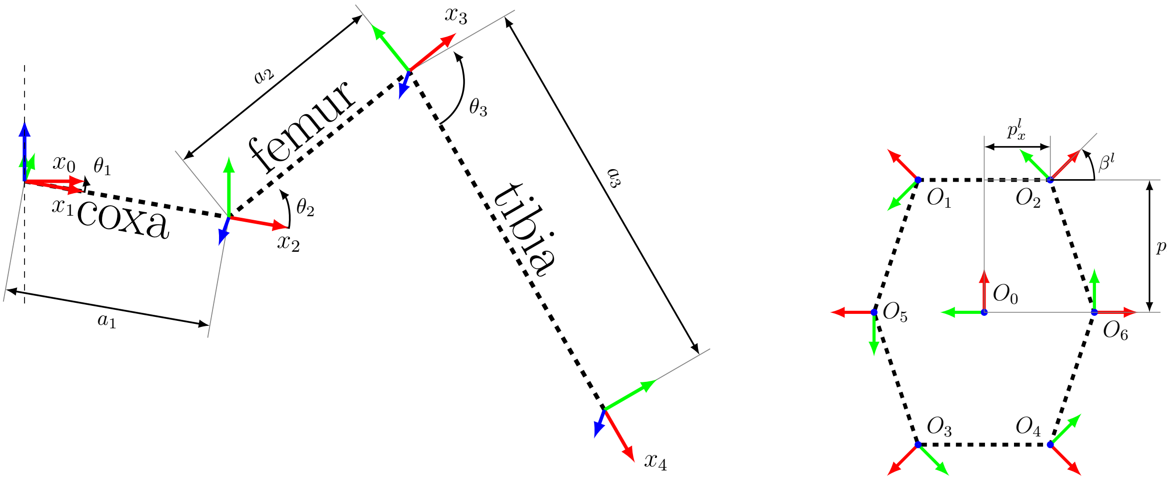
Figure 2: Measuring present position and load of an XL-320 servomotor.

Therefore, we are searching for a way to estimate such data off-line, e.g., prior to real-world experimental deployment so as to minimize the risk of damage to the robot. We propose a method of creating, enhancing and consecutive verification of a six-legged robot model in Virtual Experimentation Robot Platform (V-REP) simulator.

The modeling methodology is firstly deployed on a PhantomX-like hexapod robot. Afterwards, it is applied to a newly constructed Marvin robot with similar properties, such as the number of legs, number of joints per leg, their configuration, and characteristics of the locomotion gait. The Marvin platform designed according to requested parameters, specifically symmetrical robot trunk, lightweight construction, used servomotors and control board while conserving the joint configuration from PhantomX. The design, mechanical construction, hardware and software implementations are further described in Chapter 5. The modeling process with all its stages is described in Chapter 6.

2.1 Hexapod anatomy

Each modeled robot is an electrically actuated hexapod robot built from off-the-shelf components. The robot consists of the base trunk with the control unit and six attached revolute joints moving around a vertical axis. Each leg is attached to a single joint and is formed by two more linked revolute joints moving around a horizontal axis. Parts of each leg are named coxa, femur, and tibia as it is visualized in Figure 3a, in which parameters and dimensions of each part are as specified as well. Parameters of the robot body are depicted in Figure 3b. This convention is used in a more detailed description of each robot in Chapter 5.



(a)

(b)

Figure 3: Schematic diagram of the hexapod robot (a) leg and (b) body.

2.2 Locomotion control

Regarding the complex morphology of legged robots, given by 18 controllable degrees of freedom (DoF) in our case, the robot locomotion control is a complex task. A locomotion strategy is needed for the robot to move in a certain direction. One type of these strategies is predefining motion patterns, called *gaits*, which state the motion of the robot. Locomotion gait defines the trajectory of the robot legs and their mutual synchronization. Each leg can be in either swinging stage, when it is lifted and moved to a new position, or supporting stage, when it supports the robot body, while other legs are in swinging stage [10]. All legs then alternate between swinging and supporting stage, which leads to the overall transit of the robot. According to the number of legs, which are in the supporting stage at the same time, we distinguish tripod, tetrapod and pentapod gaits [11]. In tripod gait, there are always three legs supporting the body, while the other three legs are swinging from their previous position to the following one. The triplets then alternate as the robot proceeds in its path. In tetrapod gait, two legs can be in the swing stage at once. And pentapod gait always moves only one leg at a time, which remains the most stable, although least efficient locomotion strategy in terms of power consumption and overall robot speed.

In this thesis, we utilize the deployed model to optimize a parametrizable tripod gait controlled by a Central Pattern Generator (CPG) Network [12], which is desribed more thoroughly in Section 6.4. We search for such setting of parameters, which speed up the locomotion concerning the robot's physical limits, including maximum joint torque and leg collision avoidance.

Chapter 3
Related work

The evolution of Computer Aided Design systems (CAD) and robotic simulations has remarkably facilitated prototyping of walking robots [13]. Different approaches are utilized in design of legged robots. The design constraints are determined by various factors - the robot working environment, application, accessible materials, morphology complexity, overall cost, etc..

While four-legged robots, such as StarlETH [14] and LittleDog [15], outstand in terms of dynamic locomotion control, often inspired by mammals, six-legged robots benefit from increased stability, which is characteristic for insects. Hexapod locomotion gaits are statically stable, as the robot is always supported by three to or more legs, that are in contact with the ground. Besides, in case of partial structural damage or limb loss, hexapod robots can still continue in locomotion with interesting performances [1], whereas for quadruped platforms, such situation would be precarious.

Six-legged robots are often inspired by insects in terms of both construction [13] and locomotion control [5]. Typically insect-like leg design comprises yaw-pitch-pitch joint configuration as in case of Ragno and Messor II by [13], and Anton [16]. It is also applied in the series of commercial Bioloid robotic kits developed by Trossen Robotics with the PhantomX, popular among hobbyists and also used in research. Such design allows the robot to walk on reasonably rough terrain with minor obstacles.

In order to surpass more challenging terrains with larger obstacles, legs with more than three DoF are designed so as to enable to set not only foothold position, but orientation as well, which is implemented in LAURON V [17] and HAntR [18]. Additional DoFs can also be used for other tasks, such as object manipulation, for example, in LEMUR II platform [19]. Solutions with advanced leg design, however, require more sophisticated control methods to fill the design potential.

The opposite approach of simplifying the leg design has also been observed. An example of a legged robot with relatively simple morphology is RHex [20], which can overcome an impressive variety of rough terrains. However, since it only has 1 DoF per leg, it is not able to modify leg posture and adjust to the terrain so as to reduce the damage to the used path.

In the past few years, a breakthrough in 3D printing technologies has reinforced custom robot design development. Despite the limited materials, such method provides an advantage of possible redesign or replacement of robot components at a low cost, whenever required. A fully 3D printed four-legged robot has been designed by [21]. Their Metabot is a low-cost legged robotics platform for education. It has 12 joints actuated by Dynamixel XL-320 servomotors, for which they provide a programming environment based on Scratch programming language. A similar robot with six legs, Hexabot, has been developed for desert ant-like navigation tasks [7]. It is based on the same actuators and leg anatomy as Metabot. Hexabot is able to walk at the maximum speed of 35 cm/s. The body of Hexabot is of hexagonal shape, which enhances its direction changing aptitudes.

Another platform, which takes advantage of 3D printed design is Aracna quadruped robot [22]. However, due to the unconventional kinematics and counter-intuitive motion control, it is not suitable for use in real-world applications. Nevertheless, it provides means to develop and test evolutionary algorithms in robotics, as suggested by [23].

Besides CAD software, robotic simulations help designing new robots by analyzing their physical properties during locomotion. They are also widely used for development of locomotion control strategies. There are many commercial and open-source robotic simulators available. According to [24], most popular simulators for legged robots are Webots, Gazebo, Robotics Studio, USARsim, SimRobot, V-REP. All these simulators provide the simulated robot with a good amount of both exteroceptive and proprioceptive signals. Exteroceptive sensing has been modeled by [25], who have designed a six-legged walking robot for mine detection and mapping of minefield by creating a simulation model with an optical proximity sensor attached to each leg. However, high-fidelity simulation of proprioceptive signals is, to the best of our knowledge, an area not commonly observed in the robotic community so far. Proprioceptive sensing provides the robot with information about its own state, e.g., power consumption, joint torques and positions, overall posture, etc. Such feedback is crucial for precise robot motion control. They are used, for example, by [26], where joint position feedback from Dynamixel AX-12A servomotors is used for terrain classification by a modified PhantomX hexapod robot. A similar approach is taken by [27] in terrain characterisation and gait adaptation by a hexapod robot, who suggests a method to characterise terrain by a six-legged walking robot. They use a modified version of PhantomX to read torques from Dynamixel AX-12A actuators, which is equivalent to reading joint position. The provided load feedback is only calculated from an internal position error value and is not based on any torque sensor measurement [8]. Besides terrain characteristics, the proprioceptive signals can be used for terrain negotiation to increase the robot autonomy in rough terrains as in [28].

Although usability of proprioceptive sensing cannot be doubted, the simulation fidelity of the proprioceptive sensing has not yet been evaluated with the focus on legged robots. Authors of [29] use a hexapod robot simulation by Solidworks to propose an application of a genetic algorithm to generate a locomotion gait for adaptation to robot structural damage. Similarly, [30] have worked on automated damage diagnosis and recovery for remote robotics. They have simulated the situations using Open Dynamics Engine (ODE) with one quadrupedal and one hexapedal robot. However, they rely on quasi-static kinematics of the locomotion and base the diagnosis on analysis of geometric displacement of the robot. An evaluation of multiple robotic simulation tools has been made by [31], who propose universal tests and metrics to compare their performance. Another general evaluation has been proposed by [32], which compares six different open-source physics engines. [33] have even focused on dynamics of the walking robots, in their comparison of ODE, PhysX and Bullet physics library. Our physics engine choice was based on the results of these evaluations with respect to the most important simulation aspects for legged robots, such as constraint handling and friction model to ensure simulation stability and minimizing leg slippage.

Chapter 4
V-REP simulator

The Virtual Robot Experimentation Platform (V-REP) is a popular versatile 3D simulator based on a distributed and modular approach [4]. Each object of the scene, can be controlled, according to the demands on dynamics and accuracy, individually, which allows significant speed-up during the simulation by avoiding unnecessary computations. This chapter provides a description of simulation characteristics and adjustable parameters, which were used during the modeling of our robots.

4.1 Object types and properties

Among the variety of scene objects in V-REP, we particularly use Shapes and Joints for robot modeling. There are many other types of scene objects useful for simulation of all kinds of robotic experiments, such as paths, cameras, lights or graphs. However, in this thesis, we will not elaborate on them.

4.1.1 Shapes

Shapes can represent any physical object kinematically or dynamically. They are used to visualize rigid bodies, e.g., robot links, eventually for calculation of dynamics, motion, collision detection, path planning, etc.. Simple shapes can be either pure, convex or random. Pure simple shapes are primitive bodies, such as cuboids, cylinders or spheres (see Figure 4).

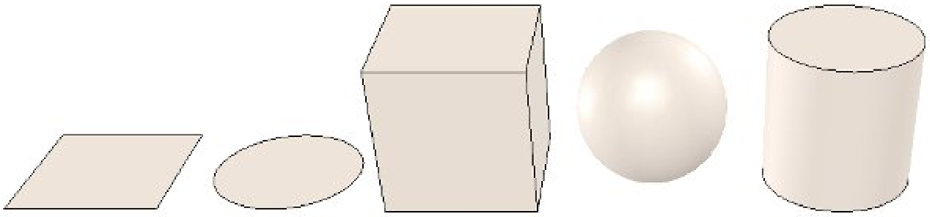


Figure 4: Types of primitive shapes in V-REP. Courtesy of [34].

Convex shape represents any convex body and is defined by a convex meshgrid. Random shapes can be defined by any meshgrid. Simple individual shapes can then be grouped into compound shapes or merged into a single simple shape. Grouping operation preserves properties of pure shapes, if and only if all components are pure shapes. Merging, on the other hand, usually leads to creation of non-pure simple shapes (see Figure 5).

It is possible to adjust or enable multiple characteristics of each shape, most importantly respondability, dynamics, and collidability. Static properties, such as proportions, can be set for any shape; however, dynamic properties (e.g. mass, inertia, friction) are specific for objects with enabled dynamics only. It is recommended to enable dynamics computation only for pure shapes (not meshes), as non-pure dynamic shapes slow down the simulation drastically.

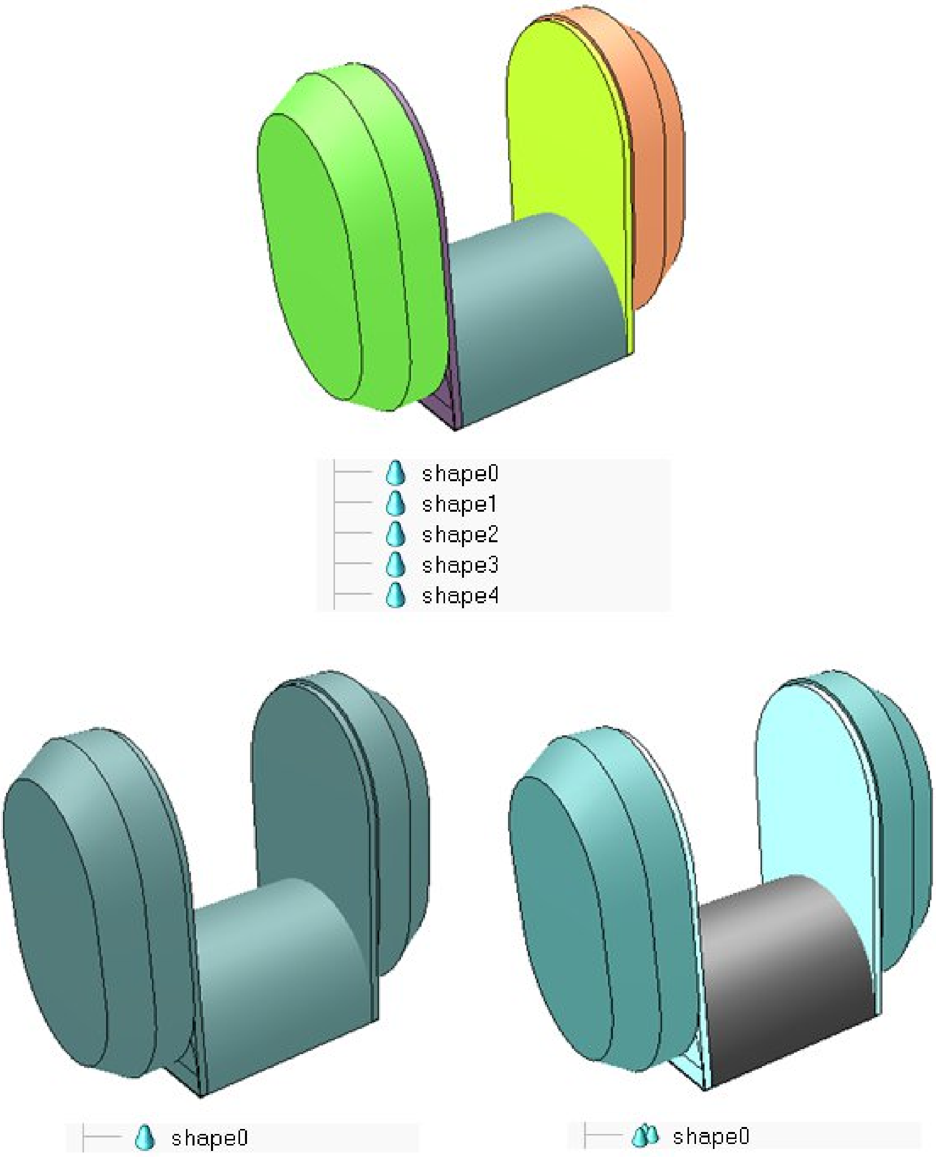


Figure 5: Example of grouping and merging shapes: (i) 5 individual convex shapes, (ii) 5 shapes merged into one random shape, (iii) 5 shapes grouped into one compound convex shape. Courtesy of [34].

4.1.2 Joints

Besides separate physical bodies, it is possible to model their dependency and linkage using joints, which bring new degrees of freedom to the simulation. Unlike other types of objects, a joint has multiple reference frames. The first one is a fixed reference frame, the other one moves relatively to the first reference frame according to the joint coordinates. There are four types of joints, that can be modeled in V-REP - prismatic, screws, revolute and spherical (see Figure 6).

Joints can operate in 5 different modes, which have a major influence on joint control. There are passive mode, inverse kinematics mode, motion mode, dependent mode and torque/force mode. The torque/force mode enables the user to set the joint target position, the simulator then performs a realistic motion from the current position to the target position according to the joint properties and applied constraints, which is suitable for modeling joint actuators. The other modes allow the user to

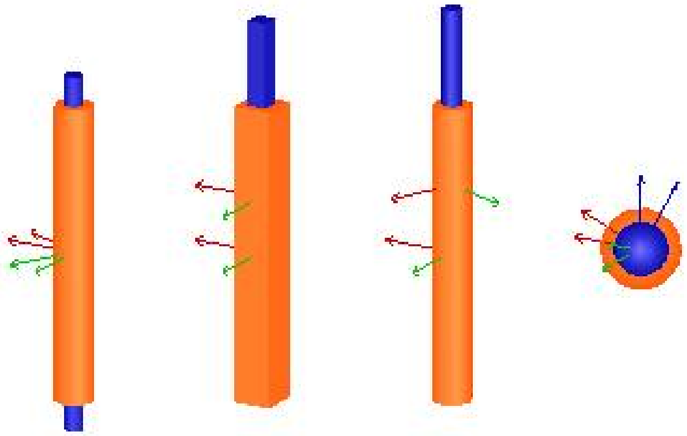


Figure 6: Joint types in V-REP with reference frames: (i) revolute, (ii) prismatic, (iii) screw, (iv) spherical. Courtesy of [34].

modify the joint position directly. The joint position is then set instantaneously without simulating the transition process, which can be used mainly for initial passive joint setup, as it moves dependently on other objects during the simulation.

It is possible to enable motor of the actuator, which corresponds to enabling actuator torque by setting up the motor current in reality. Properties of the motor can then be set, which are specific for each physics engine. The Bullet physics engine allows to set up Normal Constraint Force Mixing (CFM) parameter, Stop CFM and Stop Error Reduction Parameter (ERP) for each joint, which can be used to soften the constraints on the objects, in order to model object stiffness, damping or to increase simulation stability. However, it should be pointed out, that these parameters are not based on any specific physical constraints, which can lead to unrealistic simulation results in some situations [35] and thus should be approached carefully. The inner friction of a joint in this work is also modeled by setting friction, linear and angular in damping the material properties of the corresponding links attached to the joint. Control properties can be enabled for joint with enabled motor and control loop. The simulator supports different control modes of the actuator including, PID control, parametrizable by proportional, integral and derivative parameters, spring-damped mode, parametrizable by spring constant and damping coefficient, and custom control via custom Lua script attached to the corresponding joint. Such a custom solution is beneficial for precise modeling of the actuator dynamics, or implementing custom control rules. Nevertheless, in our case, the built-in PID controller has properties, which correspond to our actuators characteristics and model their behavior with sufficient fidelity.

4.1.3 Physics Engines

Behavior of each dynamic shape is handled by a physics engine. V-REP offers a choice of four different ones - Bullet physics library [36], Newton Dynamics [37], Open Dynamic Engine (ODE) [38], Vortex dynamics [39]. For our purposes, we have chosen to use the Bullet physics engine, also widely used as a game engine. Our choice was based on the accessibility, as the Bullet engine is open-source and it has detailed documentation provided online [40]. The other criteria were the friction model and constraint handling, which are important factors for modeling legged locomotion. The Bullet engine performs well in both friction and constraint handling, as well as their combination, according to comparison in [33]. One of the weaknesses of Bullet lies in collision checking, which is more problematic, than in case od ODE. This has to be taken into account when using the developed model for, e.g., motion planning tasks.

4.2 Simulation control

The V-REP simulator can be accessed via multiple ways; from the simulator GUI, robotic operating system (ROS) or application programming interfaces (API) (see Figure 7).

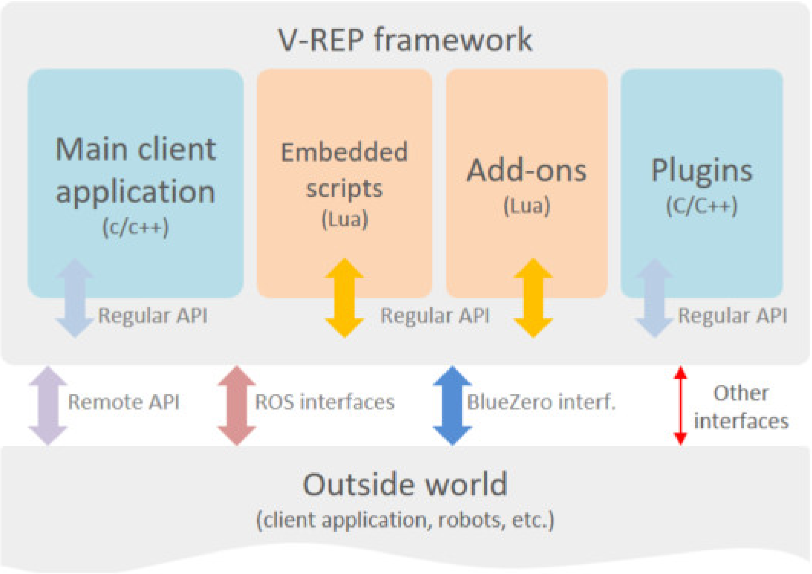


Figure 7: Available interfaces in V-REP. Courtesy of [34].

For our purposes, we use remote API client. It can be written in C/C++, Python, Java, Lua, Matlab or Octave scripts, which can be run simultaneously in a threaded or non-threaded fashion. As for parameters of the simulation, such as visualization speed, physics engine, accuracy level or simulation step time, they can be set directly in the simulator GUI or by a remote API client, e.g., when running simulator in headless mode. Outside control, the dependency of dynamic shapes can be also reached and changed in the simulator GUI or via remote API client.

The simulation can then be run in synchronous or asynchronous operation mode. Asynchronous simulation runs in a separate thread, which leads to faster execution. However, the simulation advances independently on the remote API client, therefore the simulation cannot be precisely defined and interpreted. It is suitable for fast overview simulations to check such kinds of behavior, which do not need to be observed or measured in detail. During the synchronous simulation, the remote API client has complete control over the simulation, as it triggers every simulation step and thus advances the simulation one step at a time. It is therefore possible to simulate time-dependent events, such as actuator communication delays. As a major inconvenience, there is a significant slow down when using this mode, especially with simulation steps smaller than 50 ms, which makes it impossible to use in real time operations. The diagram describing synchronous simulation execution is shown in Figure 8.

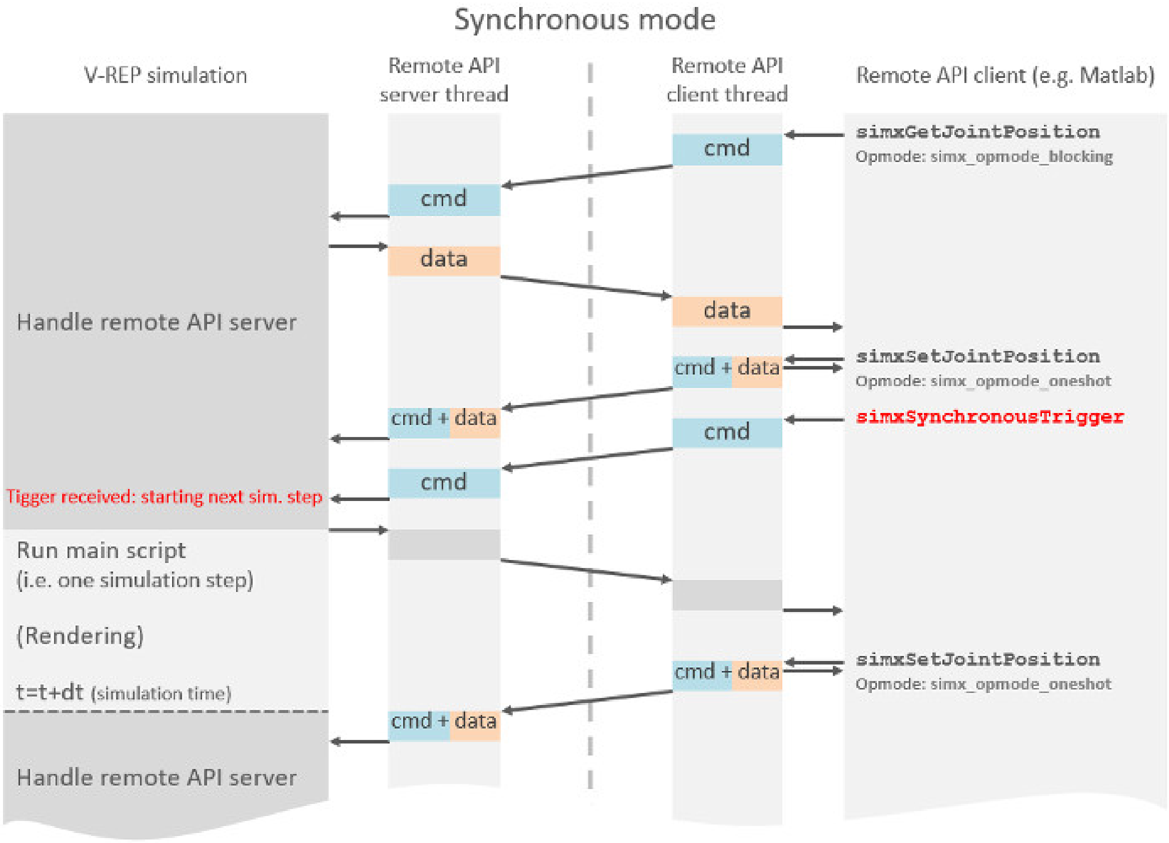


Figure 8: Diagram of synchronous simulation in V-REP. Courtesy of [34].

It is also necessary to specify the operation mode in each function call from a remote API client in order to manage the communication between the remote API client and the V-REP server, which runs the simulation. There are three operation modes - blocking, non-blocking, and data streaming. In blocking function calls, the program waits for the reply from the V-REP server before it returns. Non-blocking function calls simply send data to the server without waiting for any reply. In data streaming mode, the server anticipates the type of data required by the client and keeps streaming such data continuously to the client. The diagrams of blocking and non-blocking function calls in an asynchronous simulation are shown in Figure 9 and Figure 10 respectively.

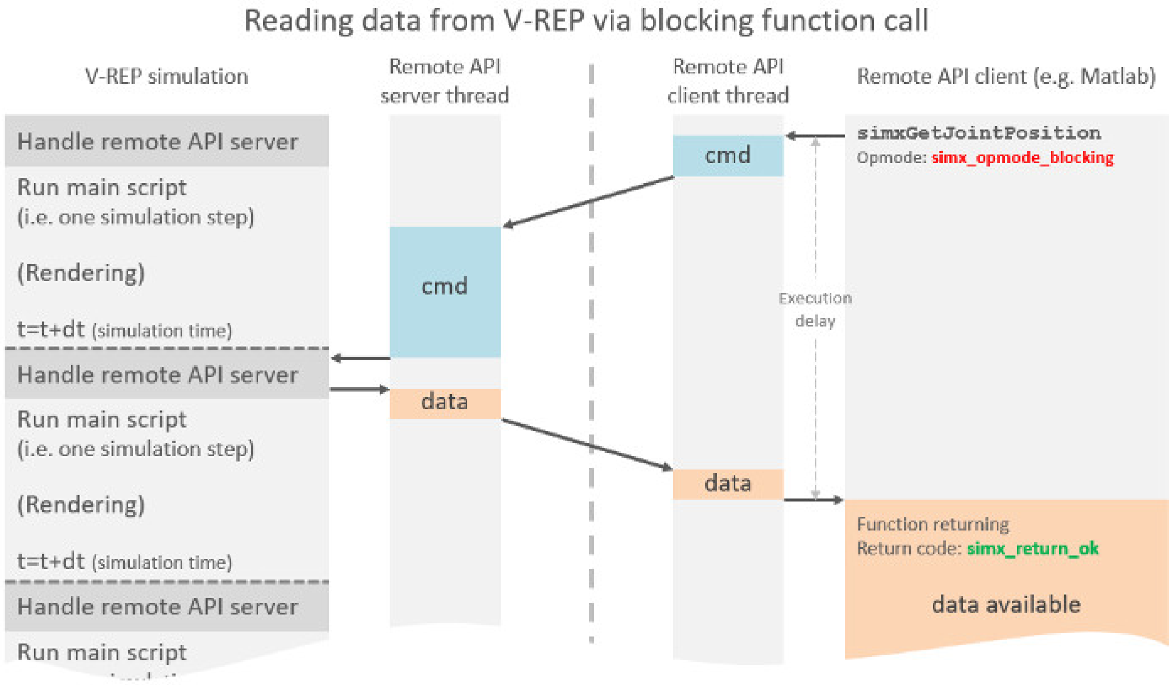


Figure 9: Diagram of a blocking function call. Courtesy of [34].

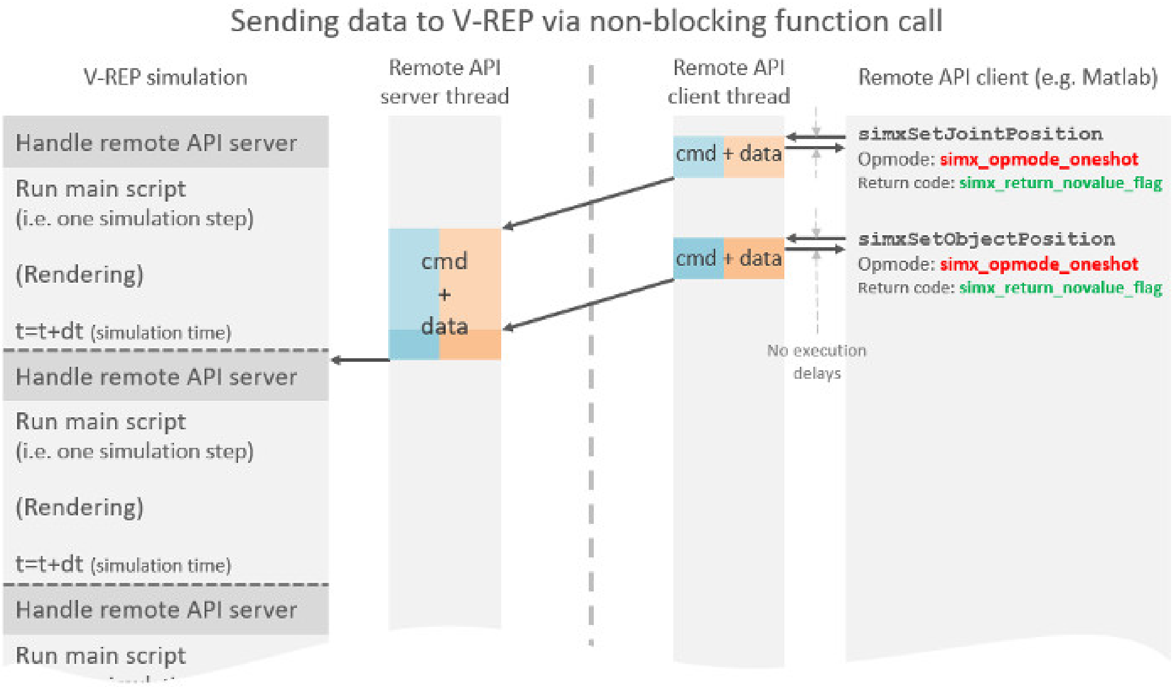


Figure 10: Diagram of a non-blocking function call. Courtesy of [34].

Chapter 5
Modeled Robots

For deployment and demonstration of our modeling methodology, we have performed the process on two different six-legged robots with 18 motorized joints. The first one is a modified version of the popular PhantomX robot manufactured by Trossen Robotics. The other one is a newly designed smaller and lightweight Marvin robot. Both of them are described more in detail in the following sections of this chapter.

5.1 PhantomX

The PhantomX robot mechanical construction derives from a rectangular axially symmetric body with six attached mirrored legs (see Figure 12). The PhantomX robot body dimensions and parameters according to the naming convention from Figure 3b are listed in Table 1. Masses of each link and Denavit-Hartenberg parameters of the robot legs according to the convention from Figure 3a are listed in Table 2.

Table 1: PhantomX body parameters.

l	1	2	3	4	5	6
β^l [rad]	$3\pi/4$	$\pi/4$	$5\pi/4$	$7\pi/4$	π	0
p_x^l [mm]	-60.5	60.5	-60.5	60.5	-100.5	100.5
p_y^l [mm]	120.6	120.6	-120.6	-120.6	0	0
Body mass (without battery)						1212 g
Mass of the battery						330 g

Table 2: PhantomX leg parameters.

Link	i	α_i [rad]	a_i [mm]	θ_i [rad]	d_i [mm]	m_i [g]
Coxa	1	$\pi/2$	52	0	0	22
Femur	2	0	66	0	0	72
Tibia	3	0	138	0	0	104

5.1.1 Hardware

Each joint is motorized by a smart serial Dynamixel AX-12A servomotor, which provides the main controller with position feedback over the half-duplex serial interface (8 bit, 1 stop, No Parity). Basic specifications are listed in Table 3. The photo of the actuator is shown in Figure 11.

Three servomotors in each leg are connected in daisy chain and to the power hub in the trunk. The configuration of actuator numbers is shown in Figure 12.

Table 3: Dynamixel AX-12A specifications

Weight	53.5 g
Dimensions [mm]	$32 \times 50 \times 40$
Resolution	0.29°
Stall torque	1.5 N·m (at 12 V, 1.5 A)
No load speed	59 rpm (at 12 V)
Running degree	$0° - 300°$ or Endless Turn
Voltage	9 - 12 V
	(Recommended Voltage 11.1 V)
Communication speed	7343 bps - 1 Mbps
Material	Engineering Plastic

(a)

(b)

Figure 11: Dynamixel AX-12A actuator (a) front, (b) back. Courtesy of [8].

5.1.2 Software

The robot control is displaced in multiple layers. The low-level layer named Hardware Abstraction Layer (HAL) is responsible for packet communication via serial interface, which sends and receives data bytes from servomotors and is responsible for compiling instruction packets according to Dynamixel Protocol 1.0. Above HAL, there is Robot Control Layer, which commands and reads from actuators according to instructions from layers above. This layer is written in C++ and Python, thus enables higher layers, such as CPG locomotion control program, to communicate with the lower ones using simple Python scripts. One of the drawbacks of used Dynamixel Protocol 1.0 is the fact that while commanding servo position can be done for all servos at once within 1 ms, it is only possible to read their position, where each also takes 1 ms. Therefore commanding all the servos with the new desired positions and reading back all current positions take altogether 19 ms, which is a crucial property that has to be taken into account when modeling the robot dynamics.

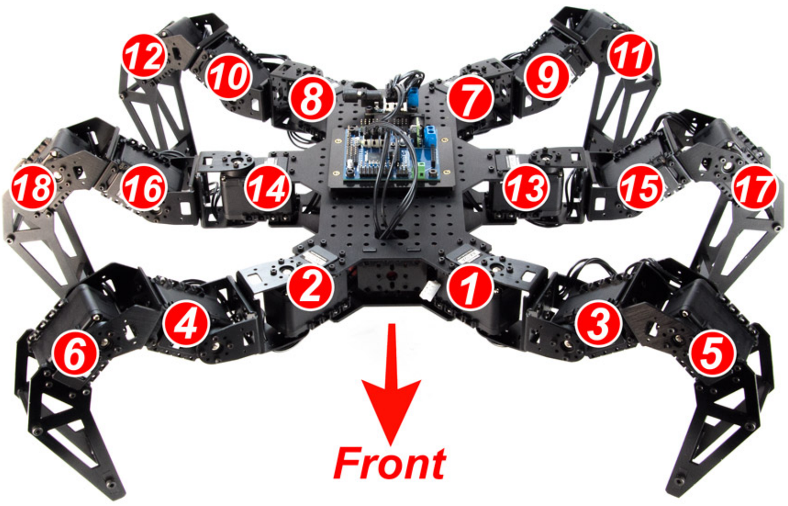


Figure 12: PhantomX actuator layout. Courtesy of [41].

5.1.3 Model properties

The robot model consists of pure shapes modeled according to the original robot parts that have to correspond with the real world proportions and morphology of the real robot. The original proportions are respected with minor geometric simplifications of individual parts to speed up the simulation and avoid unnecessary computations. The dimensions and masses of the individual shapes have been set according to Table 1 and Table 2 that list the properties of the robot leg and trunk, respectively, see Figure 3. The total mass of the real robot is distributed evenly in each of the utilized elementary shapes. The inertia matrices for each link have been computed by the simulator under the assumption of evenly distributed mass; however, the precise inertia matrices could be further set to increase the accuracy.

Next, the individual shapes have been connected to move dependently on each other according to a specific hierarchy, i.e., the kinematic chains are formed with the hierarchical structure starting from the hexapod body and ending in the tibia link. The real hexapod robot, its resulting visual model, and simplified simulation model are shown in Figure 13.

Revolute joints in torque/force control mode with enabled motor and control loop were used to model the actuators because they fit best our scenario. Their properties have been set according to AX-12A specifications listed in Table 3, specifically stall torque, maximum speed, position minimum, and position range. The PID control mode has been selected, as it corresponds to the actuator behavior with positional control.

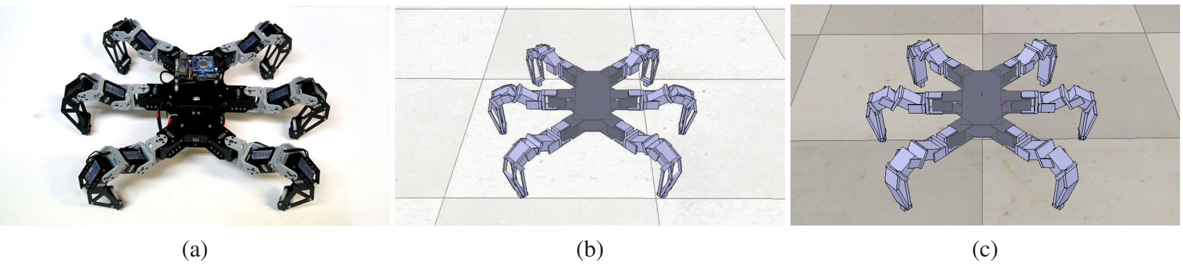


Figure 13: Comparison of (a) real, (b) simulated, and (c) simplified PhantomX robot created from pure compound shapes only.

5.2 Marvin

The Marvin platform has been designed according to the following requirements. The configuration of joints (yaw, pitch, pitch) and links (coxa, femur, tibia) has been conserved from the preceding robot PhantomX. The body has been modeled to be centrally symmetric, which should enhance robots direction changing aptitudes according to [7]. Mass of the robot should not exceed 1kg. The constructed prototype can be seen in Figure 14.

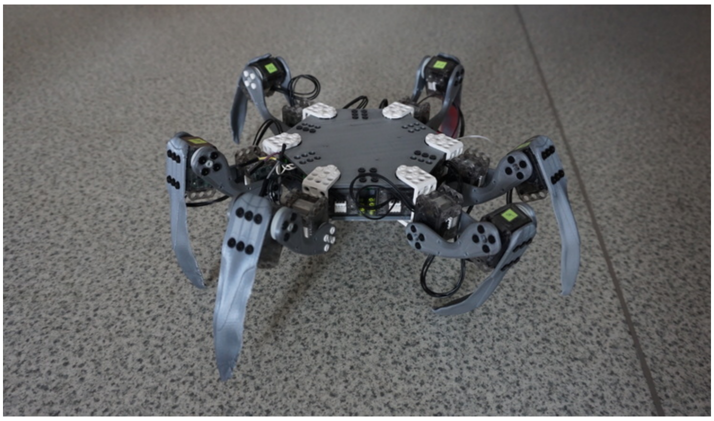


Figure 14: Constructed Marvin prototype.

The robot body consists of two boards in the shape of a regular hexagon, which hold six actuators and all electronics equipment in between.

One robot leg is connected to each corner of the hexagonal boards. Unlike the PhantomX robot, the legs are not mirrored, so that the configuration of joints is the same in all directions, which is more beneficial regarding the central symmetry of the whole robot and possible omnidirectional movements. The joint configuration in robot legs, as well as actuator numbering, has been preserved from the PhantomX so as to keep the portability of software from one robot to the other. Robot body dimensions and parameters according the to the naming convention from Figure 3b are listed in Table 4. Masses of each link and Denavit-Hartenberg parameters of the robot legs according to the naming convention from Figure 3a are listed in Table 5.

All body parts have been designed in the Autodesk Fusion 360 software [42], in which they were as well virtually assembled and the primary simulations were performed to predict the load applied to the material. The curved shape of tibia has been designed in order to minimize the torque on tibia joint, by reducing the angle between tibia vector and the ground reaction force vector in the default position (see Figure 15a). The preliminary static stress simulation in the case of 100 N force applied

Table 4: Marvin body parameters.

l	1	2	3	4	5	6
β^l [rad]	$2\pi/3$	$\pi/6$	$7\pi/6$	$5\pi/3$	π	0
p_x^l [mm]	-28.0	28.0	-28.0	28.0	-56.0	56.0
p_y^l [mm]	48.5	48.5	-48.5	-48.5	0	0
Body mass (without battery)						322 g
Mass of the battery						46 g

Table 5: Marvin leg parameters.

Link	i	α_i [rad]	a_i [mm]	θ_i [rad]	d_i [mm]	m_i [g]
Coxa	1	$\pi/2$	29	0	0	21
Femur	2	0	60	$\pi/2$	0	8
Tibia	3	0	105	0	0	28

to one leg shows the stress displacement in Figure 15b. The overal force of 100N is rather exagerated, while the real servomotors are not able to hold such a force, however, the static stress simulation helps to showcase the potentially weak parts of the robot construction directly in the CAD software.

The mechanical parts of the prototype of the robot have been 3D printed. The used printing material is polylactic acid (PLA) filament, which is a biodegradable printing material mostly produced from corn starch with low melting point temperature. [43] Unfortunately, this material is not available in the Fusion Software, thus load and stress simulations were substitued by Acrylonitrile butadiene styrene (ABS) plastic polymer for computation, which has to be considered when discussing material strength, as well as the layer structure caused by the printing process, which leads to higher fragility in certain directions.

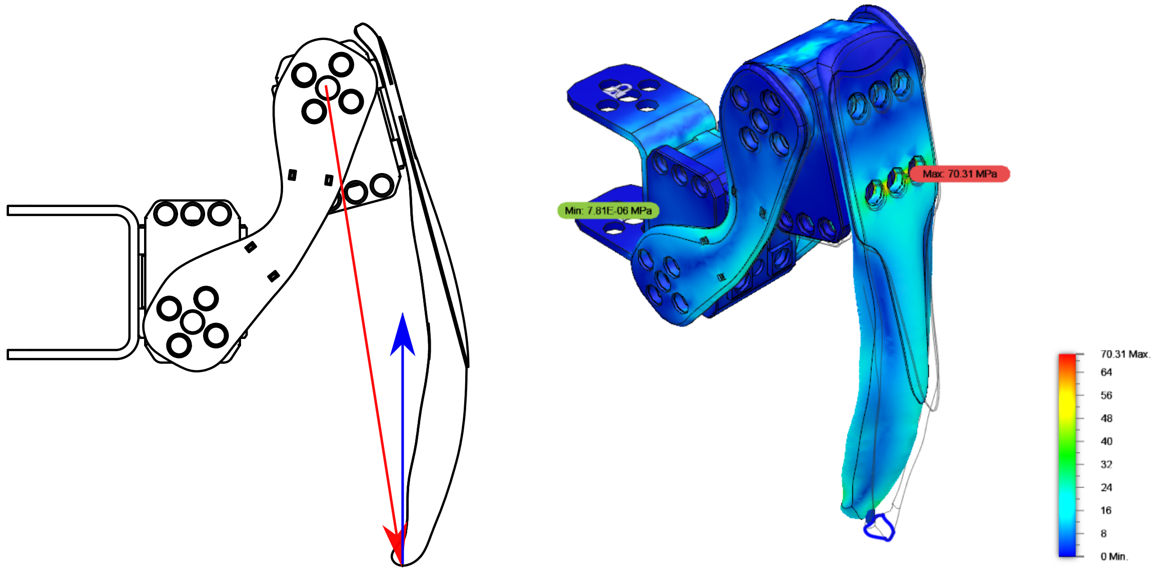
The printed parts were assembled together with the XL-320 servomotors using patented OLLO rivet system. The designed coxas were replaced by pre-manufactured OLLO coxas due to complications during the printing process. The resulting model of the robot together with the built prototype are shown in Figure 16.

5.2.1 Hardware

All 18 actuators are daisy-chained through each leg and connected in parallel to the common power hub, which is connected to USB to TTL converter leading to the main control board. The actuator numbering has been kept from the PhantomX robot (see Figure 12). The robot is powered by two Lithium-ion batteries connected in serial each with nominal voltage 3.7 V, maximum continuous discharge current 20 A and nominal capacity of 2500 mAh. Batteries power the servomotors through their common power hub and the control board through a DC-to-DC power converter, which converts the 7.4 V to 5 V. The diagram of electronics and their interconnection is shown in Figure 17.

Used actuators

All 18 joints are motorized by the smart Dynamixel XL-320 servomotors, which communicate via half duplex asynchronous serial communication (8 bit, 1 stop, No Parity) with adjustable baudrate.



(a)

(b)

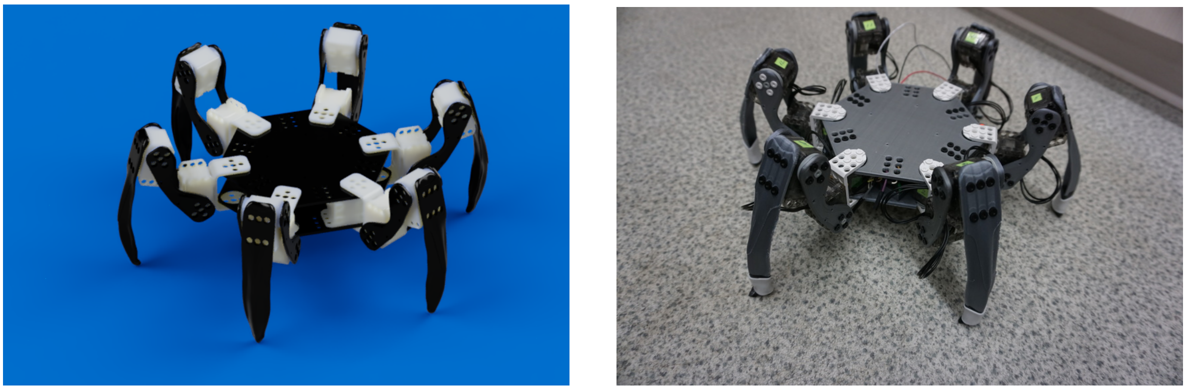
Figure 15: Designed Marvin leg (a) schematic drawing with depicted tibia (red) and ground reaction force (blue) vectors, (b) static stress simulation.

Figure 18 shows the actuator front and back. This choice has been made regarding its small size and weight, high resolution, position feedback, and a built-in safety clutch. The goal position of the servomotor is a 10 bit (1024 steps) value and can be set when using the Joint Mode. The angle range is $0 - 300°$ (see Figure 19). Basic specifications of the servomotor are listed in Table 6.

Table 6: Dynamixel XL-320 specifications.

Weight	16.7 g
Dimensions [mm]	24 × 36 × 27
Resolution	0.29°
Stall torque	0.39 N·m (at 7.4 V, 1.1 A)
No load speed	114 rpm (at 7.4 V, 0.18 A)
Running degree	$0° - 300°$ or Endless Turn
Voltage	6 - 8.4 V
	(Recommended Voltage 7.4 V)
Communication speed	7343 bps - 1 Mbps
Material	Engineering Plastic

From the mechanical point of view, the XL-320 has bindings on both front and back. Unlike AX-12A, it is designed to bind with OLLO rivets mechanism by Trossen Robotics instead of screws, which makes the assembling of the whole robot easier. However, the construction then suffers from high compliance in the connections between joints and links, especially when using OLLO pulleys to bind a link to the back of an XL-320.



(a)

(b)

Figure 16: Assembled Marvin (a) rendered from Autodesk Fusion 360 CAD software, with 3D printed parts colored in black, (b) in reality.

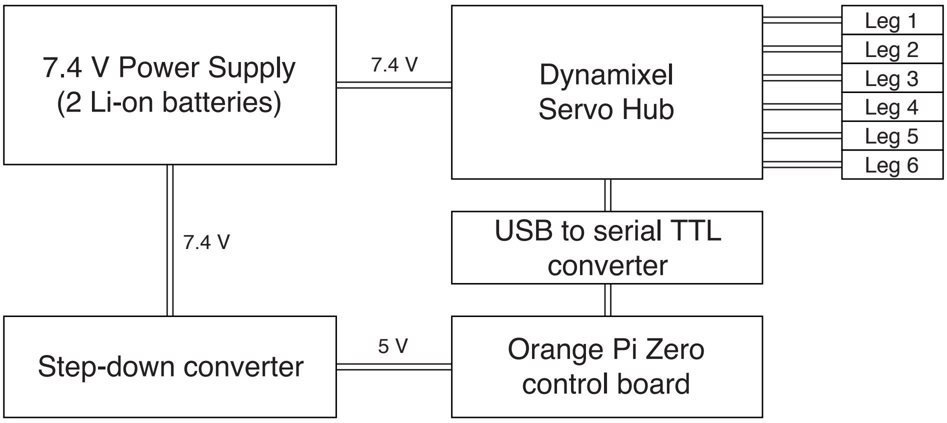
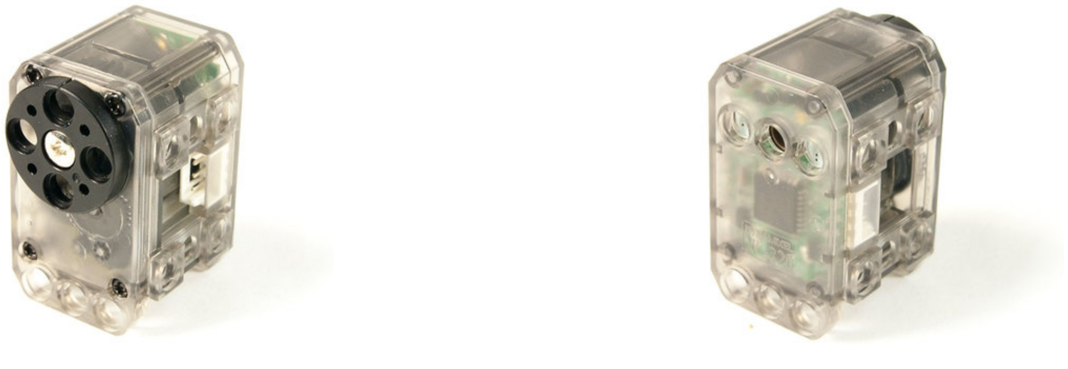


Figure 17: Diagram of electronics interconnection inside Marvin robot trunk.



(a)

(b)

Figure 18: Dynamixel XL-320 actuator (a) front, (b) back.

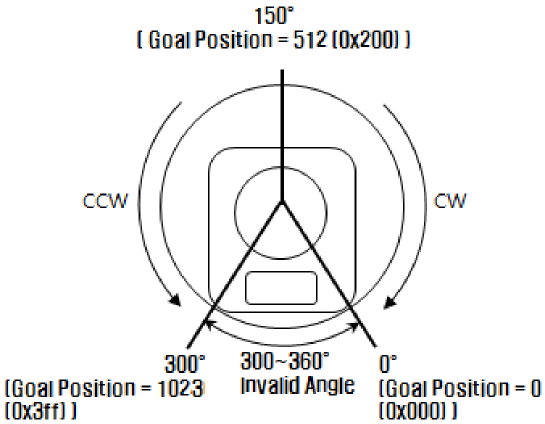


Figure 19: Range of positions of Dynamixel XL-320.

Control board

The robot is controlled by OrangePi Zero control board [44]. It is an open-source single-board computer. Basic specifications are listed in Table 7. The utilized operating system is Armbian Bionic with mainline kernel 4.19.y.

Table 7: Orange Pi Zero specifications.

Dimension [mm]	48 x 46
Weight	26 g
CPU	H2 Quad-core Cortex-A7 H.265/HEVC 1080P
GPU	Mali400MP2 GPU @600MHz
Memory	256MB/512MB DDR3 SDRAM(Shared with GPU)
Onboard Storage	TF card (Max. 32 GB)
Onboard Network	10/100M Ethernet RJ45
Onboard WiFi	XR819, IEEE 802.11 b/g/n
Power Source	Power Supply via USB OTG
USB	One USB 2.0 Host, one USB 2.0 OTG
Supported OS	Android, Lubuntu, Debian

5.2.2 Software

The higher control layers of Marvin are the same as of PhantomX, already described in Section 5.1 only with different set of constants representing joint offsets and robot dimensions. The low-level layer, which communicates directly with the actuators via TTL serial interface, has been implemented so as to ensure compatibility with Robot Control Layer and higher levels.

The communication with servomotors is done by reading and writing to the Control Table in either EEPROM or RAM area. Data in the EEPROM area are maintained even if the device is powered off, whereas data in the RAM area are reset to initial values after restart. Commands are sent to actuators via instruction packets with structure as in Table 8, which correspond to Dynamixel Protocol 2.0 packet communication. Actuators respond via status packets with structure as in Table 9.

The hardware abstraction layer is built on Dynamixel Software Development Kit (SDK), which contains control functions for packet communication, provided by Robotis Inc. It is open-source and

Table 8: Protocol 2.0 Instruction Packet.

Byte number	0	1	2	3	4	5	6
Data type	Header1	Header2	Header3	Reserved	Packet ID	Length1	Length2
Value	0xFF	0xFF	0xFD	0x00	ID	Len_L	Len_H

Byte number	7	8	9	10	11	12
Data type	Instruction	Param	Param	Param	CRC1	CRC2
Value	Instruction	Param 1	...	Param N	CRC_L	CRC_H

Table 9: Protocol 2.0 Status Packet.

Byte number	0	1	2	3	4	5	6
Data type	Header1	Header2	Header3	Reserved	Packet ID	Length1	Length2
Value	0xFF	0xFF	0xFD	0x00	ID	Len_L	Len_H

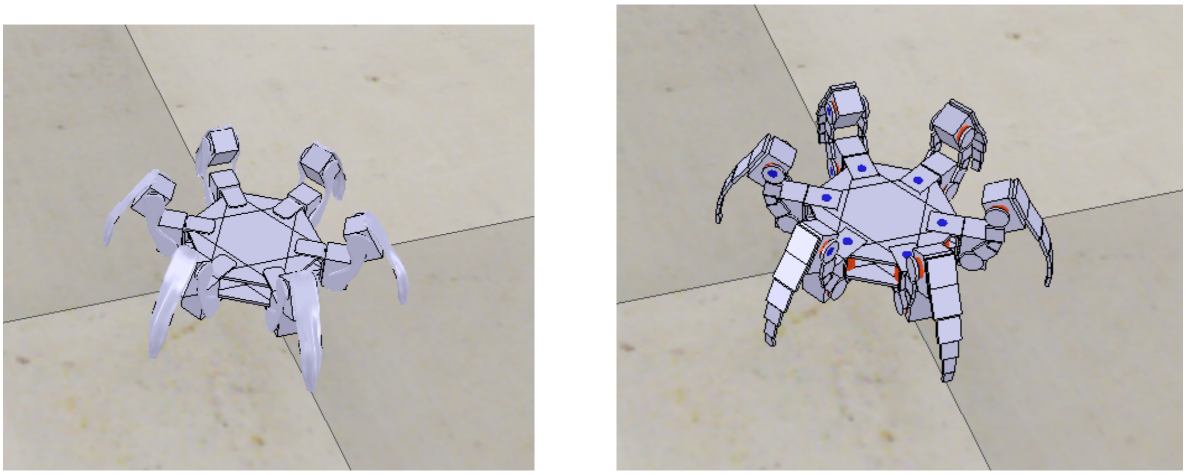
Byte number	7	8	9	10	11	12	13
Data type	Instruction	Error	Param	Param	Param	CRC1	CRC2
Value	Instruction	Error number	Param 1	...	Param N	CRC_L	CRC_H

accessible online [45]. Supported languages include C, C#, Java, MATLAB, LabVIEW, C++, Python, ROS. It can be compiled for three operating systems - Windows, Linux and macOS. For Marvin, we used C++ Dynamixel SDK for Linux operating system with ARM architecture to implement the Hardware Abstraction Layer, which included operations of setting and reading position of servomotors individually with *read* and *write* instructions or all at once using *sync_read* and *sync_write* instructions implemented in the Dynamixel Protocol 2.0. While writing to all 18 actuators takes altogether less than 1 ms, when using either of the methods, reading position takes 3 ms for an individual servomotor and 13 ms for all servomotors using *sync_read* instruction, which has to be taken into account when modeling the robot dynamics.

5.2.3 Model properties

Designed parts were imported into V-REP as mesh objects. Although V-REP supports importing mesh objects, geometric simplifications were necessary to perform the simulation. The links were created as compound pure shapes for dynamics simulation. The main dimensions and masses were set according to Table 4 and 5. The inertia matrices have been computed for each robot link by the simulator under the assumption of evenly distributed mass, as in the PhantomX case. Imported triangular meshes were kept for visualization and collision checking. The hierarchy of all parts has been conserved from the PhantomX robot model as well as the naming convention in order to keep the portability of software for both simulation and real experiments from one robot to the other.

The actuators were modeled as revolute joints in motor and control loop enabled, as in case of PhantomX. Their basic properties, such as maximum joint torque, maximum velocity, position range and position minimum were set according to Table 6. The PID control mode has been selected, as it corresponds to the PID controller implemented in the actuator.



(a)

(b)

Figure 20: Comparison of pure and non-pure model (a) Non-pure model created from imported parts, (b) Simplified pure model.

Chapter 6
Methodology

In this chapter, the developed methodology of creating a high-fidelity model of a walking six-legged robot is presented. As the created robot models in terms of basic parameters: dimensions, masses and link dependency have been discussed in Chapter 5, this chapter focuses on modeling of the actuators characteristics and verifying the overall behavior of the robot afterwards. Section 6.1 contemplates about the dynamic model of the actuator and the initial joint settings, which fit best our purposes, including a simple simulation scenario to verify the basic principle of the simulated joint. Afterwards, the joint parameters are verified or additionally adjusted according to the comparison of real and simulated data. The process is divided into three stages consisting of an experiment on the real robot for measuring real data, a series of experiments in the simulator with a range of adjustable parameters set to obtain data estimation during simulation and finally a comparison and evaluation of such data, which leads to model parameter optimization. The selected verification scenarios are (i) the static torque analysis experiment that further verifies the torque-position error relation; (ii) the dynamic leg movement experiment that verifies the overall performance of the dynamic simulation; and (iii) an experiment on the parametrization of the locomotion controller that supports the overall deployability of the presented model in real-world robotic tasks.

6.1 Dynamic Model of Servomotors

It is a priori necessary to verify the actuator basic behavior principles. The real actuator is composed of the motor and reduction gear which dynamics can be described by the equation [46]

$$J\ddot{q} + B\dot{q} + F(\dot{q}) + R\tau = K\,V, \tag{1}$$

where q is the rotor position angle before reduction, J is the rotor inertia, B is the rotor damping, F is a sum of static, dynamic and viscous friction, R is the gearbox ratio, τ is the servomotor torque, K is the back electromotive force, and finally, V is the motor voltage.

Although a precise model of the servomotor can be obtained by the identification of the parameters [46], the most influencing parts of the model are the parameters of the motion controller and frictions, which, in fact, can be directly modeled in the V-REP as a PID controller and link frictions, respectively. Therefore, there is no need for writing a custom actuator control script as the same behavior can be achieved by a proper parametrization of the existing solutions already provided by the simulator. The particular values of the parameters are obtained by grid-search in the space of feasible constants, with the objective function of the root mean square error (RMSE) between the simulated and real data.

In particular, the PID controller has been parametrized according to the real servomotor and the inner friction of joints has been simulated by setting friction and angular damping in the material properties for the Bullet 2.83 engine for the links attached to each joint.

Note, the fact that the Dynamixel AX-12A servomotor provides only position feedback and uses the P-type controller allows us to estimate the static torque according to the documentation provided by the actuator manufacturer. In the static case, the position error err between the desired and current positions of the actuator is proportional to the joint torque τ according to Figure 21. The stall torque $\tau_{stall} = 1.5$ Nm is reported by the actuator manufacturer, which we use as a hard limit for the torque values in our dynamic locomotion experiment.

As for the Dynamixel XL-320 servomotor, which provides PID control, the integral and derivative parameters were set to zero. Similar behavior to the P-type controller then has been assumed when setting the positional parameter to a positive value.

The real behavior and considered simplifications have been verified by the following experiment. The femur link of a single leg has been made static in the simulation, and the tibia link has been set with a large mass. Then, the actuator has been commanded to swing between the boundary positions which does not make either of links move but verifies the studied behavior. Note, all the experimental results are reported in Chapter 7

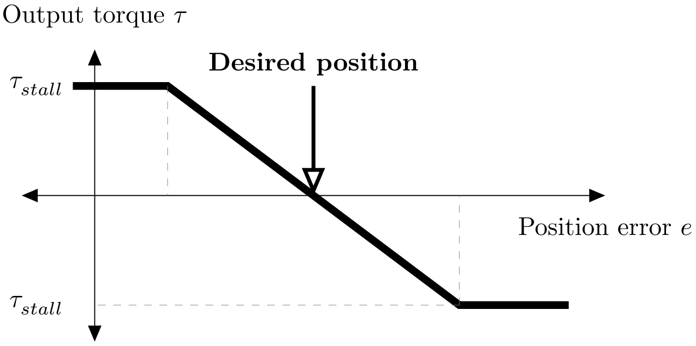


Figure 21: The relation of the torque τ and the position error err of the Dynamixel AX-12A servomotor. The value of τ is limited by a stall torque value τ_{stall}. Courtesy of [46].

6.2 Torque Analysis

In this experiment, three legs of the hexapod robot are lifted to create a support polygon in the form of a triangle as it is shown in Figures 22 and 23. Subsequently the angle of the tibia joint on the supporting middle leg goes from original position to an extended position, and thus gradually transfers a larger part of the robot body weight to this leg, and therefore, increasing torque τ on the leg joints.

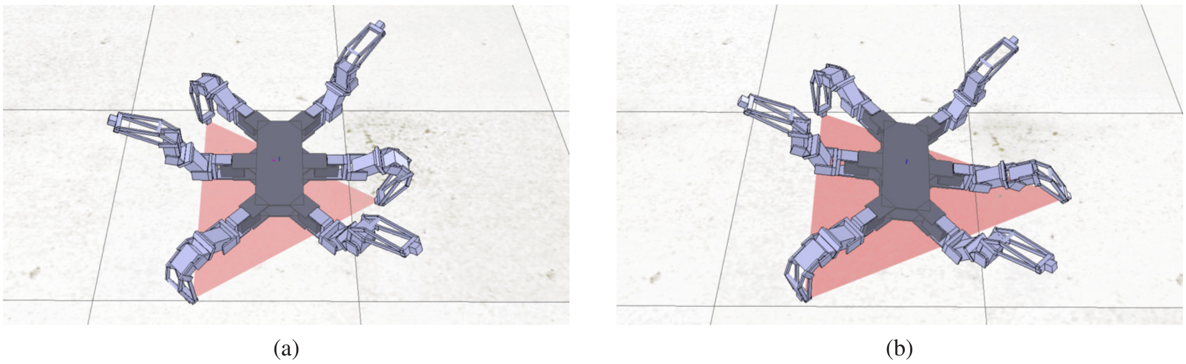
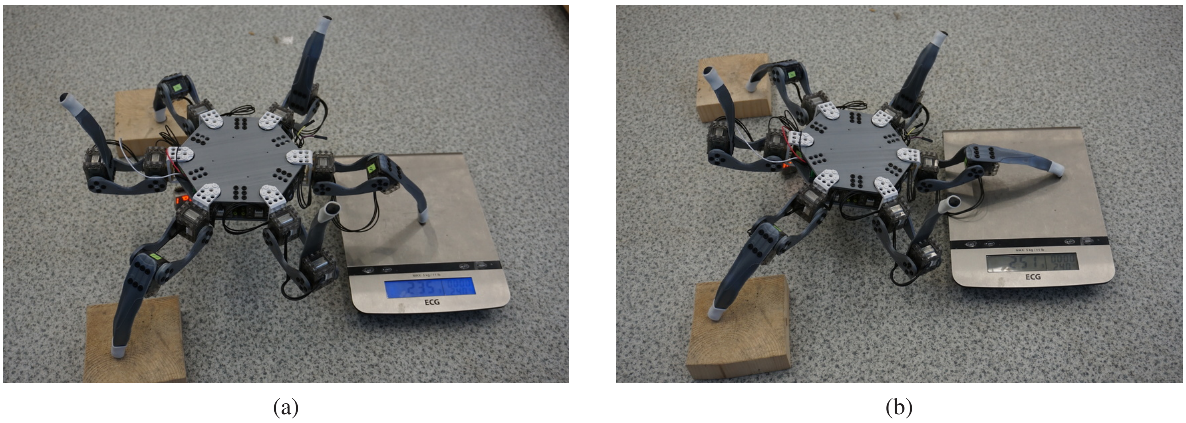


Figure 22: Torque analysis experiment of PhantomX in simulator. (a) Starting position and (b) end position.

The experiment is executed and analyzed quasi-statically for which the motion is sufficiently slow to consider each state of the robot be static. The experiment has been done three times with different weights of the robot trunk, that were adjusted by attaching battery packs to the robot body. For specification of weights see Table 1 and 4. The mass of the trunk is adjusted in the simulation accordingly. The ground truth for the torque values τ_{real} is calculated as the magnitude of the cross product of



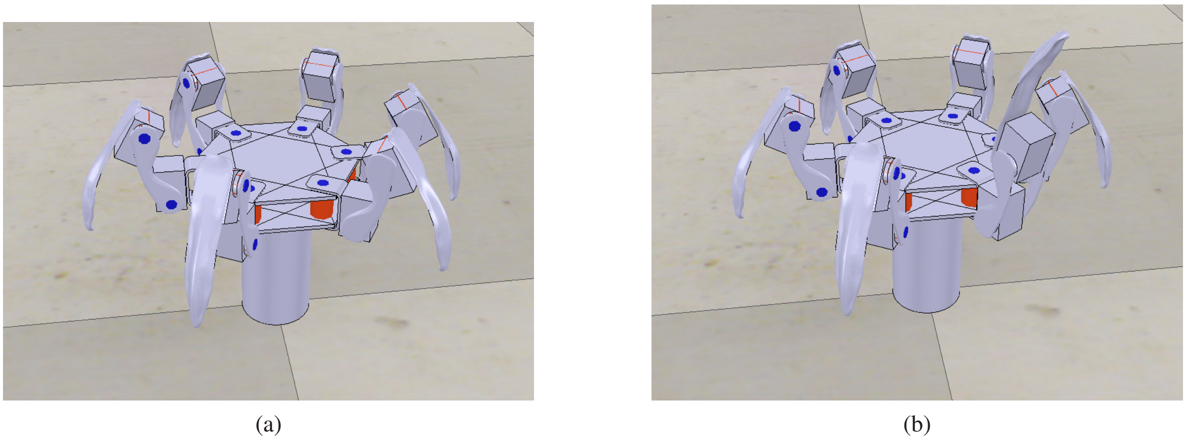
(a) (b)

Figure 23: Torque analysis experiment of Marvin. (a) Starting position and (b) end position.

ground reaction force at the leg endpoint, computed from measured weights, and the tibia vector (see Figure 15a). The simulated torque τ_{sim} is read directly from the simulator.

6.3 Dynamic Leg Movement

In the case of the quasi-static movement, the relationship between the joint position error and the torque visualized in Figure 21 holds. However, in the case of dynamic movement, further factors start to influence the evolution of the torque at the individual joints, e.g., velocities and moments of inertia. Therefore, we designed an additional experiment for analyzing the dynamic movement of a single leg as follows. In this experiment, the robot is supported under the body, so that it is enabled a free movement of the legs in both simulator and real experiments. A single actuator is set with a new position followed by the immediate consecutive position readings from that particular actuator. As it takes longer than 1 ms to reach the desired position, we obtain a swing profile of the actuator. The experiment is performed on three leg joints - coxa, femur, and tibia to verify the dynamics of the kinematic chains.



(a) (b)

Figure 24: Visualization of Dynamic leg movement experiment on tibia joint (a) initial position, (b) end position.

6.4 Dynamic Locomotion

Finally, we utilize the derived model in the locomotion control to verify the practical usability of the derived simulation model. Therefore, the model is employed in the problem of parameter search of a bio-inspired dynamic locomotion controller based on a central pattern generator (CPG), which generates synchronized rhythmic signals to control locomotion of multi-legged robots [47].

There are six CPGs, one per each leg, interconnected with symmetric inhibitive connections. The overall scheme of the controller is shown in Figure 25. Each CPG is a pair of Matsuoka's [48] adaptive neurons: extensor neuron and flexor neuron, and is connected through a simple output shaping network directly to the desired joint angle values. Therefore, the CPG network is responsible for maintaining stable oscillations and inter-leg synchronization, while the output shaping network shapes the CPG oscillations and allows simple differential steering by the requested v^{left} and v^{right} velocities for left and right half of the body, respectively. The signals generated by the networks are shown in Figure 26. One of the advantages of this approach is the fact, that it does not require identification of the robot kinematic model [5].

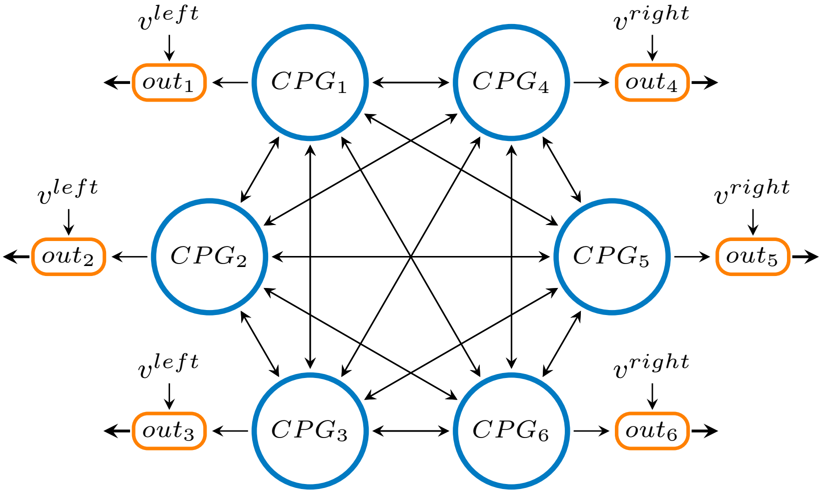
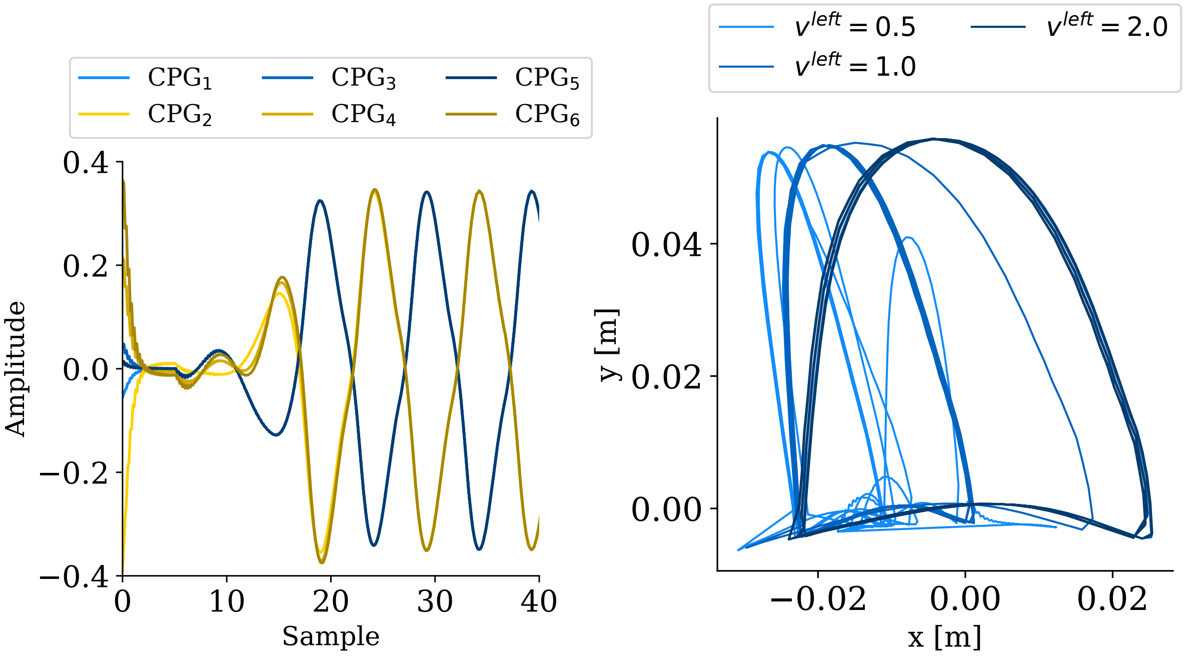


Figure 25: The overall scheme of the CPG-based locomotion controller. Courtesy of [5].

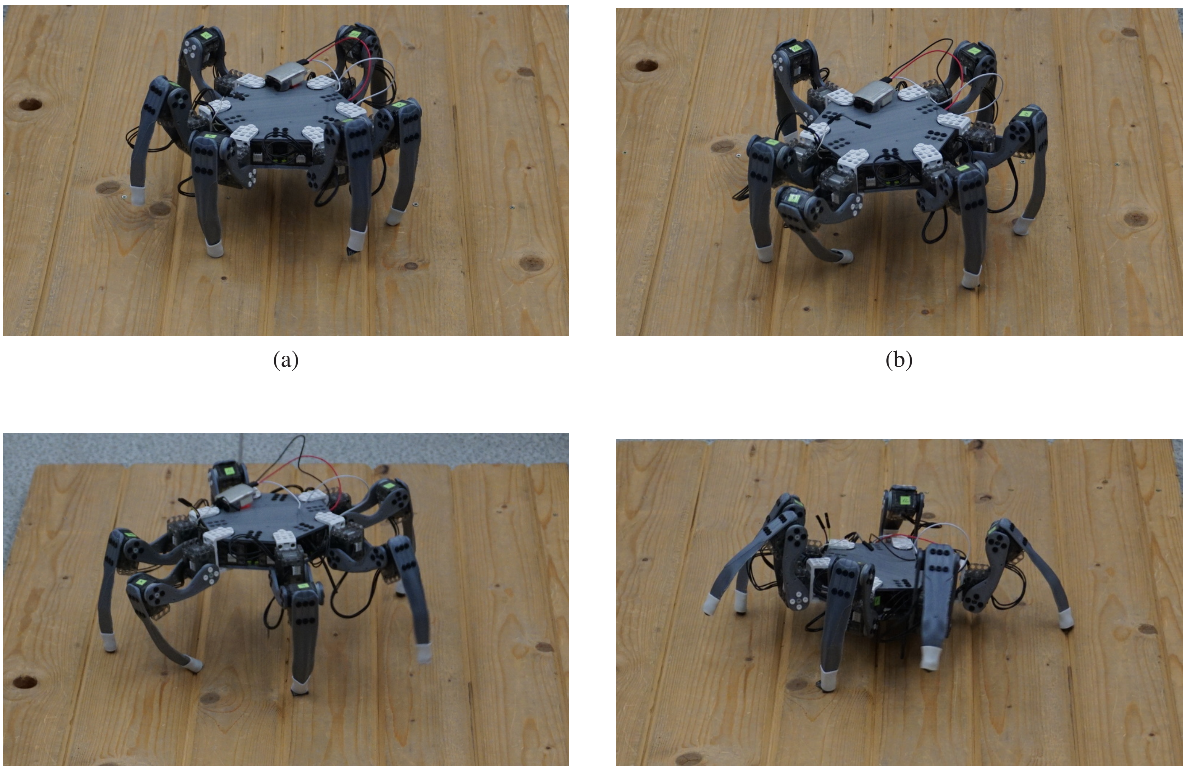
We searched for such parametrization of v^{left} and v^{right}, which directly influences the stride length of the gait cycle so as to maximize the robot forward velocity while satisfying the robot motion limits, joint torques and leg collision avoidance in particular. In Figure 26, CPG oscillations are shown along with the effect of adjusted parameters on the leg trajectory and therefore, overall robot forward motion. Walking Marvin using different locomotion parameter setup is captured in Figure 27. For each tested parametrization, the locomotion has been performed for 10 s in the V-REP simulator. During the simulation, self-collisions are monitored together with the achieved torque values τ_{sim}.



(a)

(b)

Figure 26: (a) CPG oscillations for a tripod gait, with visible transition effect before stable oscillations, due to random state initialization. (b) Leg foot-tip trajectory of the first leg for selected parameters v^{left}. Courtesy of [5]



(a)

(b)

(c)

(d)

Figure 27: Locomotion of Marvin using different parametrizations.

Chapter 7
Results

We have modeled two six-legged robots - PhantomX and Marvin (see Figure 28), which are both described in detail in Chapter 5. This chapter is divided into two sections; each contains results of modeling one of the mentioned robots. Each experiment has been performed five times. Representative measurements are visualized in each section along with the root mean square error of estimated data for the particular run. Parts of the herein presented results have been published in [49]

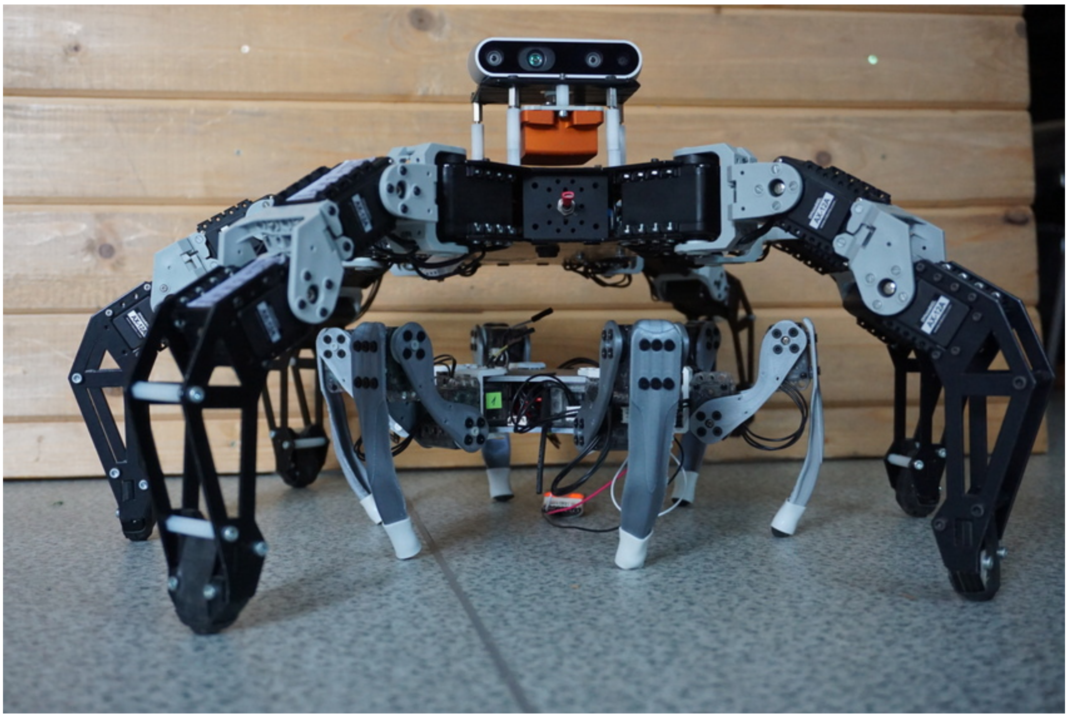


Figure 28: Modeled robots - PhantomX (above) and Marvin (below).

7.1 PhantomX

Firstly, the behavior of the adjusted joint has been verified using the experiment described in Section 6.1. The result is visualized in Figure 29 that complies with Figure 21. A small offset of the torque values is most likely caused by the experimental setup, where it is not possible to mitigate effects of the gravity acting on the dynamic link.

The selected hand-tuned parametrization is shown in Table 10. The search was performed to enhance Friction, Linear damping, Angular damping and Proportional parameter. Original default values were set for the joint Bullet properties - Normal CFM, Stop ERP and Stop CFM. The root mean square error (RMSE) was used as a criterion to evaluate simulation fidelity.

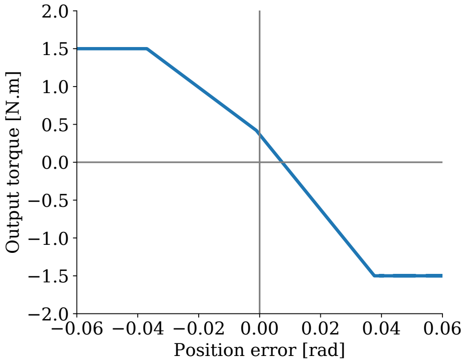


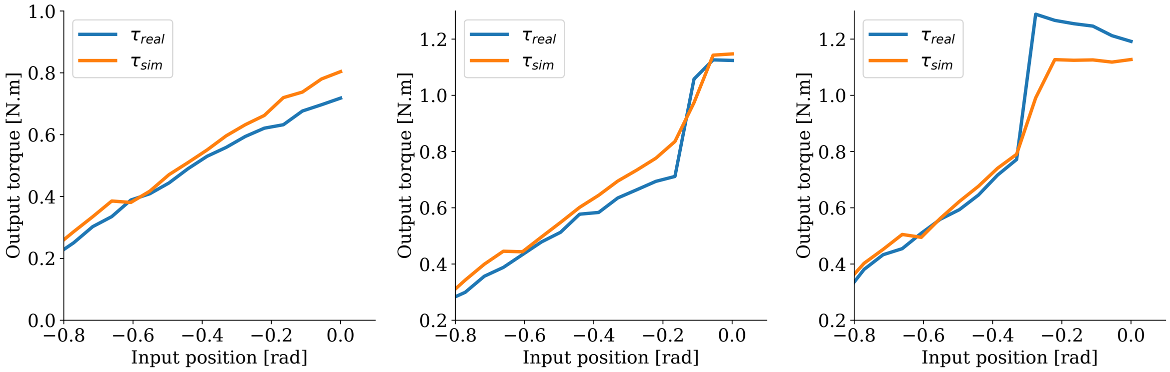
Figure 29: Relation of the torque τ and the position error err of Dynamixel AX-12A actuator model.

Table 10: Parameters to model a Dynamixel AX-12A actuator

Proportional parameter	1.0
Friction	1.0
Restitution	0.0
Linear damping	0.0
Angular damping	0.0
Normal CFM	0.0
Stop ERP	0.2
Stop CFM	0.0

7.1.1 Torque Analysis

The achieved results of the torque analysis for the selected parametrization are visualized in Figure 30a for the unloaded robot. Figure 30b and Figure 30c capture torques in the same experiment with the robot loaded with one and two battery packs, respectively. The experimental results indicate that the simulation can provide enough precision for the estimation of the joint torque values. Notice, in experiments with increased body weight, the femur servomotors went over the stall torque limit and turned off, which can be identified as a plateau in the plots, that is visible for both the simulation and real-world experiments.



(a) RMSE = 0.034 N.m

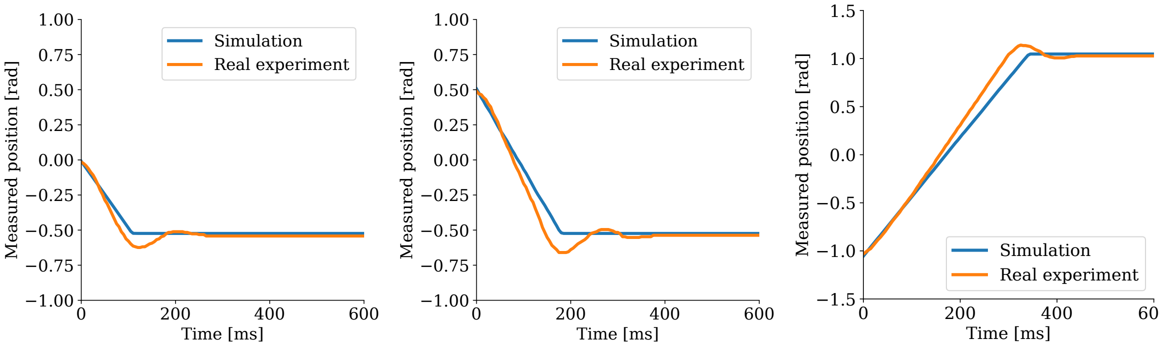
(b) RMSE = 0.065 N.m

(c) RMSE = 0.258 N.m

Figure 30: Comparison of the real torque values τ_{real} and the simulated torque τ_{sim} for (a) unloaded robot; (b) robot body loaded with additional 330 g; and (c) robot body loaded with 660 g.

7.1.2 Dynamic Leg Movement

Plots in Figure 31 represent a comparison between the behavior of the real joint and the model. The results indicate that the overall shape of the motion is correct for the proper speed that also indicates a correct setting of the P-type controller. However, the simulation does not cover the dynamic overshoots probably due to the setting of high friction in individual links. Unfortunately, we have not been able to overcome this issue.

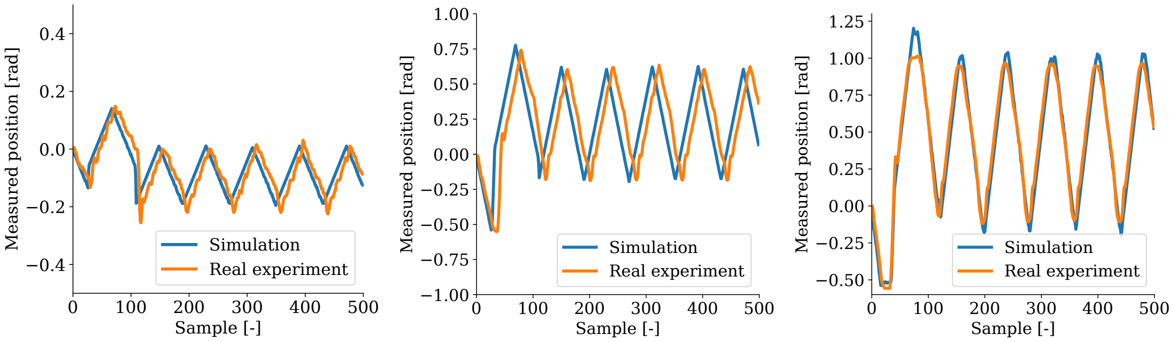


(a) RMSE = 0.033 rad (b) RMSE = 0.055 rad (c) RMSE = 0.079 rad

Figure 31: The position setting on (a) coxa, (b) femur, and (c) tibia joint.

7.1.3 Dynamic Locomotion

The found parametrization has been used in dynamic locomotion scenario. It has allowed four times longer leg stride length, while the robot parts did not collide and the torque was below the threshold of $0.75\tau_{stall}$, which is a safe value for continuous robot operation [8]. The comparison of leg trajectories for the coxa joint with the original locomotion parameters and coxa joint with four times speed up and the coxa joint with six times speed up, where self-collisions are already visible in both the simulated and real data, are visualized in Figure 32. Besides, when the speed-up is higher than the estimated parameter, the individual servomotors on the robotic platform start to overheat. Therefore, the experiment supports usability of the developed simulation and its deployability in planning and optimization of the locomotion control of the hexapod crawling robot.



(a) RMSE = 0.047 (b) RMSE = 0.221 (c) RMSE = 0.336

Figure 32: The dynamic locomotion scenario. (a) Trajectory of the coxa joint in the original parametrization. (b) Trajectory of the coxa joint in 4 times speed-up locomotion and (c) trajectory for 6 times speed-up locomotion, for which self-collisions are presented in the collected data.

7.2 Marvin

The behavior of the adjusted joint has been verified using the experiment described in Section 6.1. The result is visualized in Figure 33 that complies with Figure 21. A small offset of the torque values is most likely caused by the same experimental setup as in the case of PhantomX robot, where we were not able to mitigate the effects of the gravity acting on the dynamic link.

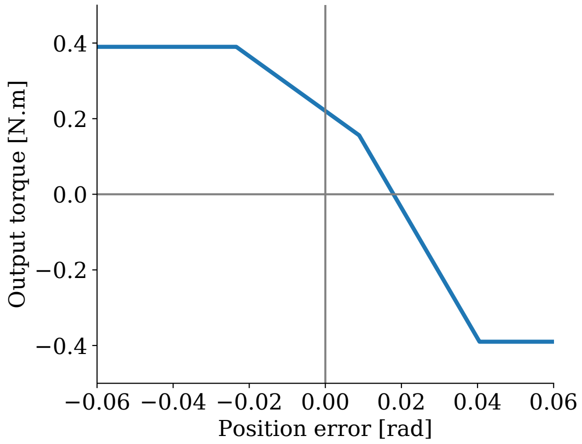


Figure 33: Relation of the torque τ and position error err of Dynamixel XL-320 actuator model.

The selected parametrization is a result of a grid search along 10 values for each parameter aiming to minimize the root mean square error of estimated data in the utilized verification scenarios. Parametrization of the simulated joints is shown in Table 10. The joint Bullet parameters, namely Normal CFM, Stop ERP, and Stop CFM, were additionally included in the grid search.

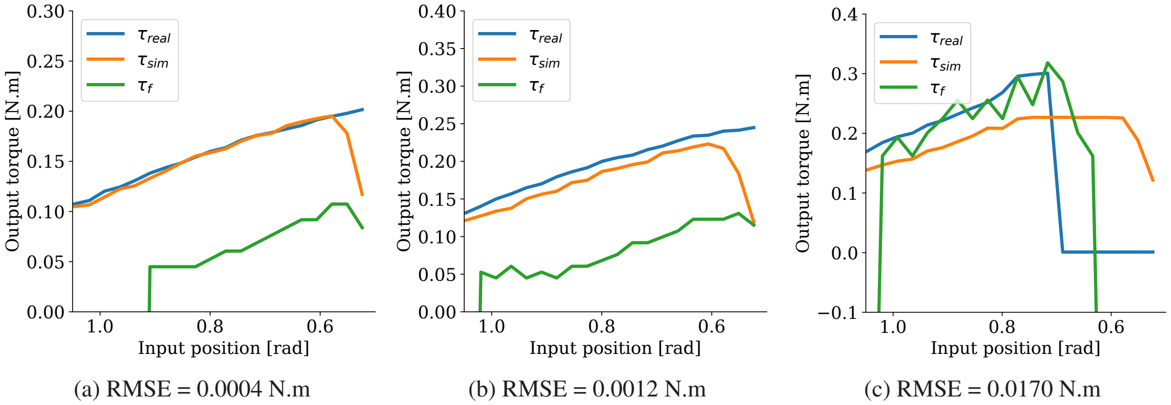
Table 11: Parameters to model a Dynamixel XL-320 actuator

Proportional parameter	2.0
Friction	0.3
Restitution	0.0
Linear damping	0.0
Angular damping	0.0
Normal CFM	0.01
Stop ERP	0.5
Stop CFM	0.0

7.2.1 Torque Analysis

The torque analysis experiment has been performed for the selected parametrization of Marvin model. The achieved results are shown in Figure 34a for unloaded robot and in Figures 34b and 34c for robot loaded with two and four Li-on batteries respectively. It has been observed, that especially Stop ERP and Proportional parameter settings had a major influence on the simulation fidelity in this scenario. In the case of unloaded and slightly loaded robot, it can be assumed that including the joint Bullet properties in the grid search has enhanced the simulation fidelity. The higher error measured for heavily loaded robot can be skewed by the missing data from the second half of the experiment due to femur actuator collapse by exceeding the stall torque $\tau_{stall} = 0.39$ N.m during the real experiment.

The position load feedback τ_f given by the servomotor provides less accurate data than provided by simulation for moderately loaded robot. Nevertheless, in case of heavy load, thus higher torques, the feedback gives more accurate measurement than simulation.

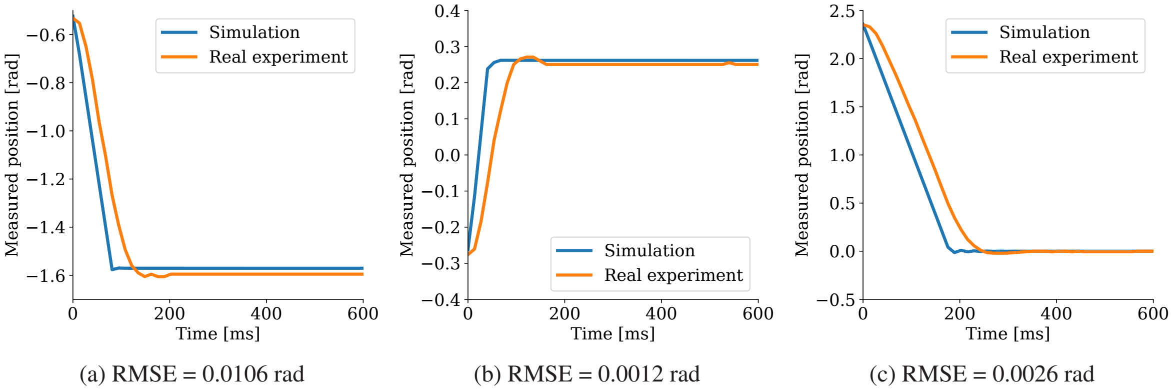


(a) RMSE = 0.0004 N.m (b) RMSE = 0.0012 N.m (c) RMSE = 0.0170 N.m

Figure 34: Comparison of the real torque values τ_{real} and the simulated torque τ_{sim} for (a) unloaded robot; (b) robot body loaded with additional 92 g; and (c) robot body loaded with 184 g.

7.2.2 Dynamic Leg Movement

The comparison of data measured on Marvin during real experiment and simulation is shown in Figure 35. The similarity of the data is significantly higher compared to the results on PhantomX (in Figure 31) due to custom settings of joint properties specific for Bullet physics engine, which were not previously taken into account. Particularly, the Normal CFM parameter has been additionally set with minor impact on previous results of Torque analysis experiment, to enhance the slope of the leg swing profile. Note, the Normal CFM parameter can be also adjusted in order to mitigate oscillations and stabilize the simulation. It can, however, lead to unrealistic simulation results, in case of strict constraints, such as application of high torques or forces. Plots in Figure 35 represent the results of found parametrization along with the eventual root mean square error values.

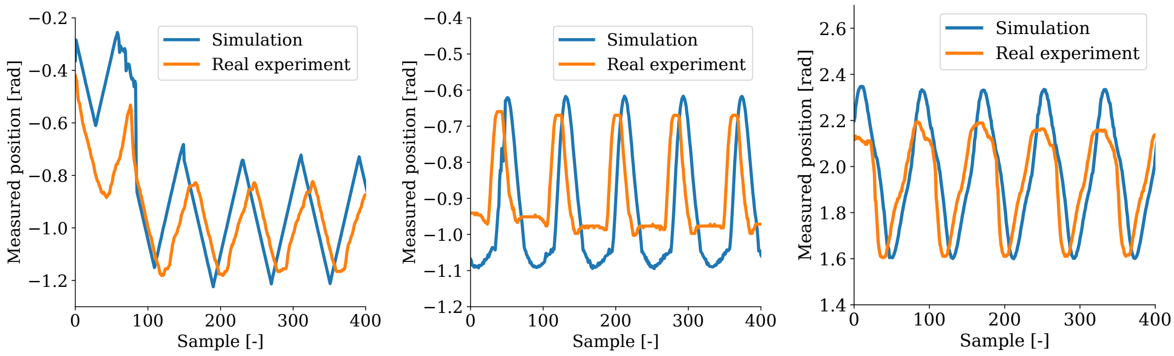


(a) RMSE = 0.0106 rad (b) RMSE = 0.0012 rad (c) RMSE = 0.0026 rad

Figure 35: The position setting on (a) coxa, (b) femur, and (c) tibia joint with with final parametrization.

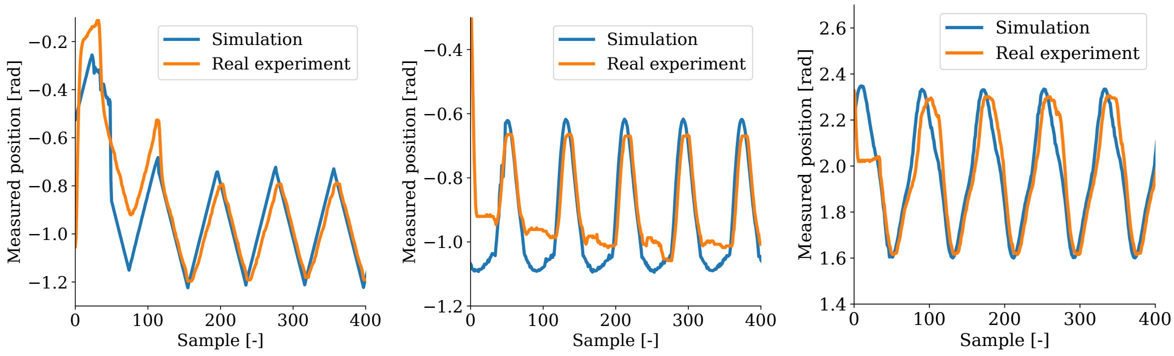
7.2.3 Dynamic Locomotion

The selected parametrization has been used in Dynamic locomotion scenario. Unlike in the case of PhantomX, it was also necessary to set other constants than v^{left} and v^{right} to adjust the output shaping network of the controller, as the previous gait parametrization did not exist. The initial set of parameters was first tested in simulation to avoid leg collisions and it was further enhanced during the real experiment in order to increase locomotion speed and efficiency. The trajectories of leg joints during locomotion are visualized in Figure 36. It is notable, that there are systematic errors in measurements of tibia and femur joint in the leg supporting phase, as can be seen in Figures 36c and 36b, respectively, which are most likely caused by high compliance in actuator and link interconnections, or by using rubber foot-tips, which were used in the real experiment, but not included in the simulation as separate objects. Comparison of simulation experiment and the real experiment having rubber foot tips removed can be seen in Figure 37. Modeling of such properties forms a significantly more challenging task, which are out of scope of thesis.



(a) (b) (c)

Figure 36: Joint trajectories during locomotion of Marvin with rubber foot tips (a) coxa, (b) femur, (c) tibia.



(a) (b) (c)

Figure 37: Joint trajectories during locomotion of Marvin without rubber foot tips (a) coxa, (b) femur, (c) tibia.

Chapter 8
Conclusion

In this thesis, the design of a six-legged robot platform, named Marvin is proposed. The robot mechanical construction has been derived from the modified version of PhantomX robot, originally created by Trossen Robotics, which is used for research at Computational Robotics Laboratory. Marvin is built from off-the-shelf components, specifically Dynamixel XL-320 servomotors, powered by two Li-on 18650 batteries and Orange Pi Zero control board with Linux Armbian Bionic system.

The main purpose of Marvin is supposed to correspond to a remotely controlled mobile sensor collecting data in spaces with limited accessibility. However, it can also be employed in education for its small dimensions and weight, lower cost, replaceable structural parts, and a built-in servomotor safety clutch, which is more resilient to misuse and overload than Dynamixel AX-12A actuators used in PhantomX. As a result of the same joint configuration and similar properties of actuators used in both robots, it was possible to employ the software developed on PhantomX on the Marvin robot with modified constants setting, e.g., leg dimensions, Denavit-Hartenberg leg parameters, or joint position range. In this thesis, we have employed a Central Pattern Generator (CPG) based locomotion controller to achieve simple locomotion of constructed Marvin prototype on flat terrain.

The structural parts of the designed robot were 3D printed to enable fast prototype modification, link shape optimization or replacement of damaged parts. However, this technology has a major drawback in a limited variety of accessible printing materials and methods. Printed parts then have several unexpected properties, such as high compliance or fragility in certain directions due to the used layer printing method. On the other hand, the high compliance can be viewed as an advantage of the robot in terms of flexibility, which may enable dynamic movements of the robot so as to increase the robot traversability, in future research.

The model of Marvin was then created in Virtual Robot Experimentation Platform (V-REP) simulator. The search for optimal model parameters was performed following the methodology, previously deployed on the PhantomX robot. The model fidelity was then evaluated in comparison of measured and simulated proprioceptive signals. It is problematic to define this accuracy of our measurements regarding missing ground truth data, due to the absence of appropriate sensors. Nevertheless, with the assumption of our real data measurements being precise enough, following conclusions were made.

The results indicate that the simulation provides enough precision, with root mean square error being as low as 6 and 0.5 times the actuator resolution in case of PhantomX and Marvin robot respectively. As for the torque measurement, the simulation measurements exceeded the accuracy of the position load feedback from servomotors, according to the Torque analysis experiment for both of the analyzed robots. The root mean square errors ranged from $0.2\tau_{stall}$ to $0.17\tau_{stall}$ for the Dynamixel AX-12A actuator and from $0.001\tau_{stall}$ to $0.04\tau_{stall}$ for the Dynamixel XL-320 actuator.

Therefore, by our work, we have obtained a high-fidelity model of the PhantomX hexapod robot that is actively used in the research activities [46], [49] of the Computational Robotics Laboratory, which can be further employed in education [6] and virtual data collection. Furthermore, we have applied the modeling methodology on the newly designed Marvin robot and verified its fidelity on a constructed prototype. Given the simulation accuracy, we were able to test and search for parametrizations of a Central Pattern Generator (CPG) locomotion controller. In comparison to the original parameter setup of PhantomX, we have achieved a four times larger coxa stride length of the gait cycle, thus corresponding increase of the overall robot forward velocity. As for the Marvin robot, there was no original parametrization of the CPG locomotion controller available, hence, it was necessary to adjust the complete initial parameter setup in the simulation so as to ensure safe locomotion. The parameters were then enhanced during experiments on the constructed prototype so as to increase locomotion speed and efficiency. There have been systematical errors observed on tibia and femur joints during the leg supporting phase, which were most likely caused by insufficiently modeled rubber foot-tips and link interconnection compliance, which form a challenging task for the future work on Marvin. Further improvement of the simulation fidelity could be made by applying specific optimization methods in parameter search instead of simple grid search.

Besides adjusting the robot model, certain modifications can be made to improve the Marvin platform properties. Such modifications include increasing the Wi-Fi connection reliability by implementing an external module instead of using the built-in version. Secondly, custom assembling system can be designed to replace OLLO rivets and thus reduce the compliance of link connections. Thirdly, battery holders could be implemented inside the robot trunk so as to allow easy battery replacement when required. Also paralelism of servomotor communication as in [50] could be employed to reduce the response time when reading several actuators at once. And finally, the software developed on the PhantomX robot could be employed for Marvin platform with minor parameter alteration, in order to deploy the robot capabilities to change locomotion gaits and adapt to the terrain.

References

[1] A. Cully, J. Clune, and J.B. Mouret. Robots that can adapt like natural animals. *Nature*, 521:503–507, 2015.

[2] V. Durr. Stereotypic leg searching movements in the stick insect: kinematic analysis, behavioural context and simulation. *Journal of Experimental Biology*, 204(9):1589–1604, 2001.

[3] D. Belter, P. Labecki, and P. Skrzypczynski. Adaptive motion planning for autonomous rough terrain traversal with a walking robot. *Journal of Field Robotics*, 33(3):337–370, 2016.

[4] M. Freese, S. Singh, F. Ozaki, and N. Matsuhira. Virtual robot experimentation platform V-REP: A versatile 3D robot simulator. In *IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, volume 6472, pages 51–62, 2010.

[5] P. Čížek and J. Faigl. Self-supervised learning of the biologically-inspired obstacle avoidance of hexapod walking robot. *Bioinspiration & Biomimetics*, page 046002, 2019.

[6] Labs and lecture notes B4M36UIR and BE4M36UIR - Artificial intelligence in robotics., Czech Technical University in Prague. `https://cw.fel.cvut.cz/b181/courses/b4m36uir/start`. cited on May 24, 2019.

[7] J. Dupeyroux, P. Grégoire, F. Ruffier, V. Stéphane, and J. Serres. Hexabot: a small 3d-printed six-legged walking robot designed for desert ant-like navigation tasks. In *IFAC World Congress*, 2017.

[8] Collective of authors. Dynamixel AX-12A documentation. `http://emanual.robotis.com/docs/en/dxl/ax/ax-12a/`. cited on May 24, 2019.

[9] Collective of authors. Dynamixel XL-320 documentation. `http://emanual.robotis.com/docs/en/dxl/x/xl320/`. cited on May 24, 2019.

[10] M. R. Fielding and G. Reginald Dunlop. Omnidirectional hexapod walking and efficient gaits using restrictedness. *The International Journal of Robotics Research*, 23:1105–1110, 2004.

[11] N. Porcino. Hexapod gait control by a neural network. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 189–194, 1990.

[12] A. J. Ijspeert. Central pattern generators for locomotion control in animals and robots: A review. *Neural Networks*, 21(4):642–653, 2008.

[13] D. Belter, P. Skrzypczynski, K. Walas, and D. Wlodkowic. Affordable multi-legged robots for research and stem education: A case study of design and technological aspects. *Advances in Intelligent Systems and Computing*, 351:23–34, 2015.

[14] M. Hutter, C. Gehring, M. Bloesch, M. Hoepflinger, C. David Remy, and R. Siegwart. Starleth: a compliant quadrupedal robot for fast, efficient, and versatile locomotion. In *15th International Conference on Climbing and Walking Robot (CLAWAR)*, pages 483–490, 2012.

[15] M. P. Murphy, A. Saunders, C. Moreira, A. Rizzi, and M. Raibert. The LittleDog robot. *The International Journal of Robotics Research*, 30:145–149, 2011.

[16] M. Konyev, F. Palis, Y. Zavgorodniy, A. Melnikov, A. Rudskiy, A. Telesh, U. Schmucker, and V. Rusin. Walking robot ”ANTON”: Design, simulation, experiments. *11th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)*, pages 922–929, 2008.

[17] A. Roennau, G. Heppner, M. Nowicki, and R. Dillmann. LAURON V: A versatile six-legged walking robot with advanced maneuverability. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 82–87, 2014.

[18] M. Zoula. Locomotion control of hexapod walking robot with four degrees of freedom per leg. Bachelor’s thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, 2019.

[19] Brett Kennedy, Hrand Aghazarian, Yang Cheng, Mike Garrett, Gregory Hickey, Terry Huntsberger, Lee Magnone, Colin Mahoney, Amy Meyer, and Jennifer Knight. LEMUR: Legged excursion mechanical utility rover. *Autonomous Robots*, 11:201–205, 2001.

[20] K. Galloway, G. Clark Haynes, D. Ilhan, A. M. Johnson, R. Knopf, G. A. Lynch, B. N. Plotnick, M. White, and D. Koditschek. X-RHex: A highly mobile hexapedal robot for sensorimotor tasks. *Technical Reports (ESE), University of Pennsylvania, Philadelphia, PA,*, 2010.

[21] G. Passault, Q. Rouxel, F. Petit, and O. Ly. Metabot: A low-cost legged robotics platform for education. In *International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 283–287, 2016.

[22] S. Lohmann, J. Yosinski, E. Gold, J. Clune, J. Blum, and H. Lipson. Aracna: An open-source quadruped platform for evolutionary robotics. In *16th European Conference on Applications of Evolutionary Computing (EvoApplications)*, pages 387–392, 2012.

[23] H. Lipson, J. Bongard, V. Zykov, and E. Malone. Evolutionary robotics for legged machines: From simulation to physical reality. In *9th International Conference on Intelligent Autonomous Systems*, pages 11–18, 2006.

[24] D. Belter, P. Skrzypczynski, K. Walas, P. Fankhauser, C. Gehring, M. Hutter, M. Hoepflinger, and R. Siegwart. Dynamic simulation of legged robots using a physics engine. In *17th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines(CLAWAR)*, pages 567–574, 2014.

[25] K. Nonami, N. Shimoi, Q.J. Huang, D. Komizo, and H. Uchida. Development of teleoperated six-legged walking robot for mine detection and mapping of mine field. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 775 – 779, 2000.

[26] G. Best, P. Moghadam, N. K.ottege, and L. Kleeman. Terrain classification using a hexapod robot. *Australasian Conference on Robotics and Automation (ACRA)*, 226:825–833, 2013.

[27] D. Williamson, N. Kottege, and P. Moghadam. Terrain characterisation and gait adaptation by a hexapod robot. In *Australasian Conference on Robotics and Automation (ACRA)*, 2016.

[28] J. Mrva and J. Faigl. Tactile sensing with servo drives feedback only for blind hexapod walking robot. In *10th International Workshop on Robot Motion and Control (RoMoCo)*, pages 240–245, 2015.

[29] A. Manglik, K. Gupta, and S. Bhanot. Adaptive gait generation for hexapod robot using genetic algorithm. In *IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, pages 1–6, 2016.

[30] J.C. Bongard and H. Lipson. Automated damage diagnosis and recovery for remote robotics. In *IEEE International Conference on Robotics and Automation*, pages 3545–3550 Vol.4, 2004.

[31] Y. Tassa. Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[32] A. Boeing and T. Braunl. Evaluation of real-time physics simulation systems. In *5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia (GRAPHITE)*, pages 281–288, 2007.

[33] A. Roennau, F. Sutter, G. Heppner, R. Dillmann, and J. Oberländer. Evaluation of physics engines for robotic simulations with a special focus on the dynamics of walking robots. In *16th International Conference on Advanced Robotics (ICAR)*, 2013.

[34] Collective of authors. Virtual Robot Experimentation Platform User Manual. `http://www.coppeliarobotics.com/helpFiles/`. cited on May 24, 2019.

[35] S. Yildrim and E. Arslan. ODE (Open Dynamics Engine) Based Stability Control Algorithm for Six Legged Robot. *Measurement*, 124, 2018.

[36] Collective of authors. Bullet physics library. `https://pybullet.org/wordpress/`. cited on May 24, 2019.

[37] J. Jerez, A. Suero et al. Newton Dynamics. `http://newtondynamics.com/forum/newton.php`. cited on May 24, 2019.

[38] Russel L. Smith et al. Open Dynamics Engine. `http://www.ode.org/`. cited on May 24, 2019.

[39] Collective of authors. Vortex Simulation Software. `https://www.cm-labs.com/`. cited on May 24, 2019.

[40] Collective of authors. Bullet physics library documentation. `https://github.com/bulletphysics/bullet3/tree/master/docs`. cited on May 24, 2019.

[41] Collective of authors. PhantomX Hexapod MK-III Metal Assembly Guide. `https://learn.trossenrobotics.com/projects/172-phantomx-hexapod-metal-assembly-guide.html`. cited on May 24, 2019.

[42] Collective of authors. Autodesk Fusion 360. `https://www.fusion360.cz/`. cited on May 24, 2019.

[43] O. Stříteský, J. Průša, and M. Bach. Základy 3D tisku. `https://www.prusa3d.cz/wp-content/uploads/zaklady-3d-tisku.pdf`. cited on May 24, 2019.

[44] Collective of authors. OrangePi Zero. `http://www.orangepi.org/orangepizero/`. cited on May 24, 2019.

[45] Collective of authors. Dynamixel software development kit. `http://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/overview/`. cited on May 24, 2019.

[46] J. Faigl and P. Čížek. Adaptive locomotion control of hexapod walking robot for traversing rough terrains with position feedback only. *Robotics and Autonomous Systems*, 116:136–147, 2019.

[47] R. J. Szadkowski, P. Čížek, and J. Faigl. Learning central pattern generator network with back-propagation algorithm. In *Proceedings ITAT 2018: Information Technologies - Applications and Theory*, pages 116–123, 2018.

[48] K. Matsuoka. Sustained oscillations generated by mutually inhibiting neurons with adaptation. *Biological cybernetics*, 52:367–76, 1985.

[49] Minh Thao Nguyenová, Petr Čížek, and Jan Faigl. Modeling proprioceptive sensing for locomotion control of hexapod crawling robot in robotic simulator. In *Modelling and Simulation for Autonomous Systems (MESAS)*, pages 215–225, 2019.

[50] M. Tržil. Fpga-based control of multi-legged walking robot. Bachelor's thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, 2019.