Development of a neuro-inspired control system for quadrupeds to emulate sensorimotor processes in animals

A project report
submitted by

SHREYAS SHANDILYA

in partial fulfilment of requirements
for the award of the dual degree of

BACHELOR OF TECHNOLOGY IN
ENGINEERING DESIGN
AND
MASTER OF TECHNOLOGY IN
BIOMEDICAL DESIGN

DEPARTMENT OF ENGINEERING DESIGN
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

MARCH 2021

CERTIFICATE

This is to certify that the project titled **Development of a neuro-inspired control system for quadrupeds to emulate sensorimotor processes in animals**, submitted by **Mr Shreyas Shandilya**, to the Indian Institute of Technology Madras, for the award of the degrees of **Bachelor of Technology in Engineering Design** and **Master of Technology in Biomedical Design**, is a *bona fide* record of the research work done by her under my supervision. The contents of this project, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr Asokan Thondiyath
Project Adviser
Professor
Department of Engineering Design
Indian Institute of Technology Madras
Chennai 600 036

Dr V.S. Chakravarthy
Co-guide
Professor
Dept. of Biotechnology
IIT Madras, 600 036

Place: Chennai

Date: March 15, 2021

ACKNOWLEDGEMENTS

ABSTRACT

KEYWORDS: Computational Neuroscience, Neural Networks, Central Pattern Generators, Deep Learning, Robotics, ROS

This project report documents studies towards the development of a neural network control system for quadruped robots. Locomotion and navigation require the perception of the surrounding space and embodied decision making to make an appropriate response to an incoming stimulus. Animals are highly adept at such tasks, and the extraordinary agility and dexterity exhibited by animals are highly desirable for legged robots. A hybrid neural network for the locomotion and navigation control system of a quadruped robot is proposed. The proposed system leverages oscillatory models of brain function to mimic a two-level hierarchy of locomotion control. The lower level is controlled by Central Pattern Generators in the spinal cord and the higher level by the Pedunculopontine Nucleus in the Brain Stem of humans. This project focuses on emulating gait with navigation when the hierarchical controller is trained using Reinforcement Learning and the developed control system's consequent deployment on a quadruped.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

FK	Forward Kinematics
IK	Inverse Kinematics
DL	Deep Learning
RL	Reinforcement Learning
DoF	Degree-of-freedom
CPG	Central Pattern Generator
DNN	Deep Neural Network
DIY	Do It Yourself
IMU	Inertial Measurement Unit
URDF	Universal Robot Description Format
Osc.	Oscillators

NOTATIONS

$\vec{r_i}$	The *i*th link vector of the MePed knee four bar linkage
θ_i	The *i*th joint angle position of the MePed knee four bar linkage
ω_i	The *i*th joint angular velocity point of the MePed knee four bar linkage
α_i	The *i*th joint angular acceleration point of the MePed knee four bar linkage
v_i	Velocity of the *i*th link of the MePed knee four bar linkage
a_i	Acceleration of the *i*th link of the MePed knee four bar linkage
T	Maximum torque from TowerPro SG90 Micro Servo
F_{ijx}	Force on the *j*th link from *i*th link along the x axis
F_{ijy}	Force on the *j*th link from *i*th link along the y axis
F_{ix}	Net Force on the *i*th link along the x axis
F_{ix}	Net Force on the *i*th link along the y axis
C_i	Net Torque acting on *i*th link
L_T	Length of the MePed Thigh
L_i	Length of *i*th link of the MePed knee four bar linkage
ϕ	Angle between $\vec{r_0}$ and the x-axis
$\vec{a}$	Resultant acceleration of the quadruped during gait
$\vec{\alpha}_R$	Resultant angular acceleration of the quadruped during gait
m	Mass of the Meped
I_R	Moment of Inertia of the MePed about the center of mass position at rest
$\vec{N}$	Normal Reaction forces acting on the quadruped legs
$\vec{F}_{fr}$	Friction forces acting on the quadruped legs
z	Complex valued dependent variable of the Hopf Oscillator
x	Real part of the dependent variable of the Hopf Oscillator
y	Imaginary part of the dependent variable of the Hopf Oscillator
ω	Frequency of a Hopf Oscillator
μ	Amplitude controlling parameter of a Hopf Oscillator
T_{sw}	Swing period of a gait cycle
T_{st}	Stance period of a gait cycle
β	Duty Factor of a gait cycle
θ_h	Amplitude of oscillation of Hip Joint Trajectory
θ_k	Amplitude of oscillation of Knee Joint Trajectory

CHAPTER 1

INTRODUCTION

1.1 Background

Animals are highly adept at locomotion and navigation under challenging terrains, capable of responding to a sudden stimulus with extraordinary agility and dexterity. They exhibit behaviour like gait switching, rapid acceleration and deceleration, evasive manoeuvres, climbing, jumping and search. The seamless transition between different behaviours and agile response to an incoming stimulus results from the evolution of neural pathways for adaptive locomotion, perception and embodied decision making. A model of such integrated sensorimotor processing can provide greater autonomy and deftness to robotic locomotion and navigation.

Among animals, quadrupedal locomotion is the most common, with some animals capable of limited bipedal locomotion. Such prevalence of quadrupedal locomotion in nature may be attributed to the inherent stability and wide range of available configurations of locomotion. These characteristics of quadrupedal locomotion make it suitable for several applications such as last-mile delivery and search and rescue to the potential to work in unstructured, severe and dangerous environments. Consequently, several quadruped robots have been developed over the years. The TITAN series [7], Tekken IV [8], MIT Cheetah [9] and BigDog [10] have been some of the successful quadruped robots capable of gait in severe and difficult environments. The aforementioned quadruped robots' critical technologies attributed to their success are their biomimetic structure, a high power density of actuator, real-time control methods, and integrated environment perception [11]. All the developments through these quadrupeds lead to an integrated environment perception and decision-making system.

Neuroscientific models of cognition have explained information processing in living organisms at varying degrees of details. Available models range from bio-molecular models operating at the synaptic levels to aggregate oscillatory models operating at the level of neuron ensembles. Since animal behaviour is desired in quadrupeds, neuroscientific models of sensorimotor processes to emulate such behaviour seem logical.

Moreover, such models also benefit from the massively parallel and efficient computations seen in the brain, making such models ideal for robotics' resource-constrained applications.

This report documents the design and development of a neural network architecture for autonomous control. The neural network combines Central Pattern Generator(CPG) theory with Deep Neural Networks to emulate different aspects of a hierarchical multi-loop sensorimotor processing with a one-to-one mapping between the proposed architecture modules and the nervous system's involved processing centres. Furthermore, a Deep Deterministic Policy Gradient is used to learn sophisticated behaviour with experience. An incremental methodology is followed to learn increasingly complex behaviour by the formulation of appropriate reward functions. The developed neural network is tested and deployed on a prototype quadruped platform, built as a part of this project.

1.2 Motivation

The primary motivation for this work is the need for adaptive gait generation for an efficient control strategy for legged robots based on biological locomotion principles. Animal locomotion requires multi-dimensional coordinated rhythmic patterns that need to be correctly modulated to satisfy multiple constraints such as generating forward motion with low energy, without falling over, adapting to possibly challenging terrain, and allowing the modulation of speed and direction [12]. In vertebrates, CPGs are the essential building blocks that generate and modulate the rhythmic patterns required for locomotion. The neuroscientific theory defines CPGs as non-linear dynamical systems with limit cycle behaviour given the dynamical system's parameters. CPGs have been successfully used to control a variety of robots. A distributed control strategy was proposed in [13] for a modular quadruped robot. The strategy proposed used N coupled amplitude-controlled Hopf oscillators to control the N degrees of freedom in the quadruped. Similarly, a four oscillator CPG model that integrates sensory feedback into the CPG for gait modulation was proposed in [3]. This model was further extended in [14] by introducing a two-level hierarchy, such that the higher level modulates the CPG model by appropriately modifying CPG parameters.

In biology, CPGs are activated and modulated by simple tonic signals from higher parts

of the brain, and they are strongly coupled with the body they control and the environment via sensory feedback. The strong coupling via sensory feedback allows for integrated environment perception and embodied decision making. Although current CPG models are successful in gait generation, gait modulation and learning with CPGs remain challenging. Moreover, the integration of higher-level control with a CPG requires modulating high dimensional parameter vectors that determine CPG behaviour instead of simple tonic signals. Furthermore, there have been very few instances where a CPG is a part of the learning system. For instance, a CPG model was implemented as a layer in a deep neural network to facilitate backpropagation in [15]. However, in most cases, behaviour like gait transition and modulation is achieved using predefined relationships to produce appropriate CPG parameters. Due to such shortcomings in CPG models, learning with experience is challenging, and genuine animal-like behaviour can not be achieved.

On the other hand, Deep Neural Networks (DNNs) are capable of universal approximation and have standard learning rules for updating parameters. Gradient-based or gradient-free techniques may be used to optimise the parameters of a DNN. The optimisation technique used constrain the structure of DNN to a specific class of networks. For instance, backpropagation requires the network to have no loops, whereas CPG coupling almost always forms loops. Although the integration of DNNs with CPGs constrains network structure choice, the ability of universal approximation added to the network allows for the transformation of a low dimensional desired motion or state vector into a tonic signal that can be used to modulate the CPG for rhythm generation. The hierarchical controller proposed in [5] uses a DNN whose outputs are used to regulate selected CPG network parameters. Though the CPG is not a part of the learned network, the use of the DNN allows for the integration of a low-dimensional feedback signal for control.

Emulation of more sophisticated behaviour like obstacle avoidance and search can be achieved using Reinforcement Learning (RL) to learn an appropriate policy. For instance, a hierarchical controller was proposed in [16] that applies RL to negotiating obstacles with a quadruped robot. Since a DNN-CPG architecture requires only low dimensional desired motion or state vector to modulate the network output, a hierarchical controller similar to those proposed in [16] and [5] can be developed using RL to emulate a particular animal behaviour. Such a controller, combining the universal

approximation and learning abilities of DNNs with the adaptive rhythm generation of CPGs, is expected to be successful in the nimble, autonomous control of a quadruped in challenging terrains.

1.3 Objectives

The primary aim of this work is to develop a generalised DNN-CPG architecture for autonomous control problems. The developed architecture is a model of the sensori-motor processing typical across vertebrates, and its functioning is to be demonstrated on a quadruped platform, also assembled as a part of this project. It is also intended to study the evolution of quadruped behaviour as it interacts with its environment. The developed model is limited to the emulation of the behaviour of obstacle avoidance and path planning.

1.4 Scope

The following have been achieved in the present work as direct objectives of its initiation.

1. Assembly of a Quadruped Platform for the testing and deployment of developed neural network
2. Development of ROS-Gazebo simulation environment for RL
3. Development of a DDPG for RL
4. Formulation reward functions to quantise different aspects of quadruped behaviour
5. Development of a DNN-CPG architecture for autonomous control

Additionally, the following are expected to result from the investigation of related topics.

1. An improved understanding of the nervous system and distributed processing seen in the brain
2. An improved understanding of quadruped kinematics and dynamics

1.5 Organisation of the Report

The remainder of this report is organised as follows.

CHAPTER 2

THE MEPED QUADRUPED PLATFORM

2.1 Introduction

The MePed is a small open-source quadruped robot, $16cm \times 16cm \times 6cm$ in size with 8 DoF, 2 on each leg. It is compatible with both Arduino and Raspberry Pi and serves as the platform for testing and deploying the DNN-CPG models to be developed. The quadruped is currently controlled using an Arduino Mega 2560, with a V2 sensor shield. Tower Pro SG90 micro servo motors are used to actuate the joints. The robot chassis weighs 196.36 g and the complete quadruped, with servos and controller weighs 345.32g.

The Tower Pro SG90 servo produces a maximum torque of 1.5 $\mathrm{kg\,cm^{-1}}$ and an operating speed of 0.143 $\mathrm{s\,rad^{-1}}$ at $6V$ power supply. The crawling type quadruped, though with a lower ground clearance has greater stability, making it ideal for testing and deployment of the DNN-CPG architecture.

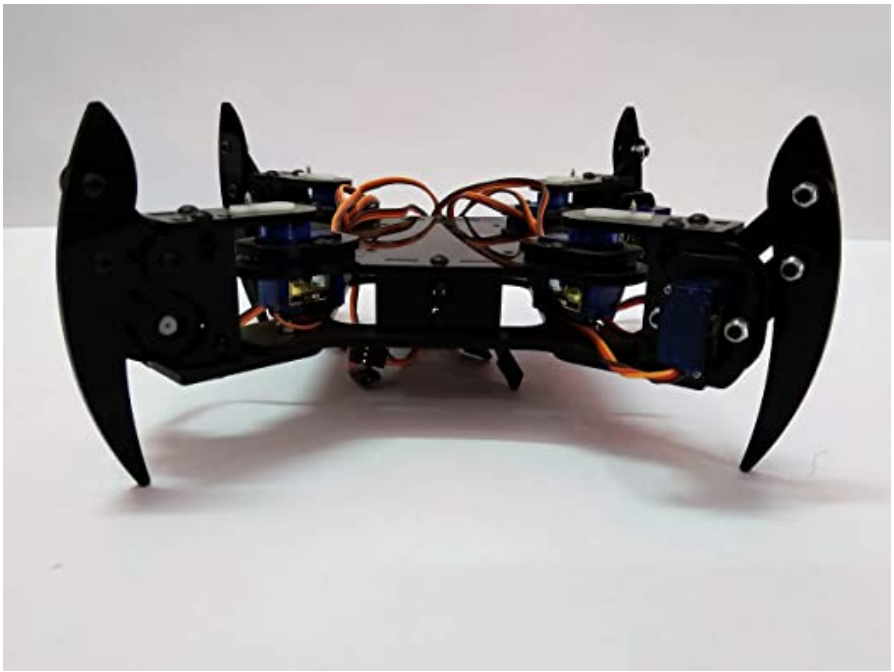


Figure 2.1: MePed Front View

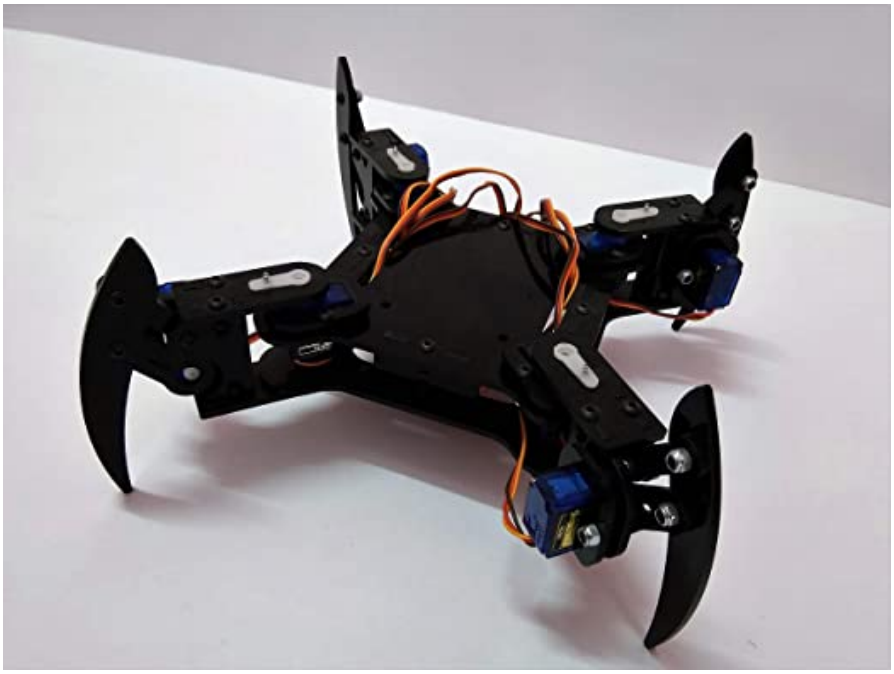


Figure 2.2: MePed Perspective View

The MePed comes as a DIY kit with the chassis and required screws. The Arduino Mega, the sensor shields and the servos were acquired based on the project's requirements. The Arduino Mega has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, four UARTS (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the sensor, communication and power supply modules needed for the quadruped. Moreover, it has 256 kB flash memory, 8 kB SRAM and 4 kB EEPROM which is just sufficient for a small neural network. The sensor shield provides enough pins to interface with all the peripherals to be mounted onto the quadruped.

The Sensor Shield provides unique pins for Ultrasonic Sensor Interface, Bluetooth Interface, SD card interface and a Wireless Module Interface. Of the available interfaces, Ultrasonic Sensor Interface and the Wireless Module Interface are currently occupied. The application of the ultrasonic sensor is described in detail in Section 2.2.1. The wireless module serves to interface the quadruped with a remote control or a computer for logging sensor information. A CMOS camera also interfaces with the sensor shield. Though there is no interface for the camera on the sensor shield, the camera connects to the controller by connecting to appropriate analog, PWM, power and ground pins.

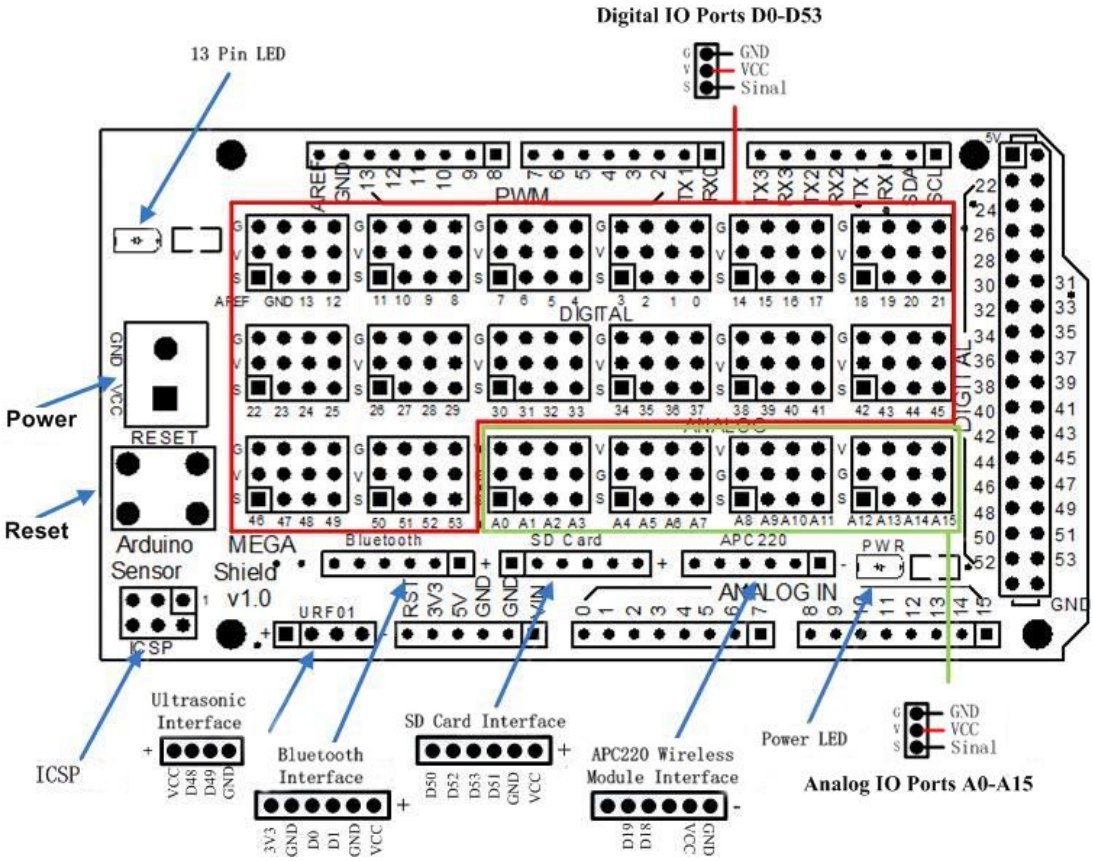


Figure 2.3: Sensor Shield V2 for Arduino Mega 2560

2.2 Sensors

The quadrupes is to be mounted with the following four sensors.

1. Ultrasonic Sensor
2. CMOS Camera
3. IMU Sensor
4. Contact Sensor

2.2.1 Ultrasonic Sensor

The ultrasonic sensor is used to detect the distance of the nearest object in front of the quadruped. The sensor emits short, high-frequency sound pulses at regular intervals. These propagate in the air at the velocity of sound. If they strike an object, then they are reflected back as echo signals to the sensor, which itself computes the distance to the target based on the time-span between emitting the signal and receiving the echo.

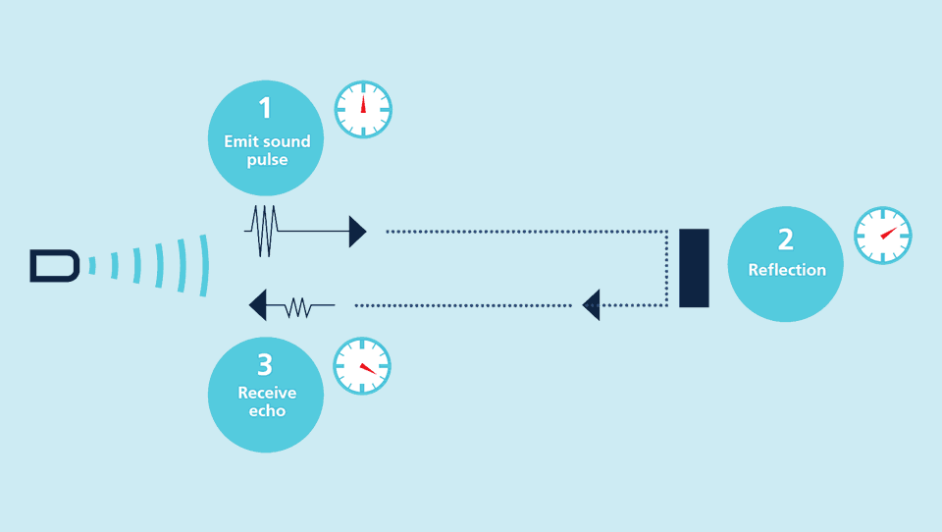


Figure 2.4: Working Principle Ultrasonic Sensor [1]

Figure 2.4 illustrates the working principle of the ultrasonic sensor. The sensor used can detect objects within a range of 2 cm to 450 cm and within an arc of $15°$.

2.2.2 CMOS camera

The REES52 OV7670 Robodo OV7670 640x480 VGA CMOS Camera Image Sensor is a high sensitivity color image sensor. The sensor has high sensitivity for low-light operation, low operating voltage for embedded applications. The sensor operates at 15 fps and produces 640×480 images. The sensor also supports additional features such as image scaling, lens shading correction, flicker 50/60Hz auto detection and color saturation level auto adjust. The sensor provides vision capabilities to the DNN-CPG sensorimotor processing model.

Figure 2.5: REES52 OV7670 Robodo CMOS Image Camera Sensor

2.2.3 IMU

IMU is a 9-axis sensor that measures orientation, velocity, and gravitational forces by combining Accelerometer, Gyroscope, and Magnetometer into one. IMU is used to obtain the following values for the construction of a quadruped state.

1. Orientation
2. Linear Acceleration
3. Angular Velocity

An IMU is not currently mounted on the MePed, but is present in the simulation model and plays a crucial role in controller learning.

2.2.4 Contact Sensor

The contact or the bump sensor is used to detect collision of the legs with the ground. It serves to sense the stance and the swing legs of the MePed. The sensor works with the help of a switch which gives the system a touch effect so that it can be used as a touch sensor. Though the bump sensor is not currently mounted on the MePed, it is present in the simulation model and plays a crucial role in controller learning.

2.3 Physics Simulation

2.3.1 CAD Model

Being a low-cost open-source hardware model, only STL(Standard Tessellation Language) were available for the MePed. Due to the need to formulate the quadruped kinematics and dynamics, a detailed description of the quadruped physical characteristics, such as the mass, dimensions and centre of mass of the constituent parts, and the quadruped, were needed. Moreover, such information was valuable for the development of a more accurate URDF description of the MePed.

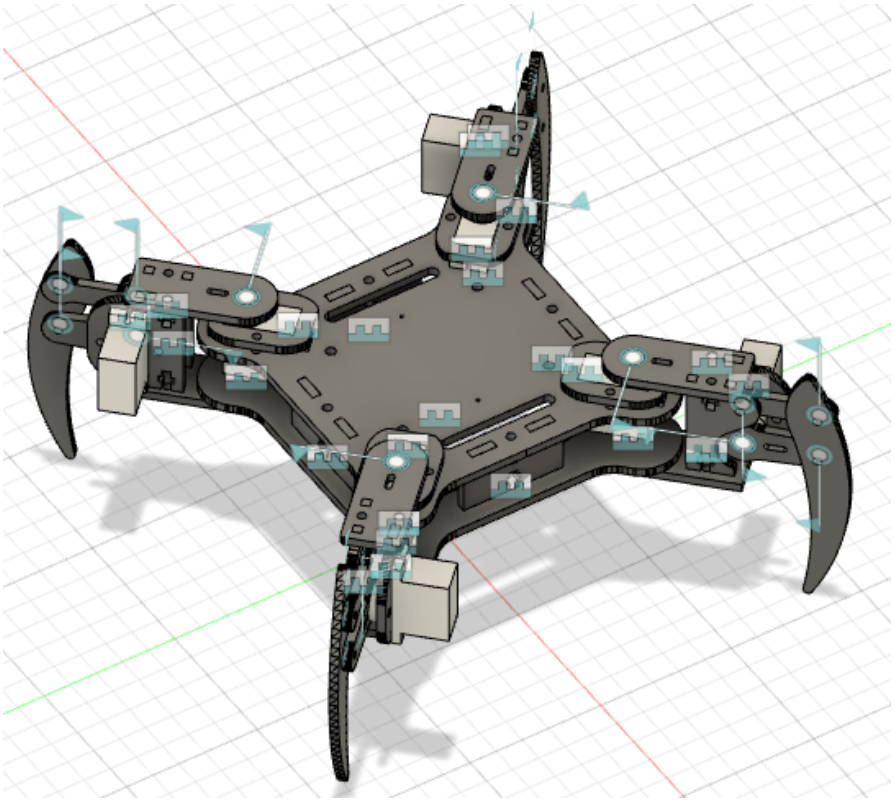


Figure 2.6: CAD model of the MePed assembled using AutoDesk Fusion 360

The MePed is 3D printed using a variant of Acrylic of density 0.00157 $\mathrm{g\,mm^{-3}}$. The CAD model of the MePed is available publicly[1].

Table 2.1: List of Components of the MePed Assembly and their physical attributes.

Component	Mass (g)	Bounding Box Dimensions ($L \times W \times H$ mm)
Body Bottom Plate	35.2	122.0×122.0×3.2
Body Spacer	4.2	38.3×3.2×25.5
Body Top Plate	51.8	122.0×122.0×3.2
Leg Rev	5.4	17.1×17.1×96.2
Leg Parallel Plate	3.1	19.3×19.3×42.8
Leg Parallel Linkage	1.6	28.2×28.2×11.3
Leg Servo Arm	2.1	29.9×29.9×15.9
Leg Servo Mount	4.03	24.8×24.8×42.8
Leg Servo Retainer	2.5	35.0×35.0×3.2
Leg Top Pivot Plate	4.4	43.8×43.8×3.2
Sensor Shield V2	40.0	100.0×54.0×18
Arduino Mega 2560	37.0	101.5×53.3×20
Tower Pro SG90 Micro Servo	9.0	32.3×32.3×30.0

[1]The CAD model can be found at the following link: https://a360.co/3eyVJS8

2.3.2 URDF Description

Universal Robot Description Format (URDF) is an XML based domain specific modelling language used to describe a robot's layout and appearance and to specify additional information for kinematic and dynamic description of the robot like joint limits, mass, friction values and so on. A URDF description is required for building any robotic simulation.

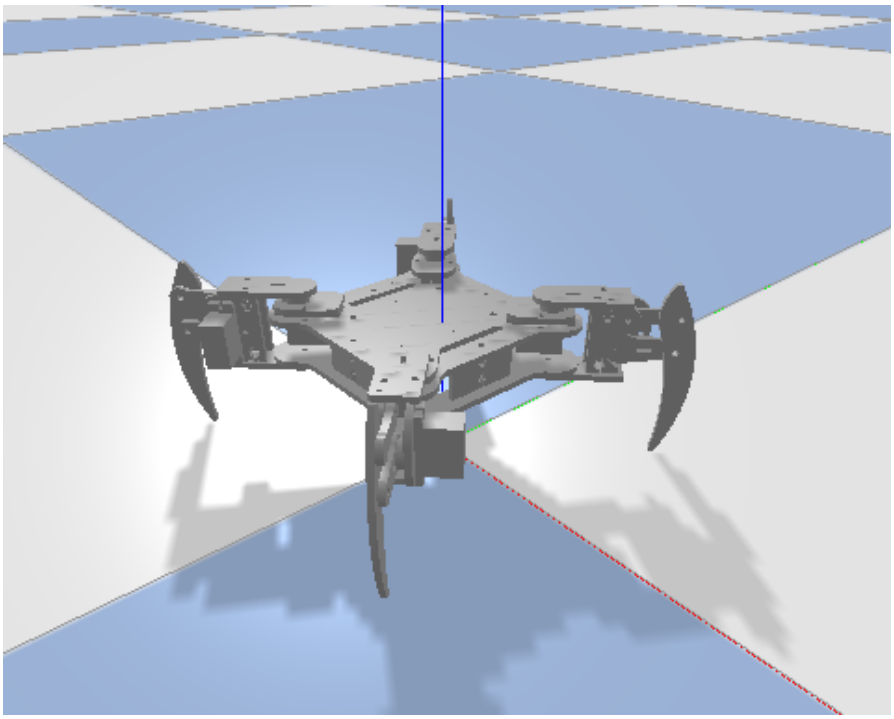


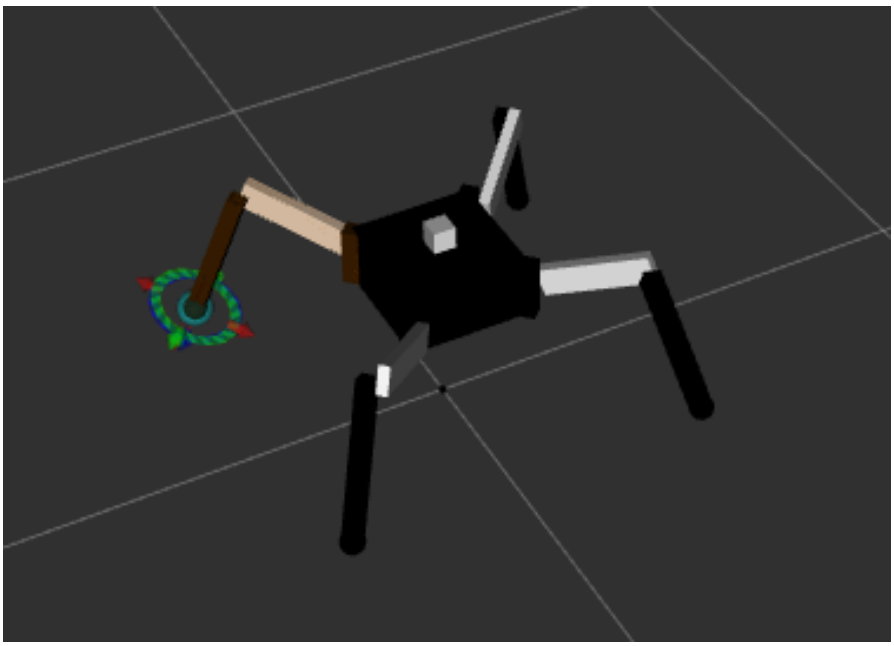
Figure 2.7: PyBullet simulation the MePed using URDF

The URDF description of the MePed was generated using a Fusion360 plugin called *fusion2urdf*[2]. The collision model was further modified to reduce the computational expense and time of computing the physics of the model. The modifications made involved the replacement of STL mesh files for generating collision boundaries with simple cuboidal shapes defined using the component bounding box dimensions in Table 2.1. Though a reduction in computational expenses and time was seen, further reduction were required for parallel execution of the physics engine and the DDPG for RL.

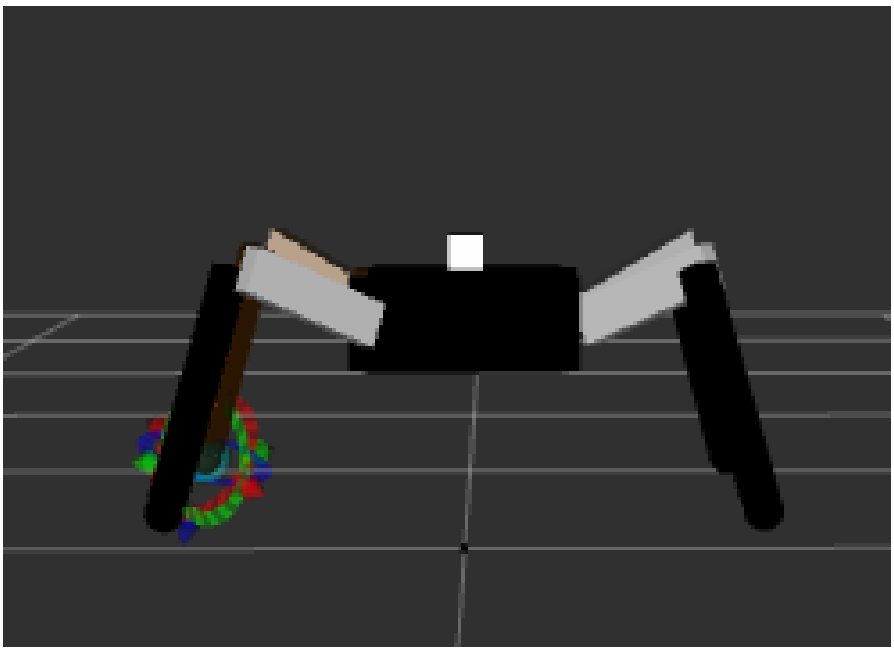
[2]The plugin is available at the following link: https://github.com/syuntoku14/fusion2urdf

2.3.3 Simplified Quadruped Description

Due to high computational expenses of the collision model of the MePed, it can not be directly used for training the DDPG. The high computational requirements are mainly due to the four bar linkages at the knee joints and a large number of small components interacting simultaneously. Thus, a simplified URDF description was developed for the MePed. Figure 2.8 illustrates the simplified URDF description.



(a) Perspective View



(b) Front View

Figure 2.8: Simplified URDF description of the MePed

The simplified description has 12 DoF instead of 8, as in the MePed. The additional degreed of freedom were introduced to emulate the kinematics of the MePed. Due to four-bar linkage as the knee joint, the end-effector motion for the MePed is not like a typical 8 DoF quadruped. Figure 2.9 illustrates the trajectories of the leg endpoints. For a typical 8 DoF quadruped, a knee joint movement of 30° would imply proportionate movement along the x, y and z axes. However, due to the four-bar linkage, the movement along the x and y axes is diminished. An additional joint was introduced in each leg to mimic the same behaviour in the simplified URDF description. The additional joint is constrained such that a trajectory similar to Figure 2.9 is obtained. For simplicity, the additional joint's movement is constrained to be 0.01 times the knee joint in the opposite direction. An unintended advantage of such a URDF description is the increased generalisability of the simulation environment developed.

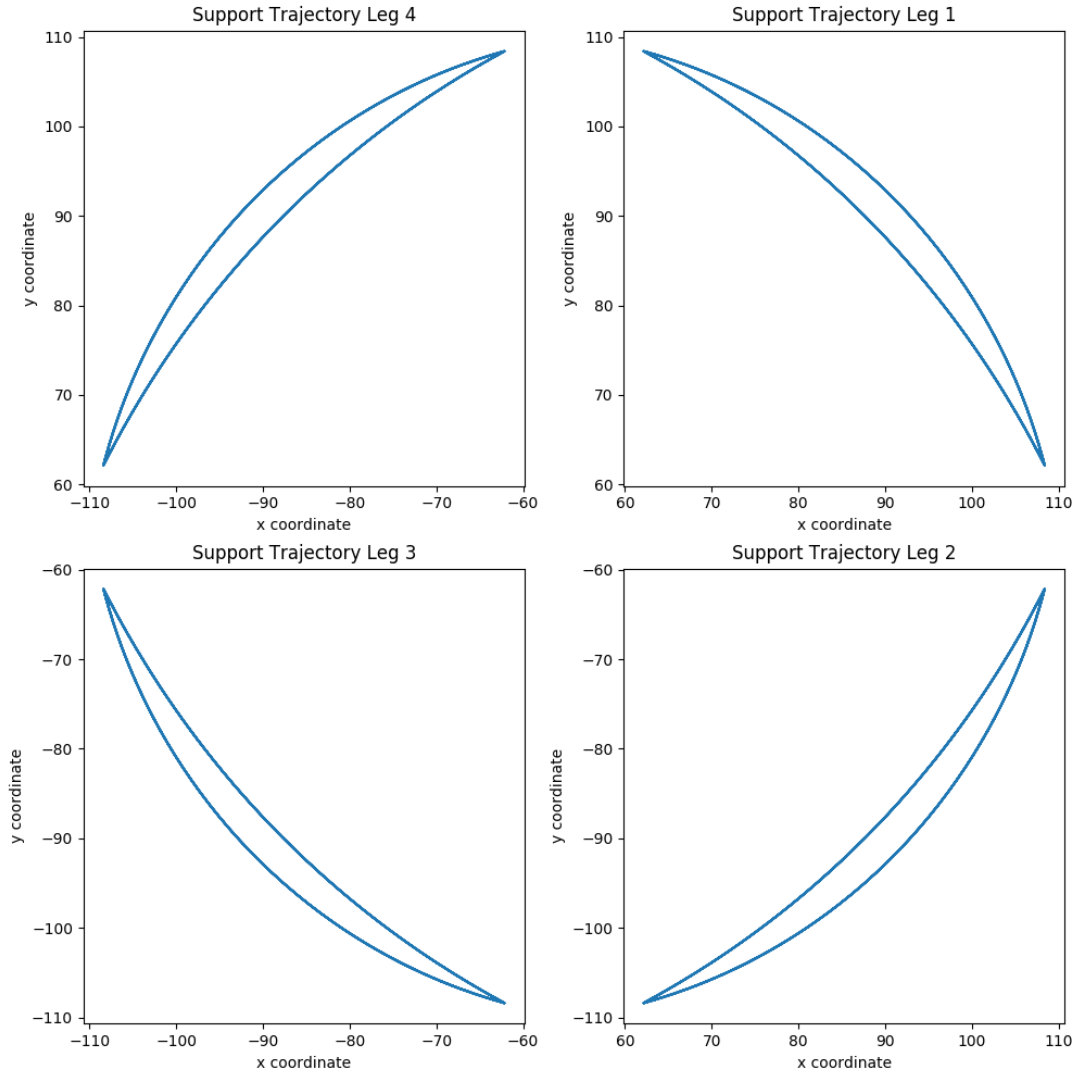


Figure 2.9: Leg End Point Trajectory in the XY plane for the MePed

2.4 Kinematics & Dynamics

Embodied Cognition spans the brain, body and environment. Perception and decision making involves the brain and the environment. A model of the body's physical behaviour must be known to include the body in the process of embodied decision making. The kinematics and dynamics of the MePed were studied to formulate the kinematics and dynamics of the simplified quadruped description.

2.4.1 MePed Kinematics & Dynamics

Quadruped motion results from forward propulsion caused by the friction forces as a leg in moves backwards during the stance phase of its gait cycle. Figure 2.10 depicts the forces acting on the quadruped due to the movement of its legs and the resulting acceleration. The resulting motion of the quadruped is not in a straight line, rather a lateral displacement also occurs as seen in Figure 2.11.

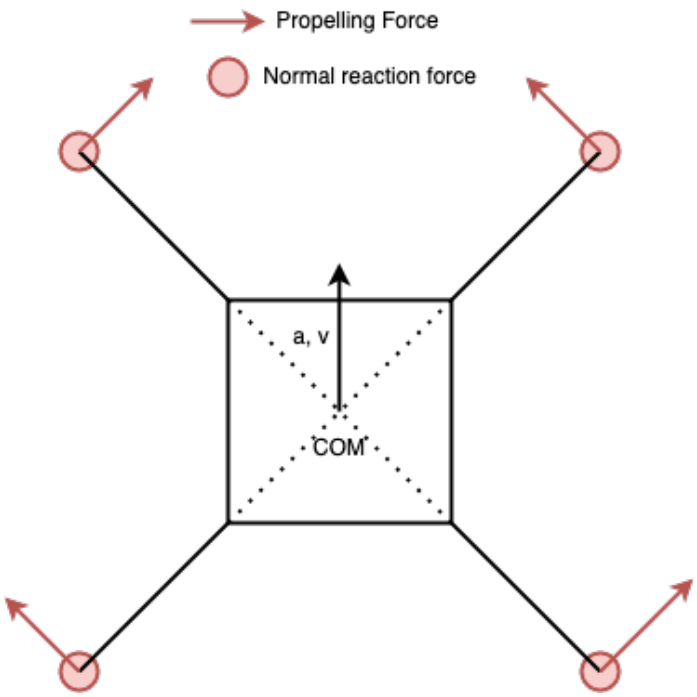


Figure 2.10: Simplified Model of MePed Dynamics as seen from the top

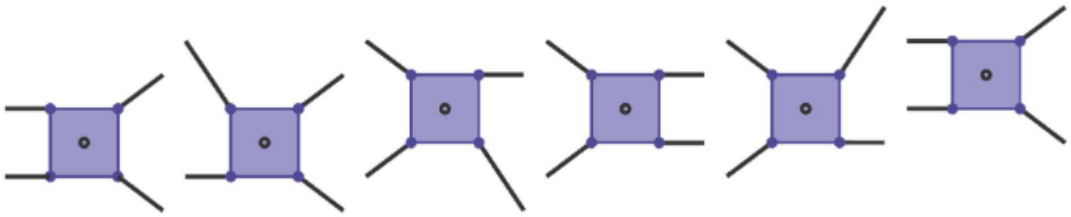


Figure 2.11: Quadruped Pose during a single period of Gait Cycle [2].

The kinematics and dynamics of the four-bar linkage in the knee were studied closely to determine the corresponding behaviour of the additional joint introduced in the simplified quadruped description. The four-bar linkage was fixed at the points F and O to the thigh. A TowerPro SG90 Micro Servo actuated the joint at O. Due to the knee joint's unique construction, a movement of an angle θ at O does not imply the same movement at the leg tip. Figure 2.12 illustrates the leg and all external and internal forces on the leg and the four-bar linkage at the knee. The equations 2.1 through 2.28 depicts the four-bar linkage's kinematics used to determine the behaviour of the additional joint in the simplified quadruped description.

$$l = \sqrt{|\vec{r_0}|^2 + |\vec{r_1}|^2 - 2|r_0||r_1|cos(\theta_1 - \phi + \pi)} \tag{2.1}$$

$$\beta_1 = \arcsin(\frac{|\vec{r_1}|}{l} sin(\theta_1 - \phi + \pi)) \tag{2.2}$$

$$\beta_2 = \arccos \frac{|\vec{r_2}|^2 + l^2 - |\vec{r_3}|^2}{2|\vec{r_3}|l} \tag{2.3}$$

$$\delta = \arcsin(\frac{l}{|\vec{r_3}|} sin\beta_2) \tag{2.4}$$

$$\theta_2 = \beta_2 - \beta_1 + \phi - \pi \tag{2.5}$$

$$\theta_3 = -\beta_1 - \delta + \phi - \pi \tag{2.6}$$

$$\vec{r_i} = [L_i \cos\theta_i, L_i \sin\theta_i, 0]^\top \quad \forall \quad i \in \{1, 2, 3\} \tag{2.7}$$

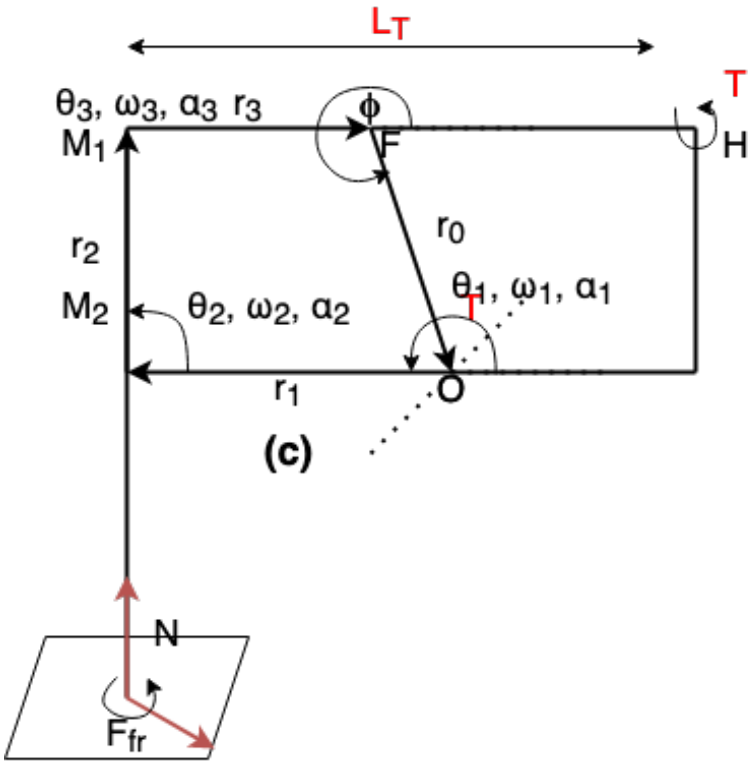
$$\vec{r_0} = [L_0 \cos\phi, L_0 \sin\phi, 0]^\top \tag{2.8}$$

$$\vec{r_1} + \vec{r_2} + \vec{r_3} + \vec{r_0} = \vec{0} \tag{2.9}$$

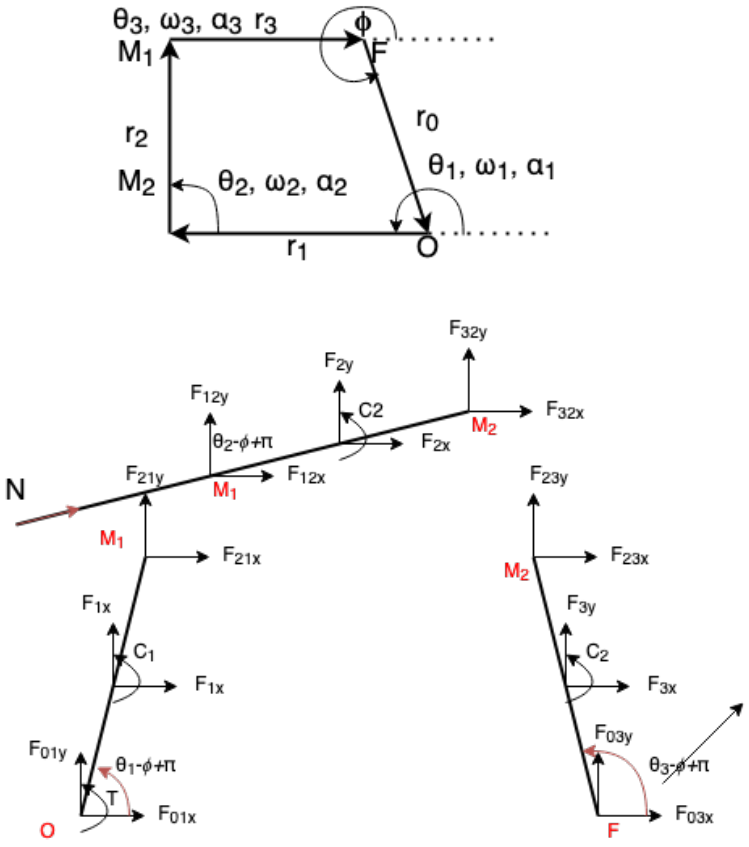
$$\vec{v_0} + \vec{v_1} + \vec{v_2} + \vec{v_3} = \vec{0} \tag{2.10}$$

$$\vec{a_0} + \vec{a_1} + \vec{a_2} + \vec{a_3} = \vec{0} \tag{2.11}$$

$$\vec{\omega_i} = [0, 0, \omega_{iz}]^\top \quad \forall \quad i \in \{1, 2, 3\} \tag{2.12}$$



(a) Model of a MePed Leg



(b) Four Bar Linkage in the MePed Knee

Figure 2.12: Model of a MePed Leg and the Four Bar Linkage at the Knee

$$\vec{\omega}_0 = [0,0,0]^\top \tag{2.13}$$

$$\vec{v}_0 = [0,0,0]^\top \tag{2.14}$$

$$\vec{v}_i = \vec{\omega}_i \times \vec{r}_i \quad \forall \quad i \in \{1,2,3\}, \tag{2.15}$$

$$\vec{v}_i = [-\omega_{iz} r_{iy}, \omega_{iz} r_{ix}, 0]^\top \quad \forall \quad i \in \{1,2,3\} \tag{2.16}$$

$$\omega_{2z} = -\omega_{1z} \frac{r_{1y} r_{3x} - r_{3y} r_{1x}}{r_{2y} r_{3x} - r_{3y} r_{2x}} \tag{2.17}$$

$$\omega_{3z} = -\omega_{1z} \frac{r_{2y} r_{1x} - r_{1y} r_{2x}}{r_{2y} r_{3x} - r_{3y} r_{2x}} \tag{2.18}$$

$$\vec{\alpha}_0 = [0,0,0]^\top \tag{2.19}$$

$$\vec{\alpha}_i = [0,0,\alpha_{iz}]^\top \quad \forall \quad i \in \{1,2,3\} \tag{2.20}$$

$$\vec{a}_0 = 0 \tag{2.21}$$

$$\vec{a}_i = \vec{\alpha}_i \times \vec{r}_i + \vec{\omega}_i \times \vec{\omega}_i \times \vec{r}_i \quad \forall \quad i \in \{1,2,3\} \tag{2.22}$$

$$\beta_3 = -\alpha_{1z} r_{1y} - \omega_{1z}^2 r_{1x} - \omega_{2z}^2 r_{2x} - \omega_{3z}^2 r_{3x} \tag{2.23}$$

$$\beta_4 = -\alpha_{1z} r_{1x} - \omega_{1z}^2 r_{1y} - \omega_{2z}^2 r_{2y} - \omega_{3z}^2 r_{3y} \tag{2.24}$$

$$\alpha_{2z} = \frac{r_{3x}\beta_3 - r_{3y}\beta_4}{r_{2y} r_{3x} - r_{3y} r_{2x}} \tag{2.25}$$

$$\beta_5 = -\alpha_{1z} r_{1y} - \omega_{1z}^2 r_{1x} - \omega_{2z}^2 r_{2x} - \omega_{3z}^2 r_{3x} \tag{2.26}$$

$$\beta_6 = -\alpha_{1z} r_{1x} - \omega_{1z}^2 r_{1y} - \omega_{2z}^2 r_{2y} - \omega_{3z}^2 r_{3y} \tag{2.27}$$

$$\alpha_{3z} = \frac{-r_{2x}\beta_5 + r_{2y}\beta_6}{r_{2y} r_{3x} - r_{3y} r_{2x}} \tag{2.28}$$

Since O is the driven joint in the knee, θ_1, ω_1 and α_1 are known and can be directly plugged into the equations above to obtain the kinematic parameters of the knee. Furthermore, the following equations determine the dynamics of the knee four bar linkage.

$$F_{01x} + F_{1x} + F_{21x} = 0 \tag{2.29}$$

$$F_{01y} + F_{1y} + F_{21y} = 0 \tag{2.30}$$

$$F_{01y} + F_{1y} + F_{21y} = 0 \tag{2.31}$$

$$F_{01x} + F_{1x} + F_{21x} = 0 \tag{2.32}$$

$$\vec{F}_{23} = -\vec{F}_{32} \tag{2.33}$$

$$\vec{F}_{21} = -\vec{F}_{12} \tag{2.34}$$

$$C_i = 0 \quad \forall \quad i \in \{1, 2, 3, \} \tag{2.35}$$

$$F_{12x} + F_{2x} + F_{32x} + N\cos\theta_2 - \phi + \pi = 0 \tag{2.36}$$

$$F_{12y} + F_{2y} + F_{32y} + N\sin\theta_2 - \phi + \pi = 0 \tag{2.37}$$

$$\begin{aligned} T + C_1 - F_{1x}\frac{|\vec{r_1}|}{2}\cos\theta_1 - \phi + \pi + F_{1y}\frac{|\vec{r_1}|}{2}\cos\theta_1 - \phi + \pi \\ + F_{21y}|\vec{r_1}|\cos\theta_1 - \phi + \pi - F_{21x}|\vec{r_1}|\sin\theta_1 - \phi + \pi = 0 \end{aligned} \tag{2.38}$$

$$\begin{aligned} C_2 - F_{2x}\frac{|\vec{r_2}|}{2}\cos\theta_2 - \phi + \pi + F_{2y}\frac{|\vec{r_2}|}{2}\cos\theta_2 - \phi + \pi \\ + F_{32y}|\vec{r_2}|\cos\theta_2 - \phi + \pi - F_{32x}|\vec{r_2}|\sin\theta_2 - \phi + \pi = 0 \end{aligned} \tag{2.39}$$

$$\begin{aligned} C_3 - F_{3x}\frac{|\vec{r_3}|}{2}\cos\theta_3 - \phi + \pi + F_{3y}\frac{|\vec{r_3}|}{2}\cos\theta_3 - \phi + \pi \\ - F_{23y}|\vec{r_1}|\cos\theta_1 - \phi + \pi + F_{23x}|\vec{r_1}|\sin\theta_1 - \phi + \pi = 0 \end{aligned} \tag{2.40}$$

2.4.2 Simplified Description Kinematics & Dynamics

The net force and torque acting on the quadruped are of more importance than the internal forces acting within the quadruped. The net force and torque are used for the quantification of the stability of quadruped gait. Equations 2.41 and 2.42 describe the overall dynamics of a quadruped.

$$\sum \vec{F}_{fr} + \sum \vec{N} - m\vec{g} = m\vec{a} \tag{2.41}$$

$$\sum_i \vec{F}_{fr} \times R_i + \sum_i \vec{N} \times R_i = I_R \vec{\alpha_R} \tag{2.42}$$

Gazebo and ROS were used for simulation of the simplified quadruped description. They provides a number of plugins for sensor integration into the simulations. Moreover, a number of kinematics and dynamics solvers are available that can use the integrated sensor information to calculate relevant kinematics and dynamics parameters for

the quadruped. The following sensor plugins are currently integrated with the simulation.

1. IMU plugin
2. Gazebo Bumper Plugin

The following ROS packages are used for calculating the kinematics and dynamics of the quadruped.

1. MoveIt!
2. Orocos KDL solver
3. tf2

Equations 2.41 and 2.42 form the basis of the all the dynamics calculations performed by the three packages. *tf2* and *MoveIt* create a real-time kinematic model of a simulated robot to calculate end-effector positions and other kinematics parameters. Figure 2.13 depicts the chains that *tf2* generates for calculating coordinate transformations.

2.5 Summary

The simulation environment developed for the MePed and its simplified quadruped description serves as the training environment for RL with the DNN-CPG controller. Moreover, the environment has been developed so that it can be used for other generic quadruped simulation tasks such as path planning. The study of kinematics and dynamics of MePed form the basis for the formulation of the reward functions for quantification of stability of gait.

Figure 2.13: *tf2* tree of the simplified quadruped description

CHAPTER 3

GAIT LEARNING

3.1 Introduction

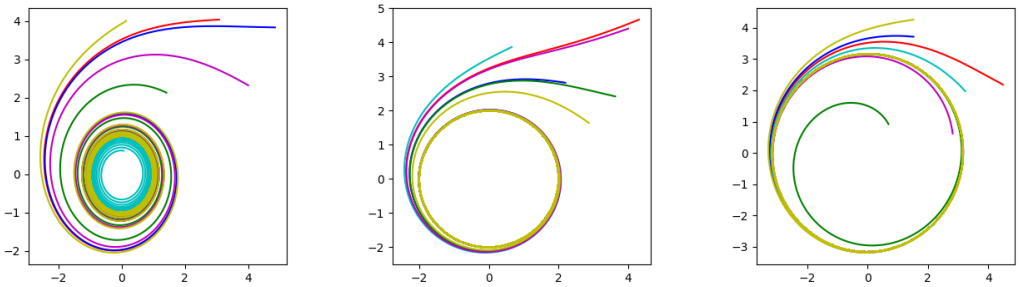
Being inspired by biology, recently researchers have proposed CPGs to generate policies for locomotion of robots ([2], [12], [13], [3]). However, the modulation of gait produced by such policies require high dimensional parameter inputs, unlike in biology, where a low dimensional tonic signal is used. Moreover, learning to produce various gait for challenging environments is difficult, and there is no standard methodology. On the other hand, DNNs enjoy standard learning techniques and the ability of universal approximation. Thus, a combination of DNNs and CPGs could be used to generate gait for robots that could be easily modulated with low dimensional inputs or feedback. The motivation for developing a combined DNN-CPG model was established in detail in Section 1.2. This chapter describes the building blocks of the DNN-CPG architecture and its construction.

3.1.1 Hopf Oscillator Model

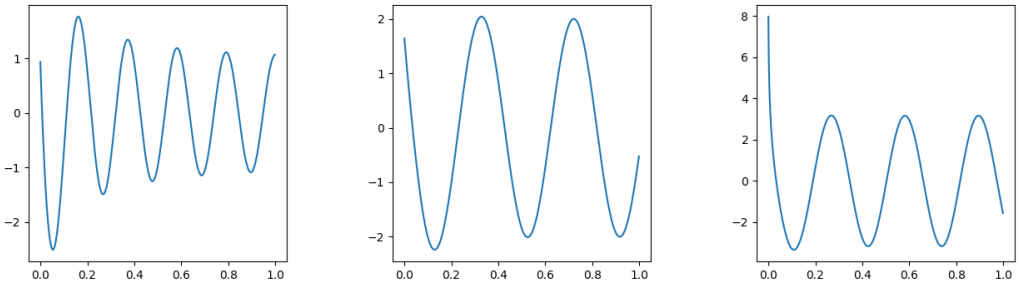
Various types of oscillator models have been robotics so far. Though such oscillators are capable of generating motion trajectories for robots, the relationship between the output variables and the model parameters is complex and hard to understand. Hopf Oscillators, on the other hand, have a simple relationship between the model parameters and the output variables, and they exhibit limit cycle and synchronization behaviour [17]. This makes Hopf Oscillator for modelling the rhythmic motion trajectories for robots. The mathematical model of the Hopf oscillator is represented as follows,

$$\dot{z} = (\mu - \|z\|^2)z + \iota\omega z, \tag{3.1}$$

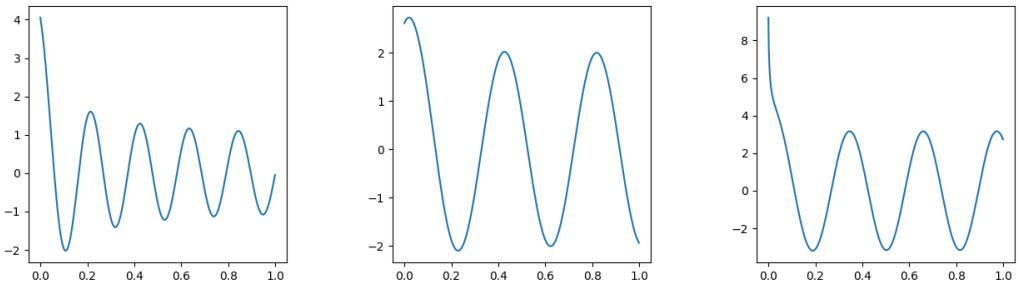
where z is a complex number. The same model can also be represented in the Cartesian coordinate system as follows,



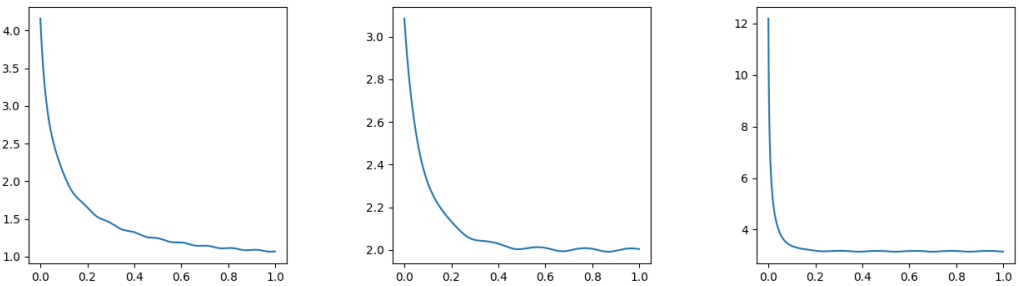
(a) caption (b) caption (c) caption



(d) caption (e) caption (f) caption



(g) caption (h) caption (i) caption



(j) caption (k) caption (l) caption

Figure 3.1: Behaviour of the Hopf Oscillator with different parameter values.

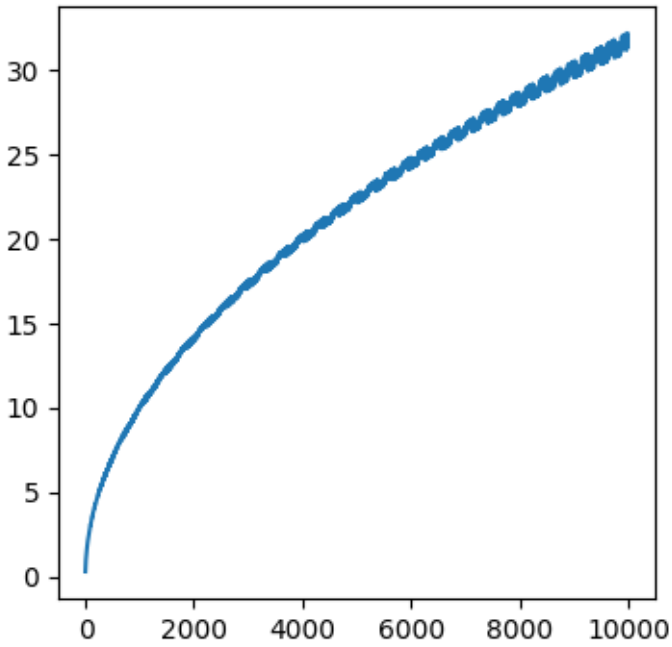


Figure 3.2: Trend of steady state amplitude of oscillations with oscillator parameter μ

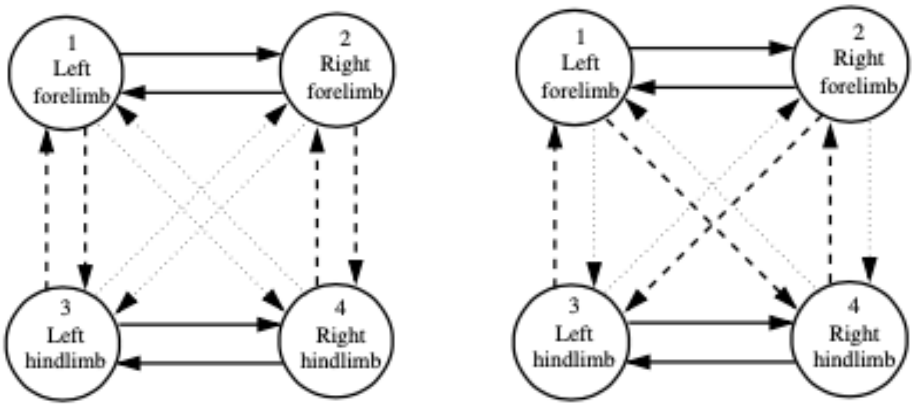
$$\dot{x} = -\omega y + x(\mu - x^2 - y^2) \tag{3.2}$$

$$\dot{y} = \omega x + y(\mu - x^2 - y^2) \tag{3.3}$$

The parameter μ controls the amplitude of steady-state oscillations, and the parameter ω controls the frequency of steady-state oscillations. The amplitude of oscillations settles to a steady-state value of $\sqrt{\mu}$.

3.1.2 Existing CPG Architecture

As previously mentioned, a number of CPG architecture have been successful in producing rhythmic patterns for generating motion trajectories. A number of such architectures for quadrupeds are successive improvements upon the fully connected network of oscillators proposed in [3]. Figure 3.3 describes the coupling structure of the 4 cell network from [3]. Such networks take advantage of the synchronization between oscillators for produce adaptive rhythmic patterns. This model was further extended in [14] by introducing a two-level hierarchy, such that the higher level modulates the CPG model by appropriately modifying CPG parameters. Another architecture of interest is the CPG model, proposed in [4], driving a salamander robot. The CPG model is based



(a) Trot Pace Bound Net (b) Walk Net.

Figure 3.3: Generic coupling structure for a 4 cell network in [3]

on the CPGs observed in a lamprey and has been successfully used to show transition from terrestrial behaviour to aquatic behaviour. Moreover, the motion trajectory generation by this model is controlled by a low-dimensional tonic signal. However, the coupling weights of the CPG are hand tuned and learning with this architecture is extremely difficult. Figure 3.4 depicts the CPG model in [4].

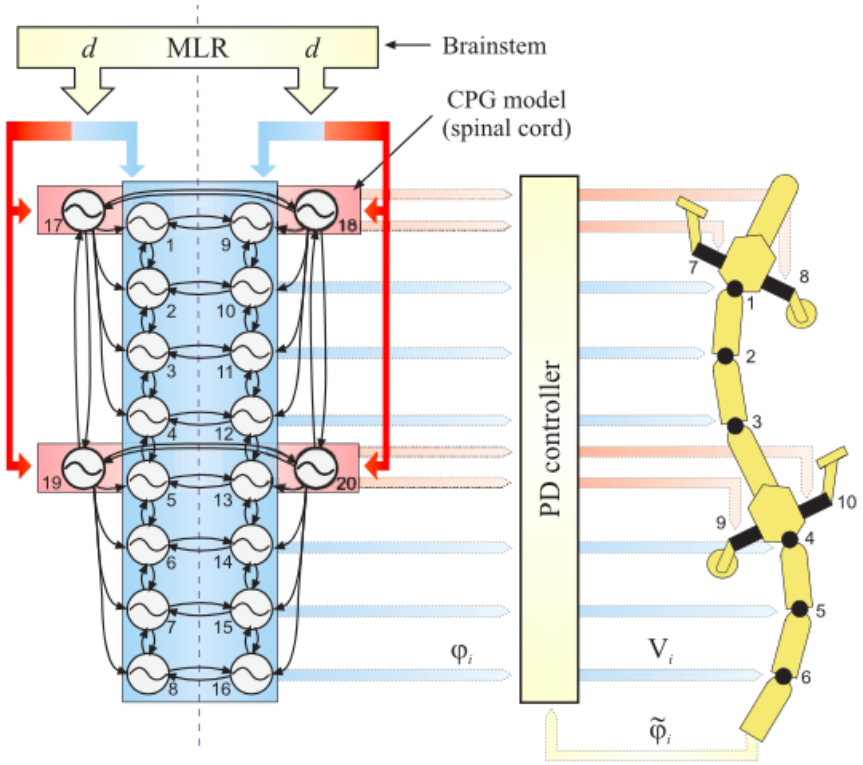


Figure 3.4: Configuration of the CPG driving the Salamander Robot in [4]

A departure from the fully-connected oscillator network architecture was proposed in [5]. CPG-based methods seek to utilise the ability of CPGs to entrain with the body dynamics of the robot. A high-level controller was introduced in [5], modulating the CPG network and leading to greater generalisation abilities and a more effective form of control. The high-level controller is implemented as a fully-connected, feed-forward neural network with two hidden layers containing 400 and 300 ReLU units, respectively and is trained to minimise the lateral deviation through RL. Figure 3.5 depicts the detailed DNN-CPG architecture for stable bipedal gait. The DNN-CPG proposed in this work builds upon the architecture in Figure 3.5 and attempts to integrate the CPG into the DNN for simultaneous optimisation of parameters.

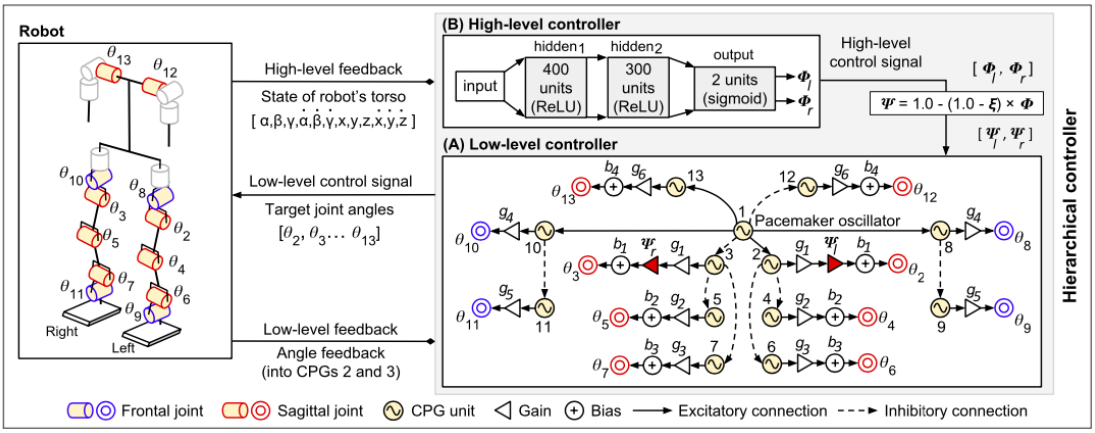


Figure 3.5: Detailed DNN-CPG architecture proposed in [5]

3.2 DNN-CPG

A DNN-CPG architecture to generate rhythmic motion trajectories for robotic control application must be able to generate rhythms, interpret a low dimensional modulating input and appropriately modulate the generated rhythms. In such an architecture, the CPG performs rhythm generation, whereas the DNN interprets inputs and appropriately modulates the CPG parameters to modulate generated rhythm. For instance, in the DNN-CPG architecture proposed in [5], the higher-level controller outputs two parameters ψ_l and ψ_r. These parameters then directly modify the amplitude of the hip joint rhythms to maintain balance.

3.2.1 Fourier Decomposition of Signals

A periodic signal may be represented as a (possibly infinite) weighted summation of harmonically related sinusoids. For instance, a square wave may be represented as a sum of sinusoids to varying accuracy. Figure 3.6 depicts reconstruction of a square wave from its fourier components.

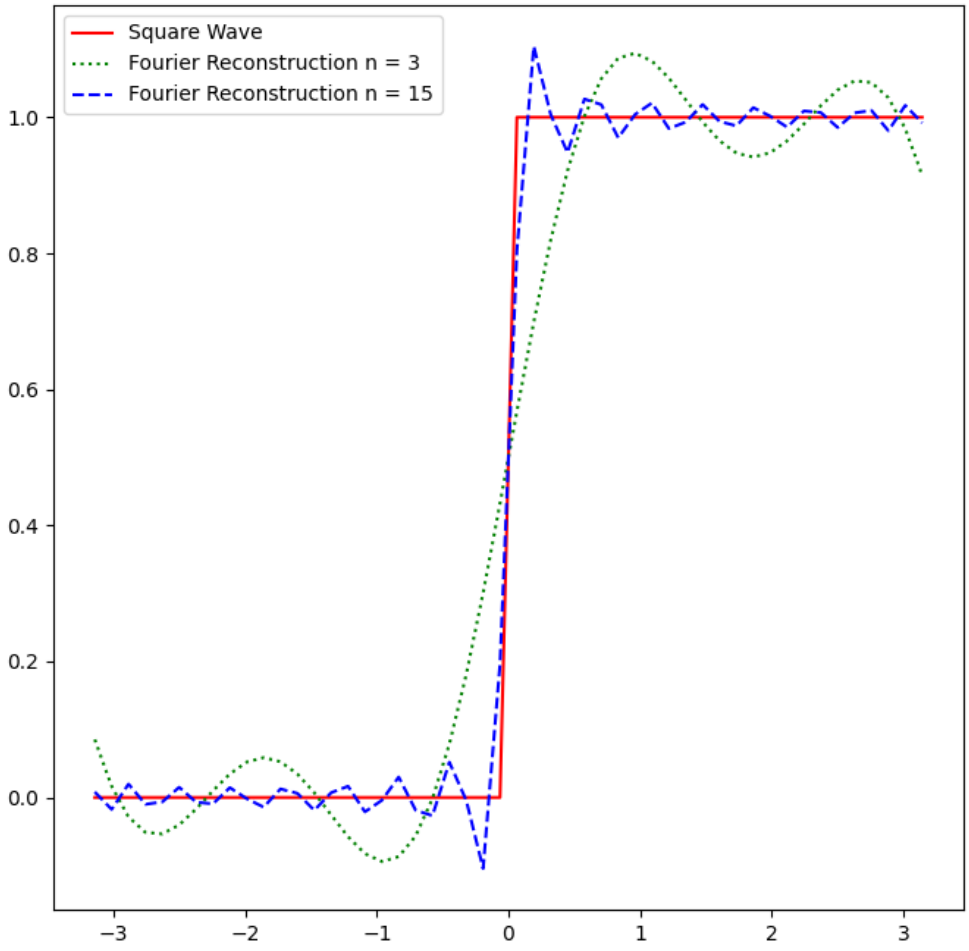


Figure 3.6: Fourier Series Reconstruction of a Square Wave with n fourier components

The limit cycle exhibited by the Hopf Oscillator at steady state can be leveraged to perform fourier reconstruction of a periodic signal, given that the fourier coefficients for the periodic signal are known. The following equation depicts fourier decomposition of a signal.

$$s(t) = \sum_{k=-\infty}^{\infty} C_k e^{\iota k \omega_o t} \tag{3.4}$$

The Hopf Oscillator, being a complex exponential at steady state, can be used as the Fourier Reconstruction basis. Thus, the Fourier Decomposition can be rewritten as

follows,

$$s(t) = \sum_{k=-\infty}^{\infty} C_k z_k(t) \tag{3.5}$$

where $z_k(t)$ is the dependent variable of a Hopf Oscillator with frequency of oscillations (ω) equal to $k\omega_o$ and ω_o is the fundamental frequency of the periodic signal $s(t)$. Thus, leveraging the limit cycle behaviour of the Hopf Oscillator, n Hopf Oscillators, each with a frequency, an integer multiple of ω_0 can be used to approximately reconstruct $s(t)$. This is similar to how the Fourier components of the square waves were used to reconstruct it to varying degrees of accuracy in Figure 3.6.

3.2.2 Motion Trajectory Generation

So far, we have established that a collection of Hopf Oscillators can be used to reconstruct any periodic signal to some degree of accuracy, given that the Fourier coefficients are known. The property of universal approximation of neural networks can be utilised to combine the simple harmonics from the Hopf Oscillator and produce rhythmic motion trajectories. Figure 3.7 depicts the neural network used to generate motion trajectories for particular values of ω and μ.

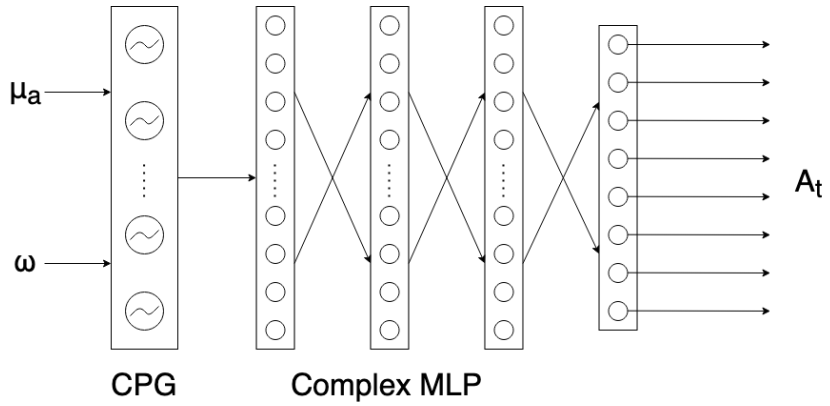


Figure 3.7: A Simple DNN-CPG Architecture for generating rhythmic motion trajectories given ω and μ

The Complex MLP takes the harmonics produced by the CPG, produces the appropriate coefficients, multiplies the harmonics, and adds the scaled harmonics to produce motion trajectories. The neural network depicted in 3.7 produces, for instance, the motion trajectories for the 8 DoF in the MePed. The network consists of a single layer of Hopf Oscillators, which constitute the CPG and several fully connected complex-valued dense layers connected in a feed-forward fashion.

3.2.3 Motion Trajectory Modulation

The simple DNN-CPG architecture depicted in Figure 3.7 can generate rhythmic motion trajectories, but has no means of modulation or interpretation of inputs. Moreover, any modulation of the needs to performed by directly modifying CPG parameters ω and μ_a. Thus, a motion encoding network is used to interpret desired motion trajectory inputs and output appropriate ω and μ. The motion encoding network consists of several fully connected feed-forward layers. The desired motion trajectory inputs is a concatenation of the following two vectors:

1. A Heading Vector
2. A Desired Robot Velocity Vector

To facilitate interpretation of feedback from sensors, a robot state encoding network is used. The robot state encoding network is also a multi layered fully connected feed-forward network. It takes the robot state as input. The robot state is a concatenation of the following vectors:

1. Current Joint Angles
2. Difference between current and last joint positions
3. Robot Orientation Vector
4. Robot Angular Velocity Vector
5. Robot Linear Acceleration Vector

Another network several fully connected and convolutional layers will also be introduced to emulate more sophisticated behaviour such as obstacle negotiation and path planning. The convolutional network will take as input the images from a camera mounted on the robot. The outputs of the motion encoding network, the robot state encoding network and the aforementioned convolutional network will be concatenated and directly passed onto the Complex MLP. Such an architecture would facilitate motion trajectory modulation without disturbing motion trajectory generation. Figure 3.8 depicts the described DNN-CPG architecture.

Each component of the CNN-CPG architecture in 3.8 may be mapped to a part of the nervous system. For instance, the CPG and the complex MLP constitute the spinal chord, the robot state encoding network constitute the proprioceptory pathways, the

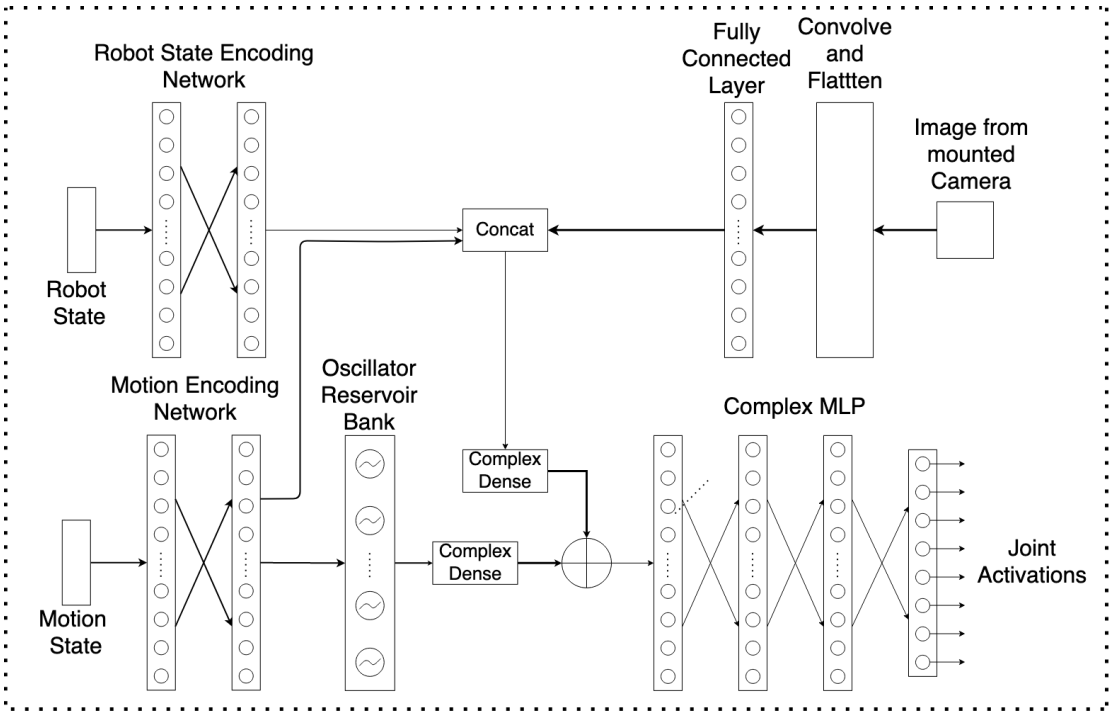


Figure 3.8: DNN-CPG architecture to ensure stable gait.

convolutional neural network constitutes the visual pathways and the motion encoding network constitutes the pedunculopontine nucleus in the lower brain stem whose outputs drives the CPG.

3.2.4 Learning

An incremental approach to learning behaviour is followed. The modular architecture of the DNN-CPG allow for addition of modules for emulation of more sophisticated behaviour in parallel to the existing architecture. This allows for a multi loop hierarchical control system. The parallel computation that such a control system executes is very similar to how the brain operates. Thus, some of the characteristics that are expected to develop naturally in the control system are as follows:

1. Multiple Goal fulfillment
2. Multiple Sensor Information Fusion
3. Additivity of sophisticated behaviour

Figure 3.9 depict a decomposition of the proposed control system. In this work, both supervised learning through DL and unsupervised learning through RL are used.

Supervised learning is used to provide a robust initialization to the DNN-CPG for quick convergence through RL.

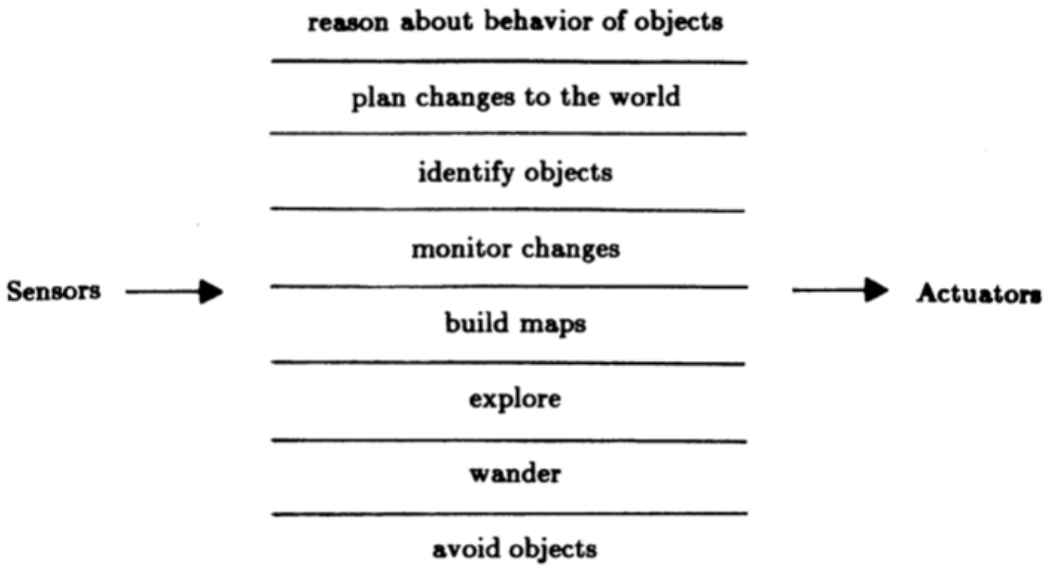


Figure 3.9: A Decomposition of a mobile robot control system based on task achieving behaviours [6]

3.3 Supervised Learning

Supervised Learning is used for the initialization of DNN-CPG for RL. It is also used to prove the ability of the network to perform as expected. Ideal Motion Trajectory and its relationship with the speed of the quadruped were formulated. Supervised Learning was then performed in two steps. First, the simple DNN-CPG in Figure 3.7 was trained then the entire architecture in Figure 3.8 (without the visual pathway) was trained.

3.3.1 Ideal Motion Trajectory

Inspired by gaits of the quadruped mammals, many different types of methods have been proposed to design a gait for the quadruped robot's locomotion like drive function and foot trajectory. The walk gait design based on sinusoidal drive functions, proposed in [18] are used for synthesis of data for supervised learning. The following equations describe the motion trajectories for the driven joints of the MePed. The trajectories may be divided into a stance phase and a swing phase. A leg is in stance phase when it is in

contact with the ground and pushes the robot in the direction of motion, whereas swing phase is during the motion of the leg when not in contact with the ground. Figure 3.10 depict the construction of the drive function for a leg. The drive function is a piece-wise sinusoidal, consisting of a stance phase of lower frequency and a swing phase of a higher frequency. Figure 3.11 depicts the motion trajectories generated from drive function. The joint trajectory for the ankle joint is the same as the joint trajectory for the knee joint scaled by -0.01. Refer to Table 3.1 for the leg index to leg name mapping used in the following equations.

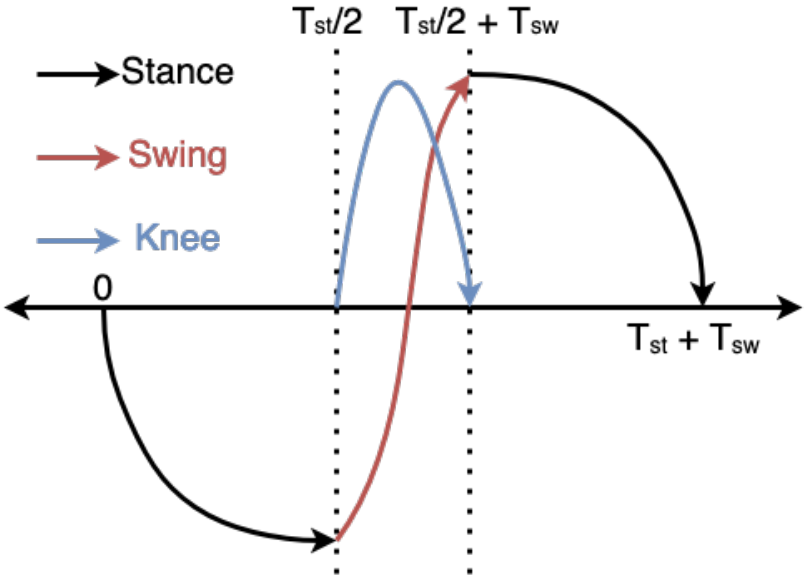


Figure 3.10: Construction of the Drive function for a leg

Table 3.1: Leg Index to Name mapping

Index	Leg Name
0	Front Right Leg
1	Back Right Leg
2	Front Left Leg
3	Back Left Leg

$$T = T_{sw} + T_{st} \tag{3.6}$$

$$\beta = \frac{T_{st}}{T_{st} + T_{sw}} \tag{3.7}$$

$$\theta_h(t) = \begin{cases} \theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \pi), & \text{if } 0 \leq t \leq \frac{\beta T}{2} \\ \theta_h sin(\frac{(t-\frac{iT}{4})\pi}{(1-\beta)T} + \frac{(3-4\beta)\pi}{2(1-\beta)}), & \text{if } \frac{\beta T}{2} \leq t \leq \frac{T(2-\beta)}{2} \\ \theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \frac{(\beta-1)\pi}{\beta}), & \text{if } \frac{T(2-\beta)}{2} \leq t \leq T \end{cases} \quad \forall \quad i \in \{0, 1\} \tag{3.8}$$

$$\theta_h(t) = \begin{cases} -\theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \pi), & \text{if } 0 \leq t \leq \frac{\beta T}{2} \\ -\theta_h sin(\frac{(t-\frac{iT}{4})\pi}{(1-\beta)T} + \frac{(3-4\beta)\pi}{2(1-\beta)}), & \text{if } \frac{\beta T}{2} \leq t \leq \frac{T(2-\beta)}{2} \\ -\theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \frac{(\beta-1)\pi}{\beta}), & \text{if } \frac{T(2-\beta)}{2} \leq t \leq T \end{cases} \quad \forall \quad i \in \{2,3\} \tag{3.9}$$

$$\theta_k(t) = \begin{cases} \theta_k sin(\frac{t\pi}{T(1-\beta)} - \frac{\beta\pi}{2(1-\beta)}), & \text{if } \dot{\theta_h}(t) \geq 0 \\ 0, & \text{otherwise} \end{cases} \quad \forall \quad i \in \{0,1\} \tag{3.10}$$

$$\theta_k(t) = \begin{cases} \theta_k sin(\frac{t\pi}{T(1-\beta)} - \frac{\beta\pi}{2(1-\beta)}), & \text{if } \dot{\theta_h}(t) \leq 0 \\ 0, & \text{otherwise} \end{cases} \quad \forall \quad i \in \{2,3\} \tag{3.11}$$

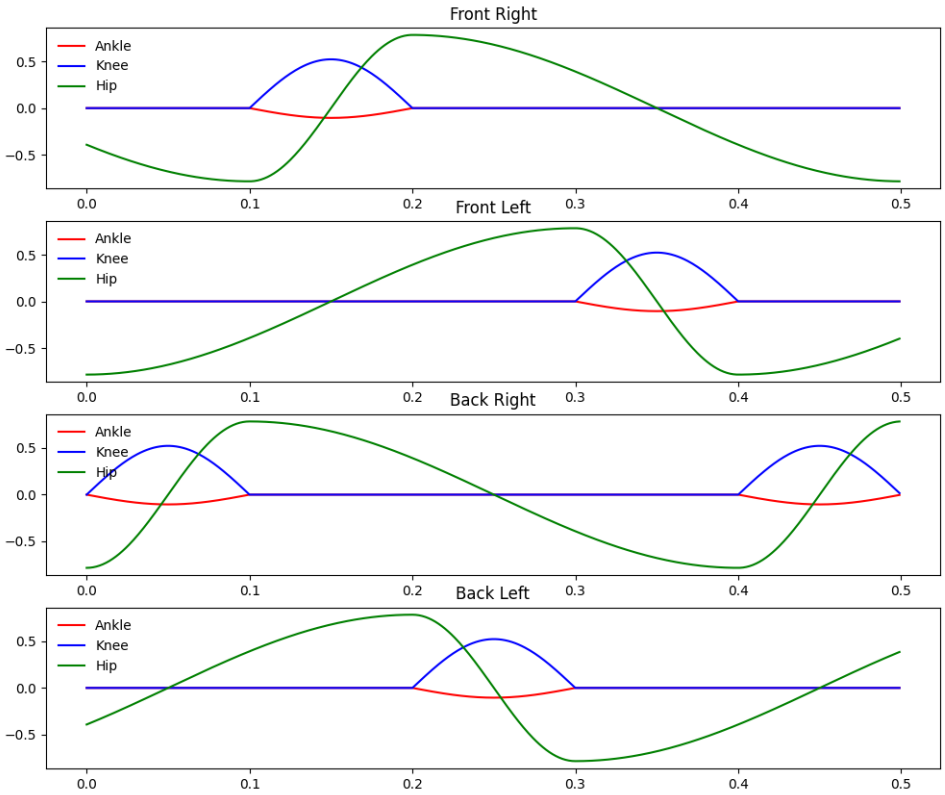


Figure 3.11: Assumed ideal gait pattern for quadruped locomotion

For an ideal motion trajectory, the following relationships hold true

1. $v \propto \frac{1}{T_{sw}}$
2. $v \propto \frac{1}{T_{st}}$
3. $v \propto \theta_h$

These relationships are evident from a superficial analysis of hip joint movement and its impact on robot movement. If the joint moves by an angle $2\theta_h$ from its initial

to final position and then back, the leg sweeps over a distance of $2L_T\theta_h$. This sweep happens over a time period of T, the period of the gait cycle. This analysis can be used to compute the average speed of movement of the robot. Since, the foot remains at a fixed position during the stance phase, the distance $2L_T\theta_h$ is actually covered by the robot in the direction of motion. Thus,

$$v = \frac{2L_T\theta_h}{T_{st} + T_{sw}} \tag{3.12}$$

A similar analysis was performed in [18] to determine the distance moved by the robot during one gait cycle.

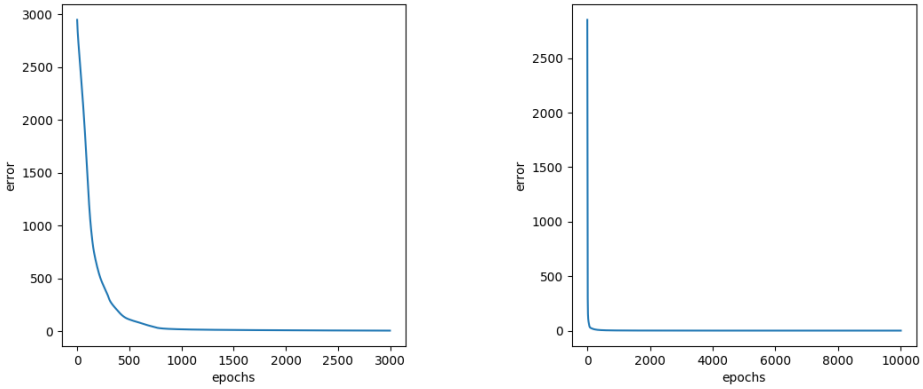
3.3.2 Training Results

Supervised training of the DNN-CPG provided the experimental proof that the architecture can successfully generate motion trajectories for a robot. First, the simple DNN-CPG architecture in Figure 3.7 was used trained using the fundamental frequency of the rhythmic ideal trajectories as input, then the DNN-CPG architecture depicted in Figure 3.8 was trained using desired motion vector and current robot state as input. The fundamental frequency was calculated using the Fourier Transform of the motion trajectories of one of the hip joint. The ideal motion trajectories defined in Section 3.3.1 were used as the ground truth or labels in both the cases.

Table 3.2: Training Results for the proof of concept simple DNN-CPG network

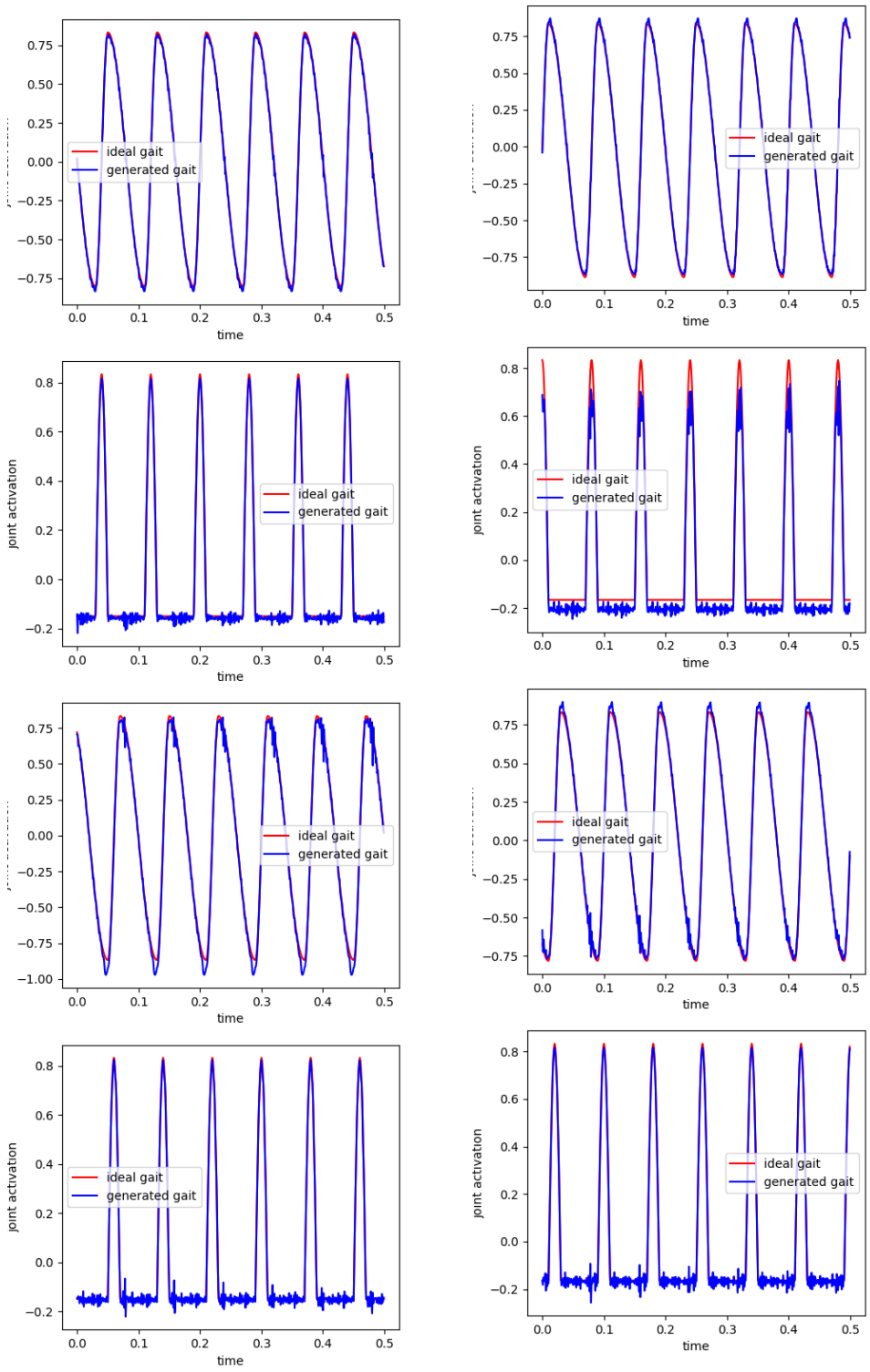
Timesteps	LR	dt	T_{st} (ms)	T_{sw} (ms)	Osc.	Hidden Layer Neurons	Plots
500	0.001	0.001	60	20	20	50	Fig. 3.12a, Fig. 3.13a, Fig. 3.13b
400, 530, 665, 800, 930, 500	0.001	0.001	60, 100, 80, 120, 140, 75	20, 33, 26, 40, 46, 25	8	16	Fig. 3.12b

3.4 Summary



(a) Trot Pace Bound Net (b) Walk Net

Figure 3.12: Generic coupling structure for a 4 cell network in [3]



(a) Trot Pace Bound Net (b) Walk Net

Figure 3.13: Generic coupling structure for a 4 cell network in [3]

CHAPTER 4

REWARD FORMULATION

4.1 Introduction

4.2 Stability Reward

4.3 Energy Efficiency Reward

4.4 Summary

CHAPTER 5

REINFORCEMENT LEARNING

5.1 Introduction

5.2 Deep Deterministic Policy Gradient

5.3 Summary

REFERENCES

[1] Microsonic, "Ultrasonic technology," https://www.microsonic.de/en/support/ultrasonic-technology/principle.htm, 2021, [Online; accessed 12-March-2021].

[2] N. Khomariah and S. Huda, "Walking Pattern for Quadruped as Observer Robot," *Journal of Physics: Conference Series*, vol. 1201, p. 012010, may 2019. [Online]. Available: https://doi.org/10.1088/1742-6596/1201/1/012010

[3] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA 2008)*, pp. 819–824, 2008. [Online]. Available: http://infoscience.epfl.ch/record/130740

[4] A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen, "From swimming to walking with a salamander robot driven by a spinal cord model," *Science*, vol. 315, no. 5817, pp. 1416–1420, 2007. [Online]. Available: https://science.sciencemag.org/content/315/5817/1416

[5] S. Auddy, S. Magg, and S. Wermter, "Hierarchical control for bipedal locomotion using central pattern generators and neural networks," in *2019 Joint IEEE 9th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, 2019, pp. 13–18.

[6] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal on Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.

[7] S. Hirose and K. Kato, "Study on quadruped walking robot in Tokyo Institute of Technology-past, present and future," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, September 2000, pp. 414–419 vol.1.

[8] H. Kimura, Y. Fukuoka, and H. Katabuti, "Mechanical design of a quadruped " tekken 3 4 " and navigation system using laser range sensor," 2005.

[9] D. Hyun, S. Seok, J. Lee, and S. Kim, "High speed trot-running: Implementation of a hierarchical controller using proprioceptive impedance control on the mit cheetah," *The International Journal of Robotics Research*, vol. 11, no. 2, pp. 1417–1445, August 2014.

[10] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "Bigdog, the rough-terrain quadruped robot," in *Proceedings of the 17th IFAC World Congress*, 2008.

[11] X. Meng, S. Wang, Z. CAO, and Z. Leijie, "A review of quadruped robots and environment perception," in *2016 35th Chinese Control Conference (CCC)*, July 2016, pp. 6350–6356.

[12] J. H. BarronS-Zambrano and C. Torres-Huitzil, "CPG Implementations for Robot Locomotion: Analysis and Design," in *Robotic Systems*, A. Dutta, Ed. Rijeka: IntechOpen, 2012, ch. 9. [Online]. Available: https://doi.org/10.5772/25827

[13] A. Sproewitz, R. Moeckel, J. Maye, and A. J. Ijspeert, "Learning to move in modular robots using central pattern generators and online optimization," *The International Journal of Robotics Research*, vol. 27, no. 3-4, pp. 423–443, 2008. [Online]. Available: https://doi.org/10.1177/0278364907088401

[14] C. P. Santos and V. Matos, "CPG modulation for navigation and omnidirectional quadruped locomotion," *Robotics and Autonomous Systems*, vol. 60, no. 6, pp. 912–927, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889012000164

[15] R. J. Szadkowski, P. Cizek, and J. Faigl, "Learning central pattern generator network with back-propagation algorithm," in *Proceedings of the 18th Conference Information Technologies - Applications and Theory (ITAT 2018), Hotel Plejsy, Slovakia, September 21-25, 2018*, ser. CEUR Workshop Proceedings, S. Krajci, Ed., vol. 2203. CEUR-WS.org, 2018, pp. 116–123. [Online]. Available: http://ceur-ws.org/Vol-2203/116.pdf

[16] H. Lee, Y. Shen, C. han Yu, G. Singh, and A. Y. Ng, "Quadruped robot obstacle negotiation via reinforcement learning," in *in In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA*, 2006.

[17] H. Liu, W. Jia, and L. Bi, "Hopf oscillator based adaptive locomotion control for a bionic quadruped robot," in *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*, September 2017, pp. 949–954.

[18] L. Zhou, W. Liu, H. Qian, and Y. Xu, "Gait design and comparison study of a quadruped robot," 07 2017.