# Self-Organized Locomotion Control of a Stick Insect Robot: From Biological Data to Machine

Gian Paolo Currà

Masters Project in Engineering (Advanced Robot Systems)
Supervisor: Poramate Manoonpong

June 2024


The Maersk Mc-Kinney Moeller Institute
University of Southern Denmark

**Abstract**

This study focuses on the development and implementation of a self-organized locomotion control system for a hexapod robot, drawing inspiration from the walking mechanics and morphology of insects. Utilizing real insect trajectory data, the research aims to create a biologically inspired control model capable of adaptive gait patterns. The primary objective is to enable the hexapod robot to achieve a self-organized locomotion by adaptation on foot feedback. In this way, a scattered control system is developed, based on distributed neural central pattern generators (CPGs) for generating basic leg movements and an adaptive sensory feedback mechanism for generating self-organized phase relationships among the local CPG circuits. The methodology includes additionally a forward model integrated with dual rate learning for adaptive control. Experimental results highlight the system's ability to produce intralimb coordination, although challenges remain in achieving a stable, self-organized gait. The findings underscore the potential of neural-based adaptive control mechanisms in robotics, while also pointing to areas for further refinement and enhanced versatilty.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

The study of adaptive control for gait cycles in bio-inspired robotic structures has been an area of active research for many years, as evidenced by numerous studies such as [1], [7], [9], [11], [15], and [16]. These studies have laid the groundwork for understanding how adaptive control can be applied to robotic systems that mimic biological organisms.

To contribute additional findings to this research field, this study focuses on a hexapod robot where each leg is represented neurally by a single control network. By implementing this structure, made by two Central Pattern Generators along with additional elements such as a radial basis function (RBF) premotor network, a dual-rate learning algorithm, and a neural forward model, it is possible to achieve a self-organized locomotion control system.

The research begins with a simulation created by previous collaborators to understand the principles of decoupled CPG control for a single leg, concentrating on the three joints of each leg. This simulation helped in developing familiarity with the system's response and performance. Following this, an analysis was conducted to observe the system's behavior when incorporating complex 3D trajectories (intesection between x-y plane and x-z plane) for each leg, using data obtained from the University of Kiel, Germany.

The Kiel data provided data detailing the movements of insect legs, allowing for a more accurate replication of natural gait patterns in the robotic system. This data was fundamental in redesigning the robot's joint trajectories based on the stick insect's foot data, ensuring a more faithful reproduction of natural locomotion.

A significant portion of the research focuses on investigating the interplay between physical communication and neural communication within the CPG systems and testing the performance of both. This involves examining how the physical properties of the robot and the neural control algorithms interact to produce stable and adaptive gait patterns.

Thus, we propose an adaptive neural control system that can mimic the adaptive capabilities seen in biological organisms through adaptive physical and neural communications. The control algorithm relies on the sensory feedback mechanism for generating self-organized phase relationships among the local CPG circuits, and an neural coupling mechanism for transferring and storing the formed phase relationships (a gait pattern) into the neural structure.

## 1.1 Problem Statement

The primary goal of this project is to develop a self-organized locomotion control system for a hexapod robot by implementing 3D foot trajectory as reference signal. The control system needs to be capable of generating adaptive gait patterns based on the interaction between the robot and its environment. It has to adapt to environmental changes and generate smooth transitions between different gait patterns, achieving stable and energy-efficient locomotion.

## 1.2 Research Objectives

The objectives of this study are to develop a control model for a biologically inspired robot, designed using real insect walking trajectory and morphology. The main tasks of this thesis include:

- Utilize a simulation environment with an insect-shaped robot based on actual insect data in CoppeliaSim V-REP.
- Obtain three-dimensional leg trajectories for the stick insect from the Department of Functional Morphology and Biomechanics, Zoological Institute at Kiel University in Germany, to investigate the leg movements and use this new data to form the intra-limb coordination.
- Employ a neural-based control method, inspired by biological mechanisms, to enable the robot to achieve self-organized locomotion on flat terrain.
- Introduce a mathematical muscle model to incorporate the concept of compliance in the robot joints, enhancing movement fluidity and adaptability.

## 1.3 Report Structure

The structure of this report is organized as follows:

- **Chapter 1: Introduction** - This chapter provides an overview of the study, including the problem statement, research objectives, and the significance of the study.
- **Chapter 2: Literature Review** - A comprehensive review of related work in the field of bio-inspired robotics, giving a context to the study and covering key concepts.
- **Chapter 3: Research Methodology** - Description of the research methodology, focusing on the design and the scientifical studies on the

implementation of the adaptive neural control system for the hexapod robot.

- **Chapter 4: Experiments** - Overview of the experiments conducted to have comparisons on different parameters' set, including the simulation environment, data collection, and hands-on implementation of the control system.
- **Chapter 5: Results and Discussion** - Interpretation of the experimental results, analysis of the findings, and discussion of the implications of the results.
- **Chapter 6: Conclusion and Future Work** - Summary of the study's findings, conclusions drawn from the research, and suggestions for future work.

This structured approach is used to ensures a logical flow of information, providing a clear understanding of the research conducted and the outcomes achieved.

# 2 Literature Review

This chapter is dedicated to the literature review of bio-inspired robotics and self-organized locomotion control. It provides an overview of the state of the art in insect biorobotics, focusing on biomechanics, control systems, cognitive models, soft robotics, materials, and manufacturing. The chapter highlights recent advancements in the field and discusses future directions for research and development.

## 2.1 State of the Art of Insect Biorobotics

Insect biorobotics is an evolving field that draws inspiration from the sophisticated locomotion, sensory systems, and neural control mechanisms of insects. This section reviews the latest advancements in insect-inspired robotics, focusing on biomechanics, control systems, cognitive models, and future directions.

### 2.1.1 Biomechanics in Hexapod Robotics

The design of robotic legs is critical for achieving insect-like locomotion. The importance of mimicking the number of degrees of freedom (DOF) found in insect legs. Most hexapod robots today are equipped with three DOFs per leg, allowing for a range of movements that include lifting, swinging, and positioning the leg tips. Advances in materials and 3D printing technology have enabled the development of more complex leg structures, some of which aim to replicate the intricate musculoskeletal systems of real insects. This has led to the creation of robots with enhanced flexibility and adaptability, capable of navigating varied terrains and performing complex tasks [10].

### 2.1.2 Locomotion Control

Locomotion control in insect biorobotics focuses on the integration of central pattern generators (CPGs), sensory feedback, and forward models. CPGs are neural circuits that generate rhythmic signals to control limb movements, essential for producing adaptive gait patterns. Incorporating sensory feedback allows the robot to respond to changes in the environment, such as uneven terrain, by adjusting its gait in real-time. Forward models predict the outcomes of motor commands and adjust movements to enhance stability and efficiency. Recent advancements have also seen the integration of machine learning algorithms to improve the adaptability and robustness of locomotion control systems [10].

### 2.1.3 High-Level Cognitive Control

High-level cognitive control in insect robotics involves emulating the neural processes that underlie insect behavior. This includes creating neural models that replicate the decision-making and adaptive behaviors observed in insects. Successful implementations have shown that these models enhance robots with the ability to navigate complex environments, make decisions based on sensory input, and learn from interactions with their surroundings. This area of research bridges the gap between biological inspiration and robotic implementation, aiming to create robots that are not only functionally efficient but also adaptable and resilient [10].

### 2.1.4 Advancements in Materials and Manufacturing

The application of advanced materials and manufacturing techniques has played a crucial role in the development of insect-inspired robots. The use of lightweight, flexible materials has allowed for the creation of robots that mimic the intricate movements of insects. Additionally, 3D printing and other additive manufacturing technologies have enabled the production of complex structures that would be difficult or impossible to create using traditional methods. These advancements have facilitated the design of robots with enhanced performance characteristics, such as improved energy efficiency and greater environmental adaptability [8].

## 2.2 Self-organizing locomotion control

As a fulcrum of this project there is the locomotion of a six legged insect robot. The morphology analyzed and simulated in this project is from the structure of a Medauroidea Extradentata or Annam walking stick, which has been studied for its locomotion control. In spite of the simple nervous system this insect is capable fulfilling a complex locomotion tasks like walking in a difficult terrain and climbing. This sensorimotor intelligence is the result of the interaction between the body, the environment and how the brain of the animal is processing both combined.
To reproduce the same intelligence in a robot, self-organized gait patterns are obtained by implementing a control system based on (1) distributed neural CPG-based control circuits, (2) a RBF premotor network, (3) a forward model to capture the body-environment interaction by foot reaction force, and (4) a dual-rate learning mechanism.

### 2.2.1 Gait Analysys in Insects

The gait cycle, is the sequence of limb movements that are required to obtain a "walking" behaviour and move the robot forward. By this definition, the

gait cycle is divided in two phases: the stance phase and the swing phase. The stance phase is defined by the interval where the leg is in contact with the ground and produce the backward motion that pulls the body forward. The swing phase, on contrary, happens when the leg is lifted from the ground and prepares the leg to generate the necessary force in the next step [2].
By studying the gait patterns of insects, researchers were able to implement control algorithms that allow robots to navigate complex environments with similar efficiency and adaptability. For instance, incorporating sensory feedback mechanisms enables robots to adjust their gait in response to changes in the terrain, enhancing their ability to travel through challenging landscapes.
The study of gait analysis in insects provides valuable insights into the development of advanced locomotion control systems for bio-inspired robots. By emulating the adaptive gait patterns and control strategies observed in insects, it is possible to create robotic systems capable of efficient and versatile movement in various environments.

# 3 Research Methodology

The chapter is dedicated to the research methodology of the project, it presents the implementation of a neural control system in a robot inspired by a Medauroidea Extradentata insect [7].
The control system is based on a decentralized architecture with the task of achieving self-organized locomotion and online adaptation through the reaction force feedback between the robot foot and the environment.
The reaction force feedback is used to adapt the control signals generated by the Central Pattern Generator (CPG) by modulating the synaptic plasticity of the CPG.

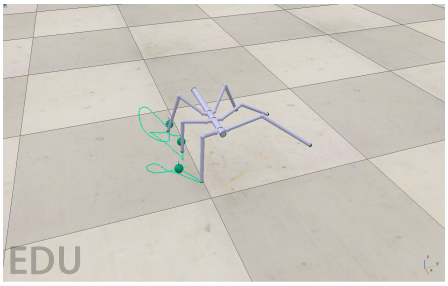
## 3.1 Simulation Environment

The simulation environment used in this project is the CoppeliaSim, a simulator that provides a wide range of functionalities to simulate the interactions between a robot and the environment via multiple dynamics engines [14]. It is a general-purpose simulator design for scalability and flexibility, it is used in this project to simulate the MedaExtra robot (further discussed in Section 3.1.1) and the control system. It integrates actuation, sensing, and control, making it suitable for a wide range of applications. Enables hybrid simulations combining kinematics and dynamics for efficient and precise simulation results. Self-organizing behavior in robots involves decentralized control where individual components interact based on simple rules, leading to complex global behavior. In CoppeliaSim, this was achieved using embedded scripts and two different simulations. The first one was responsible of recording the joint trajectories of the robot, while the second one was used to test the control system. The MedaExtra robot was modeled by using the real insect data provided by the team of Prof. Gorb from the University of Kiel.
Each leg of the insect in the first simulation was controlled by a child script attached to the foot objects. By this way, it was possible to setup an Inverse Kinematics algorithm to follow a 3D trajectory and record the joint trajectories that the robot needs to follow to perform a gait cycle. The second simulation used the local sensing data from the foot-floor interaction to make decisions, enabling the legs to coordinate with each other without a central controller. The main script managed the overall simulation loop and ensured a fine tuned control system to manage the different parts of the model.

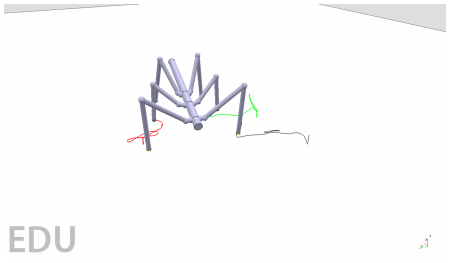
### 3.1.1 MedaExtra in Coppelia

Thanks to the data collected from Kiel University, it was possible to create a robot with the same morphology and physics of a real Medauroidea Extradentata. The resultant robot, the MedaExtra, has been modeled in CoppeliaSim. For this experiment, two simulations were used:

- **MedaExtra**: The robot is modeled with the same morphology and trajectories of the real Medauroidea Extradentata.
- **MedaExtra Trajectories Recording**: The robot is modeled as Medaextra Simulation, but thanks to a inverse kinematics algorithm, the robot is able to follow a 3D trajectory and record the joint trajectories that the robot needs to follow to perform a gait cycle.




(a) Simulation used to record the joint trajectories of the robot.




(b) Simulation used to test the control system.

Figure 3.1: CoppeliaSim simulations used in this project.

## 3.2 Control Structure

The control system is composed by a Central Pattern Generator (CPG) [3] that generates the control signals for the legs, a Radial Basis Function Network (RBFN) [13] that maps the CPG signals to motor positions and foot elevation, a forward model that predicts the foot contact from the elevation estimation and a dual-rate learner algorithm that adapts the force feedback strength on the CPG. It is based on a decentralized architecture where every leg has its own control system that is responsible for intra and interlimb coordination in order to achieve self-organized locomotion and online adaptation.[7]
To achieve the self-organized locomotion, every leg has its own control system that is responsible for intra and interlimb coordination.

### 3.2.1 Central Pattern Generator

The model of the CPG-based control circuit is realized by using the SO(2)-network which is a neural structure based on two neuron network with recur-

| Joint Keyword | Description |
|---|---|
| TC- | thoraco-coxal joint |
| CTr- | coxo-trochanteral joint |
| FTi- | femoro-tibial joint |

Table 3.1: List of the joints taken in analysis for this study, every leg has three motors that are responsible for the gait cycle.

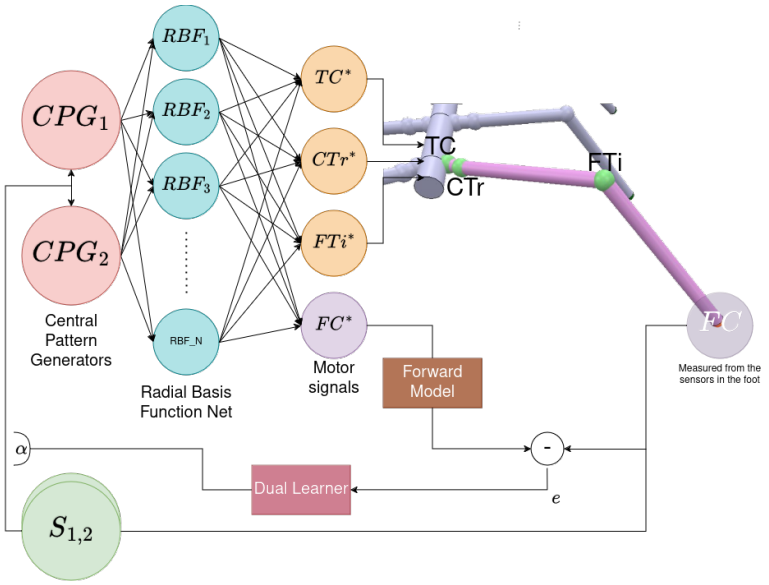



Figure 3.2: Control structure of the MedaExtra robot [7], two outputs from the Central Pattern Generator (CPG) serve as inputs to the Radial Basis Function (RBF). These inputs are then transmitted to the hidden neurons of the RBF, which utilize 2D radial basis or Gaussian activation functions. The CPG signals are mapped with an equal distribution of kernels, and the formula incorporates the distance between the center and the input vector.

rent connection. This net allows the generation of a periodic signal pseudo sinusoidal with a phase shift of $\pi/2$ [12].

$$a_1(t+1) = w_{11}o_{N_1}(t) + w_{12}o_{N_2}(t) + \alpha S_1(t), \quad (3.1)$$

$$a_2(t+1) = w_{22}o_{N_2}(t) + w_{21}o_{N_1}(t) + \alpha S_2(t), \quad (3.2)$$

$$o_{N_i}(t+1) = tanh(a_i(t+1)), i = 1, 2 \quad (3.3)$$

where:
$w_{11}, w_{22}$ = self-connection weights of $N_1, N_2$
$w_{12}, w_{21}$ = mutual weights between $N_1, N_2$.
$S_1, S_2$ = CPG inputs (foot contact feedback)

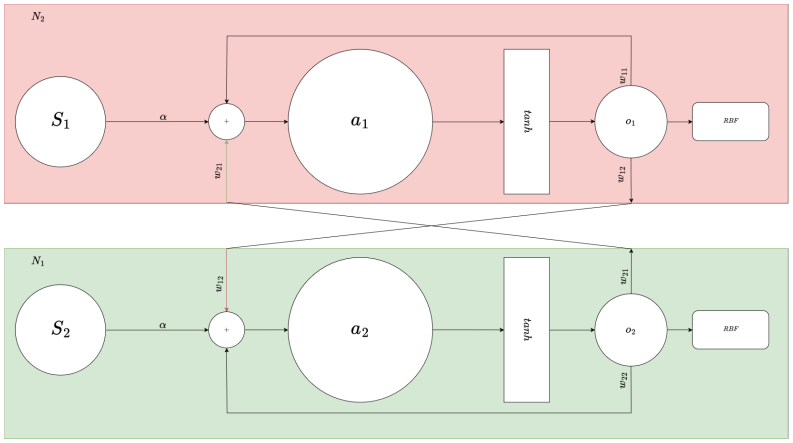



Figure 3.3: Inser caption

$\alpha$ = synaptic plasticity, automatically adjusted by dual-rate learning

There are two different types of weights in the CPG, the self-connection weights $w_{11}, w_{22}$ and the mutual weights $w_{12}, w_{21}$. where: $w_{11} = 1.4$
$w_{22} = 1.4$
$w_{12} = 0.18 + MI$
$w_{21} = -0.18 - MI$ [1]
$MI$ is the extrinsic modulatory input. It modifies the value of the mutual weights of the CPG and therefore the output frequency
The CPG is responsible for the generation of the signals for the legs and the body. It acts as a brain of the control system, generating a perpetual pulse signal.

### 3.2.2 Premotor Network for intralimb coordination

The premotor network is a Radial Basis Function Network (RBF-Net) that transforms the input space into a higher-dimensional space using radial basis functions. In this study they are used to map the CPG signals to the joint motors and the expected reaction force on the foot.
It is composed by three layers, the input, hidden and output layer.

1. **Input Layer**: The number of neurons is equal to the dimensionality of the input data. In this case is the CPG signals.

2. **Hidden Layer**: The hidden layer is composed by a preset quantity of kernels that own the Gaussian function as their activation function. Other alternatives such as the Multiquadric and Inverse Multiquadric functions are also viable but not used in this study. Every neuron in the

[1] These values are chosen from [7]

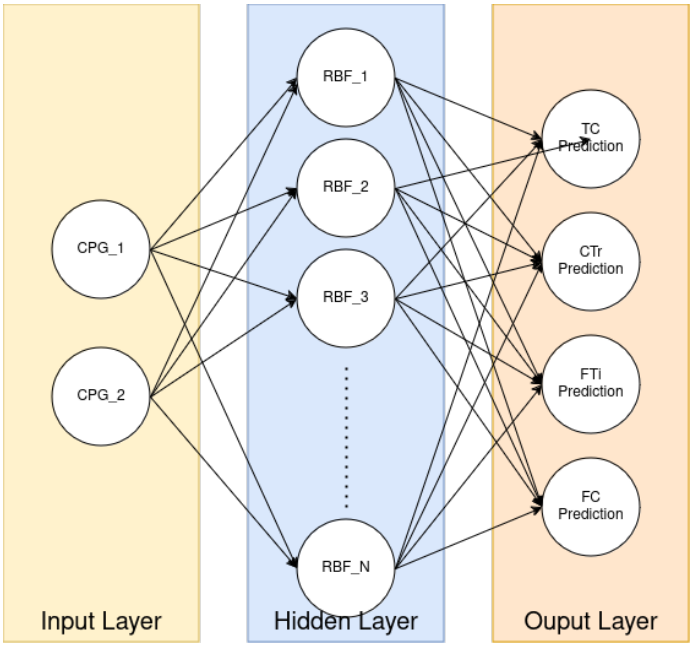



Figure 3.4: Gaussian functions as activation functions, the input space is transformed into a different dimensional space.

hidden layer is linked to a "center" and the neuron's output is determined by the distance between the input vector and this center.

3. **Output Layer**: The final layer is composed by the linear combination of the hidden layer outputs into the four outputs (3 joint motors trajectories and 1 expected reaction force).

The activation function is defined as:

$$\phi_n(t) = e^{-\left(\frac{(o_{N_1}(t)-\mu_{n,1})^2+(o_{N_2}(t)-\mu_{n,2})^2}{2\sigma^2}\right)}, n \in 1, ..., N \tag{3.4}$$

where:
$o_{N_1}, o_{N_2}$ = input signals
$\mu_{n,1}, \mu_{n,2}$ = center coordinates used to calculate the euclidean distance between them and the input signals.
$\sigma$ = parameter that controls the spread of the Gaussian distribution

The distribution's centers $\mu_{n,1}, \mu_{n,2}$ are chosen from the input space as points evenly distributed. By this way the input is discretized with a precision dependant on the number of centers (and therefore the number of kernels in the hidden layer). As it can be seen in the equation 3.4, the Euclidean distance is used to calculate the similarity between the input and the center of the kernel.

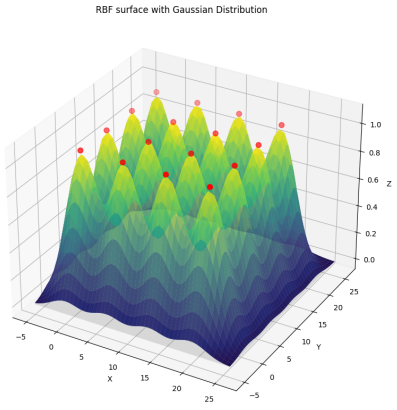



Figure 3.5: Gaussian functions as activation functions, the input space (CPG) is transformed into a higher-dimensional space.

Based on this similarity, the output of the hidden layer is calculated as:

$$o_{RBF_j}(t) \quad = \quad \sum_{n=1}^{N} w_{jn}\phi_n(t), j \in 1, ..., 4 \tag{3.5}$$

where:
$o_{RBF_j}(t)$ = output of the hidden layer
$w_{jn}$= weights of the neurons
$\phi_n(t)$ = activation function

To produce a more accurate estimation of the joint motors trajectories and the expected reaction force, the delta rule is used to update the weights of the hidden layer:

$$\Delta \mathrm{w}_{jn} \quad = \quad \eta(T_j(t) - \sum_{n=1}^{N} w_{jn}\phi_n(t)), j \in 1, ..., 4, n \in 1, ..., N \tag{3.6}$$

where:
$\Delta \mathrm{w}_{jn}$ = weight update
$\eta$ = learning rate
$T_j(t)$ = target output

By doing this, we are initializing the weights and centers of the RBF network, which will be used in the online learning phase.

### 3.2.3 Forward Model

As presented in the previous section, the outputs of the RBF-Net are the joint motors trajectories and the expected reaction force on the foot. From a broader perspective of the gait cycle, in order to obtain a more accurate estimation of the foot contact and pursue an optimal inter-limb coordination, a forward model is introduced.
It is used to predict the foot contact through two post processing neurons $P_1$ and $P_2$. $P_1$ is a recurrent neuron that behaves as a low-pass filter, while the $P_2$ neuron function as threshold to convert the shaped signal to a discrete foot contact, suitable format for a comparison real foot contact recorded from the robot.

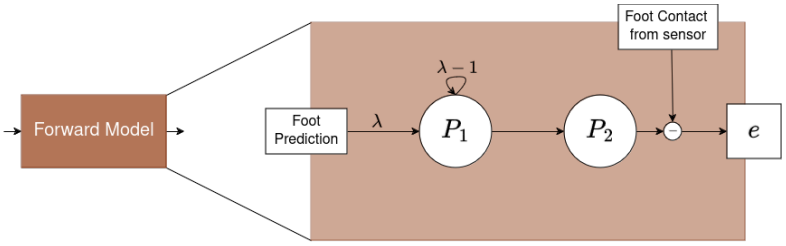



Figure 3.6: The forward model

$$o_{P_1}(t) = \lambda o_{FP}(t) + (1-\lambda) o_{P_1}(t-1), \tag{3.7}$$

$$o_{P_2}(t) = \begin{array}{ll} 1, \text{ if } & o_{P_1}(t) \leq \epsilon, \\ 0, \text{ if } & o_{P_1}(t) > \epsilon, \end{array} \tag{3.8}$$

$$e(t) = |o_{P_2}(t) - o_{FC}(t)| \tag{3.9}$$

where:
$o_{P_1}(t)$ = output of the $P_1$ neuron
$o_{P_2}(t)$ = output of the $P_2$ neuron
$\lambda$ = shaping factor between 0,1 for $o_{P_1}(t)$
$\epsilon$ = shaping threshold value for $o_{P_1}(t)$

These shaping steps are necessary because in this project we are trying to obtain gait phases which are binary transitories from swing and stance (0 and 1). While the reaction force is a value expressed in Newtons, the foot contact is a binary and discrete signal.

| Gait Phase | Description |
|---|---|
| 0 | Swing phase |
| 1 | Stance phase |

By this way, it is possible to achieve the online sensory motor coupling and the adaptation of the force feedback strength on the CPG using the$\alpha$ term (see $S_1$ and $S_2$ in Equation 3.1,3.2).

### 3.2.4 Dual Rate Learning

A dual rate learning mechanism is used to adapt the synaptic plasticity of the CPG (see $\alpha$ in Equation 3.1,3.2). It is responsible for a modulation of the CPGs phase in order to achieve the self-organized locomotion. To adjust this factor during the locomotion, two different mechanisms with different learning rate are used [9].

- **Slow learning mechanism**: Long-term adaptation.
- **Fast learning mechanism**: Short-term adaptation.

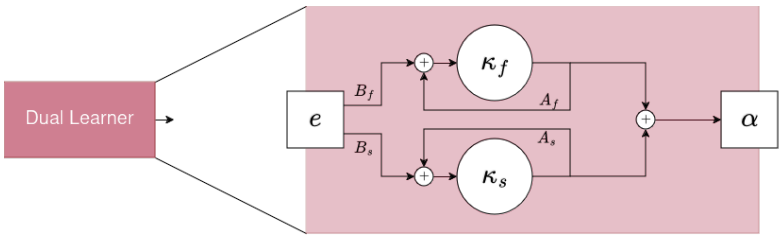



Figure 3.7: Dual Learner Structure

Both mechanisms are modulating the error (Equation 3.9) and the scalars on the following equations are sets as $A_s > A_f$ and $B_s > B_f$:

$$\kappa_f(t) = A_f\kappa_f(t-1) + B_f e(t), \tag{3.10}$$

$$\kappa_s(t) = A_s\kappa_s(t-1) + B_s e(t), \tag{3.11}$$

$$\alpha = \kappa_f(t) + \kappa_s(t) \tag{3.12}$$

where:
$\kappa_f(t)$ = fast learning rate
$\kappa_s(t)$ = slow learning rate
$A_f, A_s$ = retention or forgetting factors
$B_f, B_s$ = learning factors
$e(t)$ = error signal from Equation 3.9

In this way, the dual rate learning mechanism is responsible for the adaptation of the CPGs phase during the locomotion by correcting $\alpha$ value with different learning frequencies.

# 4 Experiment

This chapter presents the experiment conducted during the study period for this project, by this way it was possible to obtain results capable of offerng insights about the performance of the approach explained in the previous section.

## 4.1 Morphology

The robot has an hexapod structure with 3 degrees of freedom per leg, the morphology is inspired by the Medauroidea Extradentata, a real insect that has been studied by Professor Gorb of Kiel University and implemented to this project. Instead of using theoretical spheres as junctions to model the intralimb structure, the robot has a more realistic morphology, were the every joint is modeled as revolute joint. By doing this, the model obtain three degree of freedom per leg:

- **Thoraco-Coxal joint**: resposible for z-axes translation
- **Coxo-Trochanteral joint** : in charge of x-axes translation
- **Femoro-Tibial joint**: handling the y-axes translation

This structure simulate more precisely the future real robot that is built with one Dynamixel DC motor per joint [7].
The morphology of the body and the legs of the robot are presented in the following tables, the length and mass of the body and the legs are taken from the real insect and imported to the simulation environment with proportional values.
Based on the objectives of this study and the research questions, the following experiments were conducted:

| **Body part** | **Length [mm]** | **Mass [mg]** |
|---|---|---|
| Caput | $7.1 \pm 0.5$ | $59.3 \pm 15.2$ |
| Thorax | $29.9 \pm 1.7$ | $317.2 \pm 89.4$ |
| Abdomen | $48.3 \pm 2.0$ | $567.6 \pm 112.2$ |
| Body | $97.7 \pm 4.1$ | $1370.0 \pm 213.0$ |

Table 4.1: Morphology of the body of a female Medauroidea Extradentata insect [7]

| | Segment | Length [mm] | Mass [mg] |
|---|---|---|---|
| FL | coxa + trochanter | $3.39 \pm 0.01$ | $4.7 \pm 0.1$ |
| | femur | $33.09 \pm 0.41$ | $20.8 \pm 0.3$ |
| | tibia | $40.37 \pm 2.17$ | $9.1 \pm 0.2$ |
| | tarsus | $8.08 \pm 0.02$ | $1.1 \pm 0.1$ |
| | **Total** | $84.94 \pm 2.61$ | $35.7 \pm 0.7$ |
| ML | coxa + trochanter | $4.46 \pm 0.01$ | $7.1 \pm 0.1$ |
| | femur | $21.61 \pm 0.36$ | $18.6 \pm 0.2$ |
| | tibia | $22.81 \pm 1.37$ | $8.1 \pm 0.2$ |
| | tarsus | $7.72 \pm 0.01$ | $1.2 \pm 0.1$ |
| | **Total** | $56.61 \pm 1.72$ | $35.0 \pm 0.6$ |
| HL | coxa + trochanter | $3.08 \pm 0.02$ | $4.2 \pm 0.1$ |
| | femur | $26.20 \pm 0.88$ | $24.1 \pm 0.3$ |
| | tibia | $33.81 \pm 1.36$ | $11.8 \pm 0.2$ |
| | tarsus | $7.99 \pm 0.04$ | $1.5 \pm 0.1$ |
| | **Total** | $71.08 \pm 2.25$ | $41.6 \pm 0.7$ |

Table 4.2: Morphology of the legs of a Medauroidea Extradentata insect [7]

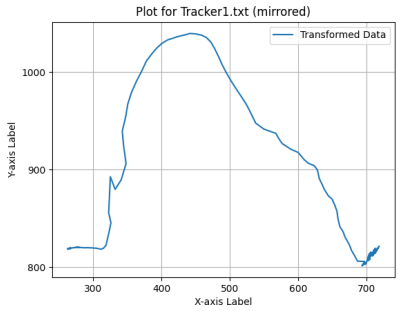
1. **Trackers from Kiel University**: The procedure conducted to process the real insect data is collected from the real Medauroidea Extradentata insect. The data is refined to extract the necessary information for the analysis and this is used to generate target joint positions for the control system.

2. **Control System**: Once the data is converted into desired joint trajectories, the control system is pretrained offine on the data to obtain the necessary centers and weights for the CPG+RBF net.

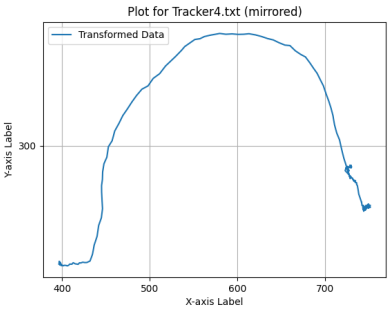
3. **Simulation** The control system is tested in a Coppelia V-REP simulator to verify its performance by comparing adaptive and fixed $\alpha$ value.

## 4.2 Trackers from Kiel University

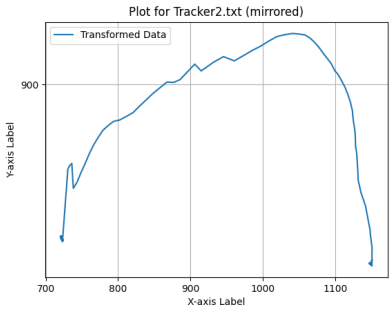
The data provided by Prof. Gorb from the University of Kiel are a cardinal point of the approach that characterize this study. The data was collected from the real Medauroidea Extradentata insect and was processed to extract the necessary information for the analysis and for the generation of the target joint positions for the control system. The data are XY and XZ coordinates of the right insect's foot. For this experiment the data is being mirrored to the left side to have a complete gait cycle.
In Figure 4.1, the original 2D trajectory of the real insect's foot is shown, casing the movement patterns in two dimensions. This raw data provides an



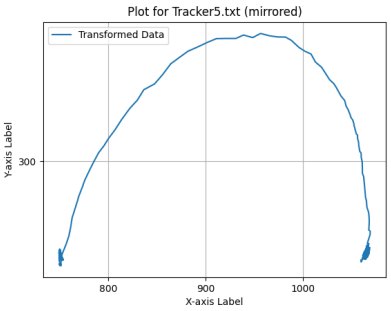

(a) Original x-z plane for Hindleg



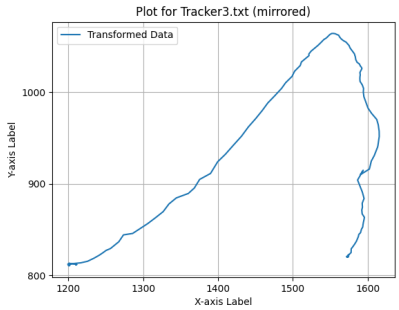

(b) Original x-y plane for Hindleg



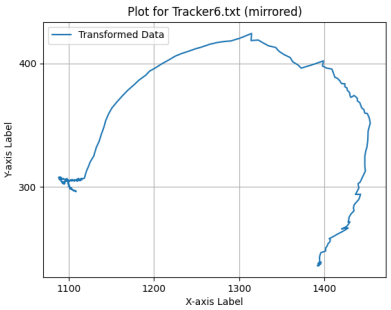

(c) Original x-z plane for Midleg




(d) Original x-y plane for Midleg




(e) Original x-z plane for Foreleg

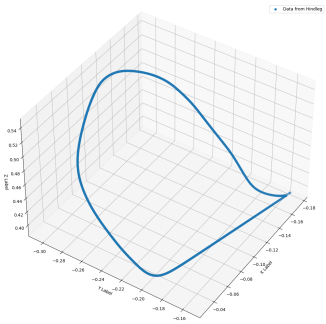



(f) Original x-y plane for Foreleg

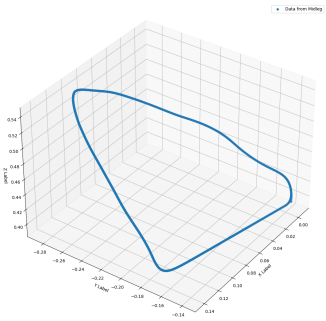
Figure 4.1: Original 2D trajectory of real insect's foot

initial understanding of the insect's locomotion, foot placement and movement over time.

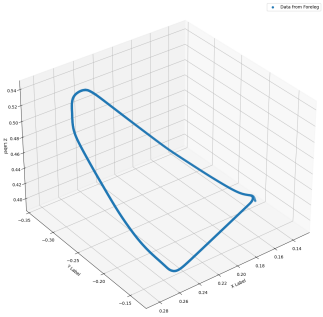
To gain a more comprehensive perspective, we processed this 2D data to extract and construct the corresponding 3D trajectories. This transformation involved combining x and y coordinates from the original 2D trajectories with the x and z coordinates, effectively mapping the insect's movement in three-dimensional space. The result of this processing is illustrated in Figure 4.2.



(a) Data combined in 3D for Hindleg

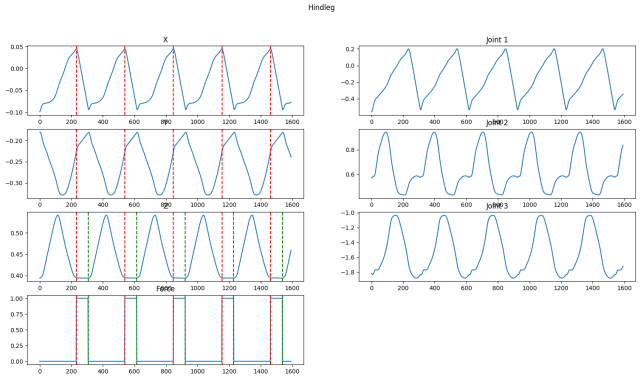


(b) Data combined in 3D for Midleg

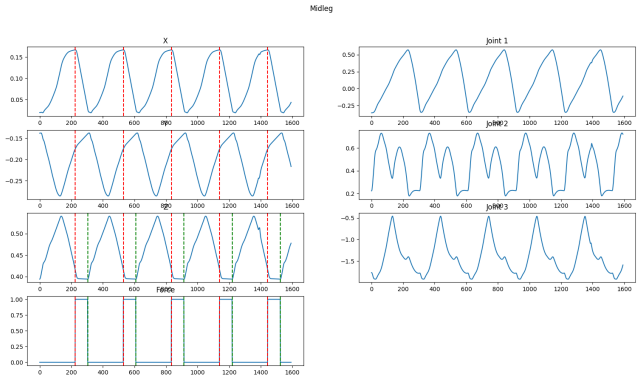


(c) Data combined in 3D for Foreleg

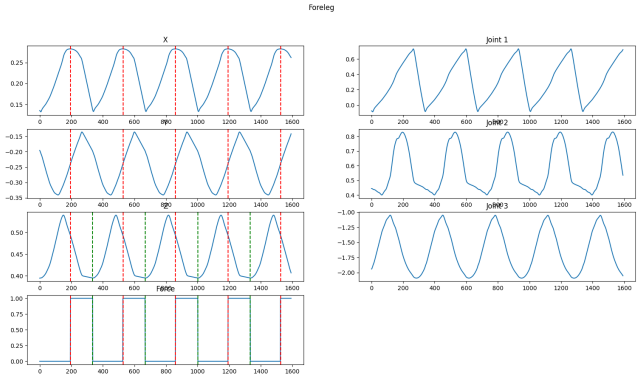
Figure 4.2: 3D closed trajectories for the different feet




(a)




(b)




(c)

Figure 4.3: Joint Trajectory for the different legs, by searching for the local minimum in multiple axis it was possible to distinguish the stance and swing phases and the period of one gait cycle

## 4.3 Offline Training of the Control System

The control system (CPG+RBF) was pretrained offine on the processed data Kiel University to obtain the necessary centers and weights for the RBF net. As part of the preprocessing step, the data was downsampled to have a numerosity of 40 datapoints. This number is chosen empirically to have a good trade off between precision and computational difficulty.

### 4.3.1 CPGs Setup

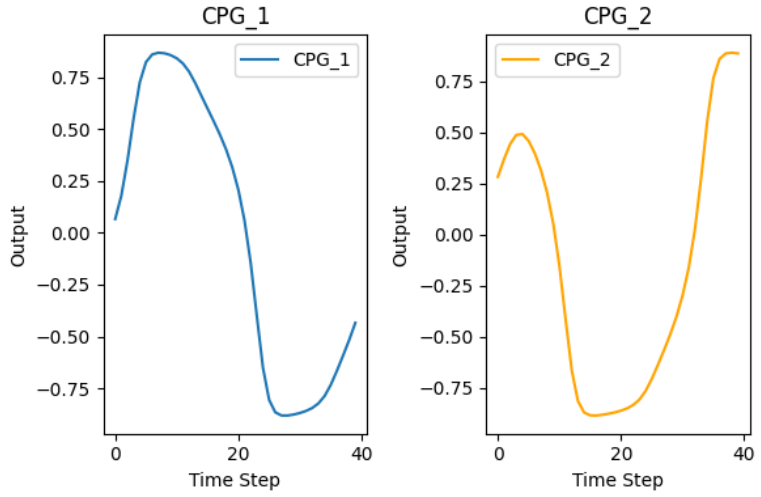
The CPGs were set up with the following parameters:

| Parameter | Value |
|---|---|
| $\alpha$ | 0.01 |
| $W_{d0}$ | 1.4 |
| $W_{d1}$ | 0.18 |
| $MI$ | 0.1 |
| $S_1$ | 1.0 |
| $S_2$ | 1.0 |
| $O_{N1}$ | 0.0 |
| $O_{N2}$ | 0.2 |

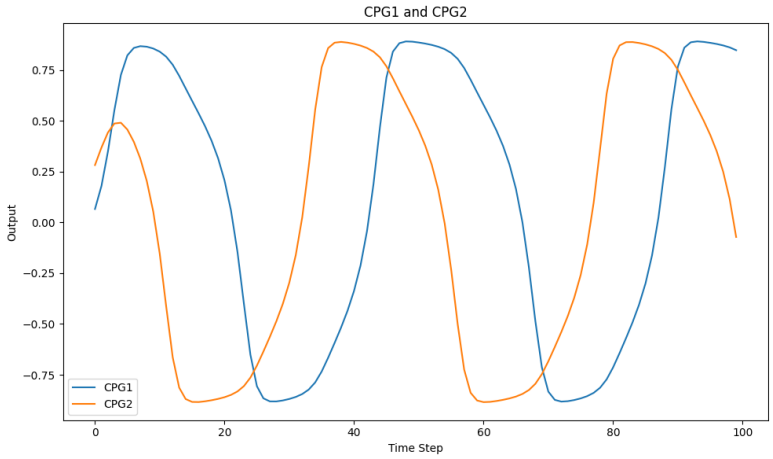
Table 4.3: CPGs Setup Parameters

By initializing the values in this manner, we instantiate a Central Pattern Generator (CPG) with a constant input value of 0.01 and a modulatory input of 0.1. The synaptic weights are set to the default values of 1.4 and 0.18, respectively. This specific configuration helps to ensure effectiveness in generating rhythmic outputs.
For our application, we employ two CPGs operating at the same frequency, but with a phase offset of 0.2 seconds. This dual-CPG setup is crucial because it generates a sinusoidal pattern that is both simple and efficient. This simplicity of the sinusoidal pattern facilitates easier mapping with Radial Basis Functions (RBFs).
Utilizing only a single CPG would result in an overly simplistic pattern. Such a configuration would lack the complexity needed to fully cover the joint trajectory space, which is essential for achieving the desired range of motion and adaptability in the robot's movements. The dual-CPG approach, on the other hand, provides a richer and diverse set of movement patterns, enabling the signal to be mapped in more complex waves, to follow sophisticated and varied trajectories.
The evolution of the CPGs over time, which demonstrates how their outputs change and interact, can be observed in Figure 4.4. This figure illustrates the dynamic nature of the CPGs and highlights their ability to produce coordinated and adaptive movements that are essential for the robot's functionality.

By carefully designing and tuning these CPGs, we achieve a balance between simplicity and computational complexity in the generated movements.
This balance is critical for ensuring that the robot can obtain adaptation and environments while maintaining efficient and effective motion control. The initialization values are obtain empirically and are based on the article [7].




(a)




(b)

Figure 4.4: Central Pattern Generators (CPGs) outputs over time

We explore and quantify the approximate error that arises when using a SO(2)-network to drive a RBF network. By increasing the modulation input, or frequency of the controller, the shape of the network output changes, introducing an error in the approximation of the joint positions.
To conduct the experiment, we collect results during the offline training, by

checking the performance of the RBF on the target.
To do so, the experiment is performed on a static Medaextra Robot (as shown in fig. 4.5) to demostrate the amount of variation that $MI$ produce on the gait. The modulation input values were increased from 0.05 to 0.3 by increments of 0.05. The simulation duration is set to 10 seconds, and the robot is held in a static position to observe the changes.

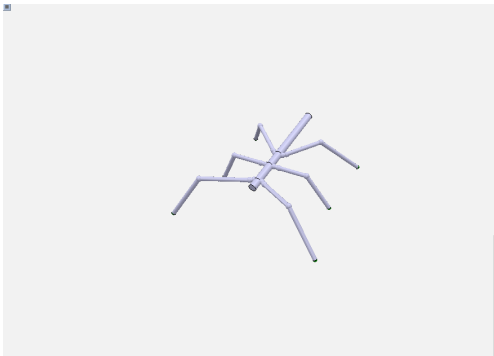


Figure 4.5: Medaextra Robot in a static position, by deactivating the dynamic properties

### 4.3.2 RBFs Setup

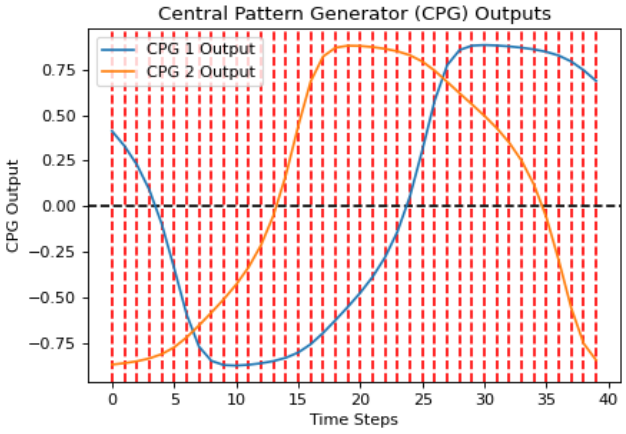
In the structure of the Radial Basis Function (RBF) network, we implemented 40 kernels, which correspond to 40 centers. These centers were uniformly distributed along the entire length of the input array, ensuring that there was precisely one center per timestep. This uniform distribution was critical for maintaining consistency and accuracy in processing the temporal data.

| **Parameter** | **Value** |
|---|---|
| $\sigma$ | 0.2 |
| $\eta$ | 0.01 |
| $N_{kernels}$ | 40 |
| *Learning rate* | 0.01 |

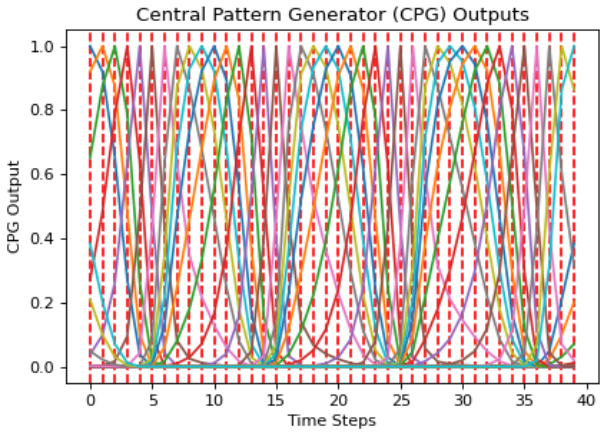
Table 4.4: RBF Setup Parameters

The RBF Nets by definition [4] have a fixed depth of three layers which is necessary to map into dimensionally different spaces.
This setup was meticulously chosen through empirical analysis to achieve optimal coverage of the input space. By ensuring that the input space was well-covered, the RBF network could effectively capture the nuances and variations in the data. Moreover, the chosen configuration provided a robust approximation of the output function, allowing for precise and reliable predictions.
The learning process is characterized by these limits:

- The number of epochs was set to 10000, which was found to be sufficient for the network to converge and reach a stable state.
- The learning rate was set to 0.01, which was determined through empirical testing to be the optimal value for achieving fast convergence without sacrificing accuracy.
- The error threshold was set to 0.001, which was chosen based on the desired level of precision and accuracy in the output predictions.
- The weights of the network were initialized to 1, which provided a good starting point for the learning process.




(a)




(b)

Figure 4.6: CPGs and RBFs outputs over time, the RBFNet is being trained to approximate the CPGs output into the target joint positions

### 4.3.3 CPG+RBF Setup

Once the CPGs and RBFs have been configured, they need to be connected to create the control system that maps the oscillatory signals generated by the CPGs as command signals for the leg joints.
This connection is achieved by using the output function of the CPGs as the input for the RBFs, which approximate the desired output function.
An additional tuning step that proved necessary was testing a displacement on the CPGs.
Each CPG created is an array of 40 elements. The first CPG started from 0, while the second started from 0.2.
During the offline training phase, it was observed that changing the initial value of the CPG output array resulted in a more stable and higher-performing control system, yielding different outcomes.
For this reason, instead of generating a 40 elements array starting from 0 (CPG 1) and 0.2 (CPG 2), we generate a 100 elements CPG array and scroll through its values with a smaller 40 elements array, selecting the most performant one.

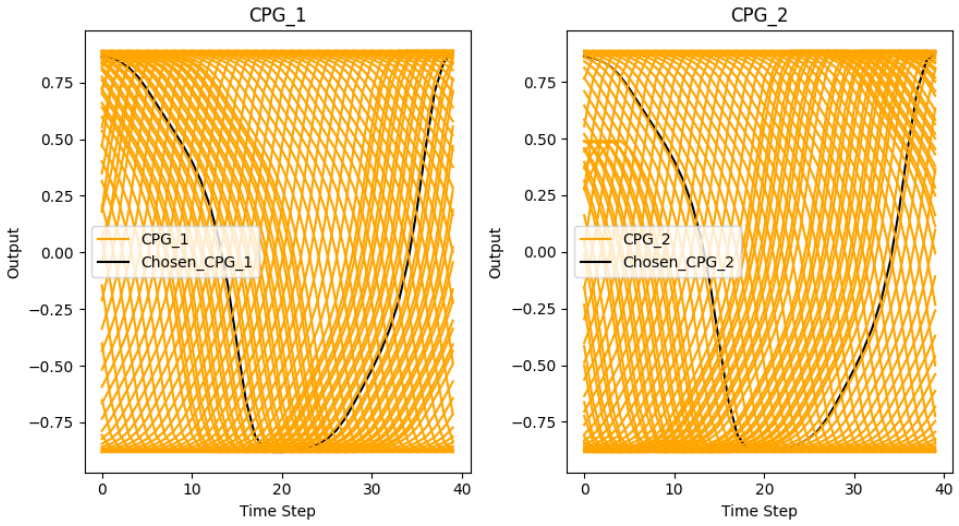



Figure 4.7: Best CPGs outputs over time, the best CPGs are selected by looking at how well the RBFs approximate the target.

## 4.4 Online Testing to obtain self-organized gaits

As introduced in chapter 3, the approach chosen for the online testing involves using a second simulation to test the interactions with the environment. This simulation allows for a more realistic evaluation of the robot's performance under dynamic conditions.

### 4.4.1 Simulation Setup

In this experiment, the robot is initially held in a mid-air position to test the offline learning and then released into the environment to test the interaction with it and the online learning.
In this second case it is released at $t = 0$ seconds and the force feedback is enabled after a delay of three seconds, allowing the robot to begin responding to environmental forces. This delayed activation of force feedback helps in stabilizing the robot and prevents immediate erratic movements upon release. The experiment runs for a total duration of 60 seconds, during which we carefully evaluate the robot's performance. The joints of the robot are modeled as revolute joints, allowing for rotational movement in a single plane. This setup enables the robot to perform a wide range of movements and interactions with the environment. The control on the joint is done on the position, trying to reach the target position given by the RBFs.
The primary metrics for this evaluation are the formation of a stable gait cycle and the distance walked by the robot within the simulation period. The formation of a gait cycle indicates the robot's ability to coordinate its limbs effectively, while the distance walked measures the efficiency and effectiveness of its locomotion.
For this experiment, we utilized Bullet 2.83 as the physics engine to ensures that our simulation environment mimics real-world physics, providing reliable data for evaluating the robot's performance.

### 4.4.2 Adaptive and fixed feedback strength

The experiment was conducted using two different feedback strength configurations: fixed and adaptive value. For the fixed configuration, the feedback strength was consistently set to $\alpha = 0.1$ throughout the entire duration of the experiment. This setup allowed us to observe the robot's performance with a constant feedback parameter, providing a baseline for comparison.
In contrast, the experiment is restarted enabling the adaptive mechanism to change $\alpha$ dynamically during the simulation in response to the robot's interactions with its environment, effectively tuning the control parameters.
By comparing these two configurations, we aimed to understand the impact of adaptive feedback on the robot's ability to maintain stability and achieve efficient locomotion. The adaptive feedback mechanism is expected to enhance the robot's performance by adjusting the CPG pattern according to varying conditions and disturbances, potentially leading to more robust behavior.

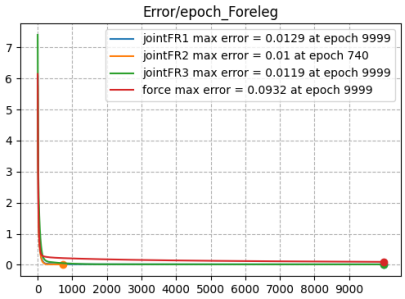
# 5 Results and Discussion

This chapter resumes the results obtained from the experiments conducted in the simulation environment.
The results are divided into two main sections: offline learning and online learning. The offline learning phase focuses on training the Radial Basis Function (RBF) network to predict the target trajectories generated by the Central Pattern Generators (CPGs). The online learning phase evaluates the performance of the CPGs and RBF network in controlling the robot's locomotion and adapting to changing conditions.

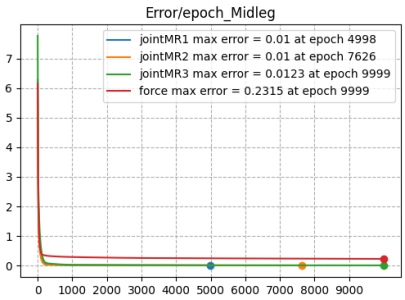
## 5.1 Offline learning

In this section, the results of the offline learning phase of the Radial Basis Function (RBF) network are presented. The outcomes of training the RBF network are obtained by using the learning algorithm offline and evaluate its performance in tracking the target trajectories.
During the offline learning, the network parameters were adjusted to minimize the error between the predicted outputs and the actual target values (Fig. 5.2).
Next, the network's ability to adapt to different input patterns from the CPGs is analyzed. By examining the response of the RBF network to various test inputs, we can assess its robustness and flexibility. The network's performance is visualized through plots comparing the predicted trajectories to the actual target paths, showcasing the network's precision in tracking.
Additionally, we explore the effect of different network configurations and hyperparameters on the learning outcomes. Various setups, including different numbers of kernels and hidden layer structures, were tested to find the optimal configuration. This comparative analysis helps in understanding how each parameter influences the network's learning capability and overall performance. By doing this, we are initializing the weights and centers of the RBF network, which will be used in the online learning phase.
List of the hyperparameters used in Figure 5.1 to obtain the best results for the RBF network are the following:

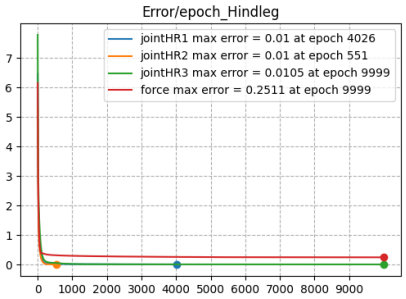
- Number of kernels: 40
- Learning rate: 0.01
- Max Epochs: 1000




(a)




(b)



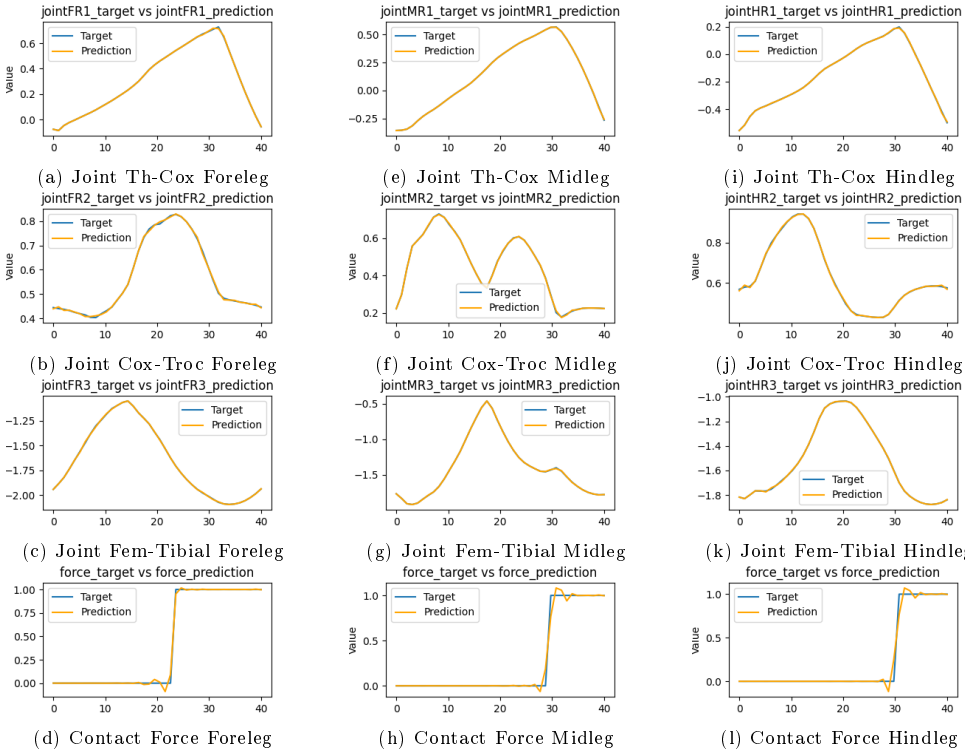

(c)

Figure 5.1: Error evolution during the training of the RBF network for the three legs of the robot.

- Sigma: 0.2
- Number of hidden layers: 1

In the Fig. 5.2 it is shown how the RBF network is able to follow the target trajectory. The target trajectory is shown in blue, while the predicted trajectory is shown in red. The RBF network is able to follow the target trajectory with a high degree of accuracy, as the predicted trajectory closely matches

the target path. This demonstrates the effectiveness of the RBF network in learning and tracking complex trajectories.




(a) Joint Th-Cox Foreleg (e) Joint Th-Cox Midleg (i) Joint Th-Cox Hindleg

(b) Joint Cox-Troc Foreleg (f) Joint Cox-Troc Midleg (j) Joint Cox-Troc Hindleg

(c) Joint Fem-Tibial Foreleg (g) Joint Fem-Tibial Midleg (k) Joint Fem-Tibial Hindleg

(d) Contact Force Foreleg (h) Contact Force Midleg (l) Contact Force Hindleg

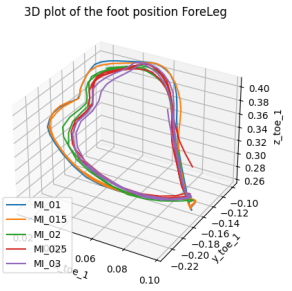
Figure 5.2: Comparison between the target trajectory and the predicted trajectory for the three legs of the robot.

### 5.1.1 Control System performances: CPGs + RBFNet

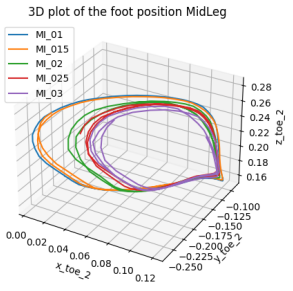
As final step of the offline testing, different values of the extrinsic modulatory input are tested to have a scope of possible solutions. After choosing the weights and centers for the RBF, Figure 5.3 shows the 3D and 2D trajectories of the robot's legs classified by different MI value. Both 3D and 2D trajectories are displayed to guaranteed a better understanding of the robot's behavior. In the final graph the error distribution of the three legs is shown in Figure 5.5 were it is possible to understand at what extent the MI value can modify the accuracy control system.

## 5.2 Online learning

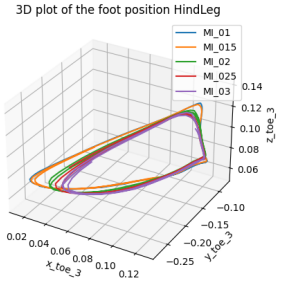
In this section, the results of the online learning phase is presented. The online learning phase focuses on evaluating the performance of the CPGs and RBF network in controlling the robot's locomotion and changing the alpha values




(a) Foreleg 3D Trajectory




(b) Midleg 3D Trajectory



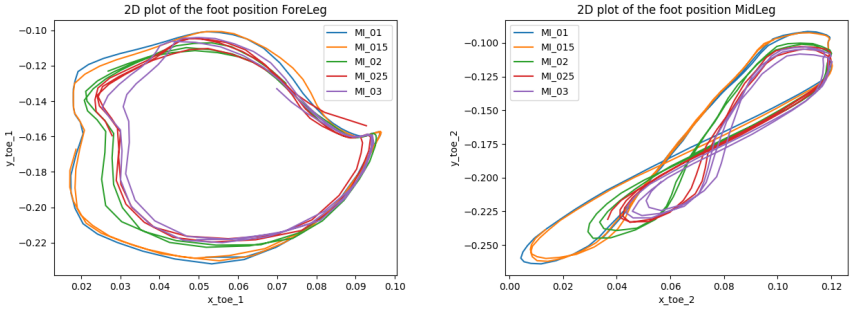

(c) Hindleg 3D Trajectory

Figure 5.3: 3D Trajectories, extrinsic modulatory input (MI) classified

according to foot contact signal. The robot's walking behavior is analyzed under different configurations and scenarios to assess the effectiveness of the control system in maintaining stability and achieving efficient locomotion.

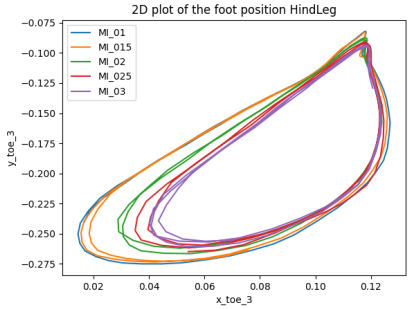
### 5.2.1 Adaptive and Fixed Feedback Strength

In Figure 5.6 it is shown how two different approaches compare to the theoretical CPGs signals.
As with the first, the alpha-fixed configuration, the feedback strength was consistently set to $\alpha = 0.1$ throughout the entire duration of the experiment. This setup allowed us to observe the robot's walking performance with a constant feedback parameter, providing a baseline for comparison.
In contrast, the adaptive configuration enabled the feedback strength $\alpha$ to change dynamically during the simulation in response to the robot's interactions with its environment. This adaptive mechanism was designed to tune the control parameters in real-time, allowing the robot to adjust to varying conditions and disturbances more effectively.
By comparing these two configurations, we aimed to understand the impact of adaptive feedback on the robot's ability to maintain stability and achieve efficient locomotion. The adaptive by providing more robust and adaptable




(a) Foreleg 2D Trajectory

(b) Midleg 2D Trajectory



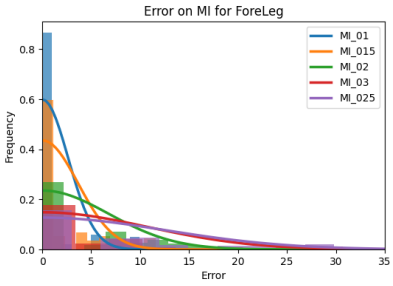

(c) Hindleg 2D Trajectory

Figure 5.4: 2D Trajectories, extrinsic modulatory input (MI) classified

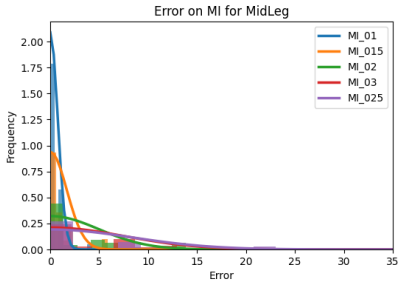
behavior, improving its response to dynamic environmental chback mechanism was expected to enhance the robot's performances.
In Figure 5.7 it is shown how the alpha values are changing during the simulation. The alpha values are changing based on the foot contact signal, which is used as feedback to adjust the phase of the CPGs. The graph represent how diffrent legs, grouped by the left and right side, are changing their alpha values during the simulation.

### 5.2.2 Gait Analysis

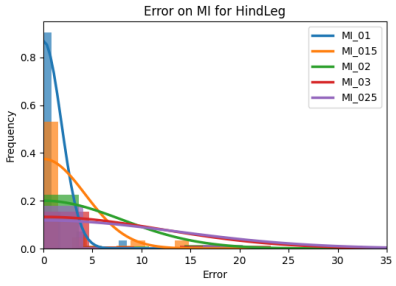
As last step of the online learning, the robot's walking behavior is analyzed under adaptive and fixed $\alpha$ value. In this way we are assessing the effectiveness of the control system in maintaining stability and achieving efficient locomotion. In Figure 5.8 it is shown how the theoretical gait is compared with the real gaits of the robot with fixed and adaptive $\alpha$ values.




(a) Foreleg Foot Contact Error Distribution




(b) Midleg Foot Contact Error Distribution

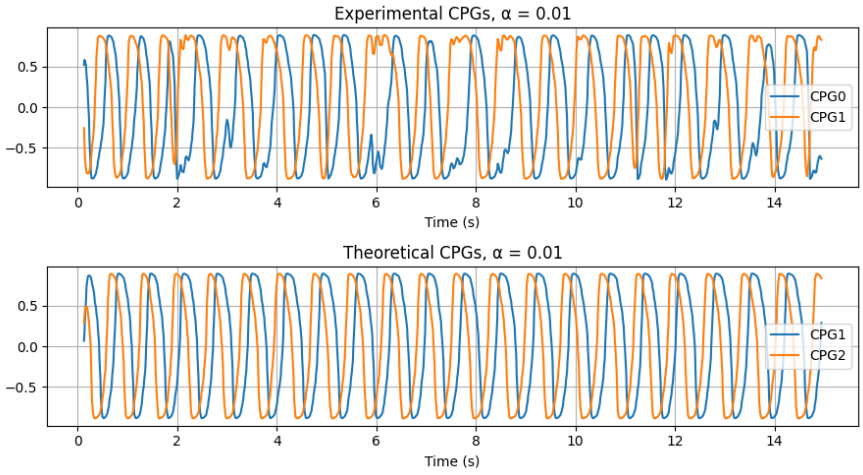



(c) Hindleg Foot Contact Error Distribution

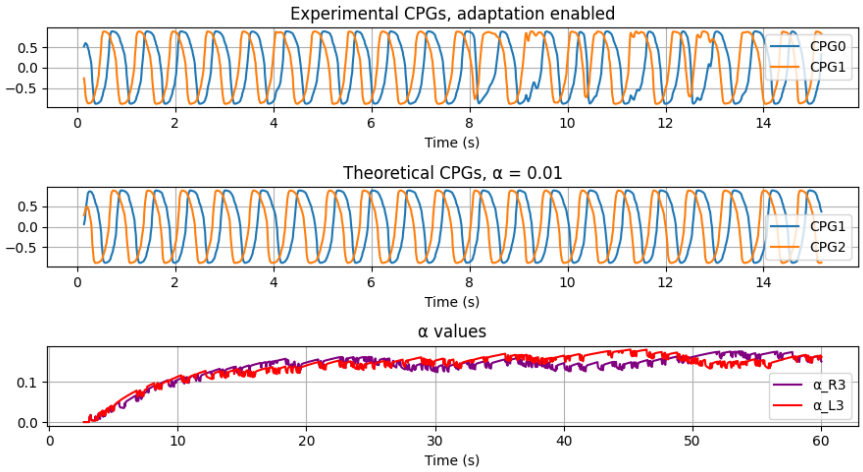
Figure 5.5: Error Distributions, extrinsic modulatory input (MI) classified

## 5.3 Discussion

The results obtained from the offline learning phase demonstrate the effectiveness of the Radial Basis Function (RBF) network in predicting the target trajectories generated by the Central Pattern Generators (CPGs)(fig. 5.2).
In Figure 5.1, it is shown the early stopping mechanism during the learning of the weights and centers of the RBF network. The error evolution during the training of the RBF network for the three legs of the robot is shown. The error decreases as the number of epochs increases, indicating that the network is learning to predict the target trajectories more accurately over time. The error evolution is similar for all three legs, demonstrating that the RBF network can learn to predict the target trajectories consistently across different legs.
Another parameter taken in consideration is the extrinsic modulatory input. The MI is used to change the frequency of the Central Patterns Generators and in this way it modifies the capability of the control system in following the feet trajectories. The results (fig. 5.3, 5.4, 5.5) show that the MI is crucial to have a good control system. It is possible to notice in the 3D and 2D trajectories that the robot is not able to follow the target trajectories while increasing the $MI$ value. The figure 5.5 shows the error distribution of the three legs of the robot. The error distribution increases as the MI value increases, indicating




(a) Comparison between a theoretical CPGs with $\alpha = 0.1$ and the experimental CPGs without adaptation but fixed $\alpha$-value.




(b) Comparison between a theoretical CPGs with $\alpha = 0.1$ and the experimental CPGs with adaptive $\alpha$-value (the graph show left and right HindLeg).

Figure 5.6

that the control system's performance deteriorates with higher MI values. As second experiment, the online learning is tested. The robot is now released into the simulation environment, and the CPGs and RBF network are used to control its locomotion.
In Figure 5.6 it is shown how the theoretical CPGs signals are compared with the experimental CPGs signals in both adaptive and fixed $\alpha$ value. The results show that the robot is able to produce a intralimb coordination similar to the theoretical CPGs signals, but the more the simulation goes on, the more the

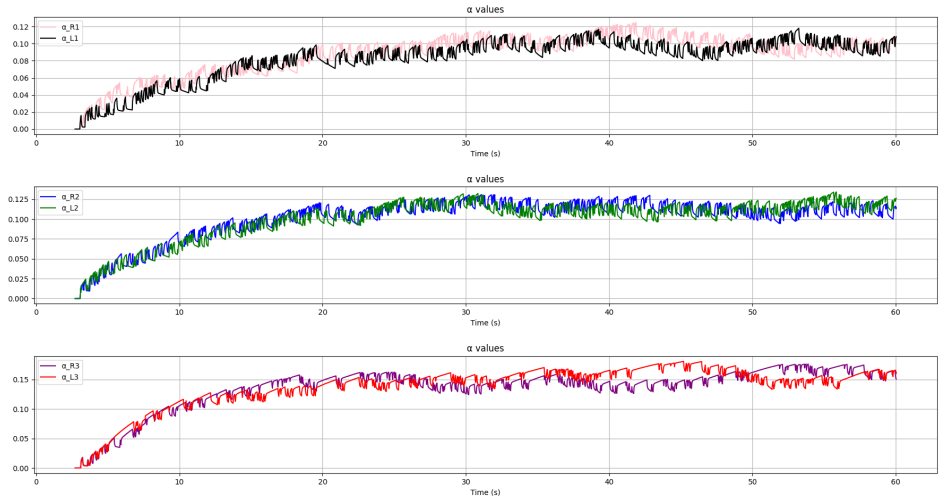
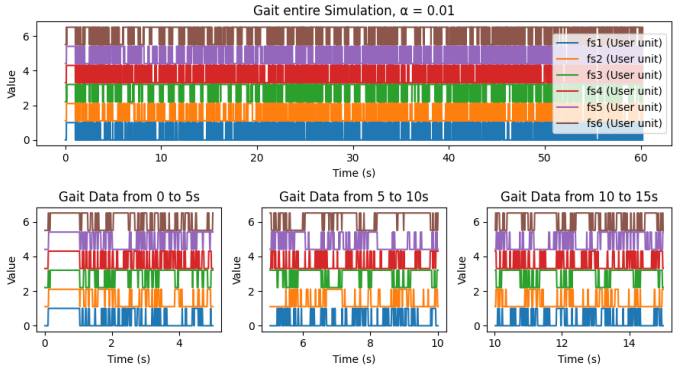
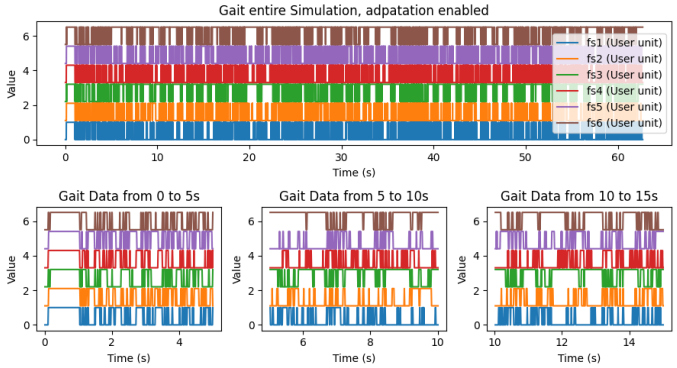



Figure 5.7: Adaptation of the alpha values during the simulation, every graph shows the alpha values for each leg, left and right.

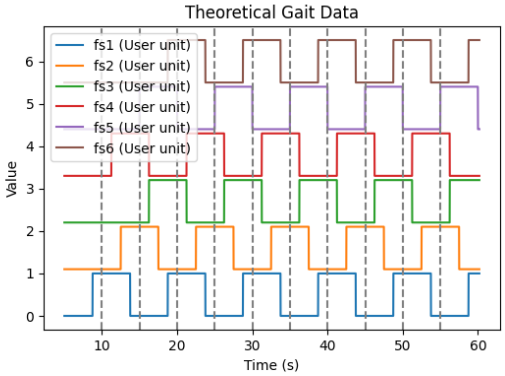
quality of the signals decreases. In the fixed $\alpha$ value, the signals, from the beginning, are not being corrected from the feedback mechanism (Forward Model + DualLeaner correction), for this reason it is possible to notice how the produced CPGs are less similar to the theoretical CPGs signals. In the adaptive $\alpha$ value case, the signals are corrected from the feedback mechanism, and the produced CPGs are more similar to the theoretical CPGs signals. In the last graph of Figure 5.6, it is shown how the alpha value is changing during the simulation based on the foot contact signal.
The behaviour of $\alpha$ values during the adaptation aligns with our expectations. On average, they show an upward trend due to the slow accumulation of error each time the robot's foot makes contact with the ground and resets its period. Additionally, as seen in Figure 5.7, the $\alpha$ values differ for each leg (Foreleg, Midleg, and Hindleg). Moreover, there are noticeable differences in the $\alpha$ values between the left and right legs, yet these values remain relatively close to each other.
This trend (Figure 5.7) is particularly important as it demonstrates the adaptive nature of the control system, based on the feedback received during the robot's locomotion.
The distinction in $\alpha$ values between different legs highlights the unique adaptation required for each leg to optimize stability and efficiency during movement. It is essential to note that each leg has a different morphology in terms of weight and dimensions and for this reason a different reference trajectory on the foot.
In addition, it has been imposed a morphological symmetry along the medial plane on MedaExtra, ensuring that the left and right sides of the robot mirror




(a) Gait signal for the entire simulation with three extracted intervals for clearer visualization. The $\alpha$ value is fixed to 0.1.

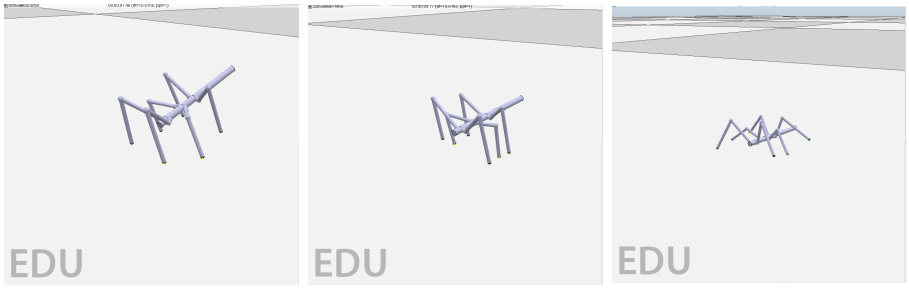



(b) Gait signal for the entire simulation with three extracted intervals for clearer visualization. The adaptive $\alpha$ value is used.




(c) Expected gait cycle in case of a self-organized gait and successful interlimb coordination.

Figure 5.8: Comparison between the theoretical gait and the real gaits of the robot with fixed and adaptive $\alpha$ values. The signals are measured from the foot contact sensors.




(a) (b) (c)

Figure 5.9: Insect positions during the gait cycle. The positions are shown at three different phases of the gait.

each other. By "morphological" we mean the physical characteristics of the robot, such as the weight and dimensions of each leg.
For instance, the forelegs might experience different loading and interaction dynamics compared to the hindlegs, necessitating different $\alpha$ adjustments.
Furthermore, the slight variations between the left and right legs indicate that while the control system is symmetric, it can still fine-tune its parameters to account for minor discrepancies in the mechanical performance or interaction with the environment. This fine-tuning capability is crucial for maintaining overall balance and coordination, especially when the robot navigates uneven terrain or encounters unexpected obstacles.
In summary, the observed behaviour of the $\alpha$ values meet our expectations. An adaptive correction is being produced.
By continually adjusting in response to real-time feedback, the system ensures that each leg performs optimally, contributing to the robot's overall stability and adaptability.
On the other hand, a study on the gait provides a more comprehensive understanding of the robot's locomotion behavior.
In Figure 5.8, the theoretical gait is compared with the real gaits of the robot with fixed and adaptive $\alpha$ values.
In this case, it is quite evident that the system fails to produce a stable and theoretically accurate gait.
Figure 5.8 shows the entire simulation period along with three significant intervals. The signals displayed are the six sensor signals from the robot's feet, which send a positive signal when in contact with the ground (stance phase) and a null signal when not (swing phase).
By looking at the comparison with the expected gait, it is clear that our control system does not achieve a self-organized gait. In the three intervals in the figure, we can observe occasional large arcs where the control system attempts to correct with a particular leg, resulting in an overshoot on the reference (Figure 5.8).

In Figure 5.8c a simplified version of the gait cycle that should be obtained from the experiment, where the three left legs are slightly out of phase, and the right legs have an almost opposite phase.
It is important to consider that the type of control we are using for the joints is based on achieving target angles. If we wanted to add more complexity to the control system, we could incorporate torque control to achieve greater resulting fluidity.
Figure 5.9 shows the robot's movement during the gait cycle, particularly highlighting the initial, intermediate, and peak positions of the obtained gait. The significant imbalance that the robot exhibits while trying to follow the target trajectories is evident.
The results of the online learning phase indicate that the control system's performance is suboptimal, with the robot struggling to maintain stability and achieve efficient locomotion. The adaptive feedback mechanism shows some promise in improving the robot's response to changing conditions, but further refinements are needed to enhance the control system's overall performance.

### 5.3.1 Future Improvements

As future development, the adaptive control system can be improved by adding a muscle model to the robot, in order to achieve a more accurate and realistic locomotion by adding a control loop on the compliance of the single leg. At the moment, the control system is based on the joint angles, but the addition of a muscle model would allow the robot to replicate muscle-like motion, enhancing its ability to withstand unexpected disturbances and adapt to changing conditions. A control on the joints torque can be applied in parallel to the existing control system.
The stiffness in a limb, in natural context, is generated by the stretch reflex and variable according to the stance/swing phases, adjusted by the neural system [6]. Muscle stiffness is high in the stance phase to support a body against gravity, and low in the swing phase for compliance against the disturbance [6]. The muscle model offered by [5] can be implemented in the neural control system to achieve the an intralimb control on the compliance. The idea is to have $K$ and $D$ dynamically adjusted online to respond to various perturbations affecting the robot's behavior. This process relies on a simple actuator position feedback instead of a complex force/torque feedback or physical compliant mechanisms like springs Consequently, the adaptive neuromechanical controller enhances the robot's ability to withstand unexpected disturbances, such as impact forces, payload variations or pulling forces.
A well known Online Adaptive Compliance (OAC) control is found in [5] where the authors propose a novel method for a quadruped robot. The OAC is derived from an impedance control method [17]. To achieve the desired compliance, it manipulates the stiffness and damping of the robot's joints based on the error value between reference and prediction of the neural system. The

desired positions are generated by the control system responsible of the locomotion (in our case the CPG+RBF structure) and by looping the derivative of the error we extend the control capacity of our system.

# 6 Conclusion

In review, this work elaborates on previous approaches to adaptive control systems for hexapod robots by implementing novel 3D trajectories from biological data. Overall, the model showed some successful features, such as an adaptive alpha controller, which adjusted to match the disturbances on the system. In addition, the neural structure (CPG+RBF) was able to learn the trajectory reference and obtain an accurate mapping of the trajectories and thus a functioning intra-limb coordination, as shown by the results section 5. However, the overall model was unable to fully replicate successful walking gaits, as the error factor grew over time and the control could not decrease it. This could be due to the lack of complexity of the model, which could be improved by refining the neural structure and adding more control models to manage different aspects of the robot.
In conclusion, this thesis has deepen the study done in [7], by choosing a 3D implementation for the trajectory reference. By exploiting the idea behind the adaptive control system applied in this study it will be possible to build a portable solution that can be applied in different fields than hexapod robots. By drawing on principles from 3D insect locomotion trajectories and neural control, we have demonstrated the potential for creating robots that can learn how to self-organize and build a structure that can navigate properly if refined with enough data. The proposed solution is a step towards a more general approach to adaptive control systems that can be applied to a wide range of applications, from robotics to prostetic to in the end industrial automation.

# Acknowledgements

I would like to thank my supervisor, Dr. Poramate Manoonpong, and our collaborator Thirawat Chuthong for their guidance and support throughout the course of this project. I would also like to thank my family and friends for their encouragement and support. Finally, I would like to thank the Southern Denmark University of Odense for providing me with the opportunity to undertake this project.

# Bibliography


[1] Subhi Shaker Barikhan, Florentin Wörgötter, and Poramate Manoonpong. Multiple decoupled cpgs with local sensory feedback for adaptive locomotion behaviors of bio-inspired walking robots. In Angel P. del Pobil, Eris Chinellato, Ester Martinez-Martin, John Hallam, Enric Cervera, and Antonio Morales, editors, *From Animals to Animats 13*, pages 65–75, Cham, 2014. Springer International Publishing.

[2] Andrew Biewener and Sheila Patek. *Animal locomotion, Chap 1-2.* Oxford University Press, 2018.

[3] Florin Dzeladini, Nadine Ait-Bouziad, and Auke Ijspeert. *CPG-Based Control of Humanoid Robot Locomotion*, pages 1099–1133. Springer Netherlands, Dordrecht, 2019.

[4] J. Ghosh and A. Nag. *An Overview of Radial Basis Function Networks*, pages 1–36. Physica-Verlag HD, Heidelberg, 2001.

[5] Carlos Viescas Huerta, Xiaofeng Xiong, Peter Billeschou, and Poramate Manoonpong. Adaptive neuromechanical control for robust behaviors of bio-inspired walking robots. In Haiqin Yang, Kitsuchart Pasupa, Andrew Chi-Sing Leung, James T. Kwok, Jonathan H. Chan, and Irwin King, editors, *Neural Information Processing*, pages 775–786, Cham, 2020. Springer International Publishing.

[6] Hiroshi Kimura, Yasuhiro Fukuoka, and Avis H. Cohen. Adaptive dynamic walking of a quadruped robot on natural ground based on biological concepts. *The International Journal of Robotics Research*, 26(5):475–490, 2007.

[7] Alexander Dupond Larsen, Thies H Büscher, Thirawat Chuthong, Thipawan Pairam, Hendrik Bethge, Stanislav N Gorb, and Poramate Manoonpong. Self-organized stick insect-like locomotion under decentralized adaptive neural control: From biological investigation to robot simulation. *Advanced Theory and Simulations*, 6(8):2300228, 2023.

[8] Michael Mangan, Dario Floreano, Kotaro Yasui, Barry A Trimmer, Nick Gravish, Sabine Hauert, Barbara Webb, Poramate Manoonpong, and Nicholas Szczecinski. A virtuous cycle between invertebrate and robotics research: perspective on a decade of living machines research. *Bioinspiration & Biomimetics*, 18(3):035005, mar 2023.

[9] Poramate Manoonpong, Ulrich Parlitz, and Florentin Wörgötter. Neural control and adaptive neural forward models for insect-like, energy-efficient, and adaptable locomotion of walking machines. *Frontiers in Neural Circuits*, 7, 2013.

[10] Poramate Manoonpong, Luca Patanè, Xiaofeng Xiong, Ilya Brodoline, Julien Dupeyroux, Stéphane Viollet, Paolo Arena, and Julien R. Serres. Insect-inspired robots: Bridging biological and artificial systems. *Sensors*, 21(22), 2021.

[11] Aitor Miguel-Blanco and Poramate Manoonpong. General distributed neural control and sensory adaptation for self-organized locomotion and fast adaptation to damage of walking robots. *Frontiers in Neural Circuits*, 14, 2020.

[12] Frank Pasemann, Manfred Hild, and Keyan Ghazi-Zahedi. So(2)-networks as neural oscillators. In *Proceedings of the International Work-Conference on Artificial and Natural Networks*, pages 144–151, 06 2003.

[13] Matheshwaran Pitchai, Xiaofeng Xiong, Mathias Thor, Peter Billeschou, Peter Lukas Mailäder, Binggwong Leung, Tomas Kulvicius, and Poramate Manoonpong. Cpg driven rbf network control with reinforcement learning for gait optimization of a dung beetle-like robot. In Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2019: Theoretical Neural Computation*, pages 698–710, Cham, 2019. Springer International Publishing.

[14] E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013. www.coppeliarobotics.com.

[15] Tao Sun, Zhendong Dai, and Poramate Manoonpong. Robust and reusable self-organized locomotion of legged robots under adaptive physical and neural communications. *Frontiers in Neural Circuits*, 17, 2023.

[16] Tao Sun, Xiaofeng Xiong, Zhendong Dai, Dai Owaki, and Poramate Manoonpong. A comparative study of adaptive interlimb coordination mechanisms for self-organized robot locomotion. *Frontiers in Robotics and AI*, 8, 2021.

[17] Xiaofeng Xiong and Poramate Manoonpong. Adaptive motor control for human-like spatial-temporal adaptation. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2107–2112, 2018.

# A Appendix

## A.1 API Documentation

The API documentation used for the simulation can be found at this URL: `https://github.com/Gideon-98/Self-Organized-Locomotion.git` The structure of the API is divided into:

- **Online Learning and Simulation**: This section contains the C++, LUA and ROS structure for the simulation and online learning.
- **Offline Learning**: This section contains the Python structure for the offline learning.