Intelligent Control of Soft Snake Robot Locomotion with Biomimic Vertebrate System

by

Xuan Liu

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

in

Robotics Department

October 2023

APPROVED:

Dr. Jie Fu, Advisor, Department of Electronic & Computer Engineering, University of Florida

---

Dr. Cagdas D. Onal, Dissertation Committee, Department of Robotics Engineering, Worcester Polytechnic Institute

---

Dr. Carlo Pinciroli, Dissertation Committee, Department of Robotics Engineering, Worcester Polytechnic Institute

---

Dr. Ming Luo, Dissertation Committee, School of Mechanical and Materials Engineering, Washington State University

Abstract

Soft snake robots have unique advantages in traversing through cluttered and confined environments because they are equipped with highly flexible body structures, deformable materials, and high sensitivity almost in any part of their body. These advantages have resulted in great expectations for the snake robots in many difficult applications, including social rescuing, cave exploration and medical operations, etc. However, planning and control of such types of robots remains a challenging problem, as these robots have infinitely many degrees of freedom (DoFs) in their body links, and soft actuators with hard-to-identify dynamics.

By taking inspiration from the cerebral and spinal control of rhythmic behaviors in natural animals, we in this thesis develop bio-inspired locomotion controllers that allow the robot to freely sense and explore the environment, and flexibly maneuver its locomotion modes with embodied intelligence in a planar workspace. These controllers are based on the concept of Central Pattern Generators (CPGs), which are a series of mathematical models that describe spinal neural circuits' activities that generate the rhythmic patterns for animals' organ contraction and locomotion.

Among the popular CPGs in bio-inspired robot control, the Matsuoka oscillator is well-known for generating high-fidelity rhythmic neural oscillation patterns for robot locomotion gaits in mimicking animal behaviors. However, its nonlinearity makes it harder to analyze the system state properties and gradually becomes less popular in robotics locomotion studies. During my study on the learning-based locomotion of soft robot snakes, we found that the rhythmic patterns in the Matsuoka oscillator can be easily learned and maneuvered by a model-free RL algorithm. On the basis of Matsuoka's theory, we further justified that the oscillation features of the Matsuoka oscillator, including oscillation bias, frequency, and amplitude have a clear relation with the specific coefficients of the Matsuoka oscillator, and therefore can be efficiently controlled by the RL agent. Such a mechanism allows the proposed control framework to easily learn flexible steering and speed control of the soft snake robot when tracking dynamically changing goals. However, as we tried to incorporate the sensory feedback mechanism in the Matsuoka CPG system to realize the contact-aware locomotion of our soft snake robot, we encountered a problem such that the conventional feedback approach could bring significant overshoot and delay to the oscillation patterns of the Matsuoka oscillator, and therefore impede the performance of the whole RL-CPG control scheme during the contact-aware locomotion of the robot. To solve this issue, we develop a novel sensory feedback mechanism for the Matsuoka CPG network. This mechanism allows the Matsuoka CPG system to work like a "spine cord" in the whole contact-aware control scheme, which simultaneously takes the stimuli including tonic input signals from the "brain" (a goal-tracking locomotion controller) and sensory feedback signals from the "reflex arc" (the contact reactive controller), and generates rhythmic signals to actuate

the soft snake robot to slither through densely allocated obstacles. In the design of the "reflex arc", we develop two distinctive types of reactive controllers – 1) a reinforcement learning (RL) sensor regulator that learns to manipulate the sensory feedback inputs of the CPG system, and 2) a local reflexive sensor-CPG network that directly connects sensor readings and the CPG's feedback inputs in a specific topology. These two reactive controllers respectively facilitate two different contact-aware locomotion control schemes.

In summary, the original contribution of this thesis can be organized in two folds:

- In theory, we have first analyzed and proved the Matsuoka CPG's steering maneuverability to allow an organic composition of the RL module and CPG module to form an efficient learning-based locomotion controller, which is also tested to be generalizable to other robotic platforms. In addition, we have developed free-response oscillation constraints (FOC) of the Matsuoka CPG system for sim-to-real transfer. Last but not least, we have completed the development of the sensory feedback mechanism in Matsuoka CPG system. Such a mechanism is combined with two feedback reactive controllers based on two different theories (hybrid control and local reflexive controller) to realize contact-aware locomotion of the soft snake robot.

- In practice, we design and build three generations of soft snake robots to optimize their performance and expand their functionality. The optimality and robustness of the proposed control design are validated in both simulated and real soft snake robots, along with a sufficient comparison to other methods (including other RL and conventional Matsuoka CPG systems). The contact-aware locomotion control schemes are tested and evaluated in both simulated and real soft snake robots, showing promising performance in the contact-aware locomotion tasks. Our experimental results have validated the advantages of the Matsuoka CPG system for bio-inspired robot controller design.

Overall, our series work is based on the theory and application of the Matsuoka oscillator, including the discussion and derivation of the special properties of the CPG system. The contribution covers hardware design and manufacturing, control scheme design (including locomotion control and sensory feedback control), and experiment design and implementation (including simulation and reality). It makes a significant breakthrough in the research of bio-inspired robot control.

Acknowledgements

Before the formal discussion of my thesis, I would like to acknowledge all the great people that I met and worked with and who made this thesis possible. First of all, I would like to infinitely thank Professor Jie Fu who hired me as a Ph.D. student in her group, and always trusted me and encouraged me to investigate my own research ideas, while always being very supportive and always giving good advice when I got stuck. Her ability to quickly find a solution to a hard problem really impressed me. Moreover, she encouraged me to present my work at different meetings, workshops, and conferences, which gave me a lot of opportunities to communicate with different groups of interesting people and expanded my vision of robotics studies. Furthermore, she is a great person to know and to work with. Her patience, enthusiasm, and immense knowledge set a shining example for everyone in the CIRL lab. Outside of the lab, she is a great mother of two little kids. Taking good care of her family while overloaded by huge amounts of work, she still manages to make sure every student is doing well in both study and daily life. The pandemic was the hardest period for me, during which my research partner left the group, I was stuck on my work for a long time, and my father's health issue back home made me almost heartbroken. It was Professor Fu who allowed me to fly back immediately to take care of my family. Without her advice and support, and the support of the staff of WPI and the RBE department, I would not be able to continue my work remotely. Without her help, I would not be able to accompany my dear father through the last year of his life.

Here I want to immensely thank Professor Cagdas Onal and Professor Ming Luo because the main piece of work of this thesis would not exist without the great collaboration I had with them. Their initiative work on the soft snake robot lays a great foundation for me to explore and test my ideas on this wonderful platform. Although they are super busy, they still managed to squeeze time to discuss with me. Both their suggestions on hardware design and their proposals on the soft snake robot research project have provided strong support for my dissertation study.

This thesis was possible thanks to the support of the National Science Foundation, project #1728412, and the teaching assistant and research assistant scholarships from WPI and Professor Fu. I would therefore like to thank all the related funding agencies for the great research opportunities they create.

Now I would like to thank the members of my thesis committee: Cagdas Onal, Carlo Pinciroli, and Ming Luo for accepting to be part of the committee, for the time they spent on it, and for giving me very good feedback on different stages of my thesis study. Except for Professor Onal and Professor Luo whom I've known for a long time, I chose to invite Professor Pinciroli because of his well-known patience, enthusiasm, and kindness to the students in the RBE department. What I really appreciate is that although soft robot control is not in Professor Pinciroli's research field, he can always provide me constructive advice with accurate feedback and supportive sentences on my writings and presentations. These interactions have left

me great impression on him for being both a good supervisor and a great friend.

Then I would like to thank Renato Gasoto for all the good times we had working on the soft snake robot project and especially on simulating the soft snake robot in Flex Nvidia, for the patience on the endless experiments and engineering discussions, and for the good technical advice that he gave me. It is a pity that our collaboration ended suddenly and I wish him all the good luck in his new job.

Then I would like to thank Yinan Sun and Zhaoyuan Ma for being extremely supportive during the hardest time during my remote study, for the help and suggestions on hardware design and experiments when I had to rebuild the soft snake robot and construct the lab environment at home from zero. Without their help, it would be much harder for me to initiate my work. I also remember our good conversations about career plans, hobbies, and other interesting stuff, which added a lot of relaxation to that hard time.

Then I would like to thank Abhishek Kulkarni for all the deep and interesting exchanges in philosophy, culture, history, food, and music. During those conversations, there were so many thoughts and ideas coming from his mind that impressed me a lot. I also remember the time when he took us to a great Indian music concert, which totally opened my mind to Indian culture. I really appreciate all the great conversations we had during our Ph.D.

Then I would like to thank my colleagues including Lening Li, Haoxiang Ma, and Zhentian Qian for being extremely supportive during the pandemic, for the help in group meetings, classes, and experiments, and most importantly for the help in daily life. Remembering the time during the pandemic, when I did not have a car and needed to go out for food and medicine, it was my colleagues who drove me around every weekend and took a break outdoors. Without them, I could not imagine how I went through that period alone. Then I would like to thank all the lecturers I have learned from at WPI, my roommates and friends during my PhD at Worcester, including but not limited to Professor Jing Xiao, Professor Andrew Clark, Professor Joseph D Fehribach, Doctor Tanek Zhang, Yudong Yu, Haowei Zhao, Shichen Cao, Doctor Kenechukwu C. Mbanisi, Doctor Shaoju Wu and Doctor Yan Wang for all the great times we had that make a tremendous difference and certainly helped me a lot during my four years at Worcester.

To the students, I supervised: Ziyi Jiang, Yash Sandeep Shukla, Thejus Jose, and Marmik Patel. It was a pleasure working with you and I surely benefited from these collaborations.

Special thanks to Doctor Kiyotoshi Matsuoka, who invented the Matsuoka oscillator. Although I haven't met him or talked to him, his great contribution to the central pattern generator studies has inspired me throughout my PhD study. Ten years ago Professor Matsuoka posted his last unpublished draft on the analysis of the Matsuoka oscillator on his personal website, and there have been no more updates about him since then. For almost twenty years, his work has been underrated. A lot of people believe that the Matsuoka oscillator is too complicated and hard to control in bio-inspired robotics. In my dissertation, a lot of work has been

done to show that the Matsuoka oscillator is indeed a very good CPG system for bio-inspired robot control and is actually easy for learning-based agents to operate. To me, Doctor Matsuoka is like a spiritual old friend. I thank him for making me discover the cool world of neural dynamical systems and its ramifications. And I hope him happy if he has a chance to see my work.

And last but certainly not least, I would like to express all my gratitude to my mother Xiaoying Chen and my father Yonghong Liu for their unfailing cultivation, support, and encouragement over the last 31 years. What I have accomplished would not have been possible without you. During the pandemic, my mother tried everything she could to prolong my father's life when he had cancer. In the meantime, she still managed to support my work and daily life. Her perseverance, love, and the spirit of never giving up are the best motivations in my life. My father was an excellent electronic engineer and project manager, who contributed to the manufacturing and sales of the first Chinese TV brand. His wisdom, responsibility, and efficiency have always been a great sample in my mind. Unfortunately, his health issue impeded his career and made him live in torture for almost twenty years. Even in the final stage of his life, he was still trying to give me some advice with his knowledge and experience. How I wish he could live to see my graduation and be proud of me. At this moment, I would like to dedicate this thesis in memory of my dearest father!

Xuan Liu
Fuzhou, China, November 2023

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

1.1.1 Why Study Snake Robot?

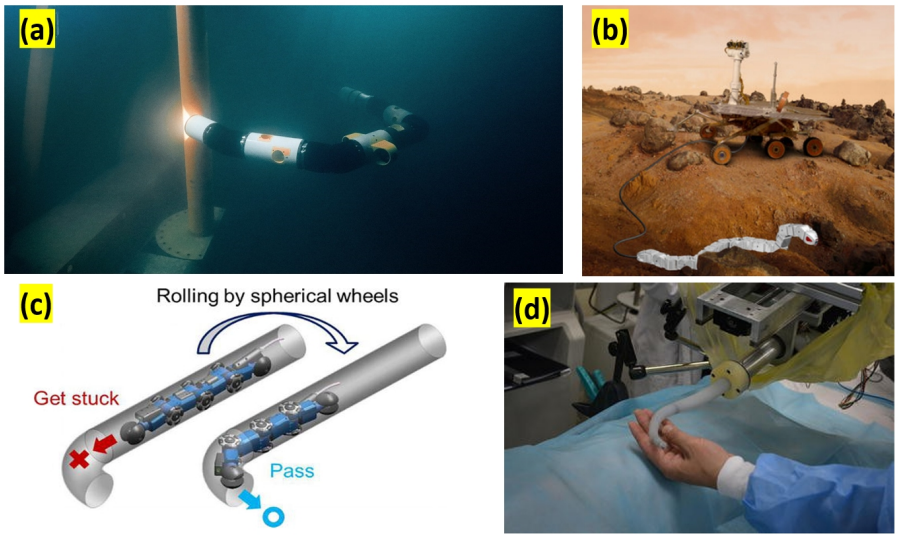


Figure 1.1: (a) Eelume snake robot for deep sea exploration and excavation. (b) Snake robot for space missions [56]. (c) Pipe inspection snake robot [30]. (d) Surgical soft snake robot [93].

Snake robots have been a hot topic in robotics studies. Due to the high flexibility and abundant gait types of biological snakes, they are adaptable to almost all unstructured solid or liquid terrains. They can climb, glide, slither, swim, and even turn into a manipulator in certain scenarios. This also results in great expectations for the snake robots in many difficult tasks. Although there have been a huge number of snake robots developed in the past decades, our knowledge of how snake

robots can be controlled efficiently remains limited. Moreover, the performance of artificial snakes is not even close to the natural snakes outside the labs. As a result, making the snake robots move and sense as good as biological snakes has become an eagerly desired goal for robotics researchers at the current stage.

Soft snake robots have unique advantages in traversing through cluttered and confined environments because they have highly flexible body structures and deformable materials. In particular, soft robotic snakes have the unique potential that any part of their body, if properly controlled, could adapt to and reduce the impact from collisions, or even benefit from the propulsion force generated by the contacts with obstacles. Based on these advantages, contact-aware soft snake robots can be potentially applied to several scenarios, including search-and-rescue [24], pipe inspection [49] and medical surgery [93], etc (see Fig. 1.1). However, planning and control of such types of robots remains a challenging problem, as these robots have infinitely many degrees of freedom (DoFs) in their body links, and soft actuators with hard-to-identify dynamics. In my dissertation study, I aim to find out a new solution for the soft snake robot locomotion which allows the robot to freely sense and explore the environment, and flexibly maneuver its locomotion modes to track targets with embodied intelligence in planar workspace.

1.1.2 Existing Snake Robot Locomotion Controllers in 2-Dimensional Space

The study of snake robot locomotion control has developed multiple different solutions in the past decades. Most contributions can be mainly categorized into model-based and model-free approaches. In particular, the existing contact-aware locomotion controller, as a class of hybrid model-based controllers, will be introduced individually.

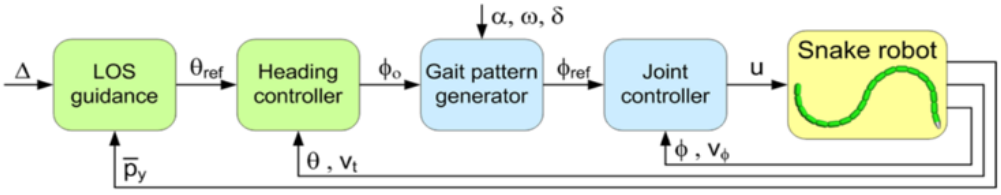


Figure 1.2: Model-based control scheme for rigid snake robot locomotion [41].

Model-based approaches: Basically, the model-based control of snake robots originated from the idea of modeling the kinematic and dynamic features of a snake robot and developing the path following controllers accordingly. For rigid snake robots, in Pettersen's work [41], a classic controller is proposed based on cascaded systems theory, which stabilizes the snake robot to follow several desired (planned)

paths in the planar workspace. For soft snake robots, a typical solution is proposed by a series of work in the Soft Robotics Lab at WPI [45–48]. Several approaches including approximated kinematic modeling (curvature sensing) [45] and feedback torque control (in joint space) [46] have been proposed to simplify the dynamic or kinematic modeling and control of the soft snake robots. Based on the approximated models, an iterative learning controller is developed with a gait correction technique (based on curvature feedback) for the soft snake robot to track the planned bounded trajectories [47]. In general, stability guarantee and interpretable dynamics are the most important advantages of the model-based approaches in certain scenarios with pre-planned paths and known environmental dynamics. However, if we want to apply the soft snake robot to more complicated scenarios, the model-based approaches could face the following limitations:

- Most of the controllers are dependent on pre-planned trajectories.
- Referring to sinusoidal joint trajectories, most of the model-based controllers cannot generate flexible gait patterns and even more natural gait transitions.
- The model-based controllers usually have difficulty in tracking dynamically changing targets, especially targets in totally different turning directions, that require a drastic switch of locomotion modes.
- The model-based approaches are usually not generalizable. Modification or change of the robot hardware would require redesign of the controllers, including re-calibration of the kinematic or dynamic model and re-tuning of the control coefficients.
- Model-based approaches have difficulty in handling the actuation delay and irregular deformation caused by the soft actuators.

Model-free approaches: Aiming to improve the problems in model-based methods, people refer to model-free approaches for solutions. Essentially, model-free controllers take a complex dynamical model (like a soft snake robot) as a black box (for example, a neural network) fitting problem, and optimize the control over such black box dynamics through learning-based algorithms. Recent work has proposed model-free methods for the control of rigid robotic systems. J. Hwangbo et al. [25] proposed an end-to-end quadruped locomotion control framework that achieves decent sim-to-real performance with multiple locomotion patterns. Q. Wu, et al.proposed an efficient sinusoidal constraint for a learning-based controller of a bipedal robot, which reduced the learning complexity. Sartoretti et al. [72] proposed a decentralized approach, where each actuator of an articulated rigid snake robot is controlled independently by a neural network (NN) controller learned with an end-to-end RL algorithm. In [6], a spiking neural net (SNN) under the regulation of reward-modulated spike-timing-dependent plasticity (R-STDP) is employed to map

visual information into desired oscillating patterns to locomote a rigid snake robot chasing a red ball. Z. Bing, et al. [5] proposed another design of a learning-based framework to realize reliable sim-to-real performance for the locomotion task of a rigid snake robot.

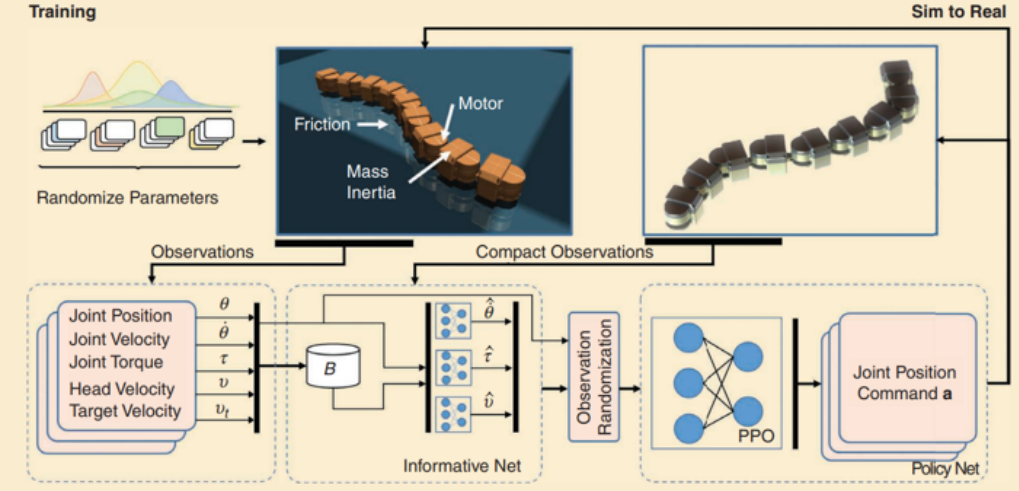


Figure 1.3: Model-free control scheme for rigid snake robot locomotion [5].

However, using vanilla reinforcement learning (RL) as an end-to-end robot locomotion controller faces several limitations:

- It always requires additional reward functions as the task objectives increase (e.g. gait patterns and constraints in locomotion), which means high data complexity for the training process.
- It is hard to adjust a converged learned policy. Techniques including transfer learning, and imitation learning are needed when the task objectives are changed and therefore increase the computation load of this approach (e.g. zero-shot sim-to-real transferring is a big challenge for end-to-end robot controllers).
- For robot locomotion control, it is usually hard to achieve gait smoothness, gait patterns, and gait stability with simple reward functions or penalty constraints through a long training process.

Contact-aware serpentine locomotion: During the serpentine locomotion of a snake robot, collisions are unavoidable as long as the obstacles are dense enough. This fact led to a special class of study based on the contact-free locomotion control of snake robots – the contact-aware locomotion control methods. So far, the solutions to the snake robots' contact-aware locomotion [19,35,71] are mainly studied and implemented on rigid snake robots and most of the control methods are model-based.

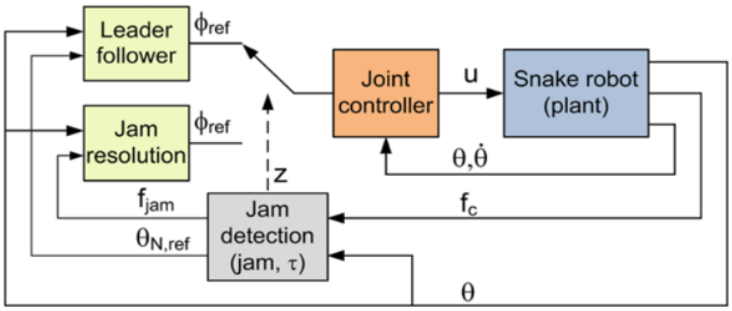


Figure 1.4: Hybrid control scheme with jam detection for obstacle-aided locomotion of a rigid snake robot [39].

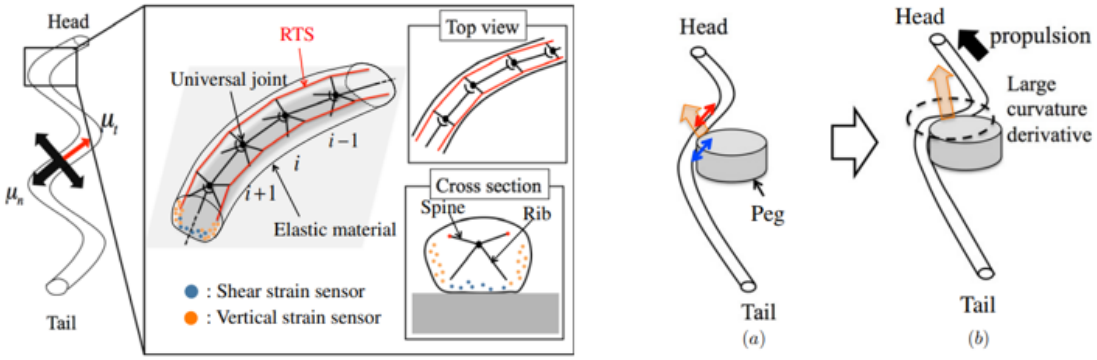


Figure 1.5: Local reflexive control method for obstacle-aided locomotion of a rigid snake robot [34].

Transeth et al. [89, 90] first defined this property as the *obstacle-aided locomotion*, wherein the snake robot actively employs external objects to generate propulsion forces during the locomotion. Their pioneer work proposed a two-module task framework of obstacle-aided locomotion with (a) a path planner that searches for a trajectory with more active contact chance for the rigid snake robot, and (b) a motion controller that controls the snake robot's real-time body movements to optimally utilize the contacts between the robot and the environment and generate desired propulsion force for the locomotion. In [22, 39, 40], a hybrid controller is developed, where a contact event is treated individually by a reactive controller that maximizes the total propulsion force at the contacting moment. This controller has been applied to a rigid snake robot and showed its reliability in maintaining beneficial propulsion force. Kano et al. [31, 34] proposed local reflexive mechanisms that interrogate the contact status between the snake robot and the obstacles to determine whether the contact is beneficial to the locomotion. In this approach, only a segment of the robot links neighboring to the link in contact react to the sensory feedback. On the basis of the local reflexive control method, a Tegotae heuristic scoring function is established by [32, 33, 35], for selecting which kind of reaction

should be applied to the contacting link of the robot given certain situations including the snake robot's shape and contacting part of the robot. From the bio-inspired perspective, inspired by the entrainment properties of the neural oscillators that allow the systems' output to be synchronized with the sensory feedback, several studies [19, 85] introduce CPG systems to the control loop of the snake robots to process the sensory feedback signals during locomotion. However, in most existing work the locomotion control inputs of the feedback CPG systems are usually constant or rather simple sinusoidal trajectories due to the difficulty of coordinating multiple complex signals through a CPG system. Conducting both intelligent locomotion control and sensory feedback control on the CPG-driven snake robot system is still a promising but rarely explored research topic.

So far, the results on contact-aware locomotion control for soft snake robots are scarce. Although a few model-free soft robot controllers [57] perform well in simulation by assuming fully proprioceptive observations, it would be appealing to enable such a capability for soft snake robots in the real world and incorporate sensory feedback into the intelligent control system. This is mainly because moving from rigid snake robots to soft snake robots faces many challenges:

- Due to the continuum of the soft actuators, it is infeasible to construct accurate kinematic/dynamic models for a soft snake robot, rendering model-based control ineffective or inapplicable.
- The pneumatic actuators in soft snake robots have nonlinear, delayed, and stochastic dynamical response given inputs, making it difficult to achieve fast responses through model-based control compared to rigid snake robots.
- It is hard to embed a tactile sensor in the soft material since the contact-free deformation of the soft body may interfere with the sensory data. As a result, the tactile sensors cannot be densely placed on the soft robot.
- Equipping tactile sensors could introduce more contact friction due to the material of the sensors, or cause more contact jamming due to the bumped shape of the sensors.
- In the scenario when a soft snake robot is traversing among unknown obstacles, the tactile sensory inputs are usually discrete and unpredictable impulses, which can result in overshoots, latency, and signal interference to a feedback control system.

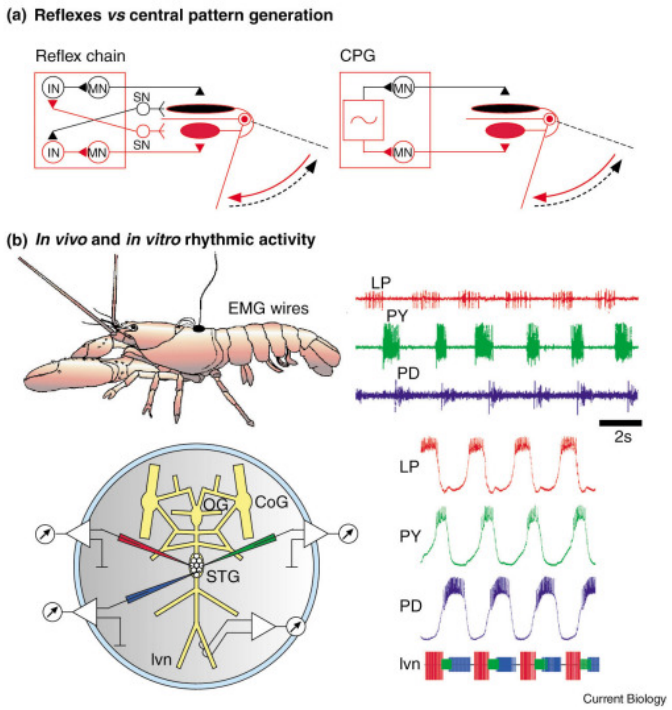


Figure 1.6: CPGs in a nature creature (recorded from the stomatogastric ganglion of the lobster *Homarus americanus*) [50].

1.2 A Sketch of the Proposed Method

1.2.1 Central Pattern Generators (CPGs) in Robot Locomotion Control

Why study CPGs: In nature, a lot of animals know how to locomote right after birth (as shown in Fig. 1.7), because they are equipped with a complete set of neural circuits that can display subconscious motion behaviors independently. Central Pattern Generator (CPG) is a mathematical model of such neural circuits (especially in the spinal cord of vertebrate animals) that can generate rhythmic and nonrhythmic activities for organ contractions and body movements in animals. Such activities can be activated, modulated, and reset by neuronal signals mainly from two directions: bottom-up ascendant feedback information from afferent sensory neurons or top-down descendant signals from high-level modules including mesencephalic locomotor region (MLR) [27] and motor cortex [67, 99].

In literature, bio-inspired control methods have been studied for the control design of rigid robots' locomotion, including legged [17, 18, 58, 60, 88] and serpentine locomotion [4, 6, 11, 12, 69, 94]. The conventional idea is to use CPG systems to generate motion patterns mimicking animals' behaviors and then track these trajectories with a closed-loop controller. In [27], the authors developed a trajectory

Figure 1.7: A baby deer picks up locomotion skills quickly after birth.

generator for a rigid salamander robot using Kuramoto CPGs and used low-level PD controllers to track the desired motion trajectories generated by the oscillator. Ryu et al. [69] established the velocity control CPG by adapting its frequency parameter with additional linear dynamics. In [94], the authors introduced a control loop that adjusts the oscillation patterns including frequency, amplitude, and phase of the oscillation to adapt to the changes in the terrain. Their results show the advantage of the Hopf oscillator on the direct access to the oscillation patterns for different locomotion purposes. In [97], the Matsuoka oscillator is combined with the amplitude modulation method to realize the steering control of a rigid snake robot. However, these approaches have not provided a way to maneuver the oscillation patterns intelligently.

Generally speaking, CPG models have the following special properties that are beneficial for robotic motion control:

- **Filtering properties**: Most CPGs are nonlinear dynamical systems with filtering properties, which lead to smooth output signals for slithering locomotion control.

- **Embedded stability properties**: Most CPG systems have stability properties that allow them to generate stable oscillation outputs and more regular gait patterns without the need for additional inputs.

- **Tunable oscillation patterns**: The oscillation patterns of most CPG systems are tunable through their specific input coefficients or parameters. This allows more natural gait transitions during the locomotion.

- **Synchronizable reflex arc**: Most CPG systems have feedback synchronization properties that are compatible with closed-loop feedback controllers.

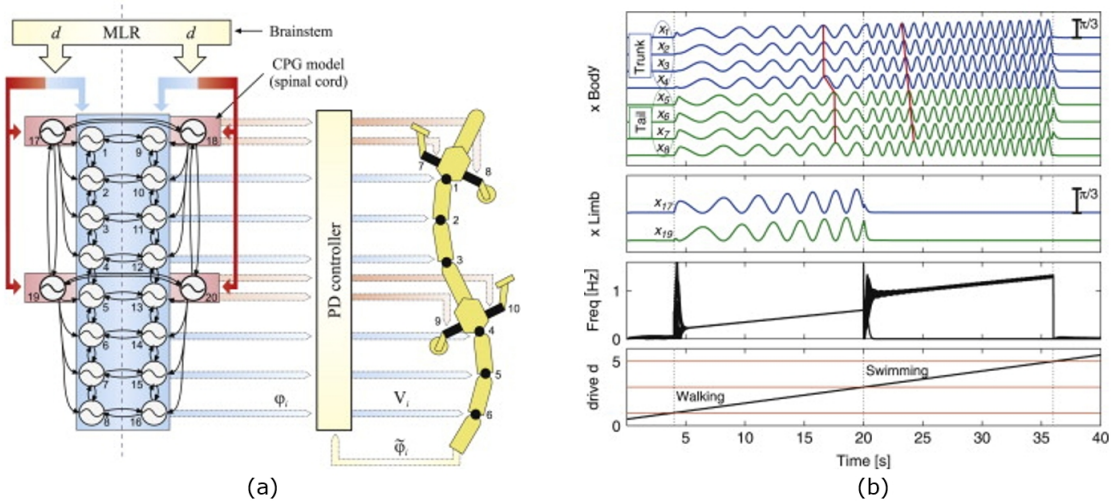


Figure 1.8: (a) CPGs for the locomotion of a salamander robot. (b) CPG output patterns or gait transition of the salamander robot [27].

Such properties bring the inspiration that if we integrate the CPG system into the model-free controller for soft snake robot locomotion, could the special properties of the CPG system help reduce the issues brought by the limitations of vanilla RL controllers?

1.2.2 Learning-based CPG Control Schemes for Robot Locomotion

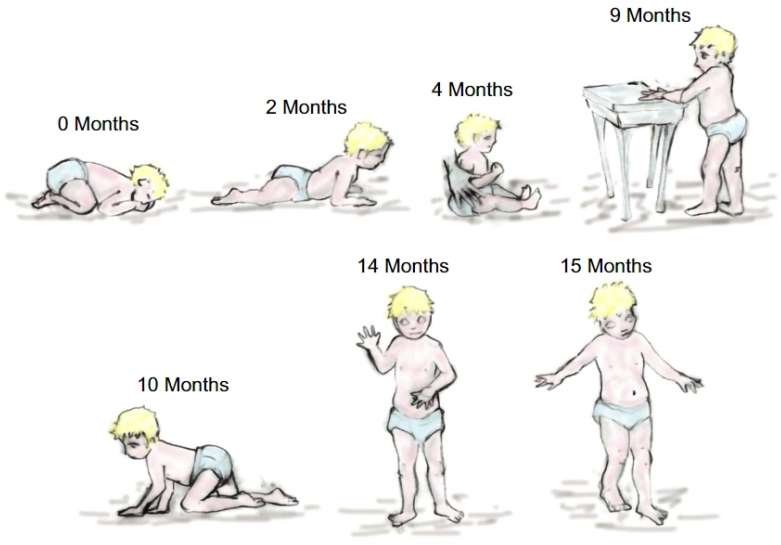


Figure 1.9: Human infant learns to locomote [23].

The idea of combining the learning-based method and CPG systems to form a bio-inspired locomotion controller is not a fresh concept in robotics studies. The biological studies have provided the evidence [23] on how natural creatures learn to coordinate with their CPG system to steer the voluntary neural oscillations after birth (as shown in Fig. 1.9). Inspired by the biological observations, T. Mori et al. [58] first proposed a hierarchical learning architecture that uses an RL controller to manipulate the input coefficients of a CPG system and output rhythmic control commands to drive bipedal locomotion in a simulator. Tran et al. [88] employed a Q-learning selector to make decisions on switching among different CPG patterns in a disturbance recovery task during bipedal locomotion. In [3], the authors proposed a CPG-RL method that directly learns the neural oscillator's intrinsic amplitude and frequency and coordinates the decoupled oscillator network to control the legged locomotion of a quadruped robot. Another recent work [8] embedded the CPG network in the policy network of an RL controller and updated the hyperparameters in the CPG system with backpropagation to optimize the overall control performance of a simulated legged robot.

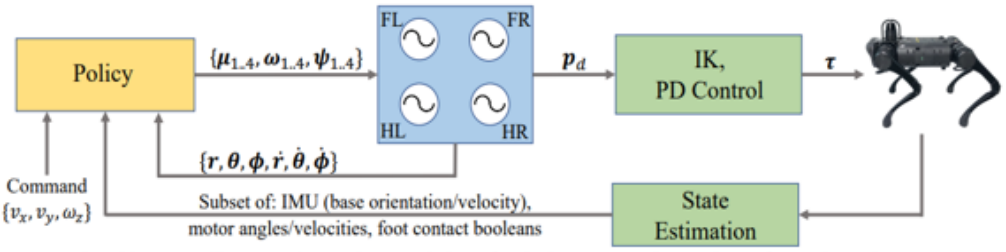


Figure 1.10: An example of CPG-RL control scheme for legged robot locomotion [3].

To the best of our knowledge, none of the investigated work in the literature has realized tracking of a randomly generated target and traversing densely distributed obstacles with a soft snake robot system. Besides the uniqueness of the application, the novelty of my work originates from the exploration of the control mechanism and theoretical analysis behind this bio-inspired control framework to leverage promising performance on the soft snake robot and other generalized platforms. To be specific, the mechanism study can be described by several important problems to solve in this study

- **Question 1:** How to select a proper CPG system that is more controllable in its dynamical properties for an RL controller?
- **Question 2:** How to build the connection between the RL module and CPG module?
- **Question 3:** Where to introduce sensory feedback in the RL-CPG framework, and how to translate the feedback signals for the target module?

1.3 Major Contribution

In this dissertation, we developed two main axes of research:

In the first part, we aim to answer **Question 1** and **Question 2** in the last section.

Why study Matsuoka CPGs? Compared to other neural oscillators used in [3,6,72], the Matsuoka oscillator has the following special properties that are highly suitable to be combined with learning-based controllers

- It is in the class of half-center [7] oscillator model that describes mutually inhibiting mechanisms in a pair of neurons. Such mechanism produces alternate activities of flexors and extensors, which can be used to directly control a pair of actuators mimicking antagonistic muscles;
- It has clear boundary conditions for the parameters such that the neurons can generate free-response oscillation when satisfying the boundary condition [52];
- On the basis of free-response oscillation, the entrainment property [53, 54] allows the intelligent controller to autonomously regulate the oscillation pattern of the system with forced-response oscillation input;
- It is a piece-wise linear system with local linearity in certain quadrants.

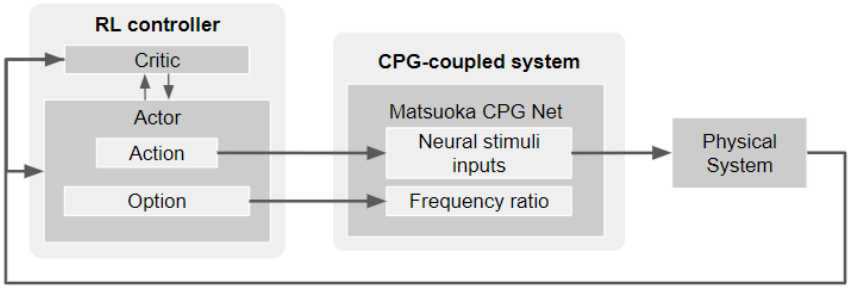


Figure 1.11: Schematic view of learning-based CPG controller.

Based on the fundamental properties of the Matsuoka oscillator, we showed that several dynamic properties of the Matsuoka oscillator can be leveraged in designing the interconnection between RL and CPG. We have proved that the *steering control* can be realized by modulating both the amplitudes bias and duty cycles of the neural stimuli inputs of the CPG network, and the *velocity control* can be realized by tuning the oscillating frequencies of the CPG net. These findings enable us to flexibly control the slithering locomotion with a CPG network given state feedback from the soft snake robot and the control objective. As a result, a bio-inspired learning-based control framework is developed for soft snake robots with two key components: To achieve intelligent and robust goal-tracking with changing goals, we use model-free

RL [74, 81] to map the feedback of soft actuators and the goal location, into control commands of a CPG network. The CPG network consists of coupled Matsuoka oscillators [51]. The Matsuoka CPG network acts as a low-level motion controller to generate actuation inputs directly to the soft snake robots for achieving smooth and diverse motion patterns. The two networks form a variant of cascade control with only one outer loop, as illustrated in Fig. 1.11. To better exhibit the performance of the control design, we build a soft snake robot based on a previous design [45] and upgrade the hardware design to realize a robust and modularized platform. A soft snake robot simulator with high fidelity is then designed and testified for the training of the control policy.

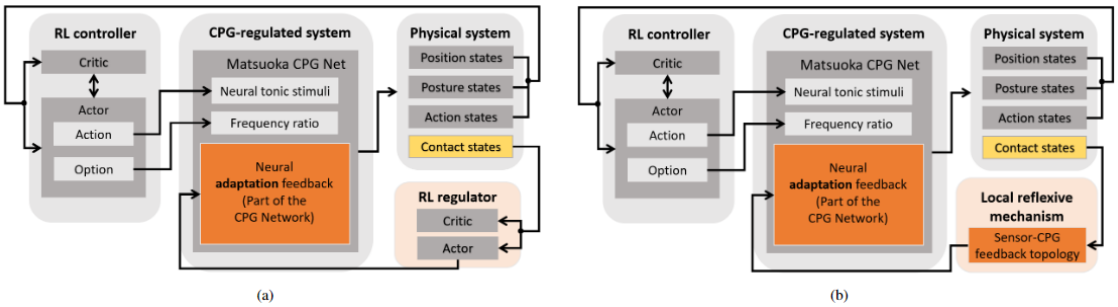


Figure 1.12: Schematic view of (a) learning reflexive PPOC-CPG, and (b) local reflexive PPOC-CPG controllers.

In the second part, we aim to answer **Question 3** in the last section. We propose an integrated approach including hardware design to enable contact sensing, as well as control design to enable contact-aware locomotion in a soft robot snake. In the hardware design, we build a group of magnetic-based tactile sensors (inspired by [92]) with scale-like cover mimicking *Scale Sensilla* [13] on the scales of real animal snakes. This structure significantly improves the sensitivity of the sensors, while making the contact friction lower and damage-free. In the controller design, we developed two control schemes, presented in Fig. 1.12. Both control schemes are composed of three major components: 1) a reinforcement learning (RL) goal-tacking controller, 2) a CPG system with feedback input, and 3) a sensory reactive controller. In the RL controller part, each control scheme includes a Deep neural network (DNN) controller that learns to maneuver the tonic input signals of the CPG system to steer the soft snake robot toward the target based on the observations including position, posture, and previous actions of the soft snake robot [44]. In the CPG system part, a novel sensory feedback mechanism of the Matsuoka CPG system is proposed. Through theoretical analysis, we reveal the desired properties of the Matsuoka oscillator's feedback mechanism in reducing the overshoot and interference of the feedback control signals compared to the conventional sensory feedback design in the Matsuoka oscillator [19, 85].

In the design of sensory reactive control, we design two different controllers that take inspiration from [39] (see Fig. 1.12a) and [34] (see Fig. 1.12b). Firstly, to enable

learning-based contact-aware locomotion, an event-triggered neural network (NN) regulator is implemented to manipulate the sensory feedback signals of the modified Matsuoka CPG system. Secondly, we incorporate the local feedback mechanism into the Matsuoka CPG network through the design of sensor-CPG network connections (see Fig. 1.12b). In this way, the CPG network generates reactive signals to the contact inputs locally and independently. Each of the two methods has its own specialties: although the AF-learning method is computationally expensive, it can achieve great performance through training. On the other hand, although the local reflexive method has a fixed heuristic rule, it is lightweight and more robust because of its local reactive property. As a result, the investigation and comparison of the above two different contact reactive control designs are necessary to comprehensively verify the advantages of the sensory feedback mechanism in the AF form Matsuoka oscillator for the soft snake robot's contact-aware locomotion.

In summary, the original contribution of this dissertation includes

1. **Theoretical analysis of Matsuoka CPG's steering maneuverability:** We analyze the property of the biased oscillation in the Matsuoka oscillator. Using describing function analysis, we show that when the tonic inputs of the Matsuoka oscillator are bounded and satisfy certain constraints, the bias of the output signal becomes linearly related to the tonic inputs. This feature makes the steering control of the snake robot easier to learn for an RL agent.

2. **Theoretical analysis of Free-response oscillation constraints (FOC) of the Matsuoka CPG system for sim-to-real transfer:** We investigate the transient property of the Matsuoka Oscillator from free-response oscillation to forced-response oscillation. Using this property, we introduce a fixed free-response tonic input signal to help regulate the amplitude and oscillation frequency of the forced tonic inputs that are generated by the RL policy. The new approach is referred to as Free-response Oscillation Constrained Proximal Policy Optimization Option-Critics with Central Pattern Generator (FOC-PPOC-CPG). This approach improves the transferability of the RL control policy learned in the simulation to the real robot.

3. **Development of sensory feedback Matsuoka CPG system for contact-aware locomotion:** We develop a novel feedback mechanism of the Matsuoka oscillator to process both the locomotion control signals and the tactile sensory feedback signals during the contact-aware locomotion of the soft snake robot. Through theoretical analysis, we reveal the desired advantages of the Matsuoka oscillator's feedback mechanism in reducing the overshoot, time latency, and interference of the feedback control signals compared to the conventional sensory feedback design in the Matsuoka oscillator [19, 85].

4. **Design of contact reactive controllers:** Based on our modification of the Matsuoka oscillator, we designed two different contact reactive controllers (*hybrid learning controller* in Fig. 1.2a and *local reflexive controller* in Fig. 1.2b)

for the modified Matsuoka CPG system to work along with the learning-based goal-tracking module developed in our previous work [44]. These lead to two different control schemes for the contact-aware locomotion of the soft snake robot under the obstacle-based goal-tracking tasks. Each method has its own unique disadvantages and advantages: the learning-based reactive controller is computationally expensive but can iteratively learn to improve its performance. The local reflexive method is lightweight and more robust because of its local reactive property but is heuristic with a fixed policy.

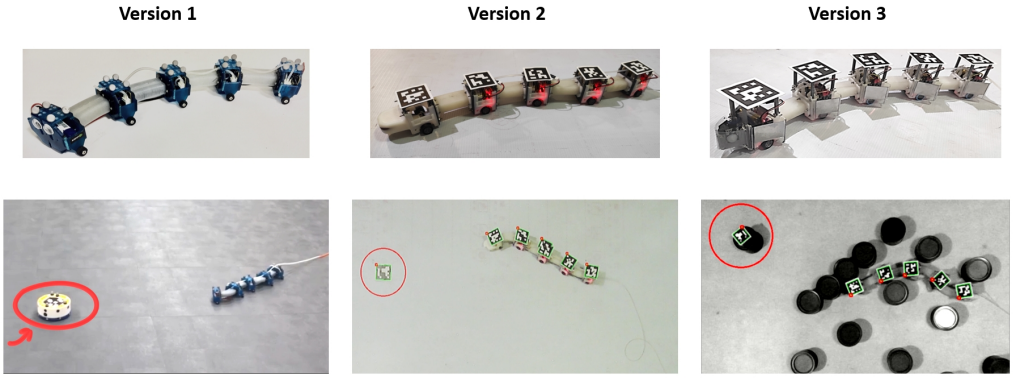


Figure 1.13: Three generations of soft snake robot hardware.

5. **Hardware designs and manufacturing:** Our soft snake robot platform has experienced three generations of development. The electronic design has evolved from a centralized design with a single micro-computation unit and sensor-free to a decentralized design with multiple independent computation and communication modules that allow multiple sensor plugins. The snake body structure has also been optimized to allow easier 3D printing production, assembling, and maintenance. To realize tactile perception in contact-aware locomotion, we design a group of magnetic-based tactile sensors (inspired by [92]) with scale-like cover mimicking *Scale Sensilla* [13] on the scales of real animal snakes. This structure can cover a larger sensing area with a small number of sensors. As a result, it improves the sensitivity to contacts with sparsely deployed sensors on the soft snake robot. In addition, the smoothness of the covering material and the scale-like structure reduce the contact friction and collision damage on the tactile sensors during contact-aware locomotion.

6. **Comprehensive sim-to-real experiments and analysis:** We added new experiments comparing the learning efficiency and adaptability of the policy between the proposed method and vanilla Proximal Policy Optimization (PPO) [74]. Based on the experimental results for both simulation and reality, we show that our soft snake robot equipped with a properly designed "vertebrate" (the CPG system) can be more easily controlled by the RL agent. Our

approach also achieves more reliable locomotion performance under various goal-reaching locomotion tasks that are unseen during the training process.

7. **Highlighting the importance of theoretical study in the bio-inspired control design:** Beyond the proposed control framework on a soft snake robot, our theoretical contribution has reemphasized the importance of studying dynamical properties of the CPG system in the RL-CPG controller design. This approach has provided a reference methodology for people who want to apply any learning-based CPG controller to robot platforms.

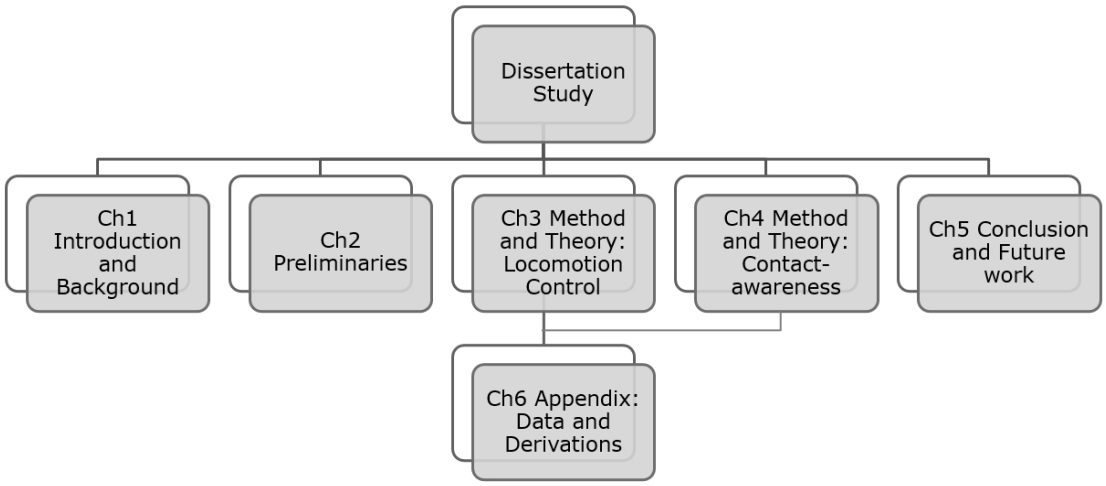


Figure 1.14: The organization of this dissertation study.

Outline: The dissertation paper is organized as Fig. 1.14 shows. In Chapter 2, I will introduce the properties of popular CPG systems and the background of model-free RL, both are combined to consist the foundations of the control method used in this study. In Chapter 3, a learning-based RL-CPG controller for soft snake locomotion will be introduced, including its design, related thoughts and properties in theory, and the experimental results that show the advantages of the proposed method. In Chapter 4, I upgrade the control framework to the contact-aware scenario, by introducing sensory feedback – the "reflex arc" to the Matsuoka CPG system to leverage the capability of the locomotion controller to traverse the densely distributed obstacle-based environment smoothly. Chapter 5 discusses and summarizes the strengths and weaknesses of the proposed work, and makes a To-do list for the possible future work to improve the performance of the current approaches. Chapter 6 provides all the data and derivations for the theories in this work.

Chapter 2

Preliminaries

2.1 Neural Oscillator Models for Robot Control

2.1.1 Kuramoto Oscillator

Different from other neural oscillators, the Kuramoto oscillator totally ignored the shape of oscillation, while only focusing on the phase and synchronization property of coupled oscillator networks. Given the governing equation

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum_{j=1}^{N} sin(\theta_j - \theta_i),$$

where K is the coupling strength, and N is the population of the network, and ω_i is the natural frequency of i_{th} oscillator sampled from some symmetric distribution, like Gaussian. According to the mean field states of a swarm of points running on a unit circle under polar coordination, with the following math description

$$re^{i\psi} = \frac{1}{N}\sum_{j=1}^{N} e^{i\theta_j},$$

where ψ is the average phase, and radius $r(t)$ measures the coherence of the population (can be taken as a vector field with the same norm, evenly distributed individuals will mostly get canceled out from nearly opposite directions). Equating the imaginary part yields

$$r\sin(\psi - \theta_i) = \frac{1}{N}\sum_{j=1}^{N} sin(\theta_j - \theta_i).$$

And therefore the governing equation becomes

$$\dot{\theta}_i = \omega_i + Kr\sin(\psi - \theta_i) \quad i = 1, ..., N.$$

This shows the very obvious mean-field character of the model, which summarizes the property of the system with statistical property with only two parameters despite the complicated interaction among a large number of nodes in the network. The early idea of statistical synchronization comes from mean-field approximation in physics. It was first discovered by Winfree and later carried forward by the Kuramoto model for its extraordinarily clear synchronization mechanism.

A special Kuramoto-type neural oscillator is the phase-amplitude oscillator applied to a rigid snake robot and a salamander robot in [27, 28]. Established by Jan Ijspeert, the oscillation for a single joint is as follows

$$\begin{aligned}
\dot{\phi}_i &= \omega_i + \sum_j (\omega_{ij} r_j \sin(\phi_j - \phi_i - \varphi_{ij})) \\
\ddot{r}_i &= a_r(\frac{a_r}{4}(R_i - r_i) - \dot{r}_i) \\
\ddot{x}_i &= a_x(\frac{a_x}{4}(X_i - x_i) - \dot{x}_i) \\
\theta_i &= x_i + r_i\cos(\phi_i).
\end{aligned}$$

Where θ_i is the oscillating set-point (in radians) extracted from the oscillator. ϕ_i, r_i, x_i are phase, amplitude, and offset (state variables). ω_i, X_i, R_i are the desired frequency, amplitude, and offset (control parameters). ω_{ij} and ϕ_{ij} are coupling weights and phase biases (influence of oscillator acted j on i). a_r and a_x are constant positive gains. Asymptotically convergence property of this oscillator shows a limit cycle converging to

$$\begin{aligned}
&x_i \to X_i \\
&r_i \to R_i \\
&\theta_i(t) \to X_i + R_i\cos(\omega_i t + \phi_0).
\end{aligned}$$

given the initial phase conditions of all oscillators.

The advantages and disadvantages of the Kuramoto oscillator when it is used for robotics applications are listed as follows:

Advantages

- It provides explicit control parameters for the controllers to modulate the oscillation patterns.
- Its stability properties provide robustness to transient perturbations.

- All control parameters can be abruptly varied without breaking the smoothness of θ (no discontinuities and jerks, a critically damped system).
- Feedback terms can be added to the state equations to maintain entrainment between control oscillations and mechanical movements.
- Never needs resetting while the control parameters are modified.

Disadvantages

- The nonlinearity increases the difficulty of analyzing the system.
- Fixed sinusoidal limit cycle behavior and limited oscillation patterns.
- Large modulation dimension increases control coefficients geometrically as the size of the network grows.

2.1.2 Hopf Oscillator

The dynamics of the Hopf oscillator are governed by the following differential equations

$$\begin{aligned}\dot{x} &= (\mu - r^2)x - \omega y + \epsilon F \\ \dot{y} &= (\mu - r^2)y + \omega x,\end{aligned}$$

where $r = \sqrt{x^2 + y^2}$, $\mu > 0$ governs the amplitude of the oscillations and ω stands for the intrinsic/inner frequency of the oscillator. This means that without perturbations (when $\epsilon = 0$), the system is oscillating at $\omega\ rad{\cdot}s^{-1}$. The oscillator is coupled with a periodic force F. When the force is zero, the system has an asymptotically stable harmonic limit cycle with radius $\sqrt{\mu}$ and frequency ω. As the limit cycle of the Hopf oscillator is structurally stable, small perturbations around its limit cycle ($\epsilon > 0$) do not change the general behavior of the system. It means that the limit cycle will still exist, only its form and time scale will change. Structural stability assures that this change is close to identity.

Rewriting the system in polar coordinates, set $x = r\cos\phi, y = r\sin\phi$, then the original system can be transformed into

$$\begin{aligned}\dot{r} &= (\mu - r^2)r + \epsilon F\cos\phi \\ \dot{\phi} &= \omega - \frac{\epsilon}{r}F\sin\phi \\ \dot{\omega} &= -\epsilon F\sin\phi.\end{aligned}$$

An important concept here is phase-lock/entrainment, which means that the oscillations synchronize with the frequency of the periodic input. Stronger coupling strength enlarges the entrainment basin.

Frequency adaptation mechanism The general form of a Hopf oscillator, perturbed by a periodic signal F, can be described as

$$\dot{x} = f_x(x, y, \omega) + \epsilon F$$
$$\dot{y} = f_y(x, y, \omega),$$

with ω some parameter that has a monotonic relation with the frequency of the oscillations (not necessarily linear). The learning rule introduced in [66] to this parameter is

$$\dot{\omega} = \pm \epsilon F \frac{y}{\sqrt{x^2 + y^2}}.$$

The sign depends on the rotating direction of the limit cycle in the (x, y) phase space. This adaptation rule generally works for various oscillators with ω converging to the value such that the frequency component of the oscillator matches the one from the input F.

The advantages and disadvantages of the Hopf oscillator when it is used for robotics applications are listed as follows:

Advantages

- It provides explicit control parameters for the controllers to modulate the oscillation patterns.
- Its stability properties provide robustness to transient perturbations.
- Input adjustable.
- All control parameters can be abruptly varied without breaking the smoothness of θ (no discontinuities and jerks, a critically damped system).
- Feedback terms can be added to the state equations to maintain entrainment between control oscillations and mechanical movements.
- Never needs resetting while the control parameters are modified.

Disadvantages

- The nonlinearity increases the difficulty of analyzing the system.
- Large modulation dimension increases control coefficients geometrically as the size of the network grows.

2.1.3 Matsuoka Oscillator

The Matsuoka Oscillator is a biologically inspired muscle contraction model that can be found in most animals. Every Matsuoka Oscillator unit contains a flexor and an extensor. The basic structure is presented in the following figure.

The rhythmic pattern of alternating bursts of flexor and extensor activities is produced by two symmetrically organized excitatory neural populations that drive alternating activity of flexor and extensor motorneurons and reciprocally inhibit each other via inhibitory interneurons.

Studies of fictive locomotion in decerebrate, immobilized cat preparations provided additional evidence for a symmetrical, halfcenter organization of the spinal locomotor CPG as well as for a critical role of reciprocal inhibition for generation and shaping of the locomotor pattern [38, 55, 68, 98]. At the same time, the specific intrinsic neural mechanisms involved in the generation of locomotor oscillations remain largely unknown.

The system formulation is

$$\begin{bmatrix} \dot{u}_i^e \\ \dot{v}_i^e \\ \dot{u}_i^f \\ \dot{v}_i^f \end{bmatrix} = \frac{1}{f_k} \begin{bmatrix} -\frac{1}{\tau_r} & -\frac{\beta}{\tau_r} & -\frac{1}{\tau_r}\mathbf{1}(u_i^f > 0) & 0 \\ -\frac{1}{\tau_a}\mathbf{1}(u_i^e > 0) & -\frac{1}{\tau_a} & 0 & 0 \\ -\frac{1}{\tau_r}\mathbf{1}(u_i^e > 0) & 0 & -\frac{1}{\tau_r} & -\frac{\beta}{\tau_r} \\ 0 & 0 & -\frac{1}{\tau_a}\mathbf{1}(u_i^f > 0) & -\frac{1}{\tau_a} \end{bmatrix} \begin{bmatrix} u_i^e \\ v_i^e \\ u_i^f \\ v_i^f \end{bmatrix} + \begin{bmatrix} \frac{s_i^e}{\tau_r} \\ 0 \\ \frac{s_i^f}{\tau_r} \\ 0 \end{bmatrix}.$$

The stability of the Matsuoka Oscillator has been studied in both frequency domain [53] and time domain [1, 51] as a piece-wise linear dynamic system. The discussion of stability depends on the results of the two indicators and several coefficients of the system. Two tables are provided to show the equilibrium point values under different cases.

$\mathbf{1}(x_1 > 0)$	$\mathbf{1}(x_2 > 0)$	x_1^*	x_2^*	v_1^*	v_2^*
T	T	$\frac{c}{\beta+\gamma+1}$	$\frac{c}{\beta+\gamma+1}$	$\frac{c}{\beta+\gamma+1}$	$\frac{c}{\beta+\gamma+1}$
T	F	$\frac{c}{\beta+1}$	$c\frac{\beta-\gamma+1}{\beta+1}$	$\frac{c}{\beta+1}$	0
F	T	$c\frac{\beta-\gamma+1}{\beta+1}$	$\frac{c}{\beta+1}$	0	$\frac{c}{\beta+1}$
F	F	c	c	0	0

Table 2.1: Equilibrium points for free vibrations by different indicator values.

Advantages

- Almost linear except the terms include the activation function.
- All control parameters can be abruptly varied without breaking the smoothness of θ (no discontinuities and jerks, property of critically damped system).

$\mathbf{1}(x_1 > 0)$	$\mathbf{1}(x_2 > 0)$	x_1^*	x_2^*	v_1^*	v_2^*
T	T	$\frac{c}{\beta+\gamma+1} - \frac{1+\beta}{(1+\beta)^2-\gamma^2}D$	$\frac{c}{\beta+\gamma+1} + \frac{\gamma}{(1+\beta)^2-\gamma^2}D$	$\frac{c}{\beta+\gamma+1} - \frac{1+\beta}{(1+\beta)^2-\gamma^2}D$	$\frac{c}{\beta+\gamma+1} + \frac{\gamma}{(1+\beta)^2-\gamma^2}D$
T	F	$\frac{c-D}{\beta+1}$	$\frac{c(\beta-\gamma+1)+\gamma D}{\beta+1}$	$\frac{c-D}{\beta+1}$	0
F	T	$c\frac{\beta-\gamma+1}{\beta+1} - D$	$\frac{c}{\beta+1}$	0	$\frac{c}{\beta+1}$
F	F	$c - D$	c	0	0

Table 2.2: Equilibrium points for forced vibrations (with constant input $g = D$) by different indicator values.

- The tonic input can accept even random processes within a certain range while maintaining the oscillation.
- Theoretically allows feedback terms to be added, but no strong guarantee for the convergence, which is rather a bounded result.

Disadvantages

- No globally closed-form solution, but piecewise solutions can be concluded respectively based on different situations.
- Not all of the controlled parameters are directly mapped to the sinusoidal wave coefficients, need some computations to find out the freq, phase, amplitude, and offset properties (especially the phase).

2.1.4 Summary

In general, the special features of the widely used CPG systems in robot motion control can be summarized as Table 2.3 shows,

Table 2.3: CPG Comparison

CPG System	**Matsuoka Oscillator**	**Kuramoto Oscillator**	**Hopf Oscillator**
Primary Structure	**Symmetric**	Unidirectionary	Unidirectionary
Linearity	**Piecewise-linear**	Nonlinear	Nonlinear
Closed Form Solution	No	**Yes**	**Yes**
Boundedness	**BIBO**	Bounded	Bounded
Periodicity	**Bifurcate**	Inherent	Structurally stable
Wave Shape	**Input adjustable**	Sinusoidal	**Input adjustable**
Oscillation Pattern Control	**Implicit**	explicit	partially explicit
Feedback States	**Unlimited**	Specific	Specific
Coupling	**State**	Phase	Frequency
Entrainment	Yes	Yes	Yes

2.2 Hierarchical Reinforcement Learning

2.2.1 Model-free RL

Policy Gradient

Policy gradient methods are a well-known group of RL techniques that are based on optimizing parameterized policies about the expected long-term cumulative reward through gradient descent [82]. The goal of policy gradient reinforcement learning is to maximize the expected return reward of a system, such that:

$$\theta^* = \arg\max_\theta E_{\tau\sim\pi_\theta(\tau)}[\sum_t r(s_t, a_t)]$$

Where $r(\cdot)$ is defined as a reward function on state action pair s_t, a_t at time t. θ is the parameter set for the function approximation method on trajectory distribution $\pi_\theta(\tau)$ calculated by policy $\pi_\theta(a|s)$ and stochastic state transition probability $Pr(s'|s,a)$. The objective can be further expanded as:

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\sum_t r(s_t, a_t)] = \int \pi_\theta(\tau) r(\tau) d\tau$$

The gradient of $J(\theta)$ then can be derived by:

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Such gradient can be applied to update the parameters of simple function approximators or backpropagate to the hidden parameters of a neural network. The formulation improves the learning performance by directly updating the policy, however, it also brings up some issues both in application and theory:

- high variance among sampled trajectories
- only converges to the local optimal solution
- the efficiency of convergence highly depends on the design of step size

Importance sampling is the first approach to mitigate the high variance and low-efficiency problem. The idea of importance sampling is to sample over an intentionally chosen distribution with a higher chance of getting the important data that might not be easily sampled from the correct distribution. Therefore, the estimator variance is reduced while keeping the expectation unbiased:

$$E_{x\sim p(x)}[f(x)] = \int p(x) f(x) dx = \int \frac{p(x)}{q(x)} q(x) f(x) dx = E_{x\sim q(x)}[\frac{p(x)}{q(x)} f(x)]$$

Taking the importance sampling, the new gradient follows the following form by chain rule:

$$\nabla_\theta J(\theta) = E_{\tau\sim\theta_{old}}[\frac{\nabla_\theta \pi_\theta}{\pi_{\theta_{old}}} r(\tau)] = E_{\tau\sim\theta_{old}}[\nabla_\theta \log \pi_\theta|_{\theta_{old}} r(\tau)]$$

Proximal Policy Optimization (PPO)

As a typical on-policy RL method using the policy gradient technique, the general goal of PPO [75] is maximizing the expected return of a system, which can be represented as:

$$\theta^* = \arg\max_\theta E_{\tau\sim\pi_\theta(\tau)}[\sum_t r(s_t, a_t)]$$

Where $r(\cdot)$ is defined as a reward function on state action pair s_t, a_t at time t. θ is the parameter set for the function approximation method on trajectory distribution $\pi_\theta(\tau)$ calculated by policy $\pi_\theta(a|s)$ and stochastic state transition probability $Pr(s'|s, a)$. The objective can be further expanded as:

$$J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\sum_t r(s_t, a_t)] = \int \pi_\theta(\tau) r(\tau) d\tau$$

The gradient of $J(\theta)$ then can be derived by:

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Such gradient can be applied to update the parameters of simple function approximators, or backpropagate to the hidden parameters of a neural network.

Let $ratio_t(\theta)$ denote the ratio between new policy and old policy $\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$, PPO modifies the importance sampling policy gradient loss function directly by adding a lower bound with clipped importance ratios, such that the modified loss function becomes:

$$L^{CLIP}(\theta) = \hat{E}_t[min(ratio_t(\theta)\hat{A}_t, clip(ratio_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)]$$

where the clip function is used to keep the policy ratio within the range of $1 \pm \epsilon$.

2.2.2 Option-Critics

The option framework [84] is one among the various fundamental frameworks and paradigms within Hierarchical RL. An option is a temporarily extended macro-action, with a series of micro-actions within that form the policy leading to sub-goals (Fig. 2.1a). This makes the RL more convenient in large-scale task space because an RL agent can now learn a policy over options rather than over primitive actions

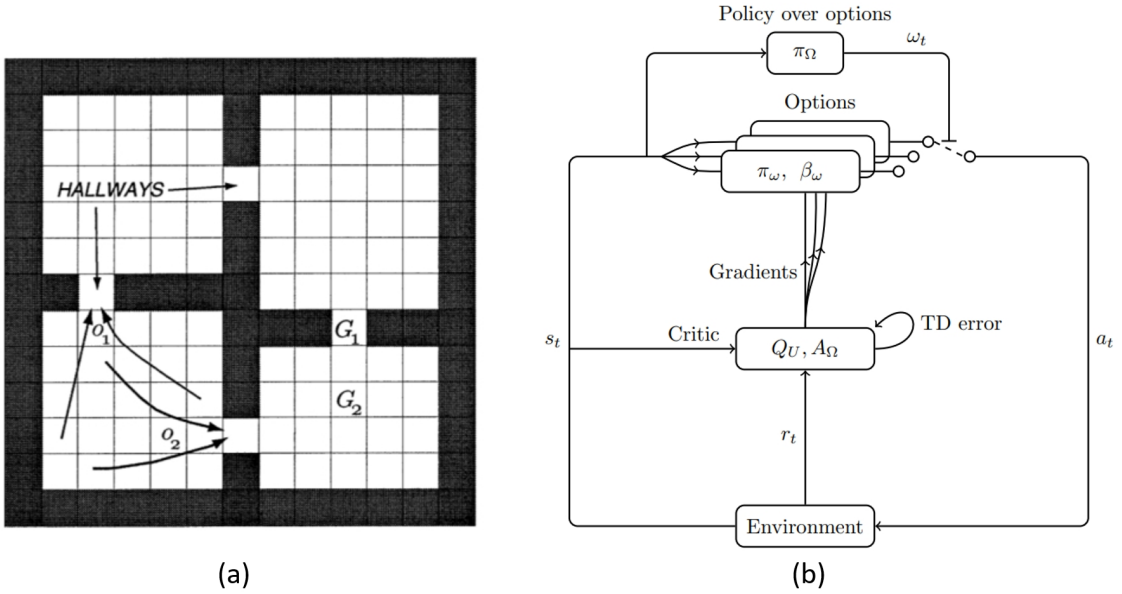


Figure 2.1: (a) Options for finding sub-goals in the four-room scenario, learning, remembering and using policies to reach those sub-goals [84].(b) Option-critic framework as a hierarchical version of actor-critic [2].

available in the environment, and avoid reward sparsity by abstracting the problem with a hierarchy.

The option-critic approach [2] is a combination of the option framework and actor-critic RL (PPO). In the option-critic framework (Fig. 2.1b), the low-level primitive actions are computed at every time step. The high-level option changes infrequently as the robot does not change velocity very often for smoothness of the macro-actions. Specifically, each option is defined by $\langle \mathcal{I}, \pi_y : S \rightarrow \{y\} \times \text{dom}(\mathbf{a}), \beta_y \rangle$ where $\mathcal{I} = S$ is a set of initial states. By letting $\mathcal{I} = S$, the macro actions are allowed to be changed at any state in the system. Variable y is a value of macro actions and $\beta_y : S \rightarrow [0, 1]$ is the termination function such that $\beta_y(s)$ is the probability of changing from the macro state to another macro state. The detailed implementation and derivations of this approach can be found in [2].

Chapter 3

Learning-based RL-CPG Controller for Soft Snake Robot Locomotion

Intelligent control of soft robots is challenging due to the nonlinear and difficult-to-model dynamics. One promising model-free approach for soft robot control is reinforcement learning (RL). However, model-free RL methods tend to be computationally expensive and data-inefficient and may not yield natural and smooth locomotion patterns for soft robots. In this chapter, we develop a bio-inspired design of a learning-based goal-tracking controller for a soft snake robot. The controller is composed of two modules: An RL module for learning goal-tracking behaviors given the unmodeled and stochastic dynamics of the robot, and a central pattern generator (CPG) with the Matsuoka oscillators for generating stable and diverse locomotion patterns. We theoretically investigate the maneuverability of Matsuoka CPG's oscillation bias, frequency, and amplitude for steering control, velocity control, and sim-to-real adaptation of the soft snake robot. Based on this analysis, we proposed a composition of RL and CPG modules such that the RL module regulates the tonic inputs to the CPG system given state feedback from the robot, and the output of the CPG module is then transformed into pressure inputs to pneumatic actuators of the soft snake robot. This design allows the RL agent to naturally learn to entrain the desired locomotion patterns determined by the CPG maneuverability. We validated the optimality and robustness of the control design in both simulation and real experiments, and performed extensive comparisons with state-of-art RL methods to demonstrate the benefit of our bio-inspired control design.

3.1 System Overview of the Soft Snake Robot

As shown in Fig. 3.1, our soft snake robot is a subtype of WPI-SRS series robot [47]. It consists of 4 pneumatically actuated soft links. The soft links are made of

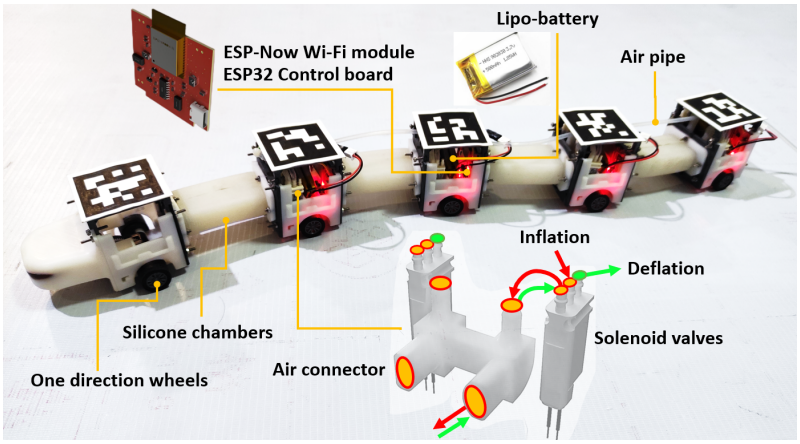


Figure 3.1: Mechatronics design of the soft snake robot.

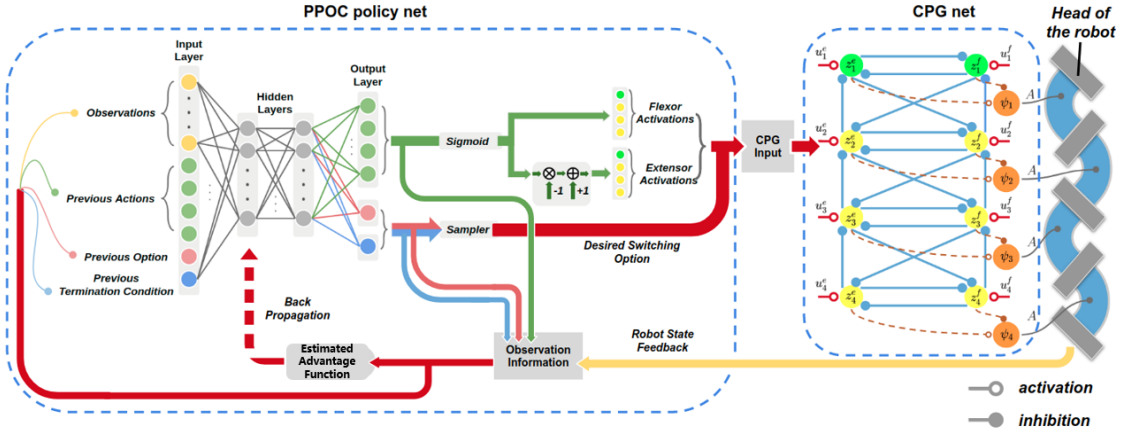


Figure 3.2: Illustrating the input-output connection of the PPOC-CPG net.

Ecoflex™ 00-30 silicone rubber. Each soft link of the robot has two air chambers mimicking antagonistic muscle (detailed structure of the soft body can be found in [46, 47]). The links are connected through rigid bodies enclosing the electronic components that are necessary to control the snake robot. Each rigid body contains an ESP32 module (powered by a Lithium-polymer battery) for control command communication and a pair of SMC-S070C-SCG solenoid valves that control the inflation and deflation of the air chambers. Only one chamber on each link is active (pressurized) at a time. In addition, the rigid body components have a pair of one-direction wheels to model the anisotropic friction of real snakes.

The configuration of the robot's coordinate is shown in Figure 3.3. At time t, state $\boldsymbol{h}(t) \in \mathbf{R}^2$ is the planar Cartesian position of the center of mass (COM) of the snake's head, $\boldsymbol{\rho_g}(t) \in \mathbf{R}^2$ is the planar displacement vector pointing from snake's head COM to the goal position, $d_g(t) \in \mathbf{R}$ is the distance traveled along the head-to-goal-direction from the initial head COM position, $\boldsymbol{v}(t) \in \mathbf{R}^2$ is the instantaneous

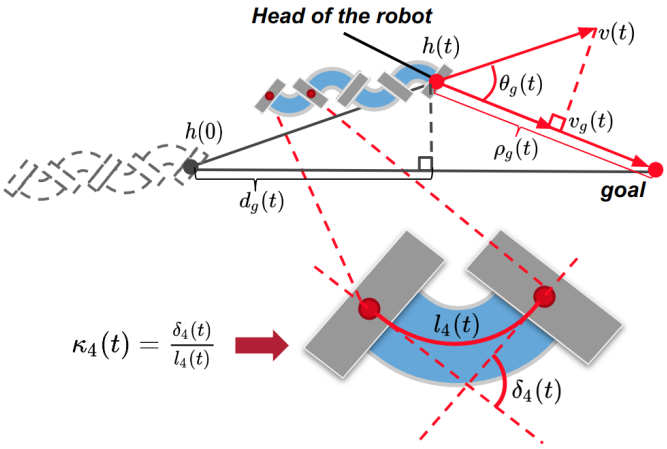


Figure 3.3: Notation of the state space configuration of the robot.

planar velocity vector of the snake's head COM, $\theta_g(t)$ is the angle between vector $\boldsymbol{\rho_g}(t)$ and vector $\boldsymbol{v}(t)$, and the locomotion speed $v_g(t) \in \mathbf{R}$ is the length of the projection of $\boldsymbol{v}(t)$ on the head-to-goal-direction. According to [45], the bending curvature of each body link at time t is computed by $\kappa_i(t) = \frac{\delta_i(t)}{l_i(t)}$, for $i = 1, \ldots, 4$, where $\delta_i(t)$ and $l_i(t)$ are the relative bending angle and the length of the middle line of the i-th soft body link.

3.2 Simulator

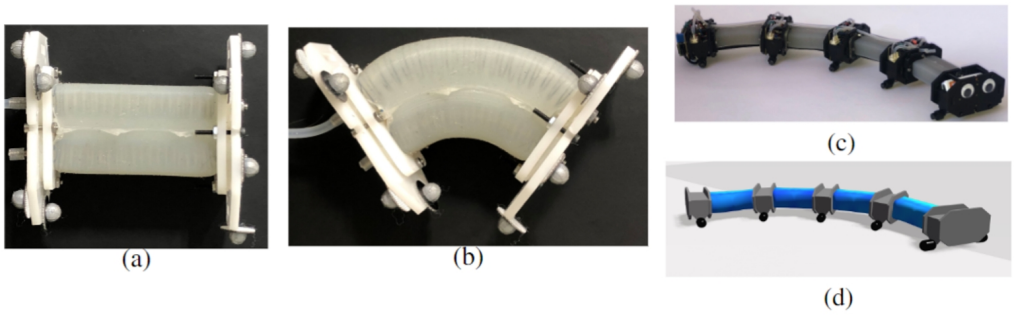


Figure 3.4: (a) Single soft link with no pressure applied. (b) 8 psi applied on left chamber. (c) Full assembly of the robotic snake with four links. (d) Snake in simulation.

To allow paralleled learning and simulated verification, we developed a physics-based simulator that models the inflation and deflation of the air chamber and the resulting deformation of the soft bodies with tetrahedral finite elements. The simulator runs in real time using GPU. We use the simulator for learning the locomotion

controller in the soft snake robot, and then apply the learned controller to the real robot.

3.2.1 Dynamic Modeling

The continuous equations of motion for the multiphysics simulator are derived from Lagrangian mechanics, and are given in general form by the following,

$$\begin{aligned}
\mathbf{M}\ddot{\mathbf{q}} - \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{J}_b^T \boldsymbol{\lambda}_b - \mathbf{J}_n^T \boldsymbol{\lambda}_n - \mathbf{J}_f^T \boldsymbol{\lambda}_f &= \mathbf{0} \\
\mathbf{c}_b(\mathbf{q}, \mathbf{p}) + \mathbf{E}\boldsymbol{\lambda}_b &= \mathbf{0} \\
\mathbf{0} \leq \mathbf{c}_n(\mathbf{q}) \perp \boldsymbol{\lambda}_n &\geq \mathbf{0} \\
\forall i \in \mathcal{A}, \quad \mathbf{D}_i^T \dot{\mathbf{q}} + \frac{|\mathbf{D}_i^T \dot{\mathbf{q}}|}{|\boldsymbol{\lambda}_{f,i}|} \boldsymbol{\lambda}_{f,i} &= \mathbf{0} \\
\forall i \in \mathcal{A}, \quad 0 \leq |\mathbf{D}_i^T \dot{\mathbf{q}}| \quad \perp \quad \mu_i \lambda_{n,i} - |\boldsymbol{\lambda}_{f,i}| &\geq 0 \\
\forall i \in \mathcal{I}, \quad \boldsymbol{\lambda}_{f,i} &= \mathbf{0}.
\end{aligned}$$

These equations describe the motion of a generic dynamics system with frictional contact forces. The state of the system is described by a vector of generalized coordinates $\mathbf{q} \in \mathbb{R}^{n_d}$ with n_d DOFs, determined by the number of particles and rigid bodies on the system. The inertial properties of the system are represented by the mass-matrix $\mathbf{M} \in \mathbb{R}^{n_d \times n_d}$, with $\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}})$ a generalized force function that includes external and gyroscopic forces. The vector $\mathbf{c}_b(\mathbf{q})$ is a set of bilateral constraints of length n_b, with $\boldsymbol{\lambda}_b$ the associated Lagrange multipliers. Elastic energy potentials are defined in terms of compliant constraints, here $\mathbf{E} \in \mathbf{R}^{n_b \times n_b}$ is a block-diagonal compliance, or inverse stiffness matrix as described by Servin et al. [76]. The target pressures are grouped into the vector $\mathbf{p}$, which are parameters to the actuation constraints described in section 3.2.4. The contact and frictional forces are based on Coulomb's model, which defines an admissible cone of contact forces [80]. Here $\mathbf{c}_n(\mathbf{q})$ are unilateral contact constraints, with n_c the number of contacts in the system, and $\lambda_{n,i}$ and μ_i the normal force Lagrange multiplier and friction coefficient for the ith contact respectively. The frictional forces for a contact are parameterized by $\boldsymbol{\lambda}_{f,i}$, with a corresponding basis $\mathbf{D}_i$ that defines the surface tangent plane at the contact point. The active contact set is defined as $\mathcal{A} = \{i \in (1, \cdots, n_c) \mid \mu_i \lambda_{n,i} > 0\}$, with inactive contacts $\mathcal{I}$ being its complement. The constraint Jacobians $\mathbf{J}_b, \mathbf{J}_n$ contain the gradient of bilateral and normal constraint functions with respect to $\mathbf{q}$, and we define the set of frictional basis vectors as the matrix $\mathbf{J}_f = [\mathbf{D}_1, \cdots, \mathbf{D}_{n_c}]^T$.

3.2.2 Particles

Each deformable link is modeled as a collection of particles connected by constraints. This is a flexible representation that allows fine-grained control over different sections of the soft body, while being efficient enough for real-time simulation. A particle

with index i adds three additional DOFs to the system,

$$\mathbf{q}_i = \begin{bmatrix} x & y & z \end{bmatrix}^T. \tag{3.1}$$

Assuming a lumped mass model, each particle is assigned a fraction of the connected tetrahedral elements mass (Section 3.2.4). The mass-block for the particle is then given by $\mathbf{M}_i = m\mathbf{1}_3$, where m is the particle mass, and $\mathbf{1}_3$ is the 3-dimensional identity matrix.

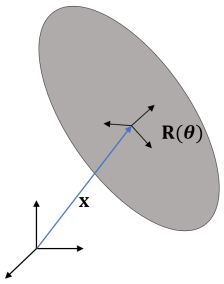


Figure 3.5: Rigid links and wheels are described by the translation of the body's center of mass from the origin $\mathbf{x}$ and, it's orientation expressed as a quaternion $\boldsymbol{\theta}$.

3.2.3 Rigid Bodies

We describe the state of a rigid body with index i using a maximal coordinate representation consisting of the position of the body's center of mass, $\mathbf{x}_i \in \mathbb{R}^3$, and its orientation expressed as a quaternion $\boldsymbol{\theta}_i = [\theta_1, \theta_2, \theta_3, \theta_4]^T \in \mathbb{R}^4$. We group these components so the state sub-block for a single rigid body is $\mathbf{q}_i = \begin{bmatrix} \mathbf{x}_i^T & \boldsymbol{\theta}_i^T \end{bmatrix}^T$.

3.2.4 Constraints

Actuation Constraints

To perform actuation we constrain particles together through equations of the form,

$$c_{dist}(\mathbf{q}, p) = |\mathbf{q}_i - \mathbf{q}_j| - r\epsilon(p) = 0, \tag{3.2}$$

where $\mathbf{q}_i$ and $\mathbf{q}_j$ are particle positions, and r is a rest length to maintain between them. The target pressure p induces a strain $\epsilon(p)$ that adjusts the rest length and causes contraction or expansion. Assuming that deformation is linear with stress, and that it occurs primarily along the chamber's main axis, the amount of expansion/contraction is given by the following relation between material stiffness determined by the Young's Modulus (Y) and pressure p,

$$\epsilon(p) = 1 + p/Y. \tag{3.3}$$

Furthermore, we use distance constraints with constant rest length to model the structural stiffness in the deformable chamber.

Tetrahedral Finite-Elements

In addition to distance constraints, tetrahedral finite-elements are used to model the solid chamber material. Assuming a constant strain element and a linear isotropic constitutive model, each tetrahedron defines a 6-dimensional constraint vector,

$$\mathbf{c}_{tetra}(\mathbf{q}) + \mathbf{E}_{tetra}\boldsymbol{\lambda} = \mathbf{0}, \tag{3.4}$$

where $\mathbf{c}_{tetra}(\mathbf{q}) = [\epsilon_{xx}\ \epsilon_{yy}\ \epsilon_{zz}\ \epsilon_{yz}\ \epsilon_{xz}\ \epsilon_{xy}]^T$ is the vector of corotational strains in Voigt notation, and $\mathbf{E}_{tetra}$ is the constant element compliance matrix, given by

$$\mathbf{E}_{tetra} = \frac{1}{V_e Y}\begin{bmatrix} 1 & -\nu & -\nu & 0 & 0 & 0 \\ -\nu & 1 & -\nu & 0 & 0 & 0 \\ -\nu & -\nu & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1+\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & 1+\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & 1+\nu \end{bmatrix}.$$

where V_e is the element volume, Y and ν are the material Young's modulus and Poisson's ratio, respectively.

Rigid Body to Particle Attachment

In order to connect soft links to rigid bodies, an attachment constraint between a particle and a point on a rigid body is defined as follows,

$$\mathbf{c}_{attach}(\mathbf{q}) = \mathbf{q}_x + \mathbf{R}(\mathbf{q}_\theta)\mathbf{r} - \mathbf{q}_p = \mathbf{0}. \tag{3.5}$$

This is a vector-valued function that adds three separate constraints, one for each x, y, z axis respectively. Here $\mathbf{q}_x, \mathbf{q}_\theta$ are the rigid body position and orientation respectively, and $\mathbf{q}_p$ is the particle position. $\mathbf{R}(\mathbf{q}_\theta)$ is the rotation matrix obtained from the body's orientation, and $\mathbf{r}$ is the attachment point expressed in the body's local frame.

Rigid Body Joints and Contact

Along with the deformable sections, we model the articulated carriage as rigid bodies, with wheels connected to the main frame using hinge joints as described by [77]. Contact between the wheels and the ground is modeled by non-interpenetration constraints of the form:

$$c_n(\mathbf{q}) = \mathbf{n}^T\left[\mathbf{a}(\mathbf{q}) - \mathbf{b}(\mathbf{q})\right] \geq 0, \tag{3.6}$$

where $\mathbf{n} \in \mathbf{R}^3$ is the contact plane normal, $\mathbf{a}$ and $\mathbf{b}$ are points on a rigid or deformable body. Frictional forces are included using a Coulomb model derived from a principle of maximum dissipation that limits the contact forces to a cone. We refer the readers to the survey paper by Stewart [79] for more detail.

3.2.5 Time-Stepping

The simulation is advanced in time with a first-order implicit time-discretization of the equations of motion similar to the method in [87]. An implicit discretization is chosen as it allows taking large time-steps and avoids constraint drift. At each time-step, the nonlinear system of equations resulting from the implicit discretization is solved using Newton's method. To solve the complementarity conditions associated with contact we use a non-smooth reformulation based on the Fischer-Burmeister function as described in [59]. Each Newton iteration requires the solution of a sparse-matrix equation of the form

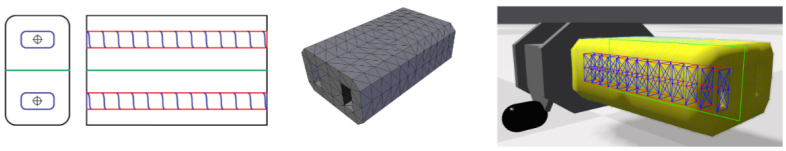
$$\left[\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T + \mathbf{E}\right] \Delta\boldsymbol{\lambda} = \mathbf{b}. \tag{3.7}$$

Where $\mathbf{J} = [\mathbf{J}_b^T \ \mathbf{J}_n^T \ \mathbf{J}_f^T]^T$ is the matrix of constraint Jacobians, $\mathbf{E}$ is a block-diagonal compliance matrix that includes the tetrahedral compliance matrices, and $\mathbf{b}$ includes the constraint function residuals evaluated at the current Newton iterate. This is a positive semi-definite system that we solve using the Preconditioned Conjugate Residual method (PCR) [70]. This is an iterative Krylov method similar to Conjugate Gradient (CG) but with smooth error reduction, making it better suited for real-time applications with a fixed computational budget. Like CG, the primary computation cost of PCR is the performing sparse matrix-vector multiplications. However, these multiplications are highly parallelizable, and can be done efficiently by assembling $\mathbf{J}, \mathbf{M}, \mathbf{E}, \mathbf{b}$ on the GPU in compressed row-storage (CSR) format, and performing the multiplication with optimized kernels [61]. In our simulator we use a simple diagonal Jacobi preconditioner since it is trivial to parallelize.

3.2.6 Construction of Soft Robotic Snake in Simulation

The soft links of the snake robot are made of Ecoflex™ 00-30 silicone rubber which has material parameters $Y = 66.243$KPa, and $\nu = 0.4999$ [15]. We construct a triangular mesh of the surface and tetrahedralize it using TetGen [78]. The link mesh was created with evenly distributed particles, we do not explicitly represent the cavity with tetrahedral elements. The mesh was carefully constructed to provide a radially symmetrical tetrahedron structure, as seen in Fig. 3.6.

Since the inextensible layer in the center of the link has a deformation threshold that is beyond the range of forces to be applied on the soft links, it is acceptable to model it as a non-deformable constraint between particles along the center plane. Similarly, the radial constraint on the chambers are defined as a set of inter-particle



(a) (b) (c)

Figure 3.6: (a) Front and top view of chamber with constraints between particles on link. (b) Soft link mesh. (c) Constraints displayed on simulation (best seen in digital format).

constraints over coplanar particles along the link. Although it would be possible to drive each link's expansion using surface pressure forces directly, the other constraints in the link allow us to simply control the chamber volume using constraints between particles along the primary axis of expansion.

Only one chamber on the link is active (*i.e.* pressurized) at a time, so this set of actuation constraints only applies to the expanding chamber. Figure 3.6 displays the constraints overlaid on the link. The link mesh was subdivided in 13 cross-sections along its length, in order to allow real-time computation, while maintaining good accuracy on the material deformation.

The links are then connected to each other through the rigid bodies that contain the electronics necessary to control the snake robot. In addition, the rigid bodies are attached to the wheels via. hinge joints. The wheels provide contact with the floor and model the anisotropic friction that a real snake has from its scales.

Type	**Quantity**
Rigid Bodies	15
Particles	1504
Distance Constraints	1460
Tetrahedral Finite Elements	4536
Rigid Joints	10
Particle Attachments	217

Table 3.1: size of the structure for one simulated snake

The type and number of all DoFs, and constraints in the simulated snake are displayed in Table 3.1.

Open-Loop Control

The snake assembly consists of four links attached together, as seen in Fig. 3.4. An undulating motion that propels the snake is given by the control equation 3.8, which outputs the pressure for the link i.

$$a_i \equiv \min(1, \max(-1, (\sin(\omega t + \alpha i) + \phi))\, A. \tag{3.8}$$

If a_i is positive, the link will inflate one chamber of the link, while if a_i is negative, it will inflate the other. The parameters ω, α, ϕ, and A are the base oscillation frequency, measured in Hz, the phase shift between links ($radians$), the offset value for the motion ($[-1, 1]$, scalar), and the oscillation amplitude ($[0, p_{max}]$, Pa), respectively. The oscillation frequency dictates how fast the actuators will switch direction, and the phase delay between links is what generates the wave pattern that propels the snake forward. These parameters set the base undulating motion. By changing the offset ϕ, applied to all links, the controller will inflate one side for longer than the other. This results in the snake propelling itself more to the opposite direction than the chambers that are more inflated, making the trajectory of the center of mass moves with a curvature radius determined by this offset and the friction with the ground. Finally, the amplitude A limits the maximum pressure during the oscillation, thereby controlling the snake's speed. The parameters that make the snake move forward depend on the physical properties of the snake, such as weight, length, friction coefficient with the floor, and were determined experimentally in [62]. This controller is an open-loop method, which generates the forward motion and allows to make turns, but no feedback is given if the trajectory is deviating from the desired trajectory.

The pressure delivery is not instantaneous but limited by the maximum airflow allowed by the valves. Assuming the pressure source can reliably maintain a constant output pressure p_s, the air flow v to the chamber is given by [95],

$$v^2 = \frac{2}{\rho}(p_t - p_s), \tag{3.9}$$

where ρ is the air density and p_t is the pressure in the chamber. This means the pressure update is proportional to the square of the difference between the current pressure and the desired pressure. The pressure update for inflation in each step is then defined based on the difference Δp_i, in Eq. 3.10.

$$\Delta p_i = \frac{a_i(t+h) - p_i(t)}{p_s}, \tag{3.10}$$

The deflation releases pressure in the atmosphere while keeping its own pressure relatively constant due to the change of volume. Therefore, the deflation ratio should be close to linear up to a threshold T_p when it is proportional to the over-pressure with a damping,

$$p_i(t+h) = \begin{cases} p_i(t) + p_s {\Delta p_i}^2 k_i & \text{is inflating} \\ p_i(t) - \min(p_i(t)k_d, T_p) & \text{is deflating} \end{cases} \tag{3.11}$$

where $k_i, k_d \in (0, 1]$ are the inflation damping parameters, and are tuned according to the experimental data.

3.3 Design of Matsuoka CPG Network for the Soft Snake Robot Locomotion

In this section, we introduce our CPG network design consisting of interconnected Matsuoka oscillators [52, 53].

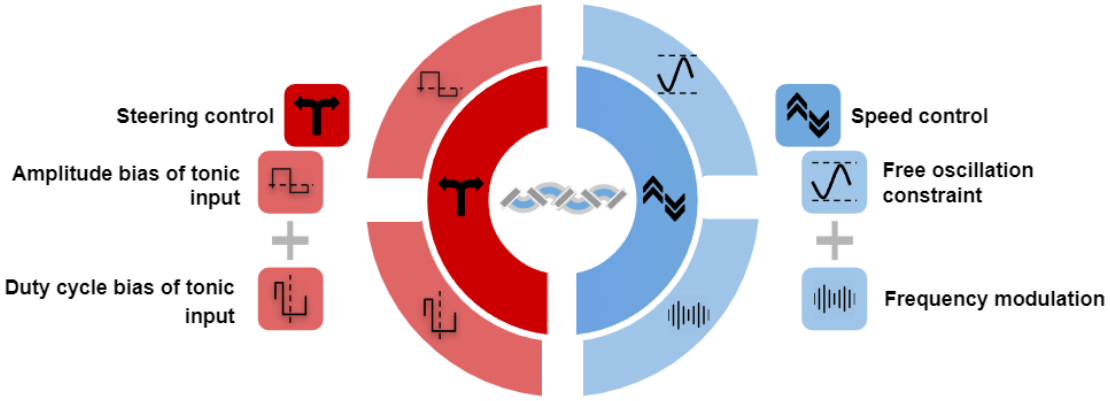


Figure 3.7: An overview of the maneuverability of Serpentine locomotion with the Matsuoka oscillator.

Primitive Matsuoka CPG: A primitive Matsuoka CPG consists of a pair of mutually inhibited neuron models. The dynamical model of the primitive Matsuoka CPG is given as follows:

$$\begin{aligned} K_f \tau_r \dot{x}_i^e &= -x_i^e - a z_i^f - b y_i^e - \sum_{j=1}^{N} w_{ji} y_j^e + u_i^e + c, \\ K_f \tau_a \dot{y}_i^e &= z_i^e - y_i^e, \\ K_f \tau_r \dot{x}_i^f &= -x_i^f - a z_i^e - b y_i^f - \sum_{j=1}^{N} w_{ji} y_j^f + u_i^f + c, \\ K_f \tau_a \dot{y}_i^f &= z_i^f - y_i^f, \end{aligned} \tag{3.12}$$

where the subscripts e and f represent variables related to the extensor neuron and flexor neuron, respectively. The tuple (x_i^q, y_i^q), $q \in \{e, f\}$ represents the activation state and self-inhibitory state of i-th neuron respectively, $z_i^q = g(x_i^q) = \max(0, x_i^q)$[1] is the output of i-th neuron, $b \in \mathbf{R}$ is a weight parameter, u_i^e, u_i^f are the forced tonic inputs to the oscillator, and $K_f \in \mathbf{R}$ is the frequency ratio. The set of parameters in the system includes the discharge rate $\tau_r \in \mathbf{R}$, the adaptation rate $\tau_a \in \mathbf{R}$, the mutual inhibition weights between flexor and extensor $a \in \mathbf{R}$, the inhibition weight $w_{ji} \in \mathbf{R}$ representing the coupling strength with the neighboring primitive

[1]The maximum function is noted as $g(\cdot) = \max(0, \cdot)$ in this thesis work.

oscillator, and the free-response oscillation tonic input $c \in \mathbf{R}$ ($c = 0$ in our previous work [42]). In our system, all coupled signals including x_i^q, y_i^q and z_i^q ($q \in \{e, f\}$) are inhibiting signals (negatively weighted), and only the tonic inputs are activating signals (positively weighted). In the current system, we have $N = 4$ primitive Matsuoka CPGs. For simplicity, we introduce a vector

$$\boldsymbol{u} = [u_1^e, u_1^f, u_2^e, u_2^f, u_3^e, u_3^f, u_4^e, u_4^f]^T \tag{3.13}$$

to represent all tonic inputs to the CPG net.

Structure of the Matsuoka CPG Network for the Soft Snake Robot: Extending from a primitive Matsuoka CPG system to the multi-linked snake robot, we construct a CPG network shown on the right of Fig. 3.2. The network includes four linearly coupled primitive Matsuoka oscillators. It is an inverted, double-sized version of Network VIII introduced in Matsuoka's paper [52]. The network includes four pairs of nodes. Each pair of nodes (e.g., the two nodes colored green/yellow) in a row represents a primitive Matsuoka CPG (3.12). The edges correspond to the coupling relations among the nodes. In this graph, all the edges with hollowed endpoints are positive activating signals, while the others with solid endpoints are negative inhibiting signals. The oscillators are numbered 1 to 4 from head to tail of the robot. In order to build the connection between the CPG network and robot actuators, we define the output of the i-th primitive Matsuoka CPG as

$$\psi_i = a_\psi z_i = a_\psi (z_i^e - z_i^f), \tag{3.14}$$

where a_ψ is a ratio coefficient of z_i. Given the Bounded Input Bounded Output (BIBO) stability of the Matsuoka CPG net [51], the outputs $\boldsymbol{\psi} = [\psi_1, \psi_2, \psi_3, \psi_4]^T$ from the primitive oscillators can be limited within $[-1, 1]$ by adjusting the ratio a_ψ. We let $\psi_i = 1$ for the full inflation of the i-th extensor actuator and zero inflation of the i-th flexor actuator, and vice versa for $\psi_i = -1$. The actual pressure input to the i-th chamber is $\lambda_i \cdot \psi_i$, where λ_i is the maximal pressure input of each actuator. The primitive oscillator with green nodes controls the oscillation of the head joint. This head oscillator also contributes as a rhythm initiator in the oscillating system, followed by the rest parts oscillating with different phase delays in sequence. Figure 3.2 shows all activating signals to the CPG network.

Configuring the Matsuoka CPG Network: To determine the hyper-parameters in the CPG network that generate a more efficient locomotion pattern, we employed a genetic programming (GP) algorithm similar to [29]. In this step, all tonic inputs are assigned with value 1 for the simplicity of fitness evaluation.

We define the fitness function–the optimization criteria–in GP as $F(v_{g,T}, \theta_{g,T}, d_{g,T}) = a_1|v_{g,T}| - a_2|\theta_{g,T}| + a_3|d_{g,T}|$, where g indicates a fixed goal initiated in the heading direction of the snake robot, T indicates the terminating time of fitness evaluation for each trial, and all coefficients $a_1, a_2, a_3 \in \mathbf{R}^+$ are constants[2].

[2]In experiments, the following parameters are used: $a_1 = 40.0$, $a_2 = 100.0$, $a_3 = 50.0$, and

To achieve stable and synchronized oscillations of the whole system, the following constraint must be satisfied [51]:

$$(\tau_a - \tau_r)^2 < 4\tau_r\tau_a b, \tag{3.15}$$

where $\tau_a, \tau_r, b > 0$. To satisfy this constraint, we can set the value of b much greater than both τ_r and τ_a, or make the absolute difference $|\tau_r - \tau_a|$ sufficiently small.

In other words, this fitness function is a weighted sum over the robot's instantaneous speed, deviation angle, and total traveled distance on a fixed straight line at the terminating time T. In this scenario, a better-fitted configuration is supposed to maintain oscillating locomotion and reaches faster locomotion speed $|v_{g,T}|$ along the original heading direction at time T. In addition, the locomotion pattern is required to have a smaller deviation between the robot's heading direction and the goal direction (with a small $|\theta_g|$), and with overall a longer traveled distance along the robot's heading direction ($|d_g|$).

The desired parameter configuration found by GP is given in Table. 6.1 in Appendix 6.1.

3.4 Maneuverability Analysis and Design of the Learning-based Controller with the Matsuoka CPG Network

When provided with equally constant tonic inputs, the designed Matsuoka CPG system can generate stable oscillation patterns to efficiently drive the soft snake robot slithering forward. However, a single CPG network cannot achieve intelligent locomotion and goal-tracking behaviors with potentially time-varying goals. For an intelligent controller, the free turning and accelerating (or decelerating) behaviors are the fundamental skills to realize autonomous locomotion in the goal-tracking tasks. In this work, we denote these two maneuverability demands as – steering control and velocity control (see Fig. 3.7). The later parts will focus on investigating the properties of the Matsuoka CPG system to prove that it is controllable from both steering and velocity control perspectives. We design a proper connection between RL actions and controllable coefficients of the Matsuoka CPG system so that both steering and velocity control of the snake robot can be efficiently learned by the RL agent.

For steering control, we prove that the bias of tonic inputs is linearly proportional to the bias of the CPG output in both amplitude and duty cycle dimensions. This property inspires a rule that transforms the action outputs of the RL policy into the tonic inputs of the CPG system.

Next, we excavate two mechanisms that are helpful for velocity control. First,

$T = 6.4$ sec.

we show that the frequency ratio coefficient K_f allows the RL agent to tune the locomotion velocity by directly adjusting the oscillation frequency. Second, by introducing the free-response oscillation constraint, we provide a way to adjust the converging amplitude of the oscillation driven by the RL agent. With experiments, we show that the free-response oscillation constraint is very helpful for reducing performance drop in the sim-to-real problem.

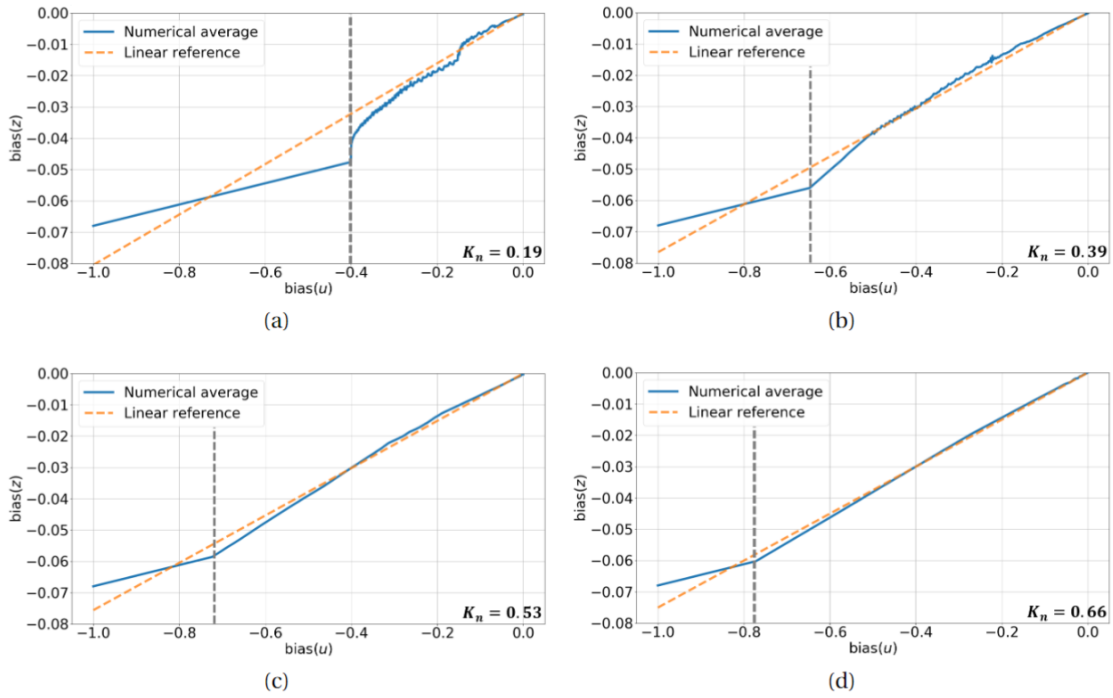


Figure 3.8: Relation between oscillation bias and extensor tonic input u^e when setting different a values to obtain (a) $K_n = 0.19$ (b) $K_n = 0.39$ (a) $K_n = 0.53$ (b) $K_n = 0.66$ (a) $K_n = 0.79$.

3.4.1 Steering control with imbalanced tonic inputs

Most existing methods based on CPG realize steering by either directly adding a displacement [27] to the output of the CPG system, or using a secondary system such as an artificial neural network to compose the weighted outputs from multiple CPG systems [58]. In this section, we present a different approach based on the maneuverability of the Matsuoka oscillator–tuning tonic inputs to realize the biased wave patterns of CPG outputs for steering the slithering locomotion of the soft snake robot[3].

[3]The fact that the biased wave output of the Matsuoka CPG system could cause the turning behavior of the snake robot comes from a previous work [97], which shows that the steering angle of a slithering snake robot on the planner ground can be linearly controlled by the bias of the oscillatory output of the Matsuoka oscillator as the command signal of joint actuators.

For the RL controller to steer our snake robot smoothly through the Matsuoka CPG system, we need to find a clever way to make the steering dynamics easy to learn for the RL algorithm. In other words, the relation between tonic inputs and the output bias of each primitive Matsuoka oscillator in the CPG network needs to be simple and clear. In the original design of the Matsuoka oscillator, the flexor and extensor tonic inputs are independent of each other. This setting not only increase the dimension of action space for the RL agent but also makes the relationship between tonic inputs and the output bias more complicated. To simplify this problem, we first introduce a new relation defined as *complementation* to reform the relation between u^e and u^f.

Definition 1. *(Complementation) For two real signals $u(t)$ and $v(t)$, and a known bounded range $\mathcal{D} : [a, b]$ where $\mathcal{D} \subseteq \mathbf{R}$, we say $u(t)$ and $v(t)$ are complementary to each other in range $\mathcal{D}$ when $u(t), v(t) \in \mathcal{D}$ for all $t \in \mathbf{R}^+$ and $u(t) + v(t) \equiv b - a$.*

Another important definition for this section is a relation between two periodic signals named *entrainment* based on the related theory in [53, 54].

Definition 2. *(Entrainment) Given a neural oscillator system with its natural frequency $\omega_n > 0$. If the neural oscillator's output is synchronized to the coupled input with frequency ω, then this system is entrained with the coupled input signal. The relation between the neural oscillator's output and the coupled input signal is called entrainment. If the two signals are perfectly entrained, they are supposed to have the same oscillation amplitude and bias in addition to the synchronized oscillation frequency.*

From our previous work [42], we have observed in experiment that the steering bias of a primitive Matsuoka oscillator is proportional to the amplitude of u^e when u^e and u^f are complementary within the range $[0, 1]$. This key observation inspires us a dimension reduction technique to the input space of the CPG net: Instead of controlling u_i^e, u_i^f for $i = 1, \ldots, n$ for a n-link snake robot, we only need to control u_i^e for $i = 1, \ldots, n$ and let $u_i^f = 1 - u_i^e$. As the tonic inputs have to be positive in Matsuoka oscillators, we define a four dimensional action vector $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]^T \in \mathbf{R}^4$ and map $\boldsymbol{\alpha}$ to tonic input vector $\mathbf{u}$ as follows,

$$u_i^e = \frac{1}{1 + e^{-\alpha_i}}, \text{ and } u_i^f = 1 - u_i^e, \text{ for } i = 1, \ldots, 4. \tag{3.16}$$

This mapping bounds the tonic input within $[0, 1]$. The reduced input dimension enables a more efficient policy search in RL.

Based on this design, we show that there are certain combinations of tonic inputs in a Matsuoka oscillator that are capable of generating imbalanced output trajectories and therefore result in the turning behavior of the robot. We present three possible cases of the forced tonic inputs that could maneuver the turning behavior of the snake robot:

1. The two tonic inputs are different constants.

2. The two tonic inputs are wave functions with imbalanced duty cycles.

3. The two tonic inputs are wave functions with imbalanced duty cycles, and both wave functions are added by different constant offsets.

It is noted that the third case is a linear combination of the first two. As a result, as long as the first two cases are proved to share the same property, the third one naturally holds. Next, we provide the frequency domain analysis of the Matsuoka oscillator to explain why the first two cases of tonic inputs enable imbalanced oscillation for the turning behavior.

Steering with biased amplitude of constant tonic inputs: To show that a pair of constant tonic inputs with different bias values can result in a biased oscillating output trajectory, we need to find out the relation between the bias of the output z and the bias of tonic inputs, when the tonic inputs are constant and complementary in $[0, 1]$. In this situation, a primitive Matsuoka oscillator needs to be a zero damping harmonic system to maintain limit cycle oscillation. When the system has zero damping, the ratio between the amplitudes of state x^q and output z^q for $q \in \{e, f\}$, referred to as K_n, is obtained from a second-order linear ordinary differential equation ((6.8) in Appendix 6.2.3) derived from (3.12):

$$K_n = \frac{\tau_r + \tau_a}{\tau_a a}, \tag{3.17}$$

where τ_r and τ_a are the discharge rate and the adaptation rate in (3.12), and parameter a is the mutual inhibition weight between flexor and extensor of a primitive Matsuoka oscillator. The derivation of (3.17) can be found in Appendix 6.2.3.

When the Matsuoka oscillator's output only consists of free-response oscillation, we can establish the following relation between the output bias(z) and the tonic input bias(u).

Proposition 1. *If a primitive Matsuoka oscillator satisfies the following three conditions: 1) the dynamical model of the primitive Matsuoka oscillator is harmonic, 2) the tonic inputs u^e and u^f are constants and complementary to each other, 3) states x^e and x^f are perfectly entrained, then the oscillation bias of outputs z and the bias of inputs u satisfies the following linear relationship,*

$$bias(z) = \frac{K_n}{(b-a)K_n + 1} bias(u), \tag{3.18}$$

where $z = z^e - z^f$, $u = u^e - u^f$, and the coefficient K_n satisfies $K_n = (\tau_r + \tau_a)/(\tau_a a)$.

Proof. See Appendix 6.3.1. □

Equation (3.18) suggests that there is a linear relationship between bias(u) and bias(z) in a primitive Matsuoka oscillator. We further validate this conclusion

through the numerical simulation of a single primitive Matsuoka oscillator. We calculate the mean oscillation bias (numerical average[4]) of the simulated state output z and compare it with the estimated bias based on (3.18) (linear reference). Figure 3.8 shows the curve of bias(z) varies with bias(u) $\in [-1, 1]$ in a primitive Matsuoka oscillator.

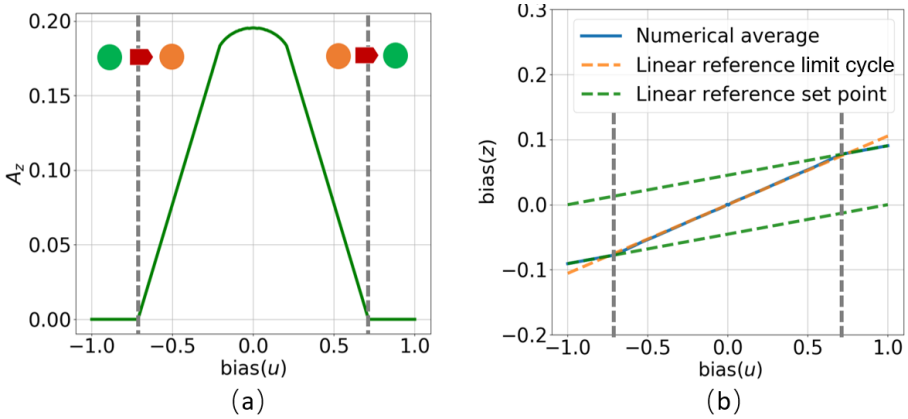


Figure 3.9: Relation between oscillation amplitude and duty cycle bias.

Figure 3.8 and theoretical analysis (see Appendix 6.3.2) show that for bias(u) $\in (\frac{2a}{a+b+1} - 1, 1 - \frac{2a}{a+b+1})$, the linear relationship mentioned in Proposition 1 is applicable to the data of bias(z) and bias(u) collected by simulating the original Matsuoka system in (3.12). It is also observed that the applicable range of bias(u) for Proposition 1 to hold increases with K_n. As shown in Fig. 3.9, when bias(u) $\in [-1, \frac{2a}{a+b+1} - 1] \cup [1 - \frac{2a}{a+b+1}, 1]$, the original Matsuoka system stops oscillating, which means the system stays at a set point equilibrium. In this case, bias(z) and bias(u) follow another linear relationship,

$$\text{bias}(z) = \begin{cases} \frac{\text{bias}(u)-(1+2c)}{2(1+b)}, & \text{if bias}(u) \in [-1, \frac{2a}{a+b+1} - 1] \\ \frac{\text{bias}(u)+(1+2c)}{2(1+b)}, & \text{if bias}(u) \in [1 - \frac{2a}{a+b+1}, 1]. \end{cases} \tag{3.19}$$

The derivation of the above relationship is provided in Appendix 6.3.2.

In the next paragraph, we show that there is also a linear relationship between bias(z) and bias(u) of the Matsuoka oscillator when u^e and u^f are periodical signals with biased duty cycles.

Steering with the biased duty cycle of periodic tonic inputs: We show a different approach to control the steering of the snake robot given that both u_i^e and u_i^f are square wave functions and are complementary to each other.

Proposition 2. *If a primitive Matsuoka oscillator satisfies the following three conditions: 1) the dynamical model of the primitive Matsuoka oscillator is harmonic,*

[4]Based on Fourier series analysis, given a continuous real-valued P-periodic function $z(t)$, the constant term of its Fourier series has the form $\frac{1}{P}\int_P z(t)dt$.

2) the tonic inputs u^e and u^f are square wave signals and are complementary to each other, 3) u^e is entrained with z^e, and u^f is entrained with z^f, then the oscillation bias of z and the bias of u satisfies the following linear relationship,

$$bias(z) = \frac{1+2m}{b-a+2} bias(u), \tag{3.20}$$

where $z = z^e - z^f$, $u = u^e - u^f$, and

$$m = \frac{1}{\pi} \frac{1}{2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n)}$$

is a constant coefficient (r indicates amplitude of state x).

Proof. See Appendix 6.3.3. □

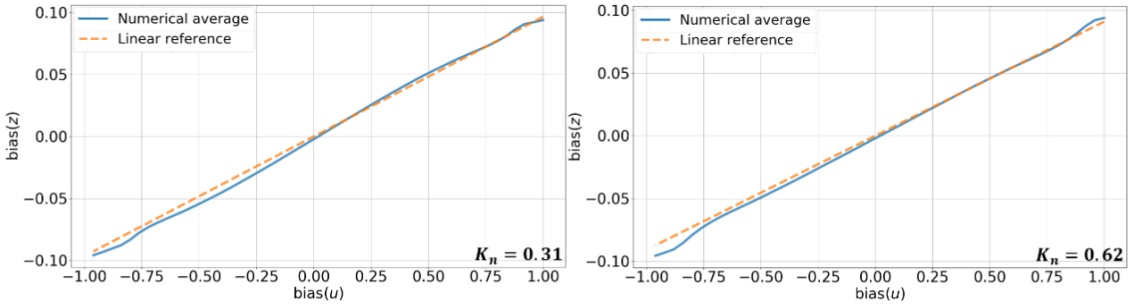


Figure 3.10: Relation between bias(z) and bias(u) for the tonic inputs satisfying Proposition 2.

The simulated results also supports Proposition 2 when the Matsuoka system is taking periodic tonic inputs with biased duty cycles. Figure 3.10 shows that with various K_n values, the linear relationship in (3.20) fits well with the curve between bias(u) and bias(z) collected by simulating the original Matsuoka system in (3.12).

Combining the conclusions in Proposition 1 and Proposition 2, we make the following remark,

Remark 1. *If a primitive Matsuoka oscillator has periodical tonic input signals u^e and u^f that are complementary to each other, with imbalanced duty cycles and both wave functions are added by different constant offsets, then bias(z) is linearly related to the bias(u), where $z = z^e - z^f$, $u = u^e - u^f$.*

Proposition 1 and 2 show that the oscillation bias of the Matsuoka CPG system is easy to maneuver through the biased tonic input signals. Since the oscillation bias is the key to steering in the snake's slithering locomotion, these two propositions provide us insight to the design of RL module so as to improve the efficiency in learning the steering behavior of the snake robot.

3.4.2 Velocity control with frequency modulation

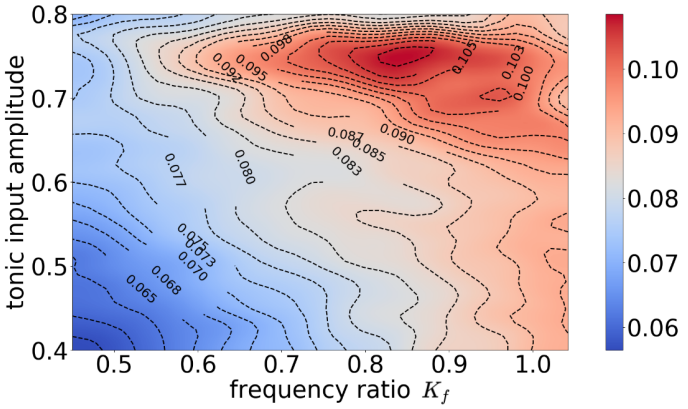


Figure 3.11: Relating oscillating frequency and amplitude to the average linear velocity of serpentine locomotion.

Generally, the linear velocity of serpentine locomotion is affected by the snake's oscillation amplitude and frequency. In this subsection, we show that the amplitude and frequency can be controlled by two coefficients of the Matsuoka CPG system to change the locomotion velocity of the soft snake robot.

First, the following relation between the frequency ratio K_f and the natural frequency $\hat{\omega}_i$ of the i-th oscillator is established in [53, (5),(6)],

$$\hat{\omega}_i \propto \frac{1}{\sqrt{K_f}}, i \in \{1, 2, 3, 4\}. \tag{3.21}$$

Second, the oscillating amplitude $\hat{A}_i$ of the i-th oscillator is linearly proportional to the amplitude of free-response oscillation tonic input c when $c > 0$ and u_i^e, u_i^f are constants [53], that is,

$$\hat{A}_i \propto c, i \in \{1, 2, 3, 4\}. \tag{3.22}$$

Equations (3.21) and (3.22) show that the frequency and amplitude of the Matsuoka CPG system are *independently* influenced by the frequency ratio K_f and the free-response oscillation tonic inputs c. Therefore, these two coefficients can be considered major factors for the Matsuoka CPG system to control the velocity of the soft snake robot's locomotion. In Fig. 3.11, we collect 2500 uniform samples within the region $c \in [0.4, 0.8]$, and $K_f \in [0.45, 1.05]$ and record the velocities generated in the simulator. We observe that with a fixed c, the average velocity increase monotonically with the frequency ratio K_f. We also observe that with the same K_f, the change of c does not affect the locomotion velocity significantly. While with different values of c, the efficiency of K_f in affecting the locomotion velocity is different. This means that we can mainly use K_f to adjust the locomotion velocity, but the value of c needs to be carefully selected. Given this analysis, we use K_f to control

the velocity of the robot. It is noted that the frequency ratio K_f only influences the strength but not the direction of the vector field of the Matsuoka CPG system. Thus, modulating K_f would not affect the stability of the whole CPG system.

3.4.3 Modulating forced-response oscillation amplitude with free-response oscillation tonic input constraint

(3.22) shows that the free-response oscillation tonic input c could affect the output amplitude of the Matsuoka oscillator when u^e and u^f are constants. We further discover that a positive value of the free response tonic input c could set a threshold for the amplitude of the force-response tonic inputs u^e and u^f, such that they need to pass this amplitude threshold in order to control the oscillation of the CPG system. In the experiment section, we show that this property of c can significantly improve the sim-to-real performance of our control framework.

In our previous work [42], when $c = 0$, there is no free-response oscillation in the system. When a Matsuoka oscillator has no free-response oscillation pattern, its output oscillation amplitude and bias are only controlled by the forced input signal given by the control tonic inputs u^e and u^f. When the inertia in the simulated learning environment is high and the contact friction force is low, the RL agent learns to generate the forced-response oscillation tonic inputs with very small amplitude to keep a more stable heading direction during the locomotion. However, if we need the RL control policy to be able to initiate the CPG oscillation with an increased amplitude on the real robot (e.g. for traversing a terrain with higher friction resistance), the learned policy would not meet the requirement.

When $c \neq 0$, we conclude that in the Matsuoka oscillator, the amplitude A_u of the force-response tonic inputs u^e and u^f must satisfy the inequality $A_u > A_0$ to completely entrain with the output z (A_0 is the entrainment threshold for u^e and u^f to synchronize the output z of the Matsuoka oscillator [54]). The equation of A_0 is given as follows

$$A_0(c, \omega) = \frac{c}{\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega_n^2-\omega^2|}\frac{c+1}{A_n}}, \tag{3.23}$$

where $A_n > 0$ is the free-response oscillation amplitude and

$$\omega_n = \frac{1}{\tau_a K_f}\sqrt{\frac{(\tau_r+\tau_a)b}{\tau_r a} - 1}$$

is the free-response oscillation frequency [53]. The detailed derivation of A_0 is pro-

vided in Appendix 6.2.4. According to [53, (30)], we have

$$A_0(c,\omega) \approx \frac{c}{\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega_n^2-\omega^2|}(2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n))}. \tag{3.24}$$

In (3.24), if $c = 0$, $A_0 \equiv 0$. In this case, there is no limiting threshold for the control policy to entrain the CPG output z with u^e, u^f. When ω is fixed and $c > 0$, then the threshold $A_0 > 0$ and A_0 increases with c. Notice that $A_u > A_0$ must be satisfied for the free-response oscillation of the Matsuoka system be attenuated by the system damping. This also means the force-response tonic inputs u^e, u^f entrain the CPG output z. In this scenario, the control policy needs to increase A_u to control the CPG system effectively. It is also noted that $A_0 \to 0$ as $\omega \to \omega_n$, therefore there are two ways for the RL agent to realize the entrainment status: one is keeping the oscillation frequency ω close to the free-response oscillation frequency ω_n, and the other is increasing the value of A_u to make $A_u > A_0$. Therefore, the combination of the two directions can encourage the intelligent controller to produce force-response tonic inputs that can not only approach desired oscillating amplitude, but also pursue frequency resonance with the original CPG system. Based on this special property of the Matsuoka oscillator, we propose a new method – FOC-PPOC-CPG to enforce better entrainment between RL control signals and the CPG states.

According to the relation between A_0 and c, we can use c to keep the oscillation amplitude of the Matsuoka oscillator at different levels. One previous work [91] has shown that the oscillation amplitude of the Matsuoka oscillator can be used to improve the slithering locomotion velocity of a rigid snake robot in different environments with different friction coefficients. Hence, we can use c to adapt the body undulation amplitude of the soft snake robot to different environments with various contact properties. With this approach, we can improve the sim-to-real performance of an RL snake controller by tuning its signal amplitude, instead of relying on the environment-based methods such as domain randomization [86] or other data augmentation techniques, which are computationally expensive.

In the later part of this chapter, our experiment results (see Section 3.6.5) verify the merit of c in improving the sim-to-real performance of our snake locomotion controller.

3.4.4 The Neural Network Controller

We have now determined the encoded input vector of the CPG net to be vector $\boldsymbol{\alpha}$ (tonic inputs) and frequency ratio K_f. This input vector of the CPG is the output vector of the NN controller. The input to the NN controller is the state feedback of the robot, given by $s = [||\boldsymbol{\rho}_g||, v_g, \theta_g, \dot{\theta}_g, \kappa_1, \kappa_2, \kappa_3, \kappa_4]^T \in \mathbf{R}^8$ (see Fig. 3.3). Next, we present the design of the NN controller.

The key insight for the design of the NN controller is that the robot needs not to change velocity very often for smooth locomotion. This means the updates for

tonic inputs and the frequency ratio can be set to be at two different time scales. With this insight, we adopt a hierarchical reinforcement learning method called the option framework [65, 83] to learn the optimal controller. The controller uses the tonic inputs as low-level primitive actions and frequency ratio as high-level options of the CPG net. The low-level primitive actions are computed at every time step. The high-level option changes infrequently. Specifically, each option is defined by $\langle \mathcal{I}, \pi_y : S \to \{y\} \times \mathbf{R}^4), \beta_y \rangle$ where $\mathcal{I} = S$ is a set of initial states, and π_y is the intra-option policy. By letting $\mathcal{I} = S$, we allow the frequency ratio to be changed at any state in the system. Variable $y \in [0, 1]$ is a value of frequency ratio, and $\beta_y : S \to [0, 1]$ is the termination function such that $\beta_y(s)$ is the probability of changing from the current frequency ratio to another frequency ratio.

The options share the same NN for their intro-option policies and the same NN for termination functions. However, these NNs for intro-option policies take different frequency ratios. The set of parameters to be learned by policy search include parameters for intra-option policy function approximation, parameters for termination function approximation, and parameters for high-level policy function approximation (for determining the next option/frequency ratio). Proximal Policy Optimization Option-Critics (PPOC) in the OpenAI Baselines [14] is employed as the policy search in the RL module.

Let us now review the control architecture in Figure 3.2. We have a Multi-layer perceptron (MLP) neural network with two hidden layers to approximate the optimal control policy that controls the inputs of the CPG net in (3.12). The output layer of MLP is composed of action $\boldsymbol{\alpha}$ (green nodes), option in terms of frequency ratio (pink node), and the terminating probability (blue node) for that option. The input of NN consists of a state vector (yellow nodes) and its output from the last time step. The purpose of this design is to let the actor network learn the unknown dynamics of the system by tracking the past actions in one or multiple steps [25,58,64]. Given the Bounded Input Bounded Output (BIBO) stability of the Matsuoka CPG net [51] and that of the soft snake robots, we ensure that the closed-loop robot system with the FOC-PPOC-CPG controller is BIBO stable. Combining with (3.16) which enforces a limited range for all tonic inputs, this control scheme is guaranteed to generate bounded inputs, which lead to bounded outputs in the system.

3.5 Curriculum and Reward Design for Efficient Learning-based Control

In this section, we introduce the design of the curriculum and reward function for efficiently learning a goal-tracking controller given the proposed FOC-PPOC-CPG scheme.

3.5.1 Task curriculum

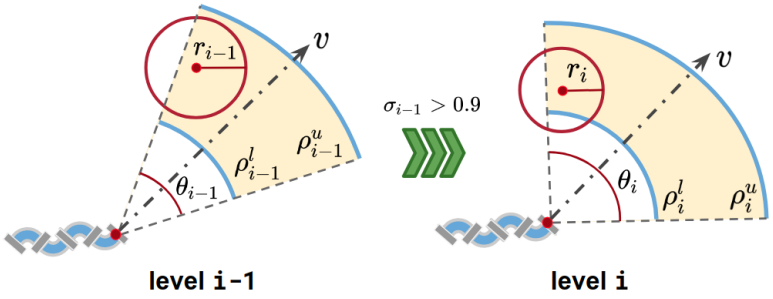


Figure 3.12: Task difficulty upgrade from level $i-1$ to level i. As the curriculum level increases, goals are sampled at a narrower distance and wider angle, and the acceptance area gets smaller.

Curriculum teaching [36] is used to accelerate motor skills learning given complex goal-tracking tasks. We design the curriculum such that the agent starts with easy-to-reach goals at level 0. As the level increases, the agent learns to perform more challenging goal-tracking tasks.

The curriculum levels are designed as follows: At the task-level, i, the center of the goal is sampled from the 2D space based on the current location and head direction of the robot. For each sampled goal, we say the robot reaches the goal if it is r_i distance away from the goal. The sampling distribution is uniform in the fan area determined by the range of angle θ_i and distance bound $[\rho^l_i, \rho^u_i]$ in the polar coordinate given by the predefined curriculum.

As shown in Fig. 3.12, when the task-level increases, we have $r_i < r_{i-1}$, $\theta_i > \theta_{i-1}$, $\rho^u_i > \rho^u_{i-1}$, and $\rho^u_i - \rho^l_i < \rho^u_{i-1} - \rho^l_{i-1}$. This means that the robot has to be closer to the goal in order to succeed and receive a terminal reward, the goal is sampled in a range further from the initial position of the robot. We select discrete sets of $\{r_i\}, \{\theta_i\}$, $[\rho^l_i, \rho^u_i]$ and determine a curriculum table. A detailed example of the learning curriculum is given in Table 6.2. We train the robot in simulation starting from level 0. The task-level is increased to level $i+1$ from level i if the controller reaches the desired success rate σ_i, for example, $\sigma_i = 0.9$ indicates at least 90 successful completions of goal-reaching tasks out of $n = 100$ trials at level i.

3.5.2 Reward design

The design of the reward function is to guide the robot to the set point goals. We consider building the artificial potential field [10] such that the robot is attracted by the goal g. We use a simple conical potential field for each goal. For any position represented by coordinate $\mathbf{x}$ in Cartesian space, let vector $\mathbf{e}_g = \mathbf{x}_g - \mathbf{x}$, the norm $||\mathbf{e}_g||$ indicates the distance between the position of the robot's head and the goal.

The constant attracting force at $\mathbf{x}$ becomes

$$\mathbf{f}_g(\mathbf{x}) = \frac{\mathbf{e}_g}{||\mathbf{e}_g||}.$$

Given the single goal-tracking scenario without obstacles, we have the potential field reward for goal-tracking as

$$U(\mathbf{x}) = \frac{\mathbf{v}_s \cdot \mathbf{f}_g(\mathbf{x})}{||\mathbf{e}_g||},$$

where $\mathbf{v}_s$ is the velocity vector of the soft snake robot.

Combining with the definition of goal-reaching tasks and their corresponding level setups, the reward at every time step is defined as

$$R(v_g, \theta_g) = c_v v_g + c_g U + c_g \cos\theta_g \sum_{k=0}^{i} \frac{1}{r_k} I(||\boldsymbol{\rho_g}|| < r_k), \tag{3.25}$$

where $c_v, c_g \in \mathbf{R}^+$ are constant weights, v_g is the length of the projection of the snake's head COM velocity $\boldsymbol{v}$ on the head-to-goal-direction, $\boldsymbol{\rho}_g$ is the linear displacement vector between the head COM of the robot and the goal position, θ_g is the angle between vector $\boldsymbol{v}$ to vector $\boldsymbol{\rho}_g$ in Fig. 3.3, r_k defines the goal range in task-level k, for $k = 0, \ldots, i$, and $I(\boldsymbol{\rho_g} < r_k)$ is an indicator function that outputs one if the robot head is within the goal range for task-level k.

This reward trades off two objectives. The first term, weighted by c_v, encourages high locomotion velocity toward the goal. The second term, weighted by c_g, rewards the learner based on the position of the robot to the goal, and the level of the curriculum the learner has achieved for the goal-reaching task. For every task, if the robot hasn't entered the goal range, it receives a potential field reward only. When the robot enters the goal range in task-level i, it receives a summation of rewards $1/r_k$ for all $k \leq i$ (the closer to the goal the higher this summation), shaped by the approaching angle θ_g (the closer the angle to zero, the higher the reward).

If the agent reaches the goal defined by the current task-level, a new goal is randomly sampled in the current or next level (if the current level is completed). There are two failing situations, where the desired goal is re-sampled and updated. The first situation is starving, which happens when the robot stops moving for a certain amount of time, referred to as the starvation time. The second case is missing the goal, which happens when the robot keeps heading in the wrong direction as opposed to moving towards the goal ($v_g(t)$ being negative) for a certain amount of time.

3.6 Experimental Evaluation

In this section, we evaluate the proposed method in both simulation and real environments. We first introduce the experimental setup to explain how the data of the

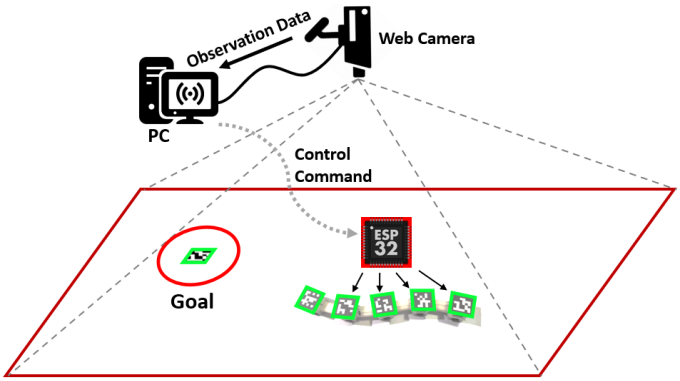


Figure 3.13: The currently used motion capture system for goal-tracking tasks.

robot is collected during locomotion, as well as the training configuration for the RL algorithm. Then we compare the properties of control signals between our method and the vanilla PPO as locomotion controllers for goal-reaching tasks in simulation. In the comparison analysis, we highlight the performance drop of each method from simulation to real to extrapolate the advantage of our method. Last, we further test and analyze the sim-to-real robustness in difficult goal-reaching tasks that are never seen at the training stage, and the real robot performance against disturbance.

3.6.1 Experimental Setup

Environment Sensing and Data Collection: The states of the real snake robot are captured by a single web camera hanging on the ceiling of the experiment room. The robot body detection is realized by using Aruco – a library in OpenCV for QR codes detection and localization [20]. These QR codes are printed and attached to the rigid bodies of the snake robot and the goal position. Figure 3.13 shows the experiment setup for the real snake robot goal-reaching tasks. Once the QR codes on the robot bodies and the goal(s) are detected, their pixel-wise coordinate vectors are calculated with distortion corrected. Given the camera calibration information, we can translate the pixel data of all QR codes into the real world 2D coordinates, and then transform it into positional information and the body posture of the robot. The control policy function running on the desktop computer receives the observation states, generates the control commands and passes them through WiFi communication. The ESP32 chips on the snake bodies translate the commands into Pulse Width Modulation (PWM) signals to activate or deactivate the valves [21, 46] on the snake robot.

Reinforcement Learning Configuration: We use a four-layered NN with 128×128 hidden layer neurons as a general configuration for the actor and critic networks of all RL methods mentioned in this section. The back-propagation of the critic net was done with Adam Optimizer and a step size of 5×10^{-4}. For data collection of each trial trajectory, the starvation time for the failing condition is 60 ms. The

missing goal criterion is triggered whenever $v_g(t)$ (the velocity on the goal-direction, see Fig. 3.3) stays negative for over 60 time steps. In order to compensate for the mismatch between the simulation and the real environment, most notably the friction coefficients, we employ a domain randomization technique [86], in which a subset of physical parameters are sampled from several uniform distributions. The range of distributions of domain randomization (DR) parameters used for training are in Table 6.3 (see Appendix 6.1).

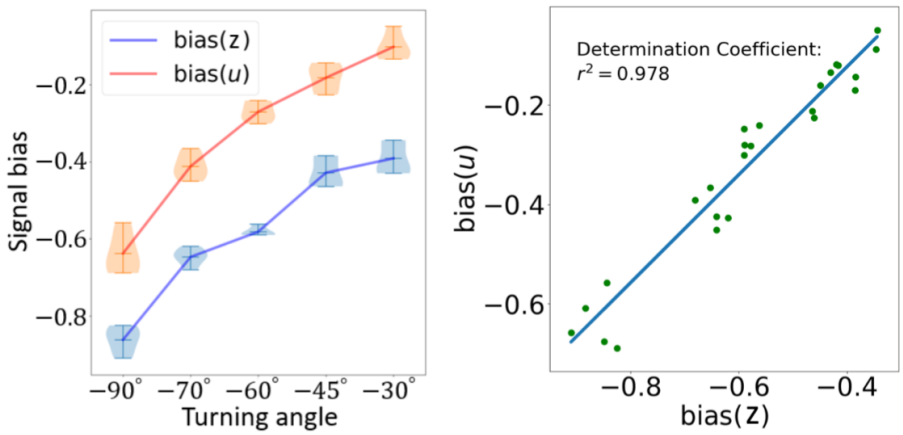
For PPOC-CPG and FOC-PPOC-CPG, we first train the policy net with fixed options (at this moment, the termination probability is always 0, and a fixed frequency ratio $K_f = 1.0$ is used). When both the task-level and the reward cannot increase anymore, we allow the learning algorithm to change the option, *i.e.*, pick a different frequency ratio K_f along with termination function β, and keep training the policy until the highest level in the curriculum is passed.

In the PPOC-CPG method, the value of the free-response tonic input c is equivalently considered zero since it is not formally introduced in the previous control design [42]. According to the definition of A_0 ((3.23)), the amplitudes of both u^e and u^f need to be greater than A_0 in order to dominate in controlling the outputs of the Matsuoka CPG system. The value of A_0 should not be greater than the upper bound of u^e and u^f, which is 1 defined by (3.16). Among a group of candidates ranging from 0.25 to 2, we choose $c = 0.75$ as our free-response oscillation constraint for the FOC-PPOC-CPG controller. This value is valid for our system because when we set $c = 0.75$ and all other coefficients of the CPG network (Table 6.1) to (3.23), the result shows $A_0 \in [0.24, 0.34] \subset [0, 1]$, with $\omega \in [3.77, 5.02]$. It is noted that the range of ω here is calculated from multiple sampled sequences of u^e and u^f recorded in the real snake goal-reaching tasks. Since we are testing the sim-to-real performance, all methods involved in this comparison are trained in the simulator for sufficiently long iterations (12500 episodes) to ensure convergence. Each method is trained 10 times with different random seeds and the controller with the best performance is selected to be tested on the real robot. All curriculum parameters (Table 6.2) and domain randomization parameters (Table 6.3) are fixed for all three methods involved.

The whole training process of each method runs on 4 simulated soft snake robots in parallel on a workstation equipped with an Intel Core i7-9700K, 32GB of RAM, and one NVIDIA RTX2080 Super GPU.

3.6.2 Verification of steering property of PPOC-CPG

We use a simulated experiment to show that our FOC-PPOC-CPG control policy has learned the turning behavior with the biased tonic input signal, and the Matsuoka CPG system can linearly map the biased tonic input to the biased actuation signal as Proposition 1 and Proposition 2 predicted. In the experiment, we test the converged FOC-PPOC-CPG policy on multiple set-point goals placed in certain directions ($-90°, -70°, -60°, -45°, -30°$) with a fixed distance (1 meter), which approximately



(a) (b)

Figure 3.14: (a) Bias input and output of the RL-driven CPG node for different turning angles (mean values connected). (b) Linear relation between input and output bias of the RL-driven CPG node during locomotion.

represent the desired turning angles of the locomotion tasks. For each goal position, we carry out 5 trials and record the tonic inputs data and CPG output data of the head CPG node of the soft snake robot. The reason for choosing the head node is because this node's behavior best reflects the desired steering direction of the RL agent. Figure 3.14a shows a violin plot of the tonic input bias and the CPG output bias for different turning angles (the bias signals are calculated by (6.38)). It is observed from Fig. 3.14a that both bias signals are monotonically related to the desired turning angle (initial goal-direction). Figure 3.14b shows the linear regression result based on all data points. We can observe a clear linear relationship between bias(z) and bias(u) of the head CPG node (with the coefficient of determination equal to 0.978, a value closer to 1 indicate higher linearity). This result provides stronger support for Proposition 1 and Proposition 2.

3.6.3 Control signal comparison between PPOC-CPG and vanilla PPO

First, we compare PPOC-CPG and vanilla PPO in terms of the smoothness of the control input learned in simulation. We train both PPOC-CPG and vanilla PPO in the same environment until convergence. Figure 3.15 shows segments of the control signal ψ_1 generated by the vanilla PPO controller and PPOC-CPG controller controlling the simulated soft snake robot in a straight line goal-tracking task. From Fig. 3.15a it is observed that the signal generated by the vanilla PPO policy oscillates at a relatively higher frequency (about 10Hz on average) with irregular oscillation patterns. Such kind of control signals are not feasible for the actuators in reality. This is because the inflation and deflation of soft air chambers on the snake robot

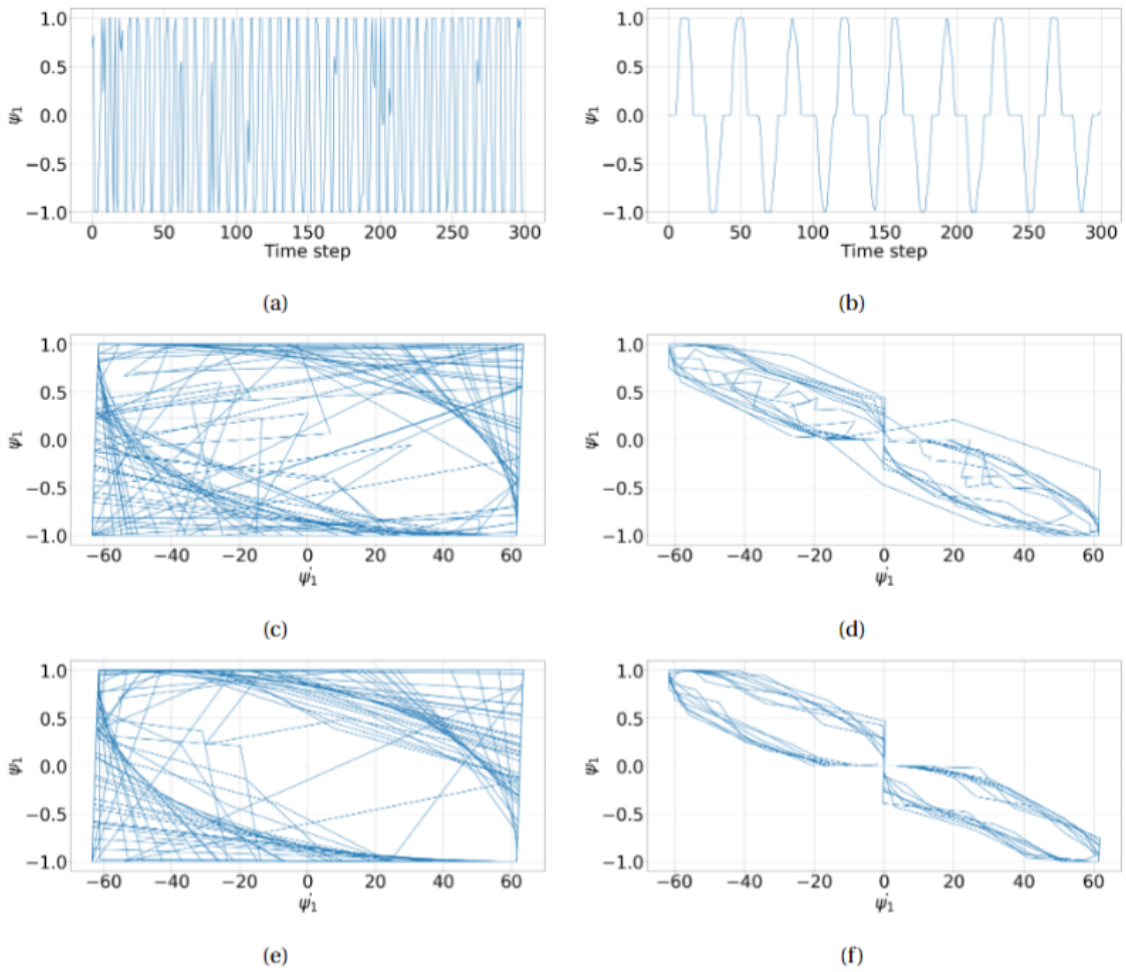


Figure 3.15: Sample actuation signal ψ_1 for the first link generated by (a) vanilla PPO and (b) PPOC-CPG from time step 0 to time step 300. Followed by phase plane portraits of ψ_1 (c) by vanilla PPO from time step 0 to 300, (d) by PPOC-CPG from time step 0 to 300, (e) by vanilla PPO from time step 400 to 700, (f) by PPOC-CPG from time step 400 to 700.

have a certain delay so that the soft pneumatic actuators are not able to track fast oscillating signals. On the other end, the curve in Fig. 3.15b shows that the agent trained with PPOC-CPG can converge to a stable limit cycle trajectory at a relatively lower but more natural frequency (1.6Hz) for serpentine locomotion. Our approach shows its advantage of being able to generate smoother oscillatory control signals even when the inputs to the CPG system are discontinuous. Fig. 3.15 also compares the phase plane portraits recorded at different time stages of the two learning methods. From Fig. 3.15c and Fig. 3.15e, we observe that the oscillating signal generated by vanilla PPO policy performs irregular oscillation in the first 300 time steps, and cannot converge to a stable limit cycle when it evolves to time step

700. While in Fig. 3.15d and Fig. 3.15f, despite a little deviation from the first 300 time steps, the outputs of the CPG network eventually converge to a stable limit cycle within 700 time steps. This experiment shows that the CPG system is capable of stabilizing the oscillation pattern in simple locomotion tasks for the soft snake robot.

3.6.4 Comparison of our reward design and a sparse reward function

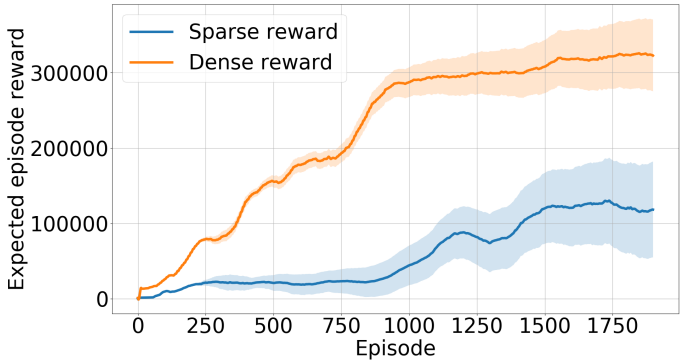


Figure 3.16: Learning process of FOC-PPOC-CPG with dense reward and sparse reward.

We compare the learning process of the revised reward function with our previous one that only rewards the agent for goal reaching events [42] (for each case we record 5 learning trials). In average, the agent with dense reward is able to reach and converge to level-12, while the agent with sparse reward only converges to level-8 (see Table 6.2). The calculated results in Fig. 3.16 show that the system trained with dense reward function outperforms that with a sparse reward design.

In the next section (Section 3.6.5), these methods are compared in the real robot to demonstrate the advantage of the proposed PPOC-CPG control.

Table 3.2: Performance Comparison of Different Approaches.

Metrics	**Vanilla PPO**	**PPOC-CPG**	**FOC-PPOC-CPG**
Simulated average speed (m/s)	**0.14**	0.137	0.135
Simulated success rate	0.95	**0.99**	0.98
Real average speed (m/s)	0.027 (↓ 80.7%)	0.063 (↓ 54%)	**0.121** (↓ 11%)
Real success rate	0.5 (↓ 42.1%)	0.82 (↓ 17.1%)	**0.9**(↓ 8.1%)

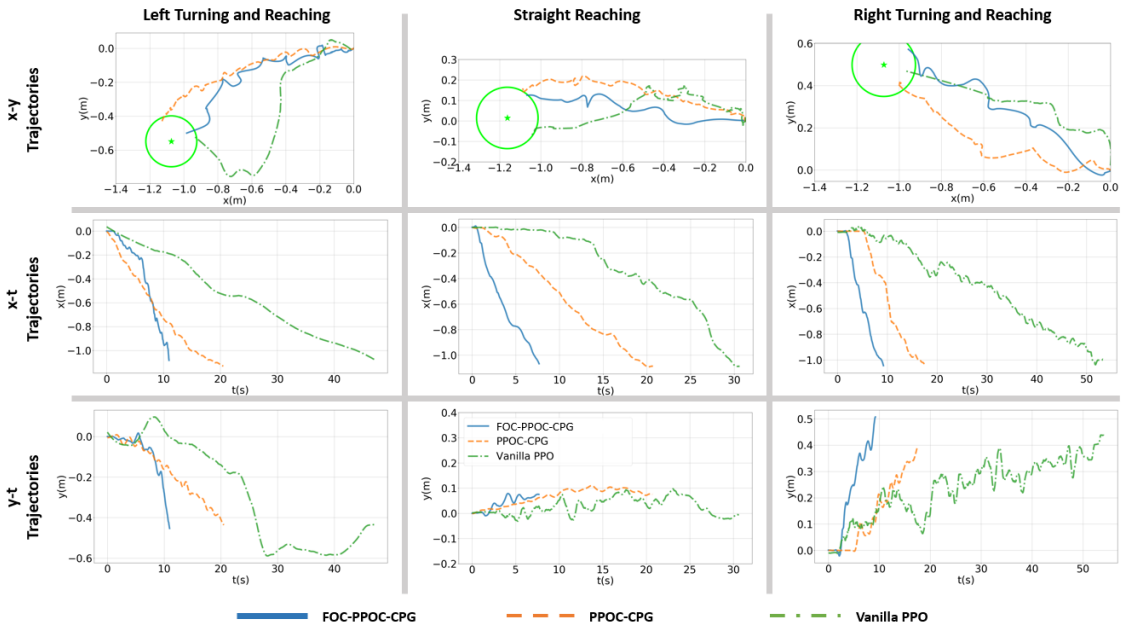


Figure 3.17: Sample comparison of trajectories generated by Vanilla PPO policy, PPOC-CPG policy, and FOC-PPOC-CPG policy in reality.

3.6.5 Sim-to-real Performance of FOC-PPOC-CPG

Performance comparison with original PPOC-CPG and Vanilla PPO

Since FOC-PPOC-CPG is designed for improving the sim-to-real transfer learning performance of the PPOC-CPG method, we first compare the sim-to-real performance of the FOC-PPOC-CPG with the original PPOC-CPG and Vanilla PPO in single goal-reaching tasks. For the real robot tests, all three controllers trained by the simulator are directly applied without further training. We test the controllers by setting goals in three directions (mid, left and right) with fixed angles, distances, and an accuracy radius of $r = 0.175$ meters. Each direction takes 10 trials for all three methods in both simulation and reality.

To evaluate the sim-to-real performance, we calculate the average locomotion speed (v_g) and the success rate for goal-reaching tasks collected from both simulation and real experiments. According to Section 3.4.3, the contact resistance forces in the simulator are smaller than in the real environment, when applying the RL control policy learned in the simulator directly to the real robot, the performance of the real robot is often worse than the simulated agent. In the rows of real robot evaluations in Table 3.2, we use down-arrows and percentage values to show the extent of performance drop compared to the simulating performance with the same method. From Table 3.2, it is observed that although the Vanilla PPO controller learns the best locomotion speed in the simulator at the cost of goal-reaching accuracy, its locomotion pattern cannot fit the real robot well. The real robot experiences

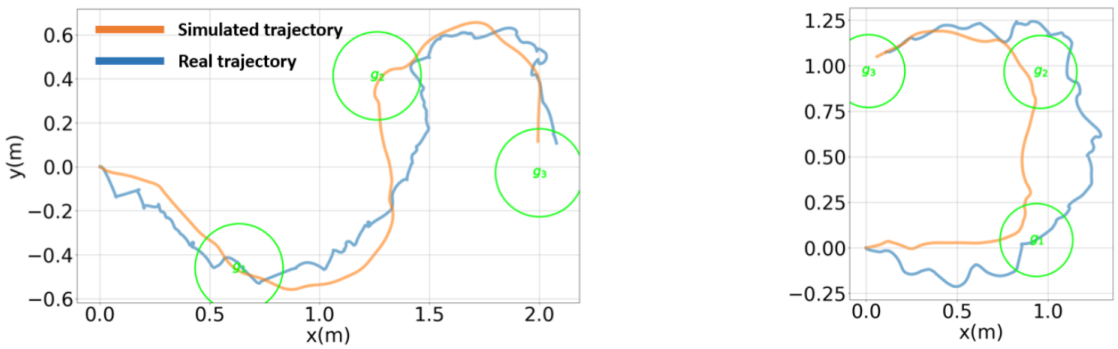


Figure 3.18: Sample way-point trajectories followed by improved PPOC-CPG controller in simulation and real in (a) zigzag and (b) square.

a drastic drop in performance on both locomotion speed (80.7%) and success rate (42.1%). For the original PPOC-CPG, though it has achieved an overall better performance than Vanilla PPO, its sim-to-real performance drop is still relatively high, with a 54% of speed drop and 17% of accuracy drop. After adding the free-response oscillation constraint to the CPG system, the new policy reaches almost the same performance as the original PPOC-CPG in the simulator. In Section 3.4.3 we have shown that the free-response oscillation tonic input $c > 0$ could help maintain the oscillation amplitude of the control signal of FOC-PPOC-CPG during the learning process. It is noticed that the maintained amplitude of the control signals does not improve the locomotion speed and goal-reaching accuracy at the training stage in the simulation. However, when the learned policy of FOC-PPOC-CPG is applied to the real robot without further training, it performs significantly better than the previous two methods in both locomotion speed and success rate.

Figure 3.17 shows a more intuitive result by comparing sample trajectories of the above three methods in different goal-reaching tasks performed on the real robot. The trajectories show that the robot controlled by Vanilla PPO policy moves much slower than the other two. And it moves in a less symmetric way for the left and right turning tasks. While the original PPOC-CPG and FOC-PPOC-CPG show similar symmetry properties in the trajectory shapes, the difference is that the controller trained with FOC-PPOC-CPG moves almost twice as fast as that trained with PPOC-CPG. This comparison is presented in the video[5] "PPO Learning methods comparison.mp4".

Since PPO is an on-policy RL algorithm and has been established for many years, we also use a more up-to-date off-policy RL algorithm – Twin Delayed Deep Deterministic policy gradient (TD3) [9] to replace the role of PPO in our framework, and train it with a shorter learning period (2000 episodes) to verify the generality of our approach. The results and a brief discussion can be viewed in Appendix 6.4.3.

[5]All videos in this chapter can be viewed from `http://shorturl.at/cgms1`

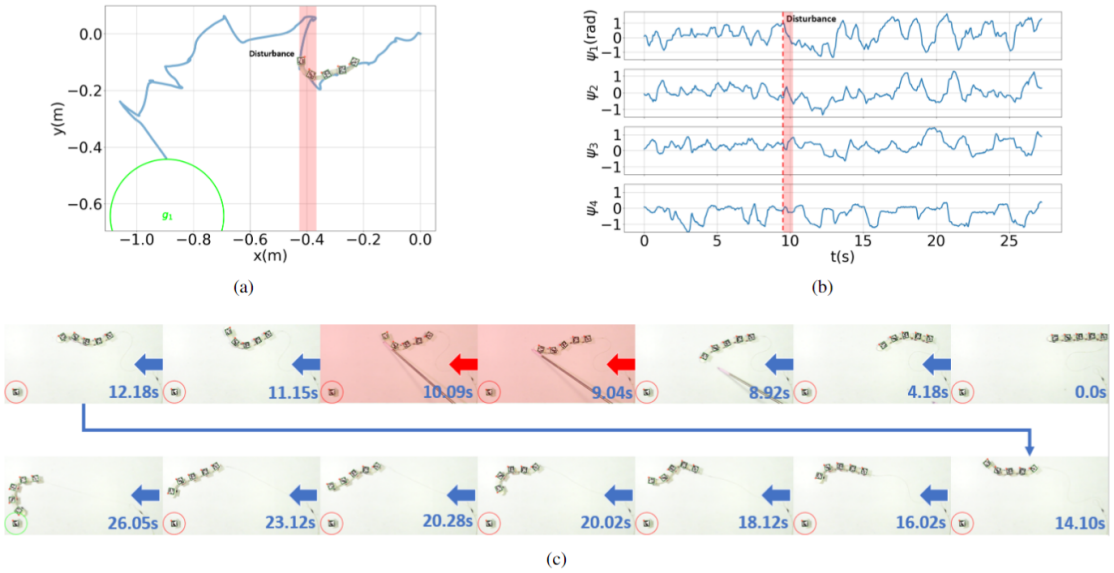


Figure 3.19: Disturbance recovery for goal-reaching task followed by FOC-PPOC-CPG controller in real experiments. The presented sub-figures are: (a) x-y plane trajectory, (b) control signals for the actuators, and (c) video snapshot of recorded robot motion.

Performance in reaching unseen goals

We also investigate the sim-to-real performance of FOC-PPOC-CPG in harder goal-reaching tasks. Figure 3.18 compares the head trajectories in Cartesian space for two different setups of way-point goals. The testing trajectories include a square turning trajectory for testing consecutive sharp turning in the same direction (Fig. 3.18b), and a zigzag trajectory for testing continuous sharp turning in opposite directions (Fig. 3.18a). Both way-point goal series have sharper turning angles than the highest level in the training curriculum in Table 6.2. Video "Half square trajectory sim2real.mp4" and "Zigzag trajectory sim2real.mp4" provide the dynamic view of Fig. 3.18a and Fig. 3.18b respectively. From the example videos, it is observed that in both trajectories, the speed drop of the real robot is still around 10%, which is not worse than single goal-reaching tasks in Table 3.2. It is noted that in both Fig. 3.18a and Fig. 3.18b, it takes the real robot longer distances to make the sharp turning. This is also due to the larger ground resistance forces in reality.

Robustness to External disturbance

We also test the FOC-PPOC-CPG controller's ability to keep tracking the desired target when the robot is disturbed by an external pushing force. Figure 3.19a and video "Disturbance recovery.mp4" shows an example trajectory of a disturbed goal-reaching task. It is observed from Fig. 3.19b that the FOC-PPOC-CPG controller reacts accordingly to its situation during the locomotion. When the deviation be-

tween the robot's head and the goal is relatively smaller before the disturbance (before $4.1s$), the robot gently oscillates and adjusts its turning direction gradually towards the goal-direction. At around $9.04s$, when the robot is pushed away from its desired direction, one can observe a clear redirection to the left-hand side of the robot's heading direction. The FOC-PPOC-CPG is able to adjust and make sharp turning to return to the correct direction and still reach the goal without wasting too much time on the recovery.

3.7 Conclusion

this chapter develops a bio-inspired controller for learning agile serpentine locomotion with a CPG net mimicking the central nervous system of natural snakes. The contribution of this chapter is two-fold: First, we investigate the properties of the Matsuoka oscillator for achieving diverse locomotion skills in a soft snake robot. Second, we construct a FOC-PPOC-CPG net that uses a CPG net to actuate the soft snake robot, and a neural network to efficiently learn a closed-loop near-optimal control policy that utilizes different oscillation patterns in the CPG net. This learning-based control scheme shows promising results in goal-reaching tasks in soft snake robots.

This control scheme can be applicable to a range of bio-mimic motion control for robotic systems and may require different designs of the CPG network given insights from the corresponding biological systems. We have been investigating the generality of the proposed control scheme on different robotic systems and obtained promising early results. The next chapter will be focusing on introducing sensory inputs into the CPG system, which enables reactive responses to contact forces with the external environment and generates an obstacle-aided locomotion controller for the soft snake robot. It is also interesting to investigate distributed control designs that can scale to high-dimensional soft snake robot or other biomimic robotic systems.

Chapter 4

Integrating Contact-aware Feedback CPG System for Learning-based Soft Snake Robot Locomotion Controllers

This chapter aims to solve the contact-aware locomotion problem of a soft snake robot by developing bio-inspired contact-aware locomotion controllers. To provide effective contact information for the controllers, we develop a scale-covered sensor structure mimicking natural snakes' *scale sensilla*. In the design of the control framework, our core contribution is the development of a novel sensory feedback mechanism for the Matsuoka central pattern generator (CPG) network. This mechanism allows the Matsuoka CPG system to work like a "spine cord" in the whole contact-aware control scheme, which simultaneously takes the stimuli including tonic input signals from the "brain" (a goal-tracking locomotion controller) and sensory feedback signals from the "reflex arc" (the contact reactive controller), and generates rhythmic signals to actuate the soft snake robot to slither through densely allocated obstacles. In the "reflex arc" design, we develop two distinctive types of reactive controllers – 1) a reinforcement learning (RL) sensor regulator that learns to manipulate the sensory feedback inputs of the CPG system, and 2) a local reflexive sensor-CPG network that directly connects sensor readings and the CPG's feedback inputs in a specific topology. Combining with the locomotion controller and the Matsuoka CPG system, these two reactive controllers facilitate two different contact-aware locomotion control schemes. The two control schemes are tested and evaluated in both simulated and real soft snake robots, showing promising performance in the contact-aware locomotion tasks. The experimental results also validate the advantages of the modified Matsuoka CPG system with a new sensory feedback mechanism for bio-inspired robot controller design.

4.1 Hardware Design for Contact-aware Soft Robotic Snake Locomotion

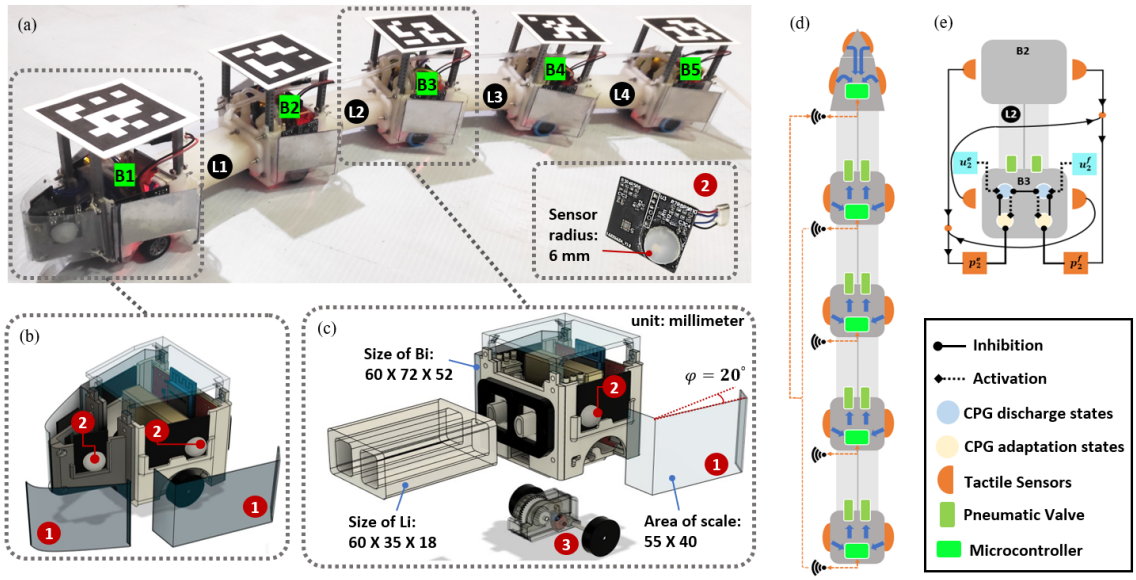


Figure 4.1: (a) Soft robotic snake (soft snake robot) in reality. (b) The 3D model of rigid head (left) and (c) rigid body (right). (d) Signal communication flow of soft snake robot circuit. (e) Example of sensor-CPG connection model for one link of an soft snake robot.

4.1.1 Design of a Contact Sensor

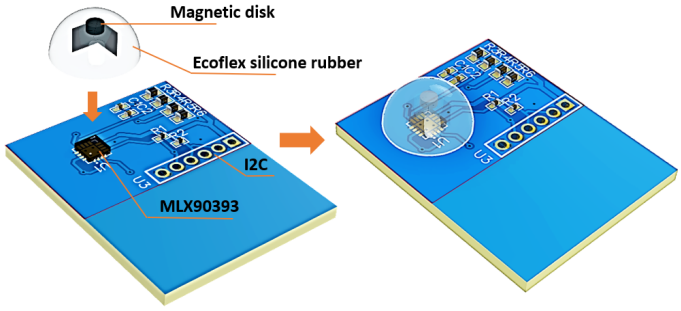


Figure 4.2: Electronic design of touch sensor.

In contact-aware robot locomotion, the tactile sensors are expected to detect the contact force timely. Other desired properties can be low-cost, small-sized, durable, accurate, deformable, and customizable. With these requirements in mind, we choose a magnetic field soft tactile sensor based on [92]. As shown in Fig. 4.2, the

major component of the soft tactile sensor is comprised of a small magnet cylinder disk (with 2 mm diameter and a height of 1 mm) and a Melexis MLX90393 Hall effect module (3mm $\times$ 3mm $\times$ 0.8mm, QFN-16 package) separated by a hemisphere shaped elastomer (made of Ecoflex™ 00-30 silicone rubber). The magnet piece is sealed in the elastomer through moulding of the silicone first and then the elastomer is glued to the top of the hall sensor on the printed circuit board (PCB). The detailed fabrication steps are similar to [92]. The working principle of this tactile sensor is based on detection of the presence and magnitude of a magnetic field using the Hall effect. The magnetic field varies when the elastomer deforms and causes positional changes of the small magnet disk inside the elastomer. These changes can be detected and calculated by the hall sensor. The data collected by the hall sensor is sent to the mother board via Inter-Integrated Circuit (I2C) bus.

According to [92, (12),(13),(14)], the three direction forces of the tactile sensor are calculated by

$$
\begin{aligned}
F_z &= \Sigma_{k=0}^{n}\Sigma_{i=0}^{k} C_{zj} B_z^i B_r^{(k-i)}, \quad j = 1, \dots \frac{(n+1)n}{2} \\
F_r &= \Sigma_{k=0}^{n}\Sigma_{i=0}^{k} C_{rj} B_z^i B_r^{(k-i)}, \quad j = 1, \dots \frac{(n+1)n}{2} \\
F_x &= \frac{B_x}{\sqrt{B_x^2 + B_y^2}} F_r, \\
F_y &= \frac{B_y}{\sqrt{B_x^2 + B_y^2}} F_r.
\end{aligned}
$$

Where F_z is normal magnetic force, F_r is shear magnetic force which can be decomposed to F_x and F_y. Parameters B_z and B_r are the normal and shear magnetic intensity. B_r can be decomposed to B_x and B_y. C_{zj} and C_{rj} are the j-th coefficients of best fitting polynomials of F_z and F_r calculated by moving least squares (MLS) method, and n is the order of the polynomial in the MLS method.

For the proposed control design, the accuracy of the force direction is not a strict requirement due to the scale structure. We thus focus on the detection of contact events and simplify measure the magnitude of the force using

$$
F = \sqrt{F_x^2 + F_y^2 + F_z^2}. \tag{4.1}
$$

Furthermore, we introduce a sigmoid function to normalize the sensory value of the soft tactile sensor, such that

$$
\sigma(F) = \frac{1}{1 - \exp^{-a|F|}}, \tag{4.2}
$$

where $a \in \mathbf{R}$ is a positive constant.

4.1.2 Deployment of Scale Sensors

The soft snake robot consists 4 pneumatically actuated soft links (L1∼L4 in Fig. 4.1(a)) [21,42,44]. The links are connected by 5 rigid bodies (B1∼B4 in Fig. 4.1(a)) enclosing the electronic components that are necessary to control the snake robot. Only one chamber on each link is active (pressurized) at a time. The mechanical design of the soft pneumatic actuators is discussed in [46]. In this chapter, we install elastic one-direction wheels on the rigid part of the soft snake robot to realize anisotropic friction property (component 3 in Fig. 4.1(c), the contribution of the elastic one-direction wheels on improving the energy efficiency of contact-aware locomotion of the soft snake robot can be found in Appendix 6.4.4.

There are in total 12 tactile sensors installed on the robot. As shown in Fig. 4.1(a),(b),(c), the components marked number 2 are the installation positions of the tactile sensors. On top of each tactile sensor, the components marked number 1 are the scales. Each scale is made of two layers of materials – an acrylic layer attached by a steel plate layer. The scales are designed for three major purposes:

- To significantly increase the contact sensitivity and effective sensing area of the snake robot (contact area expand about 20 times, according to Fig. 4.1(c)).
- To reduce friction resistance on the contact surface (friction coefficient reduced from 1.7 of dry silicone to around 0.3 of polished acrylic board).
- To protect the silicone tactile node from frequent collisions.

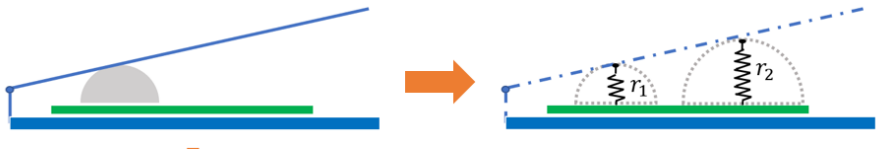


Figure 4.3: Tactile sensor+scale structure (top) versus its approximation in simulation (bottom).

In order to simulate the robot for reinforcement learning and sim-to-real transfer of the learned controller, we developed a physics-based high-fidelity simulator that models the inflation and deflation of the air chamber and the resulting deformation of the soft bodies with tetrahedral finite elements [21]. To simplify tactile sensing function of the scale structure in simulation, we use two hemisphere elastic force fields to model the tactile sensor node+scale structure in reality (as shown in Fig. 4.3). The elastic force fields have equilibrium positions (where elastic force equals zero) everywhere on the surface of the hemispheres and has no friction on the hemispheres. The tactile readings are modeled by the elastic forces when an object's distance is smaller than the radius of any simulated tactile node. In the simulation, the reading of the two hemisphere force fields are added together to simulate the contact force signal of one tactile sensor in the real robot.

Inspired by a previous study on obstacle-aided locomotion of rigid snake robots [40], we approximate the total contact force acted on each rigid body B_i based on the inputs sampled from the force sensors. For simplicity, we reduce the element of the sensory force representation by subtracting the two inputs from a pair of diagonal sensors. Therefore, for the i-th rigid body (counted from the head as 1st rigid body), we have

$$N_i = \begin{cases} N_{i1}^e + N_{i2}^e - N_{i1}^f - N_{i2}^f, & i = 1 \\ N_i^e - N_i^f, & i = 2, 3, 4, 5. \end{cases} \tag{4.3}$$

According to (4.2), let F_i^e, F_i^f represent the magnitude of contact force detected from the left and right sensor respectively on the i-th rigid body of the soft snake robot, then $N_i^e = \sigma(F_i^e)$, and $N_i^f = \sigma(F_i^f)$ (left, right in reference to the heading direction of the soft snake robot). The head joint is a special case since it has two pairs of sensors installed. The collection of contact forces in the soft snake robot from head to tail forms a vector

$$\mathbf{N} = [N_1, N_2, N_3, N_4, N_5]^T.$$

When the robot is in contact with an obstacle, the contact force $N_i^q = N$ on each tactile scale occurs as shown in Fig. 4.1c. However, due to the smoothness of the scale and reduction of φ (the angle between the scale and the rigid body) during the contact, the N_t component of N_i^q on the tangent direction of the scale is small. As a result, we assume $|N_t|$ to be always smaller than the maximum torque of the torsion spring, so that τ and N_t are in balance. Therefore, N_t is neglected in the simulator and we take $N_i^q \approx N_n$ for simplicity.

4.2 Modified Matsuoka Oscillator with Sensory Feedback

In order to effectively integrate the tactile information into the contact-aware locomotion framework of soft snake robot, we study the effect of an additional variable on the Matsuoka oscillator for handling the feedback force signals from the tactile sensors. In this section, we analyze the properties of our method and the conventional approach [19, 85] from theoretical perspective.

In our previous work [44], we presented a control scheme that employs sensor-free Matsuoka oscillators to generate undulating control signals as actuation inputs for the soft snake robot to perform Serpentine locomotion. The original Matsuoka

oscillator is a piece-wise linear dynamical system, which has the following form:

$$\begin{aligned}
K_f\tau_r\dot{x}_i^e &= -x_i^e - az_i^f - by_i^e - \sum_{j=1}^{N} w_{ji}y_j^e + u_i^e + c,\\
K_f\tau_a\dot{y}_i^e &= z_i^e - y_i^e,\\
K_f\tau_r\dot{x}_i^f &= -x_i^f - az_i^e - by_i^f - \sum_{j=1}^{N} w_{ji}y_j^f + u_i^f + c,\\
K_f\tau_a\dot{y}_i^f &= z_i^f - y_i^f,
\end{aligned} \tag{4.4}$$

where the subscripts e and f represent variables related to extensor neuron and flexor neuron, respectively. The tuple (x_i^q, y_i^q), $q \in \{e, f\}$ represents the activation state (or membrane potential) and self-inhibitory state (or adaptation state [51,53]) of i-th neuron respectively, $z_i^q = \max(0, x_i^q)$ is the output of i-th neuron. Tonic inputs u_i^e, u_i^f are the major coefficients that can be controlled to affect the output bias and amplitude of the Matsuoka oscillator. The frequency ratio $K_f \in \mathbf{R}$ can be manipulated to affect the natural oscillation frequency of the system. The free-response input introduced in [44] is denoted as parameter c in the equation, which is used for amplifying free-response oscillation of the CPG system. The remaining parameters are all constant weights. In system (4.4), all coupled signals including x_i^q, y_i^q and z_i^q ($q \in \{e, f\}$) are inhibiting signals (negatively weighted), and only the tonic inputs are activating signals (positively weighted).

Based on the form of original Matsuoka oscillator, the question is how to integrate sensory feedback to the Matsuoka CPG system to affect its outputs efficiently?

A conventional approach is to directly add positive force feedback (as activation signals) to the membrane potential state equations $(\dot{x}_i^e, \dot{x}_i^f)$ of the original Matsuoka oscillator [85, (5)]. Such form of feedback Matsuoka oscillator has been used in some snake robot locomotion studies [19,85], where the tonic inputs (for locomotion control) of the CPG systems in these applications are mostly constant or regular sinusoidal waves. We summarize the dynamic equations of the conventional feedback Matsuoka oscillator as follows:

Membrane Potential Feedback Form Matsuoka Oscillator:

$$\begin{aligned}
K_f\tau_r\dot{x}_i^e &= -x_i^e - az_i^f - by_i^e - \sum_{j=1}^{N} w_{ji}y_j^e + u_i^e + bp_i^e + c,\\
K_f\tau_a\dot{y}_i^e &= z_i^e - y_i^e,\\
K_f\tau_r\dot{x}_i^f &= -x_i^f - az_i^e - by_i^f - \sum_{j=1}^{N} w_{ji}y_j^f + u_i^f + bp_i^f + c,\\
K_f\tau_a\dot{y}_i^f &= z_i^f - y_i^f,
\end{aligned} \tag{4.5}$$

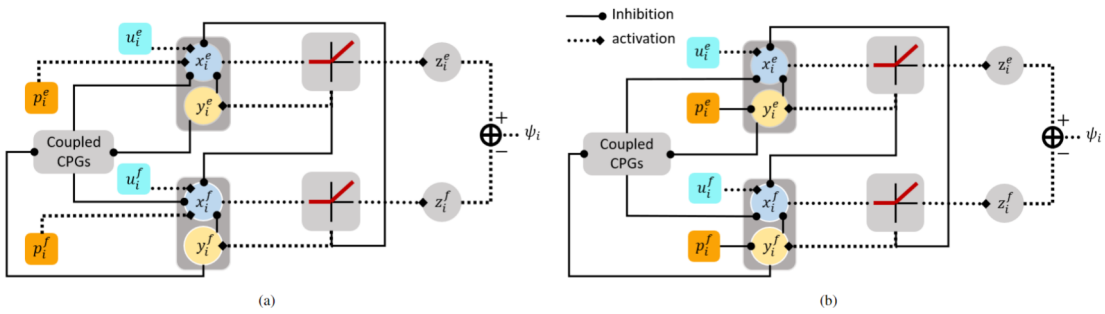


Figure 4.4: Scheme of modified Matsuoka oscillator with different allocation of feedback signals: (a) the conventionally used form (we name as MPF) Matsuoka oscillator and (b) the Adaptation feedback (AF) form Matsuoka oscillator proposed by us in this work.

where the sensory force feedback signals are represented by p_i^e and p_i^f. As shown in Fig. 4.4a, the reason of naming "membrane potential feedback form" (MPF) to this type of Matsuoka oscillator is because that the sensory feedback signals are directly added to the membrane potential states as activation (positive) signals.

However, we have a concern about the above conventional form. In the case when the tonic inputs are complicated wave signals and the sensory feedback are irregular signals, these two types of inputs may intervene each other, and therefore fail to effectively present the impact of sensory feedback to the system output.

In [73], the authors mentioned the addition of the CPG state coupling terms not only to the fast dynamic states' equations (potential membrane $\dot{x}_i^e, \dot{x}_i^f$) but also to the slow dynamic states' equations (adaptation states $\dot{y}_i^e, \dot{y}_i^f$) of the Matsuoka oscillator with opposite signs to improve the dynamic impact of the coupling signals. Given the inspiration, we consider whether it is possible to add sensory feedback signals which are external impulse signals to the adaptation states of the Matsuoka oscillator? And should the feedback signals be activating or inhibiting in the adaptation state? Could this modification resolve our previous concerns? Why people didn't try this direction in their contact-aware locomotion studies? After in-depth theoretical analysis and experimental comparison, we construct a novel branch of feedback mechanism in the Matsuoka oscillator as follows.

Adaptation Feedback Form Matsuoka Oscillator:

$$
\begin{aligned}
K_f \tau_r \dot{x}_i^e &= -x_i^e - a z_i^f - b y_i^e - \sum_{j=1}^{N} w_{ji} y_j^e + u_i^e + c, \\
K_f \tau_a \dot{y}_i^e &= z_i^e - y_i^e - p_i^e, \\
K_f \tau_r \dot{x}_i^f &= -x_i^f - a z_i^e - b y_i^f - \sum_{j=1}^{N} w_{ji} y_j^f + u_i^f + c, \\
K_f \tau_a \dot{y}_i^f &= z_i^f - y_i^f - p_i^f,
\end{aligned}
\tag{4.6}
$$

In this design, the tonic inputs u_i^e, u_i^f as well as the free oscillation tonic input c are still added to the potential membrane states (x_i^e, x_i^f) as fast dynamic inputs, while the sensory feedback p_i^e and p_i^f are added to the equations of adaptation states (y_i^e, y_i^f) of Matsuoka oscillator as slow dynamic feedback inputs (see Fig. 4.4b). In this work, we name this version of the Matsuoka oscillator as the adaptation feedback (AF) form Matsuoka oscillator.

To explore the feasibility of AF form Matsuoka oscillator, and find out the advantage of the AF form design, we discuss the difference between AF and MPF form of Matsuoka oscillator when the sensory feedback signals are variables. The discussion is organized by the following derivations:

Considering AF form Matsuoka oscillator described in system (4.6) and MPF form Matsuoka oscillator method described in system (4.5).

First, we set $x_i = x_i^e - x_i^f$, $y_i = y_i^e - y_i^f$, $z_i = z_i^e - z_i^f$, $u_i = u_i^e - u_i^f$, $p_i = p_i^e - p_i^f$. By taking subtraction between flexor and extensor in (4.6) and neglect phase related coupling terms from other primitive CPGs, we have

$$\begin{aligned} K_f \tau_r \frac{d}{dt} x_i &= -x_i + a z_i - b y_i + u_i \\ K_f \tau_a \frac{d}{dt} y_i &= z_i - y_i - p_i. \end{aligned} \tag{4.7}$$

Similarly, (4.5) can be simplified to

$$\begin{aligned} K_f \tau_r \frac{d}{dt} x_i &= -x_i + a z_i - b y_i + u_i + b p_i \\ K_f \tau_a \frac{d}{dt} y_i &= z_i - y_i. \end{aligned} \tag{4.8}$$

If x_i^e and x_i^f satisfy the *perfect entrainment assumption* [53], we have $z_{\mathcal{F}i} = K(r_x) x_{\mathcal{F}i}$, where r_x is the amplitude bias of x_i, and $K(\cdot)$ is the amplitude coefficient function of $x_{\mathcal{F}i}$ [44, (B.4)]. The subscript $_{\mathcal{F}i}$ indicates the fundamental sinusoidal and constant component in Fourier expansion of the corresponding variable. Without loss of generality, let $K_f = 1$, Eq. (4.7) can be further simplified to

$$\begin{aligned} \tau_r \frac{d}{dt} x_{\mathcal{F}i} + x_{\mathcal{F}i} &= a K(r_x) x_{\mathcal{F}i} - b y_{\mathcal{F}i} + u_{\mathcal{F}i} \\ \tau_a \frac{d}{dt} y_{\mathcal{F}i} + y_{\mathcal{F}i} &= K(r_x) x_{\mathcal{F}i} - p_i. \end{aligned} \tag{4.9}$$

And (4.8) can be further simplified to

$$\begin{aligned} \tau_r \frac{d}{dt} x_{\mathcal{F}i} + x_{\mathcal{F}i} &= a K(r_x) x_{\mathcal{F}i} - b y_{\mathcal{F}i} + u_{\mathcal{F}i} + b p_i \\ \tau_a \frac{d}{dt} y_{\mathcal{F}i} + y_{\mathcal{F}i} &= K(r_x) x_{\mathcal{F}i}. \end{aligned} \tag{4.10}$$

Next, an ordinary differential equation can be obtained by merging the two equations in (4.9) as,

$$\tau_r\tau_a\frac{d^2}{dt^2}x_{\mathcal{F}i} + (\tau_r + \tau_a - \tau_a aK(r_x))\frac{d}{dt}x_{\mathcal{F}i} \tag{4.11}$$
$$+ ((b-a)K(r_x)+1)x_{\mathcal{F}i} = \tau_a\frac{d}{dt}u_{\mathcal{F}i} + u_{\mathcal{F}i} + bp_i.$$

And merging the two equations in (4.10) yields

$$\tau_r\tau_a\frac{d^2}{dt^2}x_{\mathcal{F}i} + (\tau_r + \tau_a - \tau_a aK(r_x))\frac{d}{dt}x_{\mathcal{F}i} \tag{4.12}$$
$$+ ((b-a)K(r_x)+1)x_{\mathcal{F}i} = \tau_a\frac{d}{dt}u_{\mathcal{F}i} + u_{\mathcal{F}i} + bp_i + \tau_a b\dot{p_i}.$$

From the right hand side of (4.12) and (4.11), the derivation (4.12) of MPF form Matsuoka oscillator has an additional free term $\tau_a b\dot{p_i}$ comparing to the derivation (4.11) of the AF form Matsuoka oscillator. According to the superpositivity of solutions of the second order ordinary differential equation (ODE), when p_i is a variable with complex wave form (e.g. collision force signals), the interference of $\tau_a b\dot{p_i}$ will be relatively large. Concluding the above discussion yields the following remark.

Remark 2. *For the two types of feedback Matsuoka system (AF form and MPF form) satisfying perfect entrainment condition [53], when the feedback inputs p_i^e, p_i^f are variables, the MPF form has an additional input disturbance caused by $\dot{p_i}$, which could bring overshoot and delay to the system. Thus, the feedback inputs of AF form Matsuoka oscillator is more effective than the feedback inputs of MPF form Matsuoka oscillator.*

The result of a simple comparison test in Fig. 4.5 further shows the disturbance caused by $\tau_a b\dot{p_i}$ in MPF form Matsuoka oscillator. In this test we input the same section of sensor signal (in orange) to both a primitive AF and a primitive MPF Matsuoka oscillator. The tonic inputs are kept constant for both form of CPGs. From the MPF output curve, much larger overshoots comparing to the AF form output are observed every time when a significant contact signal is detected. In every recovery phase after the sensory input vanishes, the output signal of MPF form Matsuoka oscillator also gets delayed before recovering to the normal oscillation, while this problem is not observed in the output of AF form Matsuoka oscillator. These observations verify the conclusion in Remark 2.

It is also worth noted that, in the AF form Matsuoka oscillator, the sensory feedback signals should be inhibiting instead of activating. The detailed illustration of this design can be found in Appendix 6.3.5.

Next, in the AF form Matsuoka oscillator, in order to compare the impact of tonic inputs u_i^e, u_i^f and sensory feedback inputs p_i^e, p_i^f to the output amplitude bias,

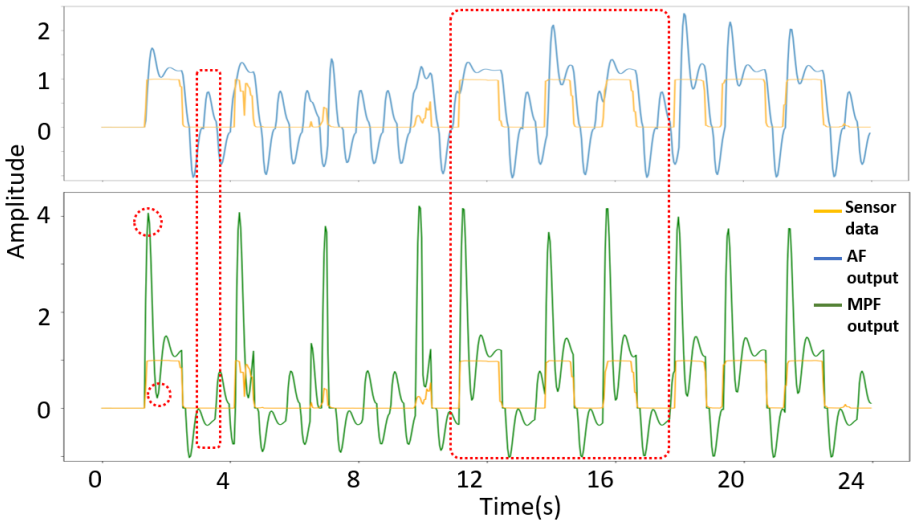


Figure 4.5: Output of AF form and MPF form Matsuoka oscillator given sensory feedback data.

we introduce the following proposition,

Proposition 3. *If an AF form Matsuoka oscillator satisfies the following conditions: 1) the dynamical model of the primitive Matsuoka oscillator is harmonic, 2) the tonic inputs u_i^e and u_i^f are square wave signals and are complementary to each other, 3) the sensory feedback signals p_i^e and p_i^f are square wave signals and are complementary to each other. 4) u_i^e is entrained with z_i^e, and u_i^f is entrained with z_i^f, then the oscillation bias of z_i and the bias of u_i satisfies the following relationship,*

$$bias(z_i) = \frac{1+2m}{b-a+2} bias(u_i) + \frac{b}{b-a+2} bias(p_i), \tag{4.13}$$

where $z_i = z_i^e - z_i^f$, $u_i = u_i^e - u_i^f$, $p = p_i^e - p_i^f$, and

$$m = \frac{1}{\pi} \frac{1}{2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n)}$$

is a constant coefficient (r_i indicates amplitude of state x_i).

Proof. (See Appendix 6.3.4.) □

Proposition 3 shows that in the AF form Matsuoka oscillator, there exists a binary linear relationship between the bias of u_i, p_i and the bias of z_i. Because the range of u_i and p_i are both limited within $[0, 1]$, the impact of p_i is larger than u_i when the coefficient of bias(p_i) is larger than the coefficient of bias(u_i). In this chapter, the constant parameters of the Matsuoka oscillator are configured according to Table 6.1. Under this condition, we have $b >> 1 + 2m$. The above discussion

indicates that when contacts occur, bias(p_i) makes major contribution to the output bias bias(z_i).

Overall, the properties of AF form Matsuoka oscillator shows its flexible and accurate capability of reacting to the contact events. Based on this, we are able to further develop contact-aware controllers for the soft snake robot locomotion.

4.3 Design of Controllers

In literature of obstacle-aided snake robot locomotion control, there are two ways to combine sensory feedback with the locomotion controller. One, referred to as *learning-based hybrid control*, is the event-triggered hybrid control [39] that utilizes an individual event-triggered controller to optimizes the control command together with the main locomotion controller only when the contacts happen. Even when only a single part of the robot body is in contact, the event-triggered controller will send control command to the whole system. Another method, referred to as *learning-based reflex control*, is to use local reflexive method [34] to setup distributed rules for the snake links such that only a few neighboring links will react to the sensory feedback, and such reactions are independent to the main locomotion controller. Both directions have their advantages. Although learning-based reflex control is computationally expensive, it can achieve great performance through training. On the other hand, the local reflexive method is light-weighted and distributed. Thus, it provides more flexibility to the controller design as well as robustness to the damage of the robot actuators.

Combining the above two directions with AF form Matsuoka CPG system respectively, we propose two different control methods: the AF-learning method and the AF-local method.

4.3.1 Event-triggered learning-based sensory reactive controller with AF Form CPG System

In the narrative of this chapter, the extensor and flexor in the CPG system are assigned with left and right side of the snake robot respectively (taking heading direction of the robot for reference).

In the AF-learning method, we introduce the concept of hybrid control to model-free learning-based control framework, which is composed of two controllers in the contact-aware goal tracking task of soft snake robot – including an RL controller for goal tracking locomotion named C1, and an event-triggered RL controller for contact reactive control named R2, which only outputs actuator signals when the contact event-triggering condition is satisfied. The scheme of the controller is shown in Fig. 4.6.

We use a goal tracking controller developed in our previous work [44], called Free-response Oscillation Constrained Proximal Policy Optimization Option-Critics

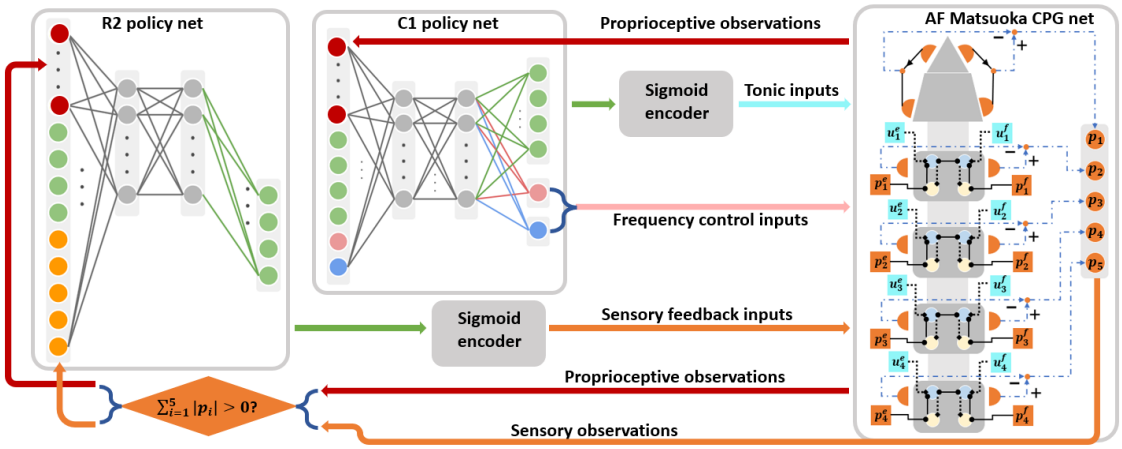


Figure 4.6: AF-learning control scheme.

with Central Pattern Generator (FOC-PPOC-CPG), as the C1 controller. The C1 controller takes fully proprioceptive observations of the soft snake robot's dynamic states and outputs control commands through manipulating the tonic inputs of the CPG network as low-level primitive actions and frequency ratio of the CPG network as high-level options.

For R2 controller, we define the *contact event-triggering condition* as follows: At each time step, given the contact force vector $\mathbf{f}$ and contact detection threshold ϵ. The event-triggering condition for the contact-aware scenario is $||\mathbf{f}|| > \epsilon$. When the event-triggering condition is satisfied, R2 is triggered to join the manipulation of the CPG system.

Although it is not necessary for R2 to use the same learning algorithm as C1, for simplicity we also train R2 with PPOC-CPG framework, which shares the same reward function and AF form CPG system with C1, but with different observation states and actions.

In the obstacle-based locomotion scenario, there are in total 19 observation states for R2, denoted as $\boldsymbol{\zeta} = \{\zeta_1, \zeta_2, ..., \zeta_{19}\}$, where $\zeta_{1:4}$ represents the dynamic state of the robot referenced on the goal position, $\zeta_{5:8}$ represents the real-time body curvature of the 4 soft links, $\zeta_{9:14}$ contains actions in the last time step including the previous option and the terminating probability, $\zeta_{15:19}$ contains the pre-processed contact forces. Similar to C1, the actions of R2 are mapped to fit the sensory feedback signals of the Matsuoka CPG network of soft snake robot. Next, we define a four dimensional action vector $\mathbf{a} = [a_1, a_2, a_3, a_4]^T \in \mathbf{R}^4$ and map $\mathbf{a}$ to sensory feedback vector $\mathbf{p}$ as follows,

$$p_i^e = \frac{1}{1 + e^{-a_i}}, \text{ and } p_i^f = 1 - p_i^e, \text{ for } i = 1, \ldots, 4. \tag{4.14}$$

This mapping bounds p_i^e and p_i^f within $[0, 1]$. The sensory feedback input vector $\mathbf{p}$

for the 4-link snake robot is an eight-dimension vector,

$$\mathbf{p} = [p_1^e, p_1^f, p_2^e, p_2^f, p_3^e, p_3^f, p_4^e, p_4^f]^T.$$

The learning process of the whole control scheme (as shown in Fig. 1.12a) is: C1 is first trained in obstacle-free environment in simulation. After C1 is converged, we fix C1 as a regular controller for goal tracking purposes. C1 policy is always effective regardless the triggering of the contact events. Then we train R2 in the environment with randomly generated obstacle mazes in simulation until convergence. R2 is effective only when the contact event-triggering condition is satisfied. According to Remark 2 and Proposition 3, when the parameters of the AF form Matsuoka CPG system satisfy Table 6.1, when R2 is effective, it will dominate the control of the CPG system (contact-awareness over goal-awareness).

4.3.2 Local Reflexive Control of Contact-aware slithering locomotion with AF Form CPG System

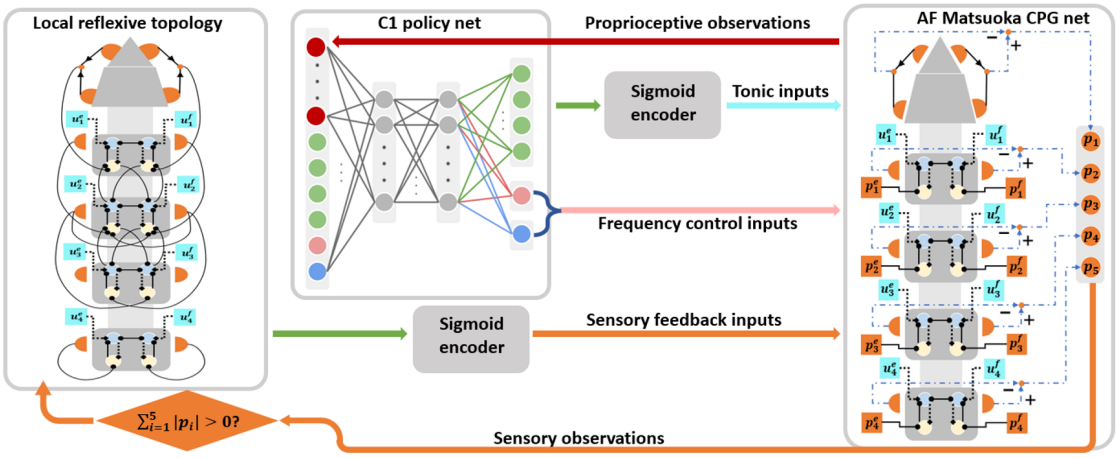


Figure 4.7: AF-local control scheme.

In this section, we describe a different learning-based controller, denoted as AF-local because the local reflexive mechanism is based on AF form Matsuoka oscillator. According to (4.13) and Proposition 3, if the parameters of an AF form Matsuoka oscillator satisfy Table 6.1, we have $b >> 1 + 2m$, so that the sensory input p_i will play a major role to influence the tonic input u_i when contact events are detected by the tactile sensors.

We validated the property through experiments. As shown in Fig. 4.8e, Case I describes a situation when there comes a p_i^q signal in only one side, the adaptation variable $y_i^q, (q \in e, f)$ on the same side is inhibited, and result in an activation of the corresponding x_i^q (at the same time the opposite x_i^q state is inhibited). At this moment, no matter what value of tonic input u_i^e, u_i^f are given within their range

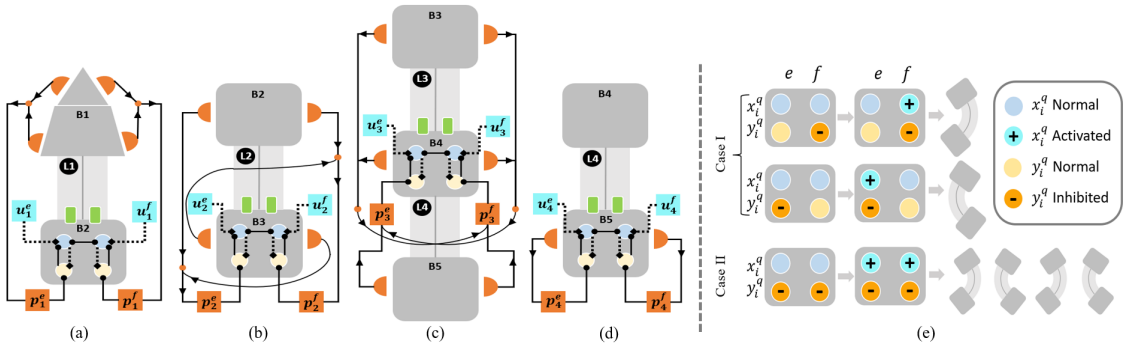


Figure 4.8: (a)-(d) Local reflexive structure of modified Matsuoka CPG network, and (e) Soft body actuation by the modified Matsuoka oscillator under contact.

$u_i^q \in [0,1]$ [44], the soft actuator will always bend towards the opposite direction of the incoming p_i^q signal. Case II shows the case when both p_i^e and p_i^f are inhibiting y_i^e and y_i^f respectively, both x_i^e and x_i^f will be strongly activated, leading to almost a free-response oscillation output regardless the values of u_i^e, u_i^f. According to Proposition 3, the oscillation bias of the CPG output in Case II situation depends on bias(p_i), where $p_i = p_i^e - p_i^f$.

We take the inspiration of local reflexive mechanism from [34,35] such that only the links that are close to a contact sensor may react to its contact events. Due to the differences of structure (antagonistic actuators, partially tunable chambers) between our pneumatically actuated snake robot and the real-time tunable spring actuated snake model described in [34], we have our specific principles for constructing the reflexive loop between the sensors and the sensory feedback inputs of the CPG network (Fig. 4.8):

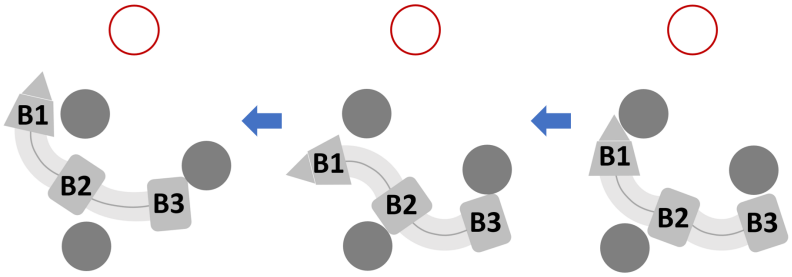


Figure 4.9: Example of reflexive mechanism on link L1.

- Since the head of the snake robot maneuvered by the goal-tracking controller is always heading to the goal direction, it could be blocked by an obstacle on the path to the target location if the head cannot properly react to the contact and turn away from the obstacle. Thus the snake robot's head should always bend to the opposite direction to the major contact event, which means that the ipsilateral chamber of L1 link to the contact side of B1's sensor will be

actuated. Figure 4.9 provides an example showing the reflexive behavior of L1 link when the head sensor on B1 touches an obstacle.

- As the rudder of steering and source of propulsion, the snake robot's tail should always push itself against the obstacles to keep oscillation with larger amplitude. So the ipsilateral chamber of L4 link to the contact side of B5's sensor will be actuated. In addition, in a 4-link soft snake robot, the role of the tail during slithering locomotion should be strengthened to generate more propulsion. Therefore, we extend the impact of the tail sensors to the actuators in L3 soft link.

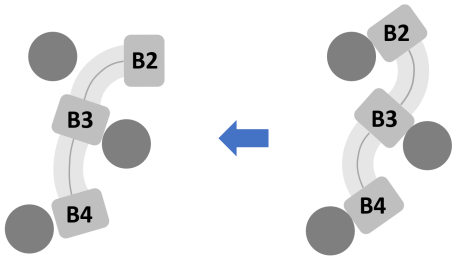


Figure 4.10: Example of reflexive mechanism on L2 and L3 links.

- Beyond head and tail links, the other soft body links' CPG nodes should refer to their neighboring sensors to determine their reflexive behaviors accordingly. To design the connection between CPG nodes of the body links and the corresponding sensors, we refer to the jamming case that mostly occurred in the contact-aware locomotion of the soft snake robot. As shown in Fig. 4.10, when B2, B3 and B4 rigid parts are in contact with the obstacles on the opposite sides, the situation leads to a typical jamming scenario for our soft snake robot in contact-aware locomotion. In this situation, the L3 link is supposed to decrease its bending curvature to avoid jamming, while L2 should actuate its ipsilateral chamber (extensor) to create more space for free oscillation controlled by the goal-reaching controller.

Based on the above features and former experience on designing local reflexive control rules [34], we design the topology of the sensor connection to each CPG node in the soft snake robot's "vertebrate" system. As shown in Fig. 4.8, the sensory feedback signals p_i^e, p_i^f are normalized when receiving non-zero inputs from the connected tactile sensors. Before formulating p_i^e, p_i^f, we first define set $\mathcal{D}_i, i = 1, 2, 3, 4$ as the set of sensor signals' numbers connected to the i-th Matsuoka CPG node. For example, for 3rd CPG node in Fig. 4.8(c), $\mathcal{D}_3 = \{3, 4, 5\}$. In addition, we define the connection marker array $\mathbf{J} = [J_1, J_2, J_3, J_4, J_5] = [-1, -1, 1, 1, -1]$. The value in $\mathbf{J}$ is assigned based on the way of connection between the sensors and the

CPG network.

$$p_i^e = \frac{\Sigma_{k\in\mathcal{D}_i} I^e(N_k)|N_k|}{\Sigma_{k\in\mathcal{D}_i}|N_k| + \delta^+},\ p_i^f = \frac{\Sigma_{k\in\mathcal{D}_i} I^f(N_k)|N_k|}{\Sigma_{k\in\mathcal{D}_i}|N_k| + \delta^+}. \tag{4.15}$$

where $\delta^+ \in \mathbf{R}^+$ is a small positive number to avoid division by zero, and

$$I^e(N_k) = \max\{0, -\text{sgn}(J_k N_k)\},\ I^f(N_k) = \max\{0, \text{sgn}(J_k N_k)\}.$$

More specifically, the mechanism of (4.15) acting on the actuators of the soft snake robot can be explained as follows:

- In L1 CPG node (Fig. 4.8a), the sensors are connected to the same side of sensory feedback inputs p_i^e, p_i^f of L1 CPG. When one side of B1 sensors are in contact, the actuation of L1 link follows Case I, which bends toward the opposite direction to the triggered sensors.

- In L2 CPG node (Fig. 4.8b), the sensors on B2 are connected to the same side of sensory feedback inputs of L2 CPG, while the sensors on B3 are connected to the opposite side of sensory feedback inputs of L2 CPG. When only B2 or B3 sensor is triggered, or both B2 and B3 receives contact feedback from the opposite side, L2 will behave in Case I. When B2 and B3 have contacts on the same side, both y_2^e and y_2^f will be inhibited, leading to Case II behavior of L2.

- In L3 CPG node (Fig. 4.8c), the sensors on B3 and B4 are connected to the opposite side of sensory feedback inputs of L3 CPG, while the B5 sensors are connected to the same side of sensory feedback inputs of L3 CPG. Consider single sensor triggered case, when only B3 or B4 or B5 sensor is triggered, L3 will also behave in Case I. For two sensors triggered case: when only (B3 and B4) are triggered on the same side, or (B3 and B5) or (B4 and B5) are triggered on the opposite side, L3 will behave in Case I; when only (B3 and B4) are triggered on the opposite side, or (B3 and B5) or (B4 and B5) are triggered on the identical side, L3 will behave in Case II. For three sensors triggered case, when B3 and B4 are triggered on the same side opposite to the contact side of B5, L3 will behave in Case I, otherwise L3 behave in Case II.

- In L4 CPG node (Fig. 4.8d), the B5 sensors are connected to the same side of sensory feedback inputs of L4 CPG. When one side of B5 sensor is in contact, the actuation of L4 link follows Case I, which bends toward the opposite direction to the triggered sensors.

The overall work flow of AF-local is showed in Fig. 1.12b. The local reflexive mechanism introduced in this section works independently and map the tactile sensor data to the sensory feedback signals of CPG system. In the mean time, the tonic input signals in the same CPG system are controlled by a C1 controller introduced

in Section 4.3.1, which only focuses on the goal tracking control of the soft snake robot.

In order to compare AF form Matsuoka CPG system with the conventional MPF form Matsuoka CPG system, we also develop MPF-local and MPF-learning controllers by replacing the AF form Matsuoka oscillator with MPF form Matsuoka oscillator in the two control methods introduced in Section 4.3.2 and Section 4.3.1. In Section 4.4, we will comprehensively compare the performance of the AF-local, AF-learning, MPF-local, and MPF-learning methods.

4.3.3 Design of the shared reward function

We now present our design for the reward function shared by both locomotion and contact-aware controllers. Our design will ensure that by maximizing the discounted sum of reward, the learned controller can achieve efficient locomotion and accurate set-point tracking.

To improve learning efficiency, we employ a potential field-based reward function. Artificial potential field (APF) is widely applied in planning problems and potential game theory [16,37,63] to accelerate the process of searching for the optimal strategy. The potential field can be classified into two categories – the attracting field for target reaching and the repulsive field for obstacle avoidance. The attracting field function is defined as follows

$$U_{att}(\mathbf{p}) = \frac{1}{2} k_{att} ||\mathbf{p} - \mathbf{p}_g||^2,$$

where $\mathbf{p}$ is the coordinate of the agent and $\mathbf{p}_g$ is the coordinate of the goal. Coefficient k_{att} is a positive constant indicating the strength of the attractive potential field. Since the attracting gravity is always pointing toward the goal coordinate from any position of the map, the value of gravity force should be negative. By taking the negative gradient of U_{att}, we have the attracting force function

$$\mathbf{F}_{att}(\mathbf{p}) = -\nabla U_{att} = -k_{att}(\mathbf{p} - \mathbf{p}_g).$$

The reward is designed to encourage the goal-reaching, guided by the artificial potential field. We design the reward to be composed of two rewards:

$$R = \omega_1 R_{goal} + \omega_2 R_{att}, \tag{4.16}$$

where $\omega_i, i = 1, 2, 3$ are constant weights. R_{goal} is the termination reward for reaching a circular accepting area centered at the goal.

$$R_{goal} = \cos\theta_g \sum_{k=0}^{i} \frac{1}{l_k} \mathbf{1}(\rho_g < l_k).$$

where θ_g is the deviation angle between the locomotion direction of the snake robot and the direction of the goal, l_k defines the radius of the accepting area in task-level k, for $k = 0, \ldots, i$. $\rho_g = ||\mathbf{p} - \mathbf{p}_g||$ is the linear distance between the head of the robot and the goal, and $\mathbf{1}(\rho_g < l_k)$ is an indicator function to determine whether the robot's head is within the accepting area of the goal. R_{att} is the reward function of the attracting potential field:

$$R_{att} = \mathbf{v} \cdot \mathbf{F}_{att}(\mathbf{p}),$$

where $\mathbf{v}$ is the velocity vector. The dot product between $\mathbf{v}$ and the potential field vector represents the extent of the agent's movement on following the potential-flow in the task space. In this reward design, though the repulsive potential function generates a cost for the contact, its combination with the other two reward terms may encourage the contact especially when the contact force can aid the locomotion. We discuss this aspect in the experimental validation.

4.4 Experiments

4.4.1 Signal Communication and Obstructed Environment Setting

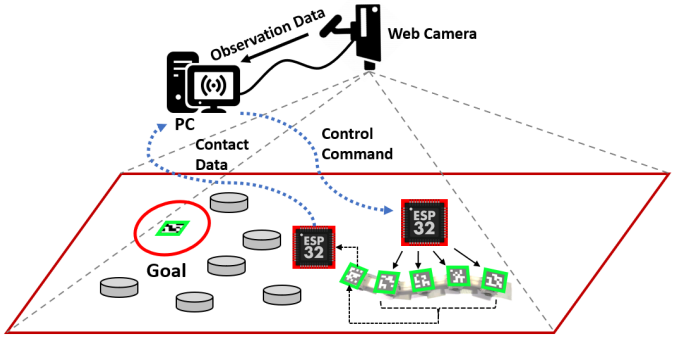


Figure 4.11: Experiment setup of the contact-aware goal tracking locomotion task in reality.

Similar to our previous work [44], the two dimensional dynamic states of the soft snake robot are captured and calculated by a web camera (works under 120 Hz) hanging on the ceiling of the experiment room. We use Aruco [20] to detect and localize QR codes attached to every rigid body of the snake robot and the goal position. Figure 4.11 shows the experiment setup for the real snake robot goal-reaching tasks. In this chapter, we update two major parts of the experiment settings:

- In the signal communication part, each ESP32 chip collects contact sensor information from local I2C and share the data with the head chip through WiFi. In every time step, the head ESP32 chip packs all the sensor data and send it back to the PC controller. The controller program running on the desktop computer receives the observation states from the web camera and the robot and generates the control commands and passes them to the body ESP32 chips on the snake robot through WiFi communication. The ESP32 chips on the snake bodies translate the commands into Pulse Width Modulation (PWM) signals to activate or deactivate the valves [21, 46] on the snake robot. The hardware communication rate between the PC controller and the snake robot is 30 Hz.

- In the environment setting, a number of tin cans filled with stones and sands are placed in the experiment field as obstacles. Each vertical peg in Fig. 4.11 represents a cylinder tin can with a diameter of 100 millimeters (mm) and height of 80 mm. The average weight of the obstacles are around 1.1 kg each, and the weight of the soft snake robot is 0.7 kg (include batteries). It has been tested to ensure that any collision caused by the soft snake robot will not move the obstacles.

4.4.2 Simulated Training and Evaluation

Reinforcement Learning Configuration: In the simulated training part, the fundamental configuration of NN is the same as [44] (four-layered with 128×128 hidden states). The goal-reaching controller C1 is a pre-trained module as is configured in [44]. In this chapter, the contact-aware regulator R1 in AF-learning and MPF-learning controllers is trained in a goal-reaching task with a randomly generated 6×5 obstacle maze. During the training process of R1, the distance between the robot and the goal is fixed to 1.5 meters. The deviation angle between the snake robot and the goal is initially sampled from $0 \sim 60$ degrees with a uniform distribution. In the simulator, the distance between every two obstacles is sampled between $120 \sim 180$ mm. The coordinate of each obstacle is added by an additional clipped standard Gaussian noise ($\omega \sim \mathcal{N}(0, 1)$, clipped by $-0.01 < \omega < 0.01$). The method of simulating contact sensors is introduced in Section 4.1.2. In order to compensate for the mismatch between the simulation and the real environment, most notably the friction coefficients, we employ a domain randomization technique [86], in which a subset of physical parameters are sampled from several uniform distributions. The range of distributions of domain randomization (DR) parameters used for training are in Table 6.3 (see Appendix 6.1). The whole training process of each method runs on 4 simulated soft snake robots (Rendered by Nvidia Flex) on a workstation equipped with an Intel Core i7-9700K, 32GB of RAM, and one NVIDIA RTX2080 Super GPU.

Task specification: In the contact-aware locomotion task, the robot is required to

traverse an array of obstacles and reach the randomly generated goals. Similar to the real world setting in Fig. 4.11, there is also an accepting radius in the simulation for each goal-reaching task, which means that the robot needs to be close enough to the goal in order to succeed and receive a terminal reward. At each time step, the robot also receives a reward from the potential field defined in Section 4.3.3. If the agent reaches the accepting region of the current goal, a new goal is randomly sampled. In the failing situation, when the robot is jammed by the obstacles for a certain amount of time, the desired goal will be re-sampled and updated. The starvation time threshold for failing condition is 900 ms. In addition, if the linear velocity of the snake robot stays negative on the goal direction for over 360 time steps (each time step is about 20 ms), the goal-reaching task is also judged as a failure and trigger the re-sampling of the new task.

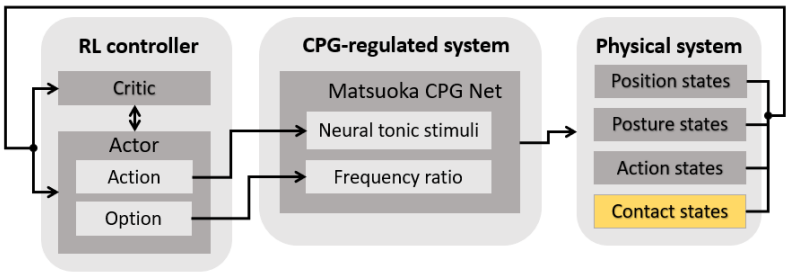


Figure 4.12: Flow chart of C1+ method. Different from C1, C1+ has contact information in its observation states, and is further trained in the obstacle-based environment.

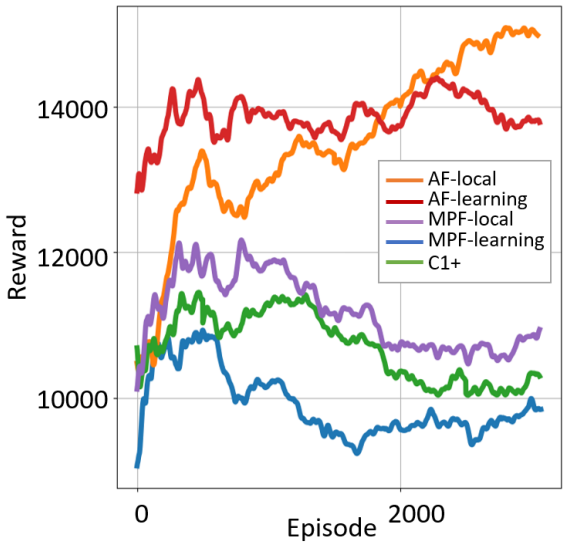


Figure 4.13: Learning process and evaluation scores comparison of the proposed method recorded in an obstacle-based training environment.

Training Score Comparison: According to the above task specification, we train the AF-learning, MPF-learning and C1+ method, and compare their training scores with the evaluation scores of AF-local, MPF-local in the same environment. It is noted that C1+ is the contact-aware version of C1 (see Fig. 4.12), which directly operates the tonic inputs of the CPG network given the full observation $\zeta_{1:19}$ from the environment. The C1 controller of all AF and MPF methods have been pre-trained in the obstacle-free environment and converged for goal-reaching tasks at the same level. The R2 controllers of AF-learning and MPF-learning methods are then trained in the obstacle-based goal tracking tasks for 3000 episodes till convergence, during which the NN parameters of their C1 controllers are fixed. The C1+ controller is first trained in the contact free environment, then transferred to the obstacle-based environment and is also trained for 3000 episodes.

From Fig. 4.13, it is observed that AF-learning method reaches the highest reward and is the only learning method that keeps improving during the learning process. Among the remaining methods, AF-local is the only method with an average reward closed to the AF-learning method. This result already shows the advantage of AF related method over the others. It is also noted that although MPF-local method is evaluated slightly better than the C1+ method, MPF-learning method cannot improve and converge to a higher score than its initial performance and end up converging to the lowest reward level. According to Remark 2, the bad performance of MPF-learning and MPF-local method is possibly due to the influence of $\dot{p}_i^e, \dot{p}_i^f$ in the MPF form Matsuoka oscillator, which makes the R2 RL controller more difficult to operate the sensory feedback signals of the CPG system.

4.4.3 Performance analysis in real robot experiments

In this section, we compare the performance of all five methods (mentioned in Section 4.4.2) in contact-aware soft snake robot locomotion tasks in the real world. Furthermore, we test the performance of the top two methods in more challenging obstacle-based environments.

Escaping experiment

In the real world contact-aware locomotion scenario, we design an escaping task to distinguish the strength and weakness of the contact-aware controllers (listed in Section 4.4.2).

Environment settings: The escaping task is designed for the following principles:

- The allocation of the obstacles should create narrow passage for the snake robot, with more contact opportunity and sharper tuning angle to test the overall capability of the controllers in escaping the jamming situations. In addition, the narrow space also limits the amplitude for regular body oscillation of the snake robot.

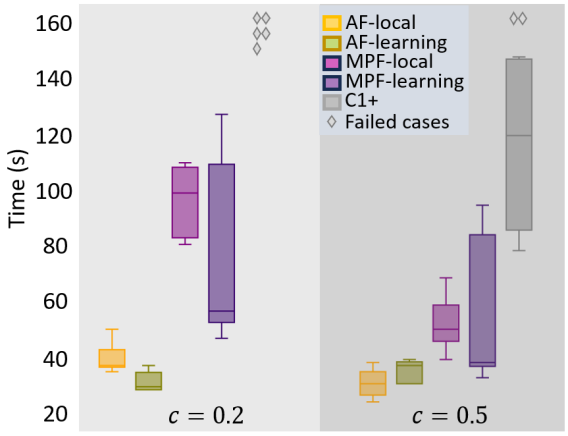


Figure 4.14: Statistics of escaping time of the proposed methods and the baseline.

- The obstacles should be allocated to obstruct the goal-reaching behavior. This is to test the coordination of the goal-reaching module (C1 controller) and contact reactive module (local reflexive or R2 method) in the compared controllers.
- The allocation of the obstacles should include the situation where only latter half links of the robot are stuck in the obstacles. This is for telling whether the controller relies mostly on its head steering to escape from the obstacles.
- The obstacles should be placed more densely in reality to test the generality of the compared controllers.

Based on the above principles, the obstacles in the escaping task are allocated as shown in Fig. 4.15. In the escaping task, the distance between every two obstacles are ranged from 85 mm to 150 mm. The robot is initially bending to its left, and placed at a position where 4 rigid bodies are in contact with the obstacles from different sides. The exit direction (left) of the obstacle region is intentionally set opposite to the goal direction (right). The distance between the exit of the obstacle region and the goal is 540 mm, which is close to the length of the snake robot.

Performance statistics: According to the free oscillation tonic input property of coefficient c in [44, Appendix B-D], as the value of c increases, it can increase the oscillation amplitude of the outputs of FOC-PPOC-CPG controller [44] and therefore improve its sim-to-real adaptability in the locomotion tasks. However, the value of c should not be larger since a higher free oscillation tonic input could decrease the goal tracking accuracy. As a result we separate the experiment into two groups with $c = 0.2$ and $c = 0.5$ respectively. For each value of c, we run five trials for each control method.[1]

[1]Performance videos for each method in the escaping task with different c values are available at: `https://shorturl.at/huBR1`.

We record and compare the finishing time of the escaping task of each controller. As shown in Fig. 4.14, AF-local and AF-learning methods outperform the other methods in the escaping task in both speed and stability. The increase of c from 0.2 to 0.5 does not significantly improve the performance of both AF methods. The main reason is that the sim-to-real adaptability of the AF methods is already good. MPF-learning method's average finishing time is shorter than MPF-local when $c = 0.2$, but is less stable than MPF-local, with the task finishing time varying from 42 seconds to 120 seconds. When $c = 0.5$, MPF-local method outperforms MPF-learning in both speed and stability. C1+ method cannot reach the goal in every trial when $c = 0.2$. However, with the increase of c to 0.5, the adaptability of C1+ controller is also improved so that it succeeds in a few of the trials. It is noted that, although MPF-learning converges to a lower reward level than C1+ method during the learning process (Fig. 4.13), its adaptability to the harder unseen task (in sim-to-real) is better than C1+ method. Generally, the results in Fig. 4.14 further verify the advantages of AF feedback Matsuoka oscillator predicted by Remark 2 and Proposition 3.

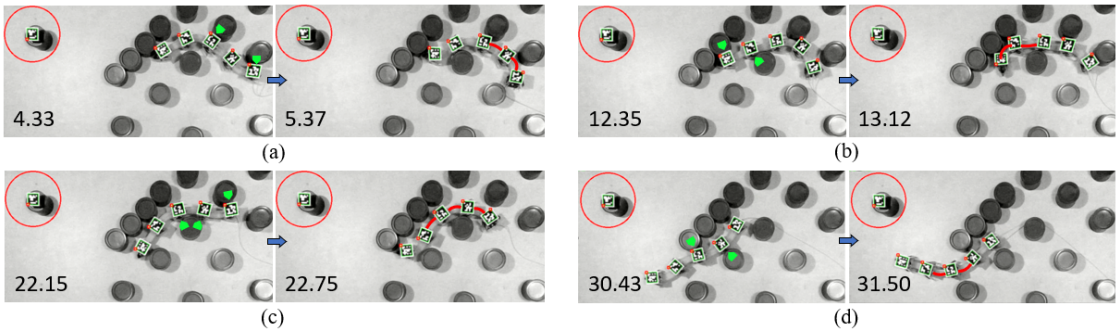


Figure 4.15: Sample screenshots of performance of the AF-local method in a goal oriented escaping task from the obstacles. Each pair of pictures shows the local reactive behavior of the soft snake robot before and after contacts.

Case analysis: We can further compare the sample trajectories of contact feedback signals and control commands for different control methods to analyze the special features of AF-local and AF-learning method (Fig. 4.16 and Fig. 4.17. It is noted that in the joint space figures, the positive and negative values are related to the extensor and flexor of the CPG system, as well as left and right of the snake body respectively.

First, we investigate the trajectory sample of AF-local method in the escaping task on the basis of AF-local mechanisms illustrated by Fig. 4.8. As shown in Fig. 4.16, we highlight four time intervals of the trajectory that present typical local reflexive control in the AF-local controller (the robot's body postures before and after contacts at intervals (a)∼(d) are captured by Fig. 4.15a∼Fig. 4.15d). Here we select time intervals (a) and (c) for discussion. At time interval (a) of Fig. 4.16, both CPG nodes at L3 and L4 are first influenced by the contact from the N_5^f

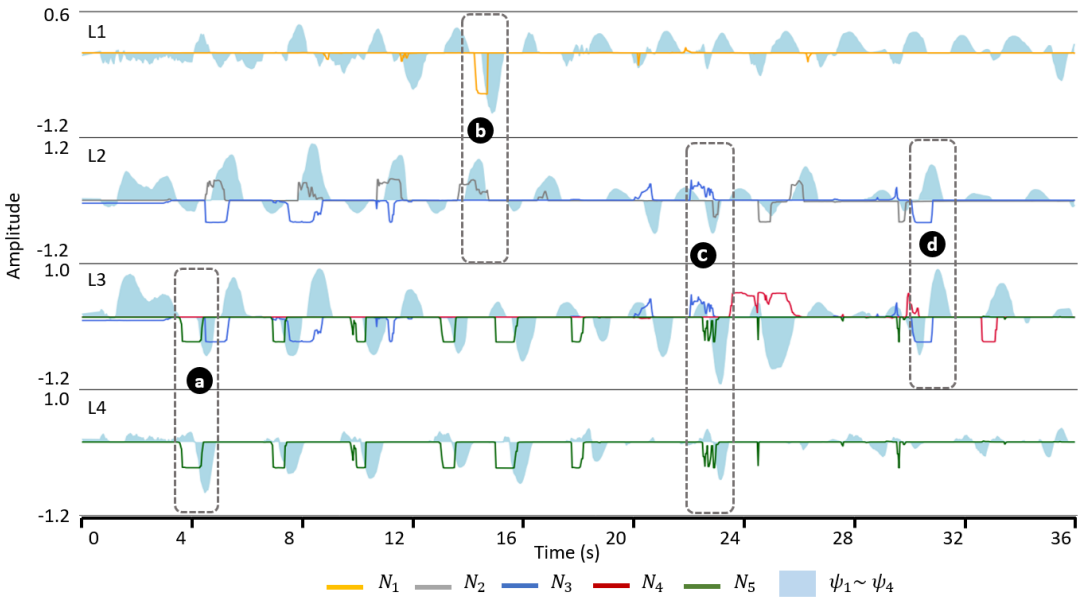


Figure 4.16: Recorded sensory input and CPG output of each body link of the soft snake robot controlled by the AF-local method in the goal-oriented escaping task.

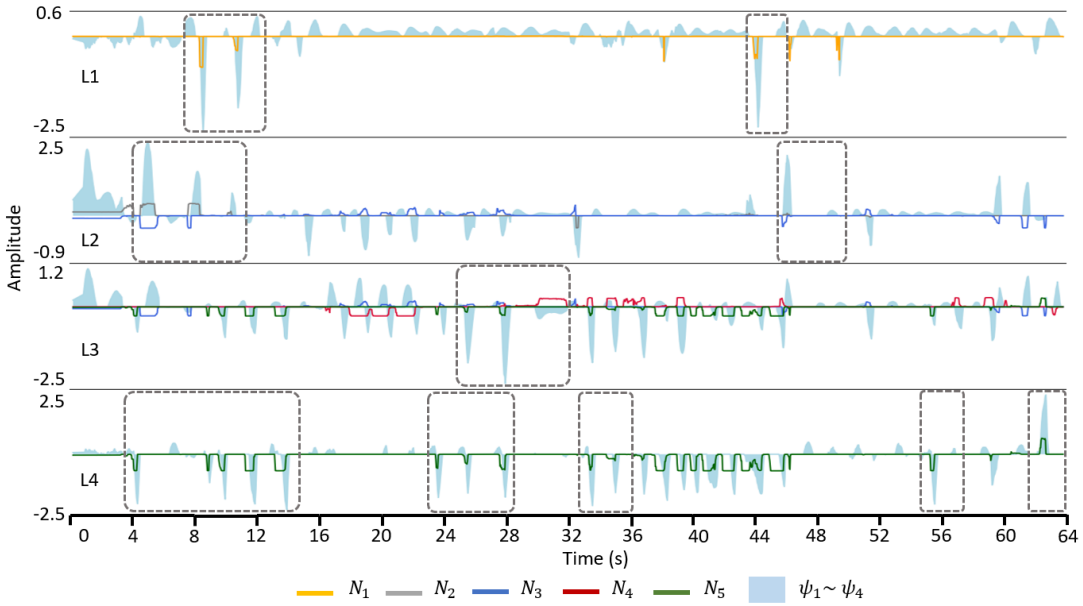


Figure 4.17: Recorded sensory input and CPG output of each body link of the soft snake robot controlled by the MPF-local method in the goal-oriented escaping task.

($N_5 < 0$), so the flexors of CPG nodes in L3 and L4 are activated to open the right valves of L3 and L4, which results in both links bend to the left in Fig. 4.15a. Then L3's CPG output is influenced by N_3^f, which will deactivate L3's CPG flexor and activates L3's CPG extensor. At time interval (c) of Fig. 4.16, CPG node at L2

influenced by the superposition of N_3^e and N_2^f, and is supposed to activate its flexor to open the right valve of L2, which results in L2 bend leftward in Fig. 4.15c. Due to the whole snake robot's tendency of turning right toward the target position, the amplitude of L2's CPG output signal is smaller than expected. The CPG node at L3 is influenced by the superposition of N_3^e and N_5^f, which also causes L3's flexor activated to bend to the left side. The CPG node at L4 is influenced by N_5^f, which activates L4's flexor and bend L4 soft chamber to the left. From the above behavior of the CPG outputs, we can verify that the experiment results match the local reflexive mechanism illustrated in Fig. 4.8.

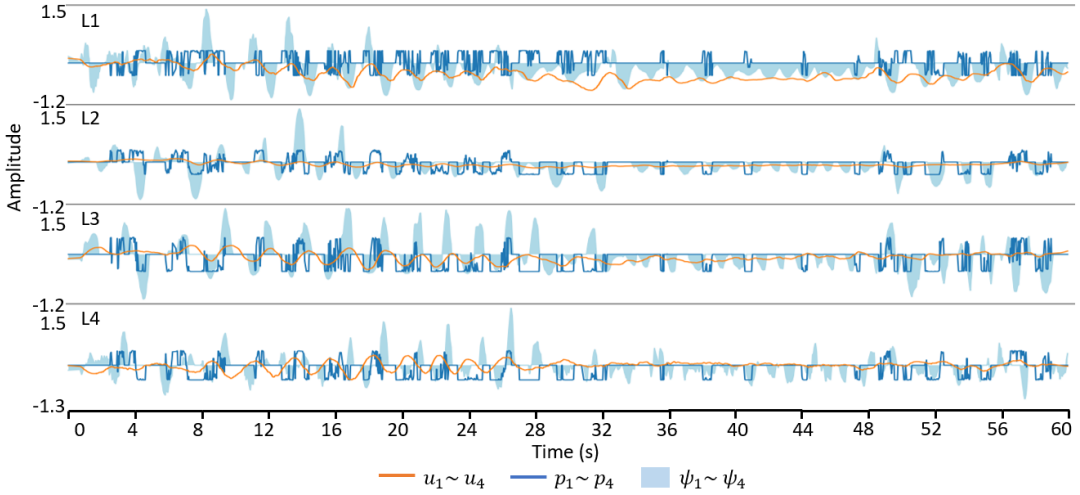


Figure 4.18: Recorded sensor feedback control signals, tonic input signals and CPG outputs of the soft snake robot controlled by MPF-learning method in the goal-oriented escaping task.

Similarly, from the sampled trajectories of MPF-local method in Fig. 4.17, we can conclude that the sensory inputs and the CPG outputs for all body links satisfy the local reflexive mechanism determined by Fig. 4.8. However, when comparing Fig. 4.17 to the AF-local behavior in Fig. 4.16, the MPF-local controller produces significantly large overshoots even when the contact signals are small. The recovery delay is also more frequently observed in the trajectories of MPF-method, such that the MPF-local controller always takes longer time to recover to its goal-reaching oscillation after the contact signals disappear. These observations further verifies Remark 2 and its derivations, that the first order derivative term $\dot{p}_i$ will seriously interfere with the control of MPF form CPG system when the contact feedback signals are densely emerging, and therefore hinder the performance of contact-aware locomotion.[2]

The issue of the output wave response can also be observed in MPF-learning (Fig. 4.18). With more chaotic sensory feedback signals from the RL event-based

[2]The locomotion performance of MPF-local method can be observed in videos `https://shorturl.at/fAGJN` and `https://shorturl.at/AORW9`.

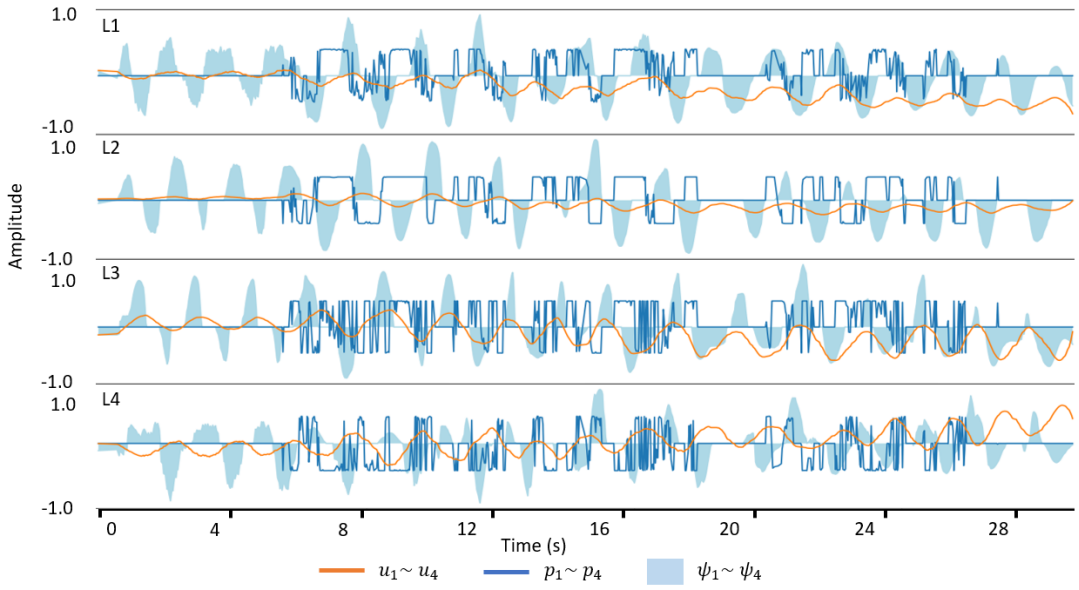


Figure 4.19: Recorded sensor feedback control signals, tonic input signals and CPG outputs of the soft snake robot controlled by AF-learning method in the goal-oriented escaping task.

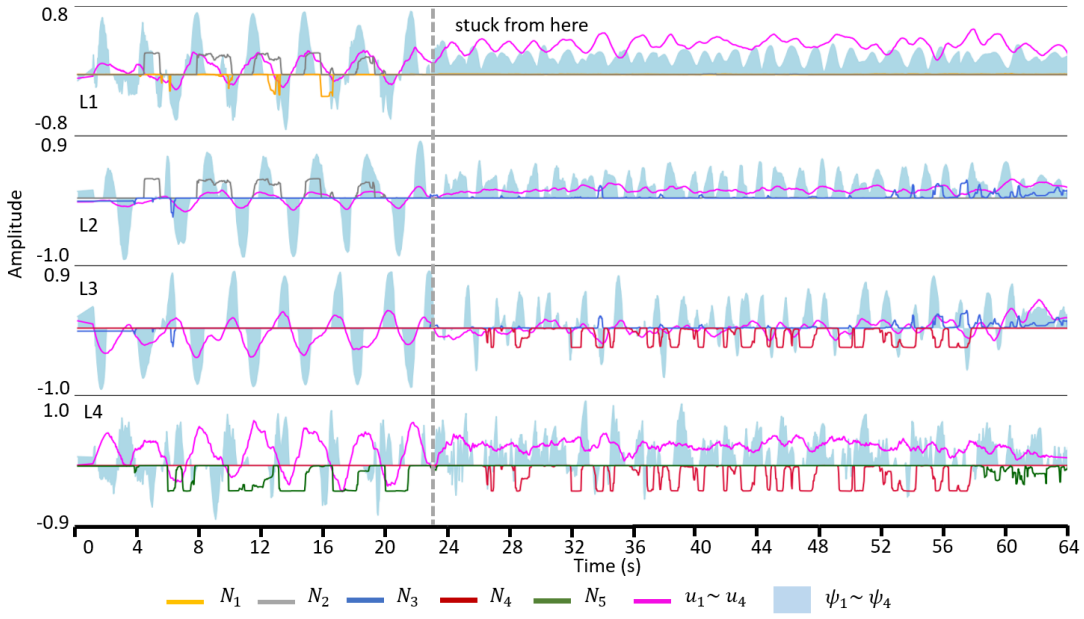


Figure 4.20: Recorded sensor feedback control signals, tonic input signals and CPG outputs of the soft snake robot controlled by C1+ method in the goal-oriented escaping task.

controller R2, the CPG outputs also shows disturbed behaviors, which significantly slow down the locomotion in the escaping task. It is worth noting that, due to the

black-box property of learning-based method, both AF-learning and MPF-learning methods send more complex sensory feedback signals to their CPG systems. However, we can still observe clear and coordinated oscillation in the sample of AF-learning trajectory in Fig. 4.19. This is also because AF series methods are free from the disturbances of the $\dot{p}_i$ term.

As shown in Fig. 4.20, C1+ method fails to learn to react to the sensory inputs. When the target moving direction of the robot is blocked by the obstacles, C1+ controller cannot pull the soft snake robot out from the jam and skirt the obstacles.

In conclusion, the results and analyses in the escaping tasks show strong evidence of the advantage of AF-local and AF-learning controllers in the contact-aware locomotion of soft snake robot.

General performance of AF series methods in difficult contact-aware locomotion tasks

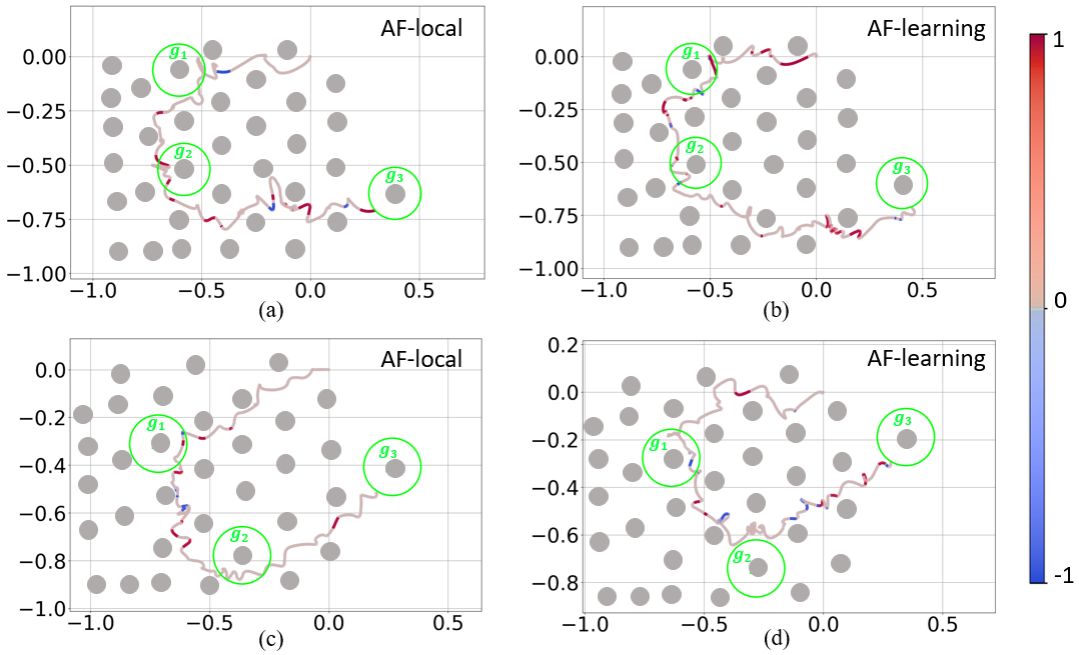


Figure 4.21: Sample way-point trajectories followed by (a) AF-local controller in square trajectory, (b) AF-learning controller in square trajectory, (c) AF-local controller in triangle trajectory and (d) AF-learning controller in triangle trajectory. The distribution of reactive signals along the trajectory to the CPG-controlled actuators from head joint of the robot are visualized.

In this section, we apply the two methods with the best performance in the escaping task to a more complicated environment with multiple targets to traverse in a dense obstacle array, with a lot of detours.

Due to the randomness of contacts in such complicated goal tracking tasks in the obstacle maze, it is harder for any methods to concentrate around certain trajectories. Without pre-planning of the path, it is possible for the same control method to traverse the target goals in different paths. The paths with fast and slow performances for each task (square and triangle) are shown in video "square.mp4" and "triangle.mp4" [3].

In Fig. 4.21, we plot the recorded paths of AF-local and AF-learning methods traversing three targets allocated in square shape and triangle shape respectively in 2 dimension space. The color map on the paths show the reactive commands of L1 actuator sent by both control methods. Both methods have achieved decent performance in the harder tasks.

4.5 Conclusion

This chapter establishes a novel framework for the contact-aware intelligent locomotion control of a soft snake robot. This framework is an organic integration of hardware design, feedback mechanism study through bio-inspired CPG system and implementation of sensory feedback control schemes. The proposed approaches are able to achieve promising performance in both simulation and real robot in several contact-aware locomotion tasks with densely allocated obstacles. Our novel method tackles jointly contact sensing and contact reacting controls in the contact-aware locomotion control of the soft snake robot. Our work brings inspiration for both distributed reflexive method and learning-based control method and forms the basis to design and control soft snake robots that can pass through environments with unpredictable and dense obstacles.

[3]The videos are available at`https://shorturl.at/huBR1`

Chapter 5

Final Conclusion

Before closing this thesis, I would like to review my understanding and contribution to the learning-based Matsuoka CPG system as a bio-inspired robot locomotion controller from a theoretical perspective.

5.1 Significance of the Original Contribution

Theoretical analysis of Matsuoka CPG's steering maneuverability: (Chapter 3) The highlight of this approach is that we reasonably related the special properties of Matsuoka CPG's tonic input to the actions of any RL agent. What we have achieved here has refreshed a preconception in bio-inspired controller research that "The neural oscillators with implicit manipulator on oscillation patterns (including Matsuoka oscillator) are too complex and usually harder to control than the neural oscillators with explicit controllable phase coefficients". Standing on the shoulder of Doctor Matsuoka, our analysis has leveraged several important properties of the Matsuoka oscillator for steering and speed control of robot locomotion tasks. These properties bridge the gap between implicit oscillation patterns and control variables of the Matsuoka CPGs. We also highlight the significance of the CPG maneuverability study in bio-inspired controller development.

Theoretical analysis of free-response oscillation constraint (FOC) of the Matsuoka CPG system for sim-to-real transfer: (Chapter 3) We investigate the transient property of the Matsuoka Oscillator from free-response oscillation to forced-response oscillation. From the motion control perspective, this property forces the RL controller to learn to either increase the oscillation amplitude of tonic inputs or adjust the oscillation frequency to synchronize the natural frequency of the CPG system properly to acquire better control of the robot. This is beneficial for the sim-to-real transferring because the key is to adapt to the changes of ground friction through adjustment of locomotion patterns. More interestingly, this approach opens up a new track for the learning process of an RL agent. Different from the

reward shaping approaches or imitation learning that uses gradient descent to guide the evolution of the RL agent, the neural oscillator requires the RL agent to learn the entrainment with its natural oscillation patterns to own the control dominance of the CPG system.

Development of sensory feedback Matsuoka CPG system for contact-aware locomotion: (Chapter 4) We develop a novel sensory feedback mechanism of the Matsuoka central pattern generator (CPG) network. This mechanism allows the Matsuoka CPG system to work like a "spine cord" in the whole contact-aware control scheme, which simultaneously takes the top-down stimuli including tonic input signals from the "brain" (a goal-tracking locomotion controller) and bottom-up sensory feedback signals from the "reflex arc" (the contact reactive controller), and generate rhythmic signals to effectively actuate the soft snake robot to slither through densely allocated obstacles with sensitive and coordinated oscillation patterns. Based on the in-depth understanding of the Matsuoka CPG system, our approach has made a breakthrough in processing sensory feedback efficiently with the Matsuoka oscillator.

5.2 Future Extensions

Although we have been investigating the generality of the proposed control scheme on different robotic systems and obtained promising early results, it remains challenging to initialize the parameters of the Matsuoka CPGs for specific robots. Moreover, the parameter optimizer for the Matsuoka CPG system should be improved to be computationally cheaper and updated online. It is also interesting to investigate distributed control designs that can scale to high-dimensional soft snake robots or other biomimetic robotic systems.

For the future study of contact-aware locomotion, the contact module and design can be enhanced with the consideration of more advanced materials and structures to improve contact sensitivity and locomotion efficiency for more challenging environments (e.g. underwater contact or uneven and compliant terrains). The tactile information and the locomotion gait can also be enriched by increasing the number of body links of the snake robot. More investigation is also needed to understand the influence of couplings among primitive AF form feedback Matsuoka oscillators in the CPG network so that the variation of couplings can be utilized to improve the performance of the contact-aware locomotion controller. In addition, one limitation is that the proposed learning-based controller is mainly reactive and may not learn to leverage obstacles to aid the locomotion without trajectory planning. Such a behavior may be achieved by combining depth visual information and tactile information of the obstacles to the deep reinforcement learning controller in the PPOC-CPG framework.

Chapter 6

Appendix

6.1 Data

This section includes the parameter configuration of the Matsuoka CPG network and the hyper parameter setting of domain randomization for the experiment.

Table 6.1: Parameter Configuration of the Matsuoka CPG Net Controller for the Soft Snake Robot.

Parameters	**Symbols**	**Values**
Amplitude ratio	a_ψ	2.0935
$*$Self-inhibition weight	b	**10.0355**
$*$Discharge rate	τ_r	**0.7696**
$*$Adaptation rate	τ_a	**1.7728**
Period ratio	K_f	1.0
Mutual inhibition weights	a_i	4.6062
Coupling weights	w_{ij}	8.8669
	w_{ji}	0.7844

6.2 Preliminary

6.2.1 Describing function analysis of the Matsuoka Oscillator

According to Fourier theory, we denote the main sinusoidal and constant component in Fourier expansion of the vanilla state $x(t)$ as

$$x_{\mathcal{F}}(t) = A\cos(\omega t) + d = A(\cos(\omega t) + r), \tag{6.1}$$

Table 6.2: Curriculum settings

Levels	**Distance range** (m)	**Turning angles** $(^\circ)$	**Goal radius** (m)
1	$1.2 \sim 1.5$	$-10 \sim 10$	0.5
2	$1.2 \sim 1.5$	$-10 \sim 10$	0.4
3	$1.2 \sim 1.5$	$-15 \sim 15$	0.3
4	$1.2 \sim 1.5$	$-20 \sim 20$	0.25
5	$1.2 \sim 1.5$	$-30 \sim 30$	0.2
6	$1.0 \sim 1.5$	$-40 \sim 40$	0.18
7	$1.0 \sim 1.5$	$-45 \sim 45$	0.15
8	$1.0 \sim 1.5$	$-50 \sim 50$	0.12
9	$0.9 \sim 1.5$	$-60 \sim 60$	0.09
10	$0.9 \sim 1.5$	$-60 \sim 70$	0.06
11	$0.9 \sim 1.5$	$-70 \sim 70$	0.05
12	$0.8 \sim 1.5$	$-80 \sim 80$	0.05

Table 6.3: Domain randomization parameters

Parameter	**Low**	**High**
Ground friction coefficient	0.1	1.5
Wheel friction coefficient	0.05	0.10
Rigid body mass (kg)	0.035	0.075
Tail mass (kg)	0.065	0.085
Head mass (kg)	0.075	0.125
Max link pressure (psi)	5	12
Gravity angle (rad)	-0.001	0.001

where $r = d/A, r \in R$ is the ratio of bias to the amplitude of the signal. We assume $x_{\mathcal{F}}(t)$ only contains cosine term for simplicity. And because this chapter only discusses amplitude and bias properties of the signals, such simplification will not affect the following derivations. We use $z_{\mathcal{F}}(t) = g(x_{\mathcal{F}}(t)) - \epsilon(t) = \max(x_{\mathcal{F}}(t), 0) - \epsilon(t)$ to represent the main sinusoidal property of $z(t) = g(x(t)) = \max(x(t), 0)$. In a single period $[-\frac{\pi}{\omega}, \frac{\pi}{\omega}]$,

$$g(x_{\mathcal{F}}(t)) = \begin{cases} 0 \quad \text{elsewhere} \\ A(\cos(\omega t) + r) \quad t \in [-\frac{\arccos(-r)}{\omega}, \frac{\arccos(-r)}{\omega}] \end{cases}.$$

Using Fourier expansion, the output state $z_{\mathcal{F}}(t)$ can also be expressed as:

$$\begin{aligned} g(x_{\mathcal{F}}(t)) &= g(A(\cos(\omega t) + r)) \\ &= Ag(\cos(\omega t) + r) \\ &= A(K(r)\cos(\omega t) + L(r)) + \epsilon(t) \\ &= z_{\mathcal{F}}(t) + \epsilon(t) \quad (n \geq 1), \end{aligned} \tag{6.2}$$

such that $z_{\mathcal{F}}(t) = A(K(r)\cos(\omega t) + L(r))$, where

$$K(r) = \begin{cases} 0 & (r < -1) \\ \frac{1}{\pi}(r\sqrt{1-r^2} - \cos^{-1}(r)) + 1 & (-1 \le r \le 1) \\ 1 & (r > 1), \end{cases} \tag{6.3}$$

and

$$L(r) = \begin{cases} 0 & (r < -1) \\ \frac{1}{\pi}(\sqrt{1-r^2} - r\cos^{-1}(r)) + r & (-1 \le r \le 1) \\ r & (r > 1). \end{cases} \tag{6.4}$$

The derivation of $K(r)$ and $L(r)$ are based on Fourier series analysis (see Appendix 6.2.2). Both $K(r)$ and $L(r)$ are constrained by $-1 \le r \le 1$ for $x_{\mathcal{F}}(t)$ to be non-negative in the period $[-\frac{\pi}{\omega}, \frac{\pi}{\omega}]$.

Function $\epsilon(t)$ is the summation of all remaining high frequency terms in the Fourier expansion of $z_{\mathcal{F}}(t)$.

When $t \in [-\frac{\arccos(-r)}{\omega}, \frac{\arccos(-r)}{\omega}]$, $z_{\mathcal{F}}(t) = x_{\mathcal{F}}(t)$, we have

$$\begin{aligned} \epsilon(t) &= x_{\mathcal{F}}(t) - A\{K(r)\cos(\omega t) + L(r)\} \\ &= -\frac{A}{\pi}\{(r\sqrt{1-r^2} - \arccos r)\cos(\omega t) + \sqrt{1-r^2} - r\arccos r\}. \end{aligned}$$

When $t \in [-\frac{\pi}{\omega}, -\frac{\arccos(-r)}{\omega}] \cup [\frac{\arccos(-r)}{\omega}, \frac{\pi}{\omega}]$, $z_{\mathcal{F}}(t) = 0$, we have

$$\begin{aligned} \epsilon(t) &= 0 - A\{K(r)\cos(\omega t) + L(r)\} \\ &= -A\{[\frac{1}{\pi}(r\sqrt{1-r^2} - \arccos r) + 1]\cos(\omega t) - \frac{1}{\pi}(\sqrt{1-r^2} - r\arccos r) - r\}. \end{aligned}$$

Then we can numerically calculate the bound of $\epsilon(t)$ for certain A and ω. For instance, if $A = 1$ and $\omega = 1$, we have

$$\begin{aligned} &\epsilon(t) \in [0, 0.2055] \text{ when } t \in [-\frac{\arccos(-r)}{\omega}, \frac{\arccos(-r)}{\omega}] \\ &\epsilon(t) \in [-2.0009, 0] \text{ when } t \in [-\frac{\pi}{\omega}, -\frac{\arccos(-r)}{\omega}] \cup [\frac{\arccos(-r)}{\omega}, \frac{\pi}{\omega}]. \end{aligned}$$

6.2.2 Calculation of $K(r)$ and $L(r)$

Given $x_{\mathcal{F}}(t) = A(\cos(\omega t) + r)$ as an even function, the general Fourier expansion of $z_{\mathcal{F}}(t) = g(x_{\mathcal{F}}(t))$ is:

$$z_{\mathcal{F}}(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega t) = \frac{1}{2}a_0 + a_1 \cos(\omega t) + \epsilon(t). \tag{6.5}$$

where

$$\begin{aligned} a_0 &= \frac{1}{\pi}\int_{-\pi}^{\pi} g(A(\cos(\omega t) + r))dt \\ a_1 &= \frac{1}{\pi}\int_{-\pi}^{\pi} g(A(\cos(\omega t) + r))\cos(\omega t)dt. \end{aligned}$$

In this case, both the bias a_0 and the amplitude a_1 become functions of r. Combining with (6.2), we use $AK(r)$ to represent a_1 and $AL(r)$ to represent a_0, which are calculated as follows:

$$K(r) = \frac{a_1}{A} = \frac{\omega}{\pi}\int_{-\pi/\omega}^{\pi/\omega} g((\cos(\omega\tau) + r))\cos(\omega\tau)d\tau.$$

Let $t = \omega\tau$, we have

$$\begin{aligned} K(r) &= \frac{1}{\pi}\int_{-\pi}^{\pi} g((\cos(t) + r))\cos(t)dt \\ &= \frac{1}{\pi}\int_{-\cos^{-1}(-r)}^{\cos^{-1}(-r)} (\cos(t) + r)\cos(t)dt \\ &= \frac{1}{\pi}(r\sqrt{1-r^2} - \cos^{-1}(r)) + 1 \quad (-1 \le r \le 1), \end{aligned}$$

and

$$\begin{aligned} L(r) = \frac{a_0}{A} &= \frac{1}{\pi}\int_{-\pi}^{\pi} g(\cos(t) + r)dt \\ &= \frac{1}{\pi}\int_{-\cos^{-1}(-r)}^{\cos^{-1}(-r)} (\cos(t) + r)dt \\ &= \frac{1}{\pi}(\sqrt{1-r^2} - r\cos^{-1}(r)) + r \quad (-1 \le r \le 1). \end{aligned}$$

6.2.3 Derivation of K_n

Based on (3.12), we first set $x_i(t) = x_i^e(t) - x_i^f(t)$, $y_i(t) = y_i^e(t) - y_i^f(t)$, $z_i(t) = z_i^e(t) - z_i^f(t)$, $u_i(t) = u_i^e(t) - u_i^f(t)$. Then by taking subtraction between flexor and

extensor in (3.12) and neglect phase related coupling terms from other primitive CPGs, we have

$$K_f\tau_r\frac{d}{dt}(x_i^e - x_i^f) = -(x_i^e - x_i^f) - a(z_i^f - z_i^e) - b(y_i^e - y_i^f) + (u_i^e - u_i^f)$$
$$K_f\tau_a\frac{d}{dt}(y_i^e - y_i^f) = (z_i^e - z_i^f) - (y_i^e - y_i^f),$$

which can be simplified to

$$\begin{aligned} K_f\tau_r\frac{d}{dt}x_i &= -x_i + az_i - by_i + u_i \\ K_f\tau_a\frac{d}{dt}y_i &= z_i - y_i. \end{aligned} \tag{6.6}$$

If x_i^e and x_i^f satisfy the *perfect entrainment assumption*, such that the amplitude $A_{x_i^e} = A_{x_i^f} = A_x$, and the bias $r_{x_i^e} = r_{x_i^f} = r_x$, and the phase delay between x_i^e and x_i^f is $\frac{\pi}{\omega}$ (half of the period). Then we have $z_{\mathcal{F}_i} = K(r_x)x_{\mathcal{F}_i}$. Similar to the notation in Appendix 6.2.1, the marker $\mathcal{F}_i$ indicates the fundamental sinusoidal and constant component in Fourier expansion of the corresponding variable. Let $K_f = 1$, (6.6) can be further simplified to

$$\begin{aligned} \tau_r\frac{d}{dt}x_{\mathcal{F}_i} + x_{\mathcal{F}_i} &= aK(r_x)x_{\mathcal{F}_i} - by_{\mathcal{F}_i} + u_{\mathcal{F}_i} \\ \tau_a\frac{d}{dt}y_{\mathcal{F}_i} + y_{\mathcal{F}_i} &= K(r_x)x_{\mathcal{F}_i}. \end{aligned} \tag{6.7}$$

Next, an ordinary differential equation can be obtained by merging the two equations in (6.7) as,

$$\tau_r\tau_a\frac{d^2}{dt^2}x_{\mathcal{F}_i} + (\tau_r + \tau_a - \tau_a aK(r_x))\frac{d}{dt}x_{\mathcal{F}_i} + ((b-a)K(r_x) + 1)x_{\mathcal{F}_i} = \tau_a\frac{d}{dt}u_{\mathcal{F}_i} + u_{\mathcal{F}_i}. \tag{6.8}$$

When the system is harmonic, the coefficient of the first-order derivative of (6.8) becomes zero, then

$$K(r_x) = \frac{\tau_r + \tau_a}{\tau_a a} \triangleq K_n. \tag{6.9}$$

Coefficient K_n is a special case of $K(r_x)$ in the harmonic Matsuoka oscillator.

6.2.4 Amplitude Threshold of Transition from Free Oscillation to Forced Entrainment

In order to extract the free-response oscillation component, let $\tilde{u} = u - 1$, $\tilde{c} = c + 1$ then (3.12) is equivalent to

$$\begin{aligned}
K_f \tau_r \dot{x}_i^e &= -x_i^e - a z_i^f - b y_i^e - \sum_{j=1}^{N} w_{ji} y_j^e + \tilde{u}_i^e + \tilde{c}, \\
K_f \tau_a \dot{y}_i^e &= z_i^e - y_i^e, \\
K_f \tau_r \dot{x}_i^f &= -x_i^f - a z_i^e - b y_i^f - \sum_{j=1}^{N} w_{ji} y_j^f + \tilde{u}_i^f + \tilde{c}, \\
K_f \tau_a \dot{y}_i^f &= z_i^f - y_i^f,
\end{aligned} \tag{6.10}$$

Since $u_i^q \in [0, 1]$ (for $q \in \{e, f\}$) and $c \geq 0$, we have $\tilde{u}_i^q \in [-1, 0]$ (for $q \in \{e, f\}$), and $\tilde{c} \geq 1$. Now $\tilde{c}$ becomes the only positive term in the primitive Matsuoka system in (6.10). According to Matsuoka's derivation in [53, (26)], from (6.10), the free-response oscillation amplitude of the Matsuoka oscillator can be written as

$$A_n = \frac{\tilde{c}}{K^{-1}(K_n) + (a + b) L(K^{-1}(K_n))}. \tag{6.11}$$

Assume the fundamental harmonic component of the vanilla action signal α_i generated by RL policy has the form: $\alpha_{\mathcal{F}_i} = A \cos(\omega t)$.

Then substitute $\alpha_{\mathcal{F}_i}$ into (3.16), we have

$$u_{\mathcal{F}_i}^e \approx \frac{1}{1 + e^{-A \cos(\omega t)}}, \ \tilde{u}_{\mathcal{F}_i}^e \approx \frac{1}{1 + e^{-A \cos(\omega t)}} - 1. \tag{6.12}$$

Because the sigmoid function in $\tilde{u}_{\mathcal{F}_i}^e$ is monotonically increasing with $\alpha_{\mathcal{F}_i}$, the frequency of $\tilde{u}_{\mathcal{F}_i}^e$ is the same as the frequency of $\alpha_{\mathcal{F}_i}$. The amplitude of $\tilde{u}_{\mathcal{F}_i}^e$ is

$$A_{\tilde{u}} = \frac{\max_t (\tilde{u}_{\mathcal{F}_i}^e(t)) - \min_t (\tilde{u}_{\mathcal{F}_i}^e(t))}{2} = \frac{1}{2} \frac{e^A - 1}{e^A + 1}. \tag{6.13}$$

And the bias of $\tilde{u}_{\mathcal{F}_i}^e$ can be calculated as

$$r_{\tilde{u}} = \frac{\max_t (\tilde{u}_{\mathcal{F}_i}^e(t)) + \min_t (\tilde{u}_{\mathcal{F}_i}^e(t))}{2} = -\frac{1}{2}. \tag{6.14}$$

It is noted that $\tilde{u}_{\mathcal{F}_i}^e$ and $\tilde{u}_{\mathcal{F}_i}^f$ are complementary to each other by Definition 1. Thus $\tilde{u}_{\mathcal{F}_i}^e$ and $\tilde{u}_{\mathcal{F}_i}^f$ share the same amplitude and bias.

By taking time average of all variables in (6.10) and ignoring the coupling term from other primitive Matsuoka oscillator nodes, we have the equation of the ampli-

tude of the inner state $x^q_{\mathcal{F}_i}$ ($q \in \{e, f\}$) as

$$A_x[r_x + (a+b)L(r_x)] = \tilde{c} + r_{\tilde{u}} = \tilde{c} - \frac{1}{2}. \tag{6.15}$$

Next, since (6.10) can be reduced to

$$\begin{aligned} \tau_r \frac{d}{dt} x_{\mathcal{F}_i} + x_{\mathcal{F}_i} &= aK(r_x)x_{\mathcal{F}_i} - by_{\mathcal{F}_i} + \tilde{u}_{\mathcal{F}_i} \\ \tau_a \frac{d}{dt} y_{\mathcal{F}_i} + y_{\mathcal{F}_i} &= K(r_x)x_{\mathcal{F}_i}, \end{aligned} \tag{6.16}$$

where $\tilde{u}_{\mathcal{F}_i} = \tilde{u}^e_{\mathcal{F}_i} - \tilde{u}^f_{\mathcal{F}_i}$. We derive the describing function from $\tilde{u}_{\mathcal{F}_i}(t)$ to $x_{\mathcal{F}_i}(t)$. Applying the Laplace transform to (6.16), we have

$$\begin{aligned} G(s, A) &= \frac{1}{\tau_r s + 1 - K(r_x)(a - \frac{b}{\tau_a s + 1})} \\ &= \frac{\tau_a s + 1}{1 + (\tau_r \tau_a \omega_n^2 - 1)\frac{K(r_x)}{K_n} + \tau_r \tau_a s^2 + (K_n - K(r_x))\tau_a a s}. \end{aligned} \tag{6.17}$$

More precisely, the frequency transfer function is

$$G(\omega, A) = \frac{j\tau_a \omega + 1}{1 + (\tau_r \tau_a \omega_n^2 - 1)\frac{K(r_x)}{K_n} - \tau_r \tau_a \omega^2 + j(K_n - K(r_x))\tau_a a \omega} \tag{6.18}$$

where $\omega_n = \frac{1}{\tau_a}\sqrt{\frac{(\tau_a + \tau_r)b}{\tau_r a} - 1}$. Because the gain from $\tilde{u}_{\mathcal{F}_i}(t)$ to $x_{\mathcal{F}_i}(t)$ is $|G(\omega, A)|$, the amplitude of $x_{\mathcal{F}_i}(t)$ is given by $|G(\omega, A)|A_u$. Since the amplitude of $x_{\mathcal{F}_i}(t)$ is twice of A_x, and the amplitude of $u_{\mathcal{F}_i}(t)$ is twice of A_u, we have the relation between A_x and A_u expressed as

$$A_x = |G(\omega, A)|A_{\tilde{u}} = |G(\omega, A)|A_u. \tag{6.19}$$

Given (6.8), $(\tau_r + \tau_a - \tau_a a K(r_x))$ is the coefficient of first-order differential variable, also known as damping coefficient. When $K(r_x) = K_n = \frac{\tau_r + \tau_a}{\tau_a a}$, the original oscillation system is harmonic. For the damped oscillation system, the damping coefficient should be positive such that $K(r_x) < K_n$, or equivalently, $\frac{K(r_x)}{K_n} < 1$. In this situation, there will be only forced-response oscillation, and all free-response oscillations diminish due to the positive damping. From (6.3) and (6.4) we know both $K(r)$ and $L(r)$ are monotonic, and therefore $K^{-1}(r)$ and $L^{-1}(r)$ are monotonic as well. When $K(r_x) < K_n$,

$$A_n = \frac{\tilde{c}}{K^{-1}(K_n) + (a+b)L(K^{-1}(K_n))} < \frac{\tilde{c}}{r_x + (a+b)L(r_x)}, \tag{6.20}$$

that is

$$r_x + (a+b)L(r_x) < \frac{\tilde{c}}{A_n}. \tag{6.21}$$

From the other end, let $K_x \triangleq K(r_x)$, we have

$$\begin{aligned} A_x &= |G(\omega, A)|A_u \\ &= \frac{A_u\sqrt{\tau_a^2\omega^2+1}}{\sqrt{[1+(\tau_r\tau_a\omega_n^2-1)\frac{K_x}{K_n}-\tau_r\tau_a\omega^2]^2+(K_n-K_x)^2\tau_a^2a^2\omega^2}} \\ &\triangleq \frac{A_u\sqrt{\tau_a^2\omega^2+1}}{\sqrt{[1+(\tau_r\tau_a\omega_n^2-1)U-\tau_r\tau_a\omega^2]^2+K_n^2(1-U)^2\tau_a^2a^2\omega^2}}, \end{aligned} \tag{6.22}$$

where $U \triangleq \frac{K(r_x)}{K_n}$, and $U \subseteq (0,1]$. Next, define a function $Q(U)$ as

$$Q(U) \triangleq [(\tau_r\tau_a\omega_n^2-1)U-(\tau_r\tau_a\omega^2-1)]^2+K_n^2(1-U)^2\tau_a^2a^2\omega^2. \tag{6.23}$$

When $\omega > \omega_n$ and $\tau_r\tau_a\omega_n^2 - 1 > 0$, or $\omega < \omega_n$ and $\tau_r\tau_a\omega_n^2 - 1 < 0$,

$$Q_{min}(U) = Q(1) = \tau_r^2\tau_a^2(\omega^2-\omega_n^2)^2. \tag{6.24}$$

Thus when $U \subseteq (0,1]$ is satisfied, we have

$$A_x < A_u\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega^2-\omega_n^2|}. \tag{6.25}$$

Combining (6.25), (6.21) and (6.15), we have

$$A_u\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega_n^2-\omega^2|}\frac{\tilde{c}}{A_n} > \tilde{c}-\frac{1}{2} > \tilde{c}-1. \tag{6.26}$$

Thus we have

$$A_u > \frac{\tilde{c}-1}{\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega_n^2-\omega^2|}\frac{\tilde{c}}{A_n}} = \frac{c}{\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega_n^2-\omega^2|}\frac{c+1}{A_n}} \triangleq A_0(c,\omega). \tag{6.27}$$

Substitute A_n in the above equation with its approximation in [53, (30)], we have

$$A_0(c,\omega) \approx \frac{c}{\frac{\sqrt{\tau_a^2\omega^2+1}}{\tau_r\tau_a|\omega_n^2-\omega^2|}(2K_n-1+\frac{2}{\pi}(a+b)\sin^{-1}(K_n))} \tag{6.28}$$

Since $c \geq 0$, when ω is fixed, A_0 linearly increases with c.

6.3 Theory

6.3.1 Proof of Proposition 1

Proof. As seen in (6.8), when u_i^e and u_i^f of the $i-$th oscillator satisfy constant constraints in Problem 1, the tonic inputs become time-invariant, such that $\frac{d}{dt}u_i(t) = 0$. If the oscillation is harmonic ($K(r_x) = K_n$), then (6.8) can be rewritten as

$$\tau_r\tau_a\frac{d^2}{dt^2}x_i + (\tau_r + \tau_a - \tau_a a K_n)\frac{d}{dt}x_i + ((b-a)K_n + 1)x_i = 2u_i^e - 1, \tag{6.29}$$

Then the above equation can be interpreted as a non-homogeneous spring-damper system with a constant load. By setting $\tilde{x}_i \triangleq x_i - (2u_i^e - 1)/((b-a)K_n + 1)$, and substitute x_i with $\tilde{x}_i$ in (6.29), we can obtain its homogeneous form as:

$$\tau_r\tau_a\frac{d^2}{dt^2}\tilde{x}_i + (\tau_r + \tau_a - \tau_a a K_n)\frac{d}{dt}\tilde{x}_i + ((b-a)K_n + 1)\tilde{x}_i = 0. \tag{6.30}$$

Here $\tilde{x}_i$ is the unbiased variable of x_i, and thus the bias of x_i naturally becomes

$$\text{bias}(x_i) = \frac{2u_i^e - 1}{(b-a)K_n + 1} = \frac{1}{(b-a)K_n + 1}\text{bias}(u_i). \tag{6.31}$$

Since z_i and x_i are entrained (Definition 2), $z_i = z_i^e - z_i^f = g(x_i^e) - g(x_i^f) = K_n x_i$, we have

$$\text{bias}(z_i) = K_n\text{bias}(x_i) = \frac{K_n}{(b-a)K_n + 1}\text{bias}(u_i). \tag{6.32}$$

□

6.3.2 Applicable range of Proposition 1

Let $x_i^e < 0$, $x_i^f > 0$, from (3.12), we have

$$z_i^e = \max(x_i^e, 0) = 0,\ z_i^f = \max(x_i^f, 0) = x_i^f.$$

Thus

$$z_i = z_i^e - z_i^f = -x_i^f.$$

Since u_i^e and u_i^f are constants in Proposition 1, we have

$$x_i^e + a x_i^f = u_i^e + c \tag{6.33}$$

$$x_i^f + b x_i^f = u_i^f + c. \tag{6.34}$$

Let $u_i = u_i^e - u_i^f, x_i = x_i^e - x_i^f$, the above two equations can be reduced to

$$u_i = x_i^e + (1 + b - a)z_i. \tag{6.35}$$

According to Definition 1, $u_i^e + u_i^f = 1$, then we have

$$u_i^e = \frac{1+u_i}{2}, u_i^f = \frac{1-u_i}{2}.$$

Substitute x_i^f in (6.33) with (6.34), and then substitute u_i^e, u_i^f with u_i, we have

$$x_i^e = \frac{1+b+a}{2(1+b)}u_i + \frac{1+b-a}{1+b}(\frac{1}{2}+c). \tag{6.36}$$

Substitute the above equation of x_i^e to (6.35) to obtain

$$u_i = \frac{1+b+a}{2(1+b)}u_i + \frac{1+b-a}{1+b} + (1+b-a)z_i,$$

which can be rearranged to

$$z_i = \frac{1}{2(1+b)}u_i - \frac{1}{1+b}(\frac{1}{2}+c), \quad (c \geq 0).$$

Similarly, for the case when $x_i^e < 0, x_i^f > 0$, we have

$$z_i = \frac{1}{2(1+b)}u_i + \frac{1}{1+b}(\frac{1}{2}+c), \quad (c \geq 0).$$

Since z_i and u_i are both constants, $\text{bias}(z_i) = z_i$ and $\text{bias}(u_i) = u_i$. In summary, we have

$$\text{bias}(z_i) = \begin{cases} \frac{1}{2(1+b)}\text{bias}(u_i) - \frac{1}{1+b}(\frac{1}{2}+c) & (x_i^e < 0, x_i^f > 0) \\ \frac{1}{2(1+b)}\text{bias}(u_i) + \frac{1}{1+b}(\frac{1}{2}+c) & (x_i^e > 0, x_i^f < 0). \end{cases} \tag{6.37}$$

The derivation in this section shows that, when the value of u_i^e and u_i^f causes the Matsuoka system fall into a quadrant such that $x_i^e x_i^f < 0$, the system converges to a set point equilibrium. At this moment the conclusion in Proposition 1 is not applicable to the system. The system should instead follow the relation described in (6.37).

The boundary case is at $x_i^e = 0, x_i^f > 0$ or $x_i^f = 0, x_i^e > 0$. For $x_i^e = 0, x_i^f > 0$, substitute $x_i^e = 0$ to (6.33) and (6.35), we can obtain the equation

$$\text{bias}(u_i) = u_i = \frac{2a}{a+b+1} - 1.$$

Similarly when $x_i^f > 0, x_i^e = 0$, we have

$$\text{bias}(u_i) = u_i = 1 - \frac{2a}{a+b+1}.$$

6.3.3 Proof of Proposition 2

Proof. For simplicity we denote $A^q \triangleq A_{x_i^q}$ and $r^q \triangleq r_{x_i^q}$ for $q \in \{e, f\}$. Instead of looking into the relation between u_i and z_i, we focus on the bias between the two states.

According to the *perfect entrainment assumption* [53] and Definition 2, let u_i be resonant to z_i. We define the duty cycle of a wave function as $D(\cdot)$. Let the period of z_i be $T = 2\pi$ (a different value of T would not affect the result of calculation), based on the Fourier expansion, the bias of u_i can be expressed as

$$\begin{aligned} \text{bias}(u_i) &= \frac{1}{T}\int_{-T/2}^{T/2} u_i(t)dt = \frac{1}{2\pi}\int_{-\pi}^{\pi} u_i(t)dt \\ &= 2\frac{1}{2\pi}\int_{-\pi}^{\pi} u_i^e(t)dt - 1 = 2D(u_i^e) - 1. \end{aligned} \tag{6.38}$$

Because the bias terms of x_i and u_i are time-invariant, from (6.6), we can extract the bias component to form a new equation as follows

$$\begin{aligned} &\text{bias}(x_i) = a \cdot \text{bias}(z_i) - b \cdot \text{bias}(y_i) + \text{bias}(u_i) \\ &\text{bias}(y_i) = \text{bias}(z_i). \end{aligned} \tag{6.39}$$

Assume x_i can be approximated by its main sinusoidal component and the period of both x_i and z_i is represented by T. From (6.1) and (6.2) we have

$$\begin{aligned} \text{bias}(x_i) &= \frac{1}{T}\int_{-T/2}^{T/2} x_i dt = \frac{1}{T}\int_{-T/2}^{T/2} (x_i^e - x_i^f)dt \\ &= \frac{1}{T}\int_{-T/2}^{T/2} A^e(\cos(\omega t) + r^e) - A^f(\cos(\omega t) + r^f)dt \\ &= A^e r^e - A^f r^f, \\ \text{bias}(z_i) &= \frac{1}{T}\int_{-T/2}^{T/2} z_i dt = \frac{1}{T}\int_{-T/2}^{T/2} (z_i^e - z_i^f)dt \\ &= \frac{1}{T}\int_{-T/2}^{T/2} (A^e(K(r^e)cos(\omega t) + L(r^e)) - A^f(K(r^f)\cos(\omega t) + L(r^f)))dt \\ &= A^e(L(r^e) - \frac{1}{\pi}) - A^f(L(r_f) - \frac{1}{\pi}) + \frac{1}{\pi}(A^e - A^f). \end{aligned}$$

Apply Taylor expansion on $L(r)$ (Appendix 6.2.2) at $r = 0$, we have

$$L(r) = \frac{1}{\pi} + \frac{r}{2} + o(r), r \in (-1, 1).$$

Then we have

$$\text{bias}(z_i) = \frac{1}{2}A^e r^e - \frac{1}{2}A^f r^f + \frac{1}{\pi}(A^e - A^f) = \frac{1}{2}\text{bias}(x_i) + \frac{1}{\pi}(A^e - A^f). \tag{6.40}$$

According to [53], the amplitude A^q (for $q \in \{e, f\}$) has the form

$$A^q = \frac{\text{bias}(u^q) + c}{r^q + (a+b)L(r^q)}.$$

When the system is harmonic, according to [53, (30)], we have $r^e = r^f = K^{-1}(K_n)$, such that

$$A^e - A^f = \frac{\text{bias}(u_i^e) - \text{bias}(u_i^f)}{K^{-1}(K_n) + (a+b)L(K^{-1}(K_n))} \approx \frac{\text{bias}(u_i)}{2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n)}. \tag{6.41}$$

Let $m = \frac{1}{\pi}\frac{1}{2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n)}$, (6.47) can be rewritten as

$$\text{bias}(z_i) = \frac{1}{2}\text{bias}(x_i) + m\text{bias}(u_i). \tag{6.42}$$

Substitute $\text{bias}(z_i)$ in (6.39) with (6.50), we can obtain the pure relation between $\text{bias}(x_i)$ and $\text{bias}(u_i)$ as

$$(1 - m(b-a))\text{bias}(u_i) = (\frac{1}{2}(b-a) + 1)\text{bias}(x_i). \tag{6.43}$$

In this case, the relation between $\text{bias}(z_i)$ and $\text{bias}(u_i)$ can be expressed as

$$\text{bias}(z_i) = \frac{1 - m(b-a)}{b-a+2}\text{bias}(u_i) + m\text{bias}(u_i) = \frac{1+2m}{b-a+2}\text{bias}(u_i). \tag{6.44}$$

□

6.3.4 Proof of Proposition 2

Proof. For simplicity we denote $A_i^q \triangleq A_{x_i^q}$ and $r_i^q \triangleq r_{x_i^q}$ for $q \in \{e, f\}$. Instead of looking into the relation between u_i and z_i, we focus on the bias between the two states.

According to the *perfect entrainment assumption* [53] and [44, Definition 1], let u_i be resonant to z_i. We define the duty cycle of a wave function as $D(\cdot)$. Let the period of z_i be $T = 2\pi$ (a different value of T would not affect the result of the

calculation), based on the Fourier expansion, the bias of u_i can be expressed as

$$\begin{aligned}\text{bias}(u_i) &= \frac{1}{T}\int_{-T/2}^{T/2} u_i(t)dt = \frac{1}{2\pi}\int_{-\pi}^{\pi} u_i(t)dt \\ &= 2\frac{1}{2\pi}\int_{-\pi}^{\pi} u_i^e(t)dt - 1 = 2D(u_i^e) - 1.\end{aligned} \tag{6.45}$$

Because the bias terms of x_i and u_i are time-invariant, we can extract the bias component to form a new equation as follows

$$\begin{aligned}&\text{bias}(x_i) = a \cdot \text{bias}(z_i) - b \cdot \text{bias}(y_i) + \text{bias}(u_i) \\ &\text{bias}(y_i) = \text{bias}(z_i) - \text{bias}(p_i).\end{aligned} \tag{6.46}$$

Assume x_i can be approximated by its main sinusoidal component and the period of both x_i and z_i is represented by T, we have

$$\begin{aligned}\text{bias}(x_i) &= \frac{1}{T}\int_{-T/2}^{T/2} x_i dt = \frac{1}{T}\int_{-T/2}^{T/2}(x_i^e - x_i^f)dt \\ &= \frac{1}{T}\int_{-T/2}^{T/2} A_i^e(\cos(\omega t) + r_i^e) - A_i^f(\cos(\omega t) + r_i^f)dt \\ &= A_i^e r_i^e - A_i^f r_i^f,\end{aligned}$$

and

$$\begin{aligned}\text{bias}(z_i) &= \frac{1}{T}\int_{-T/2}^{T/2} z_i dt = \frac{1}{T}\int_{-T/2}^{T/2}(z_i^e - z_i^f)dt \\ &= \frac{1}{T}\int_{-T/2}^{T/2}(A_i^e(K(r_i^e)cos(\omega t) + L(r_i^e)) - A_i^f(K(r_i^f)\cos(\omega t) + L(r_i^f)))dt \\ &= A_i^e(L(r_i^e) - \frac{1}{\pi}) - A_i^f(L(r_i^f) - \frac{1}{\pi}) + \frac{1}{\pi}(A_i^e - A_i^f).\end{aligned}$$

Apply Taylor expansion on $L(r)$ at $r = 0$, we have

$$L(r) = \frac{1}{\pi} + \frac{r}{2} + o(r), r \in (-1, 1).$$

Then we have

$$\text{bias}(z_i) = \frac{1}{2}A_i^e r_i^e - \frac{1}{2}A_i^f r_i^f + \frac{1}{\pi}(A_i^e - A_i^f) = \frac{1}{2}\text{bias}(x_i) + \frac{1}{\pi}(A_i^e - A_i^f). \tag{6.47}$$

According to [53], the amplitude A_i^q (for $q \in \{e, f\}$) has the form

$$A_i^q = \frac{\text{bias}(u_i^q) + c}{r_i^q + (a+b)L(r_i^q)}.$$

When the system is harmonic, according to [53, (30)], we have

$$r_i^e = r_i^f = K^{-1}(K_n),$$

such that

$$A_i^e - A_i^f = \frac{\text{bias}(u_i^e) - \text{bias}(u_i^f)}{K^{-1}(K_n) + (a+b)L(K^{-1}(K_n))} \tag{6.48}$$

$$\approx \frac{\text{bias}(u_i)}{2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n)}. \tag{6.49}$$

Let $m = \frac{1}{\pi}\frac{1}{2K_n - 1 + \frac{2}{\pi}(a+b)\sin^{-1}(K_n)}$, (6.47) can be rewritten as

$$\text{bias}(z_i) = \frac{1}{2}\text{bias}(x_i) + m\text{bias}(u_i). \tag{6.50}$$

Substitute $\text{bias}(x_i)$ in (6.46) with (6.50), we can obtain the pure relation between $\text{bias}(z_i)$ and $\text{bias}(u_i)$, $\text{bias}(p_i)$ as

$$\text{bias}(z_i) = \frac{1+2m}{b-a+2}\text{bias}(u_i) + \frac{b}{b-a+2}\text{bias}(p_i).$$

□

6.3.5 The Reason of Using Inhibiting Sensory Feedback Input in the AF form Matsuoka Oscillator

In the AF form Matsuoka oscillator, if the sensory feedback coefficients p_i^e, p_i^f are activating, then by changing the sign of p_i term in Eq. (4.11), we have

$$\tau_r\tau_a\frac{d^2}{dt^2}x_{\mathcal{F}_i} + (\tau_r + \tau_a - \tau_a a K(r_x))\frac{d}{dt}x_{\mathcal{F}_i} \tag{6.51}$$

$$+ ((b-a)K(r_x) + 1)x_{\mathcal{F}_i} = \tau_a\frac{d}{dt}u_{\mathcal{F}_i} + u_{\mathcal{F}_i} - bp_i.$$

Let $m_1 = \tau_r\tau_a$, $m_2 = \tau_r + \tau_a - \tau_a a K(r_x)$ and $m_3 = (b-a)K(r_x) + 1$, $\omega_0^2 = \frac{m_3}{m_1}$. We have Eq. (6.51) simplified to

$$m_1 \frac{d^2}{dt^2} x_{\mathcal{F}_i} + m_2 \frac{d}{dt} x_{\mathcal{F}_i} + m_3 x_{\mathcal{F}_i} = \tau_a \frac{d}{dt} u_{\mathcal{F}_i} + u_{\mathcal{F}_i} - b p_i. \tag{6.52}$$

According to [44, Eq. (B.15), Eq. (B.16)],

$$u_{\mathcal{F}_i} \approx A_u \cos(\omega t) - \frac{1}{2}, \text{ where } A_u = \frac{1}{2}\frac{e^A - 1}{e^A + 1}.$$

Therefore we have

$$m_1 \frac{d^2}{dt^2} x_{\mathcal{F}_i} + m_2 \frac{d}{dt} x_{\mathcal{F}_i} + m_3 x_{\mathcal{F}_i} = -\tau_a \omega A_u \sin(\omega t) + A_u \cos(\omega t) - \frac{1}{2} - b p_i. \tag{6.53}$$

During a contact segment, assume $p_i \approx 1$ and p_i is a constant in this segment (e.g. the first half period of a square wave). We consider the solutions of Eq. (6.53) in the following two scenarios:

- When $K(r_x) < \frac{\tau_r + \tau_a}{\tau_a a}$, the particular solution is

$$\begin{aligned} x^*(t) = &-\frac{\tau_a \omega A_u}{\sqrt{m_1^2(\omega_0^2 - \omega^2)^2 + m_2^2\omega^2}} \cos(\omega t + \theta_1) \\ &+ \frac{A_u}{\sqrt{m_1^2(\omega_0^2 - \omega^2)^2 + m_2^2\omega^2}} \cos(\omega t + \theta_2) \\ &- \frac{\frac{1}{2} + b}{m_3}, \end{aligned} \tag{6.54}$$

  and the general solution is

$$x(t) = c_1 \exp(\lambda_1 t) + c_2 \exp(\lambda_2 t) + x^*(t) \approx x^*(t),$$

  where $\lambda_1, \lambda_2 < 0$ are the eigenvalues of Eq. (6.53).

- When $K(r_x) = K_n = \frac{\tau_r + \tau_a}{\tau_a a}$ and $\omega \neq \omega_0$, the system becomes harmonic. The particular solution is

$$x^*(t) = -\frac{\tau_a \omega A_u}{m_1(\omega_0^2 - \omega^2)} \sin(\omega t) + \frac{A_u}{m_1(\omega_0^2 - \omega^2)} \cos(\omega t) - \frac{\frac{1}{2} + b}{m_3}, \tag{6.55}$$

  and the general solution is

$$x(t) = c_1 \cos(\omega_0 t) + c_2 \sin(\omega_0 t) + x^*(t),$$

  where the parameters c_1, c_2 are related to the initial condition of Eq. (6.53).

When the snake robot is jammed by the obstacles, the oscillation frequency ω will decrease, and therefore leads to the increase of $\omega_0^2 - \omega^2$. In this case, as long as ω is small enough, in both harmonic and non-harmonic situations the solution of Eq. (6.53) will be consistently negative. In the Matsuoka oscillator, a consistently negative membrane potential will cause $z_i = 0$. Then the output of the Matsuoka oscillator becomes zero. Therefore the CPG node in contact will always stop oscillating for a certain period of time, which extend the jamming period and harm the locomotion performance in contact-aware scenario.

6.4 Implementations

6.4.1 Extension of the PPOC-CPG controller for a soft quadruped robot

In this section, we use a demo to show that our approach can be easily generalized to more tasks than the locomotion of a soft snake robot. Specifically, we show that the same procedure of PPOC-CPG can be applied to designing controllers for a soft quadruped robot (Fig. 6.1), demonstrating the universality of PPOC-CPG.

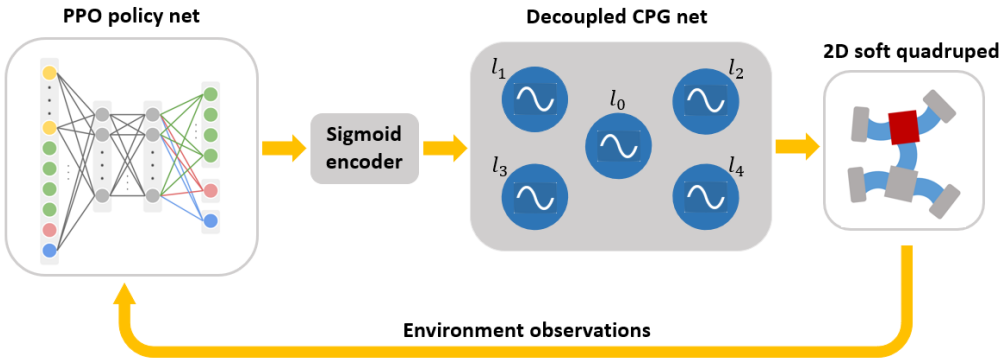


Figure 6.1: PPOC-CPG scheme for 2D soft quadruped robot locomotion control.

Figure 6.1 shows the basic scheme of PPOC-CPG framework [42] applied on a 2D soft quadruped robot (we call it soft turtle bot for simplicity). The Sigmoid encoder function is defined by [43, Eq.(5)]. The CPG network includes 5 decoupled primitive Matsuoka oscillators [43, Eq.(1)], with l_0 being connected to the body trunk joint, l_1, l_2 being related to the front legs, and l_3, l_4 related to the hind legs.

In [42, 43], it is concluded that for the proposed PPOC-CPG controller, the oscillation patterns including frequency, amplitude and bias are all maneuverable to the RL agent. To show the universality of the above conclusion, we train the same controller with the soft quadruple robot for the random goal tracking task. The fundamental configuration for this experiment is the same as the presented in [43].

Table 6.4: Curriculum settings

Levels	Distance range (m)	Turning angles ($^\circ$)	Goal radius (m)
1	$0.9 \sim 1.5$	$-10 \sim 10$	0.5
2	$0.9 \sim 1.5$	$-20 \sim 20$	0.45
3	$0.9 \sim 1.5$	$-30 \sim 30$	0.4
4	$0.9 \sim 1.5$	$-40 \sim 40$	0.35
5	$0.9 \sim 1.5$	$-40 \sim 40$	0.3
6	$0.9 \sim 1.5$	$-45 \sim 45$	0.3
7	$0.9 \sim 1.5$	$-45 \sim 45$	0.25

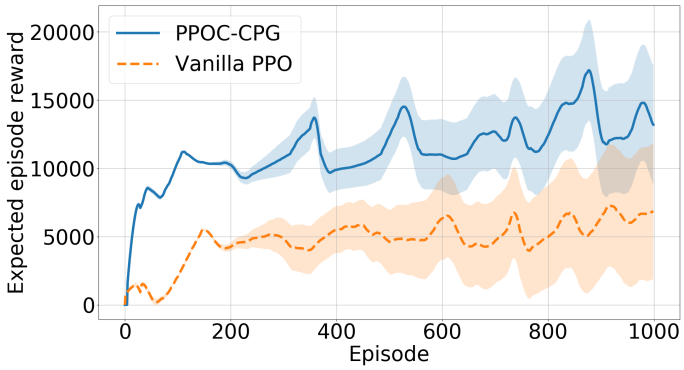


Figure 6.2: Learning process comparison, both methods are trained under the same condition for 5 rounds.

The curriculum is adjusted accordingly for better convergence of the learning process (see Table 6.2). The training result shows that our approach is able to converge to the highest curriculum level with stable locomotion pattern (The goal tracking performance video of PPOC-CPG on the soft turtle bot is available at: `https://youtu.be/ByMreo7Re18`). The success of PPOC-CPG controlled soft turtle bot in random goal tracking demonstrates the maneuverability of our CPG system. In addition, Fig. 6.2 shows that PPOC-CPG still outperforms the vanilla PPO in learning soft quadruple robot locomotion. It takes less learning episodes to converge and converges to a higher reward level.

Moreover, the same advantages of our approach on the soft snake robot can still be easily observed on the control command of the soft quadruped robot. Fig. 6.3 compares the control commands sent to all 5 joints of the soft quadruped robot by different controllers when they are approaching the same goal. The agent with vanilla PPO controller converges to a control policy that generates gait pattern with considerably high frequency and amplitude, which puts the real robot actuators under high risk of being damaged in the sim-to-real tasks. On the other hand, the PPOC-CPG controller is able to generate smoothier control commands, with adjustable frequency and amplitude, which naturally fits the real robot actuator.

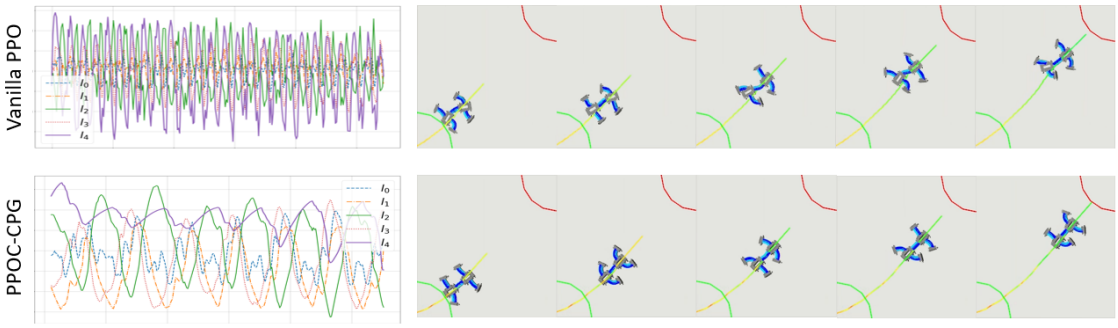


Figure 6.3: Trajectory comparison, the left column are joint space control commands, and the right are body motion snapshots of the 2D soft quadruped robot.

6.4.2 Extension of the PPO-CPG controller for a bipedal robot (Cassie)

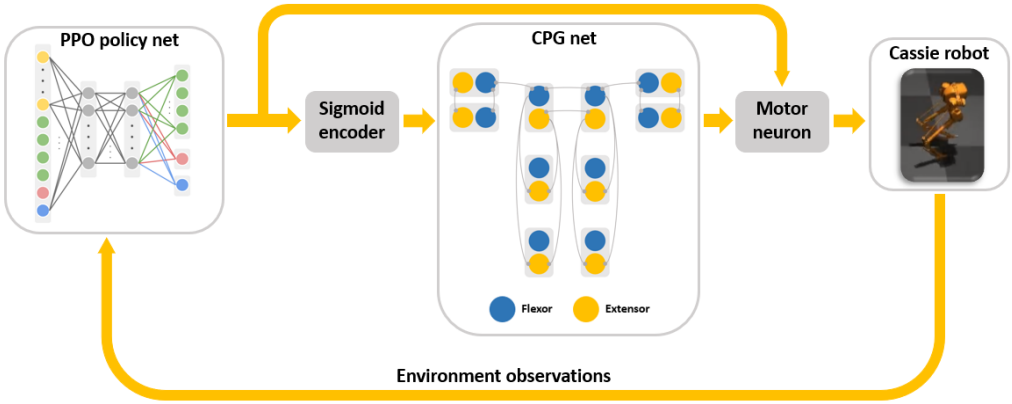


Figure 6.4: PPO-CPG scheme for Cassie robot locomotion control.

This section shows the application of PPO-CPG method on the Cassie robot. Because the 3D bipedal robot is more vulnerable to unreliable dynamics while maintaining self-balancing during locomotion tasks, and the consequence of changing locomotion frequency of a bipedal robot may cause the change of locomotion gait and therefore influence the self-balancing dynamics. As a result, we will use a fixed locomotion frequency for simplicity in this study. Despite the frequency control, all other features including amplitude-based velocity tuning and changing walking direction with biased tonic inputs that have been implemented on the soft snake robot in [42, 43] are proved to be applicable to the 3D bipedal walking in this section.

Due to the complexity of bipedal locomotion, few modifications are applied to the control scheme of PPO-CPG method. For the configuration of the CPG network, we take the empirical parameters from [26], with additional primitive CPG nodes

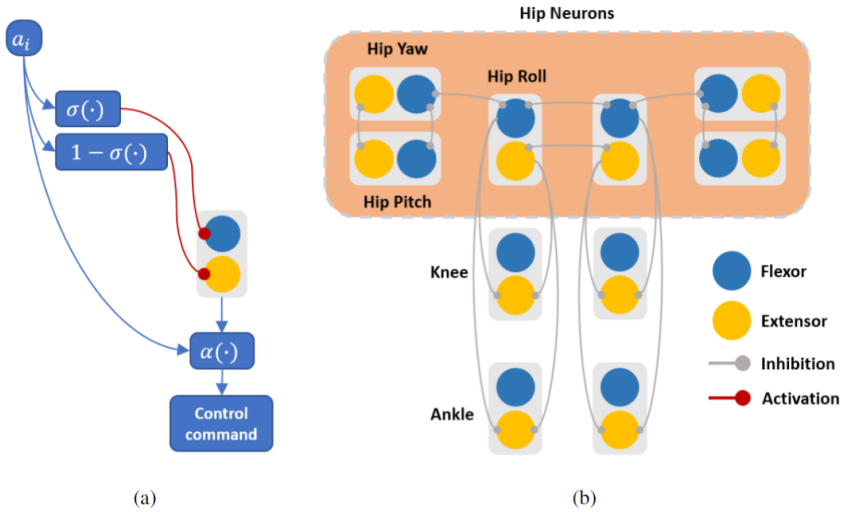


Figure 6.5: (a) Signal flow chart of each primitive joint action in the Cassie robot controller case, where $\sigma(\cdot)$ is the sigmoid function, and $\alpha(\cdot)$ is the motor neuron function. (b) CPG network design for bipedal locomotion.

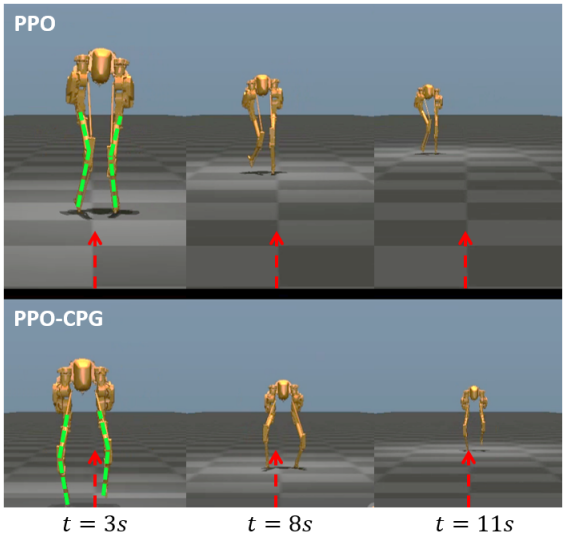


Figure 6.6: Performance comparison in straight line following task with target velocity $v_x = 0.6\text{m/s}, v_y = 0.0\text{m/s}$.

for hip yaw and hip pitch connected to the network intuitively (the configuration of the CPG network can be found in Table 6.1 and Fig. 6.5b). And we will show that such kind of CPG network can be applied to the PPO-CPG scheme to realize stable 3D bipedal walking. An additional motor neuron is added to the whole scheme to help the self-balancing control. For each action signal a_i generated by the RL policy,

Table 6.5: Parameter Configuration of the Matsuoka CPG Net Controller for Cassie Robot.

Parameters	**Symbols**	**Values**
Amplitude ratio	a_ψ	1.0
∗Self-inhibition weight	b	**2.5**
∗Discharge rate (hip joints)	τ_{r_1}	**0.1**
∗Adaptation rate (hip joints)	τ_{a_1}	**1.2**
∗Discharge rate (other joints)	τ_{r_2}	**0.05**
∗Adaptation rate (other joints)	τ_{a_2}	**0.6**
Period ratio	K_f	1.0
Mutual inhibition weights	a_i	2.0
Coupling weights	w_{ij}	1.0
	w_{ji}	1.0

we define our motor neuron function as follows

$$\alpha(a_i) = c_1 a_i + c_2 \psi_i, \tag{6.56}$$

where $c_1, c_2 > 0$ are constant coefficients, and ψ_i is the output of the i-th Matsuoka CPG node.

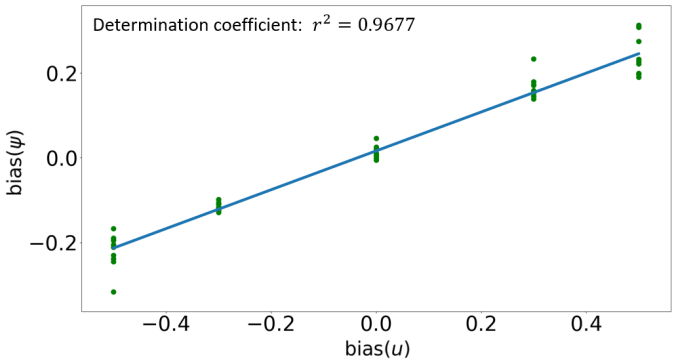


Figure 6.7: Linear relation between input and output bias of the RL-driven CPG node during Cassie's locomotion. The performance video is available at: `https://youtu.be/wODTrnln0Xw`

Our method is tested in Mujoco simulator. The learning process is modified from the configuration in [96]. With the same learning configuration of PPO in [96], our method is trained for 2 million sampling episodes. Our method is able to reach a higher performance score than the original method in straight line following task (maintaining certain velocity on a fixed direction) – the claimed performance score on tracking goal velocity $v_x = 0.6\text{m/s}, v_y = 0.0\text{m/s}$ was around 0.6 [96] and our score is 0.7, which indicates stronger ability of the agent in accurate direction keeping. The difference of line keeping performance is also visually observable during locomotion.

The upper part snapshot in Fig. 6.6 shows that the Cassie robot trained with vanilla PPO approach [96] has significant deviation from the desired locomotion direction after around 8-second walking in simulation, while the agent trained with PPO-CPG (lower part in Fig. 6.6) is able to follow the desired direction in a longer time. This is mainly because, although the walking pace can be determined by the imitation reward, the compared approach with vanilla PPO [96] cannot specify the desired body posture during locomotion, so that the hip yaw joints end up turning in as the robot walks (the leg postures are highlighted by the green dashed lines in Fig. 6.6), which causes the turning ability of the robot highly restricted. In our approach, since the CPG system provides a fundamental rhythm for free oscillation as a natural reference to the RL agent, the control policy converges to move in a more appropriate range when driving hip yaw angles in straight walking tasks. The comparison video is available at: `https://youtu.be/Ql3cby06Ddo`.

In addition to the body posture reason, the maneuverability of the Matsuoka CPG system may also contribute to the direction following performance. Based on what we have mentioned about the maneuverability of the Matsuoka CPG system with biased tonic inputs in our previous work [42], we further verify that the same property still holds for the PPO-CPG scheme on Cassie robot during bipedal locomotion. Figure 6.7 shows the linear relationship between bias(u) and bias(ψ) when we add different bias values to the hip pitch joints' tonic inputs. And this leads to stable side walking without further training from the straight walking controller, which shows the sensitivity of the Matsuoka CPG system to the biased directional control command.

It is noted that steering with hip yaw requires more investigation on balancing control, which is not studied yet in this document. We leave this case for future study.

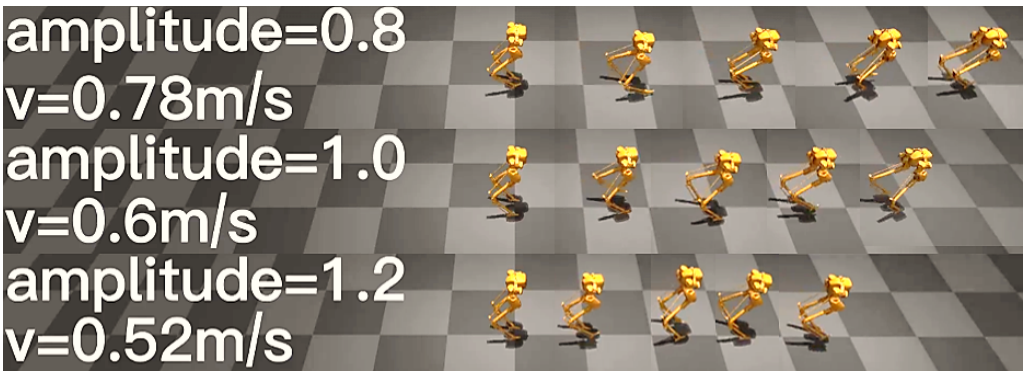


Figure 6.8: Amplitude control of the CPG network for velocity tuning of the Cassie robot. The performance video is available at: `https://youtu.be/_RHqiQrb2mM`

Moreover, as Fig. 6.8 shows, our method allows a range of direct velocity tuning through amplitude tuning of tonic inputs of the CPG system without harming the

performance of locomotion, while the compared method [96] require re-training of the policy if the desired locomotion speed is changed on the same direction. In this case, the reason of lower oscillation amplitude leading to higher locomotion speed is because that the locomotion period (frequency) is fixed, so that the foot lifting phase will occupy the time of swinging phase for the robot to step forward.

6.4.3 Experiment result with an off-policy method (TD3) on the Soft Snake Robot

Table 6.6: Performance Comparison of Different Approaches (TD3 version).

Metrics	Vanilla TD3	TD3-CPG	FOC-TD3-CPG
Simulated average speed (m/s)	0.149	**0.162**	0.155
Simulated success rate	0.97	**0.99**	**0.99**
Real average speed (m/s)	0.023 (↓ 84.6%)	0.032 (↓ 80.2%)	**0.045** (↓ 70.9%)
Real success rate	0.43 (↓ 55.6%)	0.76 (↓ 23.2%)	**0.84**(↓ 15.1%)

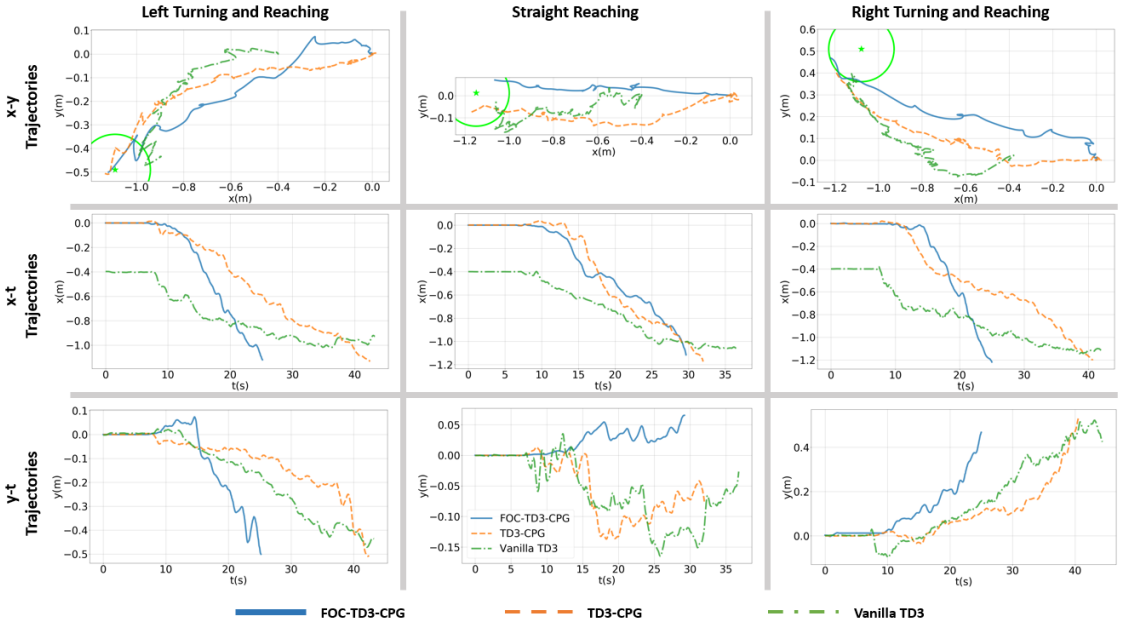


Figure 6.9: Sample comparison of trajectories generated by Vanilla TD3 policy, TD3-CPG policy, and FOC-TD3-CPG policy in reality. The connection between TD3 and CPG is the same as PPOC-CPG, and FOC also means "Free-response Oscillation Constrained".

Figure 6.9 and Table 6.6 show that a different RL algorithm can easily fit into our framework and generate similar results as PPO did. This comparison is also presented in the video "TD3 Learning methods comparison.mp4" [1].

[1]The video can be viewed from `http://shorturl.at/cgms1`

6.4.4 Dynamic analysis of elastic pull-back wheels

This section analyzes and calculates the torque of the elastic pull-back wheels in the scenario of snake robot locomotion.

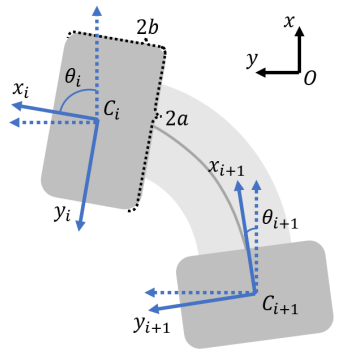


Figure 6.10: Coordinate notation of the rigid bodies of the soft snake robot.

As shown in Fig. 6.10, let the 2D state representation in world coordination be $\{x, y\}$, for the i-th rigid body, the local coordinate of the i-th rigid body is denoted as: $\{x_{c_i}, y_{c_i}\}$, with θ_i being the angle between global coordinate axis and local coordinate axis of the i-th rigid body. The position of the i-th rigid body can be determined by $\{x_{c_i}, y_{c_i}, \theta_i\}$. The length and width of each rigid body are set as $2b$ and $2a$.

From rigid body kinematics, the corresponding velocity of the left wheel in the local heading direction x_i of the i-th rigid body is

$$v_{x_i,L} = \dot{x}_{c_i} \cos\theta_i + \dot{y}_{c_i} \sin\theta_i - a\dot{\theta}_i.$$

Respectively, we have the right wheel velocity

$$v_{x_i,R} = \dot{x}_{c_i} \cos\theta_i + \dot{y}_{c_i} \sin\theta_i + a\dot{\theta}_i.$$

Thus the velocity of the selected wheel can be expressed as

$$v_{x_i} = v_{x_i,L}^{(1+\mathrm{sgn}(\theta_i-\theta_{i+1}))/2} v_{x_i,R}^{(1-\mathrm{sgn}(\theta_i-\theta_{i+1}))/2}, \quad (\theta_6 = 0).$$

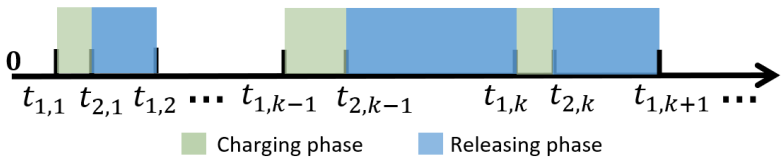


Figure 6.11: The time series illustration of energy charging phase and releasing phase of the elastic torsion spring presents in turn.

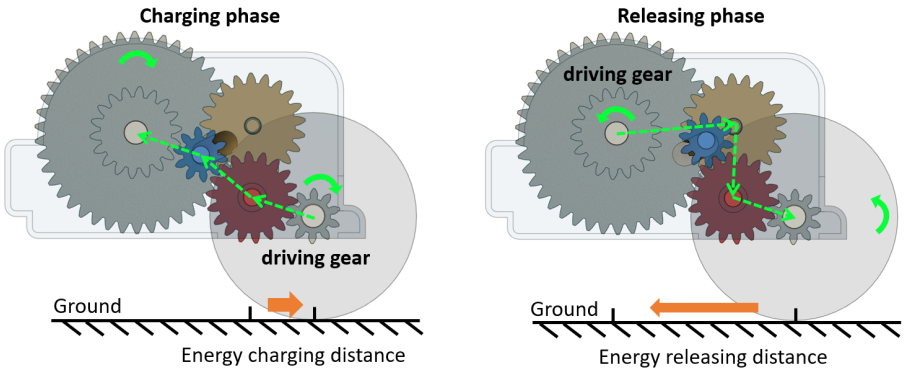


Figure 6.12: The driving gear illustration of energy charging phase and releasing phase of the elastic torsion spring.

For the torsion spring torque calculation, as shown in Fig. 6.11, in every k-th ($k = 0, 1, 2, ...$) contact event of the elastic torsion spring in the i-th rigid body, when $t \in (t_{1,k}, t_{2,k})$, $v_{x_i}(t) < 0$, when $t \in (t_{2,k-1}, t_{1,k})$, $v_{x_i}(t) > 0$. We denote the time interval satisfying $v_{x_i}(t) < 0$ as energy charging phase. While energy releasing phase indicates the time interval when $v_{x_i}(t) > 0$. Since the transition of spring energy is smooth, in between the two phases we have $v_{x_i}(t_{1,k}) = v_{x_i}(t_{2,k}) = 0$.

Assume the energy waste is negligible. As shown in Fig. 6.12, at the k-th charging phase, the energy charging distance of the i-th rigid body is the distance of the wheels pulled backward, gaining potential energy on the torsion spring, which can be expressed as

$$P_{i,k}(t) = \min \left\{ \frac{M_{i,k-1}(t_{1,k})}{k_0} - \int_{t_{1,k}}^{t} v_{x_i}(h)dh, P_{\max} \right\}, \ t \in (t_{1,k}, t_{2,k}).$$

Where k_0 is the backward-forward ratio of the elastic pullback gearbox, and

$$M_{i,k}(t) = \max \left\{ 0, \ k_0 P_{i,k}(t_{2,k}) - \int_{t_{2,k}}^{t} v_{x_i}(h)dh \right\}, \ t \in (t_{2,k}, t_{1,k+1}), M_{i,0}(t_{1,1}) = 0.$$

In the above equation, $M_{i,k}$ is the energy releasing distance of the i-th rigid body at k-th releasing phase, which is the distance of the wheels rolling forward, driven by the loaded torsion spring in the gearbox.

Let $\beta_i(t) = \text{sgn}(v_{x_i}(t))$, we have

$$\tau_{i,k}(t) = M_{i,k}(t)^{1-\frac{(1-\beta_i(t))|\beta_i(t)|}{2}} \cdot (k_0 P_{i,k}(t))^{\frac{(1-\beta_i(t))|\beta_i(t)|}{2}}, \quad \text{for } t \in [t_{1,k}, t_{1,k+1}].$$

Therefore,

$$\tau_i(t) = \begin{cases} \tau_{i,1}(t) & t \in (0, t_{1,1}) \\ \vdots \quad , & \\ \tau_{i,k}(t) & t \in (t_{1,k}, t_{1,k+1}) \\ \vdots \quad . & \end{cases}$$

$$\boldsymbol{\tau}(t) = [\tau_1(t), ..., \tau_5(t)]^T.$$

This model shows that the implementation of the elastic pull-back wheels can improve the locomotion energy-efficient by transmitting the pull-back forces caused by the obstacles to the positive torques that drives the soft snake robot forward.

Bibliography

[1] Arsenic, A. M. On stability and tuning of neural oscillators: application to rhythmic control of a humanoid robot. In *IEEE International Joint Conference on Neural Networks* (July 2004), vol. 1, pp. 99–104.

[2] Bacon, P.-L., Harb, J., and Precup, D. The option-critic architecture. In *AAAI* (2017).

[3] Bellegarda, G., and Ijspeert, A. CPG-RL: Learning central pattern generators for quadruped locomotion. *IEEE Robotics and Automation Letters 7*, 4 (2022), 12547–12554.

[4] Bing, Z., Cheng, L., Chen, G., Röhrbein, F., Huang, K., and Knoll, A. Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot. *Bioinspiration & Biomimetics 12*, 3 (2017), 035001.

[5] Bing, Z., Cheng, L., Huang, K., and Knoll, A. Simulation to real: Learning energy-efficient slithering gaits for a snake-like robot. *IEEE Robotics & Automation Magazine 29*, 4 (2022), 92–103.

[6] Bing, Z., Jiang, Z., Cheng, L., Cai, C., Huang, K., and Knoll, A. End to end learning of a multi-layered SNN based on R-STDP for a target tracking snake-like robot. *International Conference on Robotics and Automation* (2019), 9645–9651.

[7] Brown, T. G. The intrinsic factors in the act of progression in the mammal. *Proceedings of The Royal Society B: Biological Sciences 84*, 572 (1911), 308–319.

[8] Campanaro, L., Gangapurwala, S., De Martini, D., Merkt, W., and Havoutis, I. *CPG-Actor: Reinforcement Learning for Central Pattern Generators*. 10 2021, pp. 25–35.

[9] Chen, S., Tang, B., and Wang, K. Twin delayed deep deterministic policy gradient-based intelligent computation offloading for iot. *Digital Communications and Networks* (2022).

[10] Choset, H. M., Hutchinson, S., Lynch, K. M., Kantor, G., Burgard, W., Kavraki, L. E., Thrun, S., and Arkin, R. C. *Principles of robot motion: theory, algorithms, and implementation.* MIT Press, 2005.

[11] Crespi, A., Badertscher, A., Guignard, A., and Ijspeert, A. J. Swimming and crawling with an amphibious snake robot. *IEEE International Conference on Robotics and Automation* (2005), 3024–3028.

[12] Crespi, A., and Ijspeert, A. J. Online optimization of swimming and crawling in an amphibious snake robot. *IEEE Transactions on Robotics 24*, 1 (2008), 75–87.

[13] Crowe-Riddell, J. M., Snelling, E. P., Watson, A. P., Suh, A. K., Partridge, J. C., and Sanders, K. L. The evolution of scale sensilla in the transition from land to sea in elapid snakes. *Open biology 6*, 6 (2016), 160054.

[14] Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. `https://github.com/openai/baselines`, 2017.

[15] Di, J., Yao, S., Ye, Y., Cui, Z., Yu, J., Ghosh, T. K., Zhu, Y., and Gu, Z. Stretch-triggered drug delivery from wearable elastomer films containing therapeutic depots. *ACS nano 9*, 9 (2015), 9407–9415.

[16] Dong, J., Zhang, X., and Jia, X. Strategies of pursuit-evasion game based on improved potential field and differential game theory for mobile robots. 1452–1456.

[17] Dzeladini, F., Ait-Bouziad, N., and Ijspeert, A. CPG-based control of humanoid robot locomotion. *Humanoid Robotics: A Reference* (2018), 1–35.

[18] Endo, G., Morimoto, J., Matsubara, T., Nakanishi, J., and Cheng, G. Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot. *The International Journal of Robotics Research 27*, 2 (2008), 213–228.

[19] Fukuoka, Y., Otaka, K., Takeuchi, R., Shigemori, K., and Inoue, K. Mechanical designs for field undulatory locomotion by a wheeled snake-like robot with decoupled neural oscillators. *IEEE Transactions on Robotics 39*, 2 (2023), 959–977.

[20] Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., and Marín-Jiménez, M. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition 47*, 6 (2014), 2280–2292.

[21] Gasoto, R., Macklin, M., Liu, X., Sun, Y., Erleben, K., Onal, C., and Fu, J. A validated physical model for real-time simulation of soft robotic snakes. In *IEEE International Conference on Robotics and Automation* (2019).

[22] Gravdahl, I., Øyvind Stavdahl, Koushan, A., Løwer, J., and Pettersen, K. Y. Modeling for hybrid obstacle-aided locomotion (hoal) of snake robots. *IFAC-PapersOnLine 55*, 20 (2022), 247–252. 10th Vienna International Conference on Mathematical Modelling MATHMOD 2022.

[23] Grillner, S., and El Manira, A. Current principles of motor control, with special reference to vertebrate locomotion. *Physiological reviews* (2019).

[24] Hawkes, E. W., Blumenschein, L. H., Greer, J. D., and Okamura, A. M. A soft robot that navigates its environment through growth. *Science Robotics 2*, 8 (2017).

[25] Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., and Hutter, M. Learning agile and dynamic motor skills for legged robots. *Science Robotics 4*, 26 (2019).

[26] Ichimura, D., Hobara, H., Hisano, G., Maruyama, T., and Tada, M. Acquisition of bipedal locomotion in a neuromusculoskeletal model with unilateral transtibial amputation. *Frontiers in Bioengineering and Biotechnology 11* (2023).

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

[28] Ijspeert, A. J., and Crespi, A. Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model. In *IEEE International Conference on Robotics and Automation* (2007), IEEE, pp. 262–268.

[29] Ijspeert, A. J., Hallam, J., and Willshaw, D. Evolving swimming controllers for a simulated lamprey with inspiration from neurobiology. *Adaptive Behavior 7*, 2 (1999), 151–172.

[30] Kakogawa, A., and Ma, S. Robotic search and rescue through in-pipe movement. In *Unmanned Robotic Systems and Applications*, M. Reyhanoglu and G. D. Cubber, Eds. IntechOpen, Rijeka, 2019, ch. 1.

[31] Kano, T., and Ishiguro, A. Obstacles are beneficial to me! scaffold-based locomotion of a snake-like robot using decentralized control. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2013), pp. 3273–3278.

[32] Kano, T., and Ishiguro, A. Decoding Decentralized Control Mechanism Underlying Adaptive and Versatile Locomotion of Snakes. *Integrative and Comparative Biology 60*, 1 (03 2020), 232–247.

[33] Kano, T., Matsui, N., and Ishiguro, A. 3d movement of snake robot driven by tegotae-based control. In *Biomimetic and Biohybrid Systems* (Cham, 2019), U. Martinez-Hernandez, V. Vouloutsi, A. Mura, M. Mangan, M. Asada, T. J. Prescott, and P. F. Verschure, Eds., Springer International Publishing, pp. 346–350.

[34] Kano, T., Sato, T., Kobayashi, R., and Ishiguro, A. Local reflexive mechanisms essential for snakes' scaffold-based locomotion. *Bioinspiration & biomimetics 7*, 4 (2012), 046008.

[35] Kano, T., Yoshizawa, R., and Ishiguro, A. Tegotae-based decentralised control scheme for autonomous gait transition of snake-like robots. *Bioinspiration & Biomimetics 12*, 4 (aug 2017), 046009.

[36] Karpathy, A., and Van De Panne, M. Curriculum learning for motor skills. *Canadian Conference on Artificial Intelligence* (2012), 325–330.

[37] Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. 396–404.

[38] Lafreniere-Roula, M., and McCrea, D. A. Deletions of rhythmic motoneuron activity during fictive locomotion and scratch provide clues to the organization of the mammalian central pattern generator. *Journal of neurophysiology 94*, 2 (2005), 1120–1132.

[39] Liljeback, P., Pettersen, K. Y., Stavdahl, Ø., and Gravdahl, J. T. Hybrid modelling and control of obstacle-aided snake robot locomotion. *IEEE Transactions on Robotics 26*, 5 (2010), 781–799.

[40] Liljeback, P., Pettersen, K. Y., Stavdahl, Ø., and Gravdahl, J. T. Experimental investigation of obstacle-aided locomotion with a snake robot. *IEEE Transactions on Robotics 27*, 4 (2011), 792–800.

[41] Liljebäck, P., Pettersen, K. Y., Stavdahl, Ø., and Gravdahl, J. T. *Hybrid Control of Obstacle-Aided Locomotion*. Springer London, London, 2013.

[42] Liu, X., Gasoto, R., Jiang, Z., Onal, C., and Fu, J. Learning to locomote with artificial neural-network and CPG-based control in a soft snake robot. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020), pp. 7758–7765.

[43] Liu, X., Onal, C. D., and Fu, J. Reinforcement learning of a cpg-regulated locomotion controller for a soft snake robot. *ArXiv abs/2207.04899* (2022).

[44] Liu, X., Onal, C. D., and Fu, J. Reinforcement learning of cpg-regulated locomotion controller for a soft snake robot. *IEEE Transactions on Robotics* (2023), 1–20.

[45] Luo, M., Agheli, M., and Onal, C. D. Theoretical modeling and experimental analysis of a pressure-operated soft robotic snake. *Soft Robotics 1*, 2 (2014), 136–146.

[46] Luo, M., Skorina, E. H., Tao, W., Chen, F., Ozel, S., Sun, Y., and Onal, C. D. Toward modular soft robotics: Proprioceptive curvature sensing and sliding-mode control of soft bidirectional bending modules. *Soft robotics 4*, 2 (2017), 117–125.

[47] Luo, M., Wan, Z., Sun, Y., Skorina, E. H., Tao, W., Chen, F., Gopalka, L., Yang, H., and Onal, C. D. Motion planning and iterative learning control of a modular soft robotic snake. *Frontiers in robotics and AI 7* (2020), 599242.

[48] Luo, M., Yan, R., Wan, Z., Qin, Y., Santoso, J., Skorina, E. H., and Onal, C. D. Orisnake: Design, fabrication, and experimental analysis of a 3-d origami snake robot. *IEEE Robotics and Automation Letters 3*, 3 (2018), 1993–1999.

[49] Majidi, C. Soft robotics: A perspective–Current trends and prospects for the future. *Soft Robotics 1*, 1 (2014), 5–11.

[50] Marder, E., and Bucher, D. Central pattern generators and the control of rhythmic movements. *Current Biology 11*, 23 (2001), R986–R996.

[51] Matsuoka, K. Sustained oscillations generated by mutually inhibiting neurons with adaptation. *Biological cybernetics 56*, 5-6 (1985), 367–376.

[52] Matsuoka, K. Mechanisms of frequency and pattern control in the neural rhythm generators. *Biological cybernetics 56*, 5-6 (1987), 345–353.

[53] Matsuoka, K. Analysis of a neural oscillator. *Biological Cybernetics 104*, 4-5 (2011), 297–304.

[54] Matsuoka, K. Frequency responses of a neural oscillator.

[55] McCrea, D. A., and Rybak, I. A. Modeling the mammalian locomotor cpg: insights from mistakes and perturbations. *Progress in brain research 165* (2007), 235–253.

[56] Merz, M., Transeth, A. A., Johansen, G., and Bjerkeng, M. Snake robots for space applications(saros).

[57] Min, S., Won, J., Lee, S., Park, J., and Lee, J. Softcon: simulation and control of soft-bodied animals with biomimetic actuators. *ACM Transactions on Graphics (TOG) 38*, 6 (2019), 1–12.

[58] Mori, T., Nakamura, Y., Sato, M.-A., and Ishii, S. Reinforcement learning for CPG-driven biped robot. *AAAI Conference on Artificial Intelligence 4* (2004), 623–630.

[59] Munson, T. S., Facchinei, F., Ferris, M. C., Fischer, A., and Kanzow, C. The semismooth algorithm for large scale complementarity problems. *INFORMS Journal on Computing 13*, 4 (2001), 294–311.

[60] Nassour, J., Hénaff, P., Benouezdou, F., and Cheng, G. Multi-layered multi-pattern CPG for adaptive locomotion of humanoid robots. *Biological cybernetics 108*, 3 (2014), 291–303.

[61] Nvidia, C. Cusparse library. *NVIDIA Corporation, Santa Clara, California* (2014).

[62] Ozel, S., Skorina, E. H., Luo, M., Tao, W., Chen, F., Pan, Y., and Onal, C. D. A composite soft bending actuation module with integrated curvature sensing. In *Robotics and Automation (ICRA), IEEE International Conference on* (2016), IEEE, pp. 4963–4968.

[63] Park, M. G., Jeon, J. H., and Lee, M. C. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. 1530–1535.

[64] Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. *International Conference on Robotics and Automation* (2018), 1–8.

[65] Precup, D. *Temporal abstraction in reinforcement learning*. University of Massachusetts Amherst, 2000.

[66] Righetti, L., and Ijspeert, A. J. Programmable central pattern generators: an application to biped locomotion control. In *IEEE International Conference on Robotics and Automation* (2006), IEEE, pp. 1585–1590.

[67] Roberts, A., Soffe, S., Wolf, E., Yoshida, M., and Zhao, F.-Y. Central circuits controlling locomotion in young frog tadpoles. *Annals of the New York Academy of Sciences 860*, 1 (1998), 19–34.

[68] Rybak, I. A., Shevtsova, N. A., Lafreniere-Roula, M., and McCrea, D. A. Modelling spinal circuitry involved in locomotor pattern generation: insights from deletions during fictive locomotion. *The Journal of physiology 577*, 2 (2006), 617–639.

[69] Ryu, J.-K., Chong, N. Y., You, B. J., and Christensen, H. I. Locomotion of snake-like robots using adaptive neural oscillators. *Intelligent Service Robotics 3*, 1 (2010), 1.

[70] Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.

[71] Sanfilippo, F., Azpiazu, J., Marafioti, G., Transeth, A. A., Stavdahl, Ø., and Liljebäck, P. A review on perception-driven obstacle-aided locomotion for snake robots. In *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)* (2016), pp. 1–7.

[72] Sartoretti, G., Paivine, W., Shi, Y., Wu, Y., and Choset, H. Distributed learning of decentralized control policies for articulated mobile robots. *IEEE Transactions on Robotics 35*, 5 (2019), 1109–1122.

[73] Sato, Y. D., Nakada, K., and Matsuoka, K. Singular perturbation approach with matsuoka oscillator and synchronization phenomena. In *Artificial Neural Networks and Machine Learning – ICANN 2011* (Berlin, Heidelberg, 2011), T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds., Springer Berlin Heidelberg, pp. 269–276.

[74] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR abs/1707.06347* (2017).

[75] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR abs/1707.06347* (2017).

[76] Servin, M., Lacoursiere, C., and Melin, N. Interactive simulation of elastic deformable materials. In *Proceedings of SIGRAD Conference* (2006), pp. 22–32.

[77] Shabana, A. *Computational Dynamics, 3rd Edition*. Wiley, 2009.

[78] Si, H. Tetgen: A quality tetrahedral mesh generator and three-dimensional delaunay triangulator. *Weierstrass Institute for Applied Analysis and Stochastic, Berlin, Germany* (2006).

[79] Stewart, D. E. Rigid-body dynamics with friction and impact. *SIAM review 42*, 1 (2000), 3–39.

[80] Stewart, D. E., and Trinkle, J. C. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering 39*, 15 (1996), 2673–2691.

[81] Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems* (1999), S. Solla, T. Leen, and K. Müller, Eds., vol. 12, MIT Press.

[82] Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems* (Cambridge, MA, USA, 1999), NIPS'99, MIT Press, pp. 1057–1063.

[83] Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems* (2000), 1057–1063.

[84] Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence 112*, 1-2 (1999), 181–211.

[85] Thandiackal, R., Melo, K., Paez, L., Herault, J., Kano, T., Akiyama, K., Boyer, F., Ryczko, D., Ishiguro, A., and Ijspeert, A. J. Emergence of robust self-organized undulatory swimming based on local hydrodynamic force sensing. *Science Robotics 6*, 57 (2021), eabf6354.

[86] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. *International Conference on Intelligent Robots and Systems* (2017), 23–30.

[87] Todorov, E. Implicit nonlinear complementarity: A new approach to contact dynamics. In *Robotics and Automation (ICRA), IEEE International Conference on* (2010), IEEE, pp. 2322–2329.

[88] Tran, D. H., Hamker, F., and Nassour, J. A humanoid robot learns to recover perturbation during swinging motion. *IEEE Transactions on Systems, Man, and Cybernetics: Systems 50*, 10 (2020), 3701–3712.

[89] Transeth, A. A., Leine, R. I., Glocker, C., Pettersen, K. Y., and Liljebäck, P. Snake robot obstacle-aided locomotion: Modeling, simulations, and experiments. *IEEE Transactions on Robotics 24*, 1 (2008), 88–104.

[90] Transeth, A. A., Liljeback, P., and Pettersen, K. Y. Snake robot obstacle aided locomotion: An experimental validation of a non-smooth modeling approach. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2007), pp. 2582–2589.

[91] Wang, G., Chen, X., and Han, S.-K. Central pattern generator and feedforward neural network-based self-adaptive gait control for a crab-like robot locomoting on complex terrain under two reflex mechanisms. *International Journal of Advanced Robotic Systems 14*, 4 (2017), 1729881417723440.

[92] Wang, H., De Boer, G., Kow, J., Alazmani, A., Ghajari, M., Hewson, R., and Culmer, P. Design methodology for magnetic field-based soft tri-axis tactile sensors. *Sensors 16*, 9 (2016).

[93] Wang, H., Zhang, R., Chen, W., Wang, X., and Pfeifer, R. A cable-driven soft robot surgical system for cardiothoracic endoscopic surgery: preclinical tests in animals. *Surgical endoscopy 31*, 8 (2017), 3152–3158.

[94] Wang, Z., Gao, Q., and Zhao, H. CPG-inspired locomotion control for a snake robot basing on nonlinear oscillators. *Journal of Intelligent & Robotic Systems 85*, 2 (2017), 209–227.

[95] White, F. *Fluid Mechanics*. McGraw-Hill series in mechanical engineering. McGraw-Hill, 2008.

[96] Wu, Q., Zhang, C., and Liu, Y. Custom sine waves are enough for imitation learning of bipedal gaits with different styles. In *2022 IEEE International Conference on Mechatronics and Automation (ICMA)* (2022), IEEE Press, p. 499–505.

[97] Wu, X., and Ma, S. Neurally controlled steering for collision-free behavior of a snake robot. *IEEE Transactions on Control Systems Technology 21*, 6 (2013), 2443–2449.

[98] Yakovenko, S., McCrea, D., Stecina, K., and Prochazka, A. Control of locomotor cycle durations. *Journal of neurophysiology 94*, 2 (2005), 1057–1065.

[99] Yuste, R., MacLean, J. N., Smith, J., and Lansner, A. The cortex as a central pattern generator. *Nature Reviews Neuroscience 6*, 6 (2005), 477.