CTU

CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

Department of Computers

Master's thesis

Gait Adjustment of Hexapod Walking Robot after Leg Damage

Jan Feber

January 2024

Supervisor: Ing. Rudolf Jakub Szadkowski

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Feber Jan** Personal ID number: **482397**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Computer Science**

Study program: **Open Informatics**

Specialisation: **Artificial Intelligence**

II. Master’s thesis details

Master’s thesis title in English:

Gait Adjustment of Hexapod Walking Robot after Leg Damage

Master’s thesis title in Czech:

Změna chůze šestinohého kráčejícího robotu po poškození končetiny

Guidelines:

The hexapod walking robot can get damaged while deployed on a long-term autonomous mission and without damage compensation, the robot will not be able to continue the locomotion. The long-distance locomotion of walking robots is often handled using the approach of rhythmical motion control based on animal-observed neural oscillators [1, 2]. A walking robot's movement is based on so-called motion patterns/gait which prescribe the rhythmic movement of individual legs. In the case of limb damage, the motion pattern has to be adjusted to the changes caused by the damage, so the robot is able to continue in its long-term mission. The causes and consequences of the damage are in general unpredictable; hence, the adjustments need to be handled autonomously [3].
- Familiarize yourself with state-of-the-art related to rhythmical locomotion of hexapod walking robots with damaged limbs.
- Propose and implement a solution for rhythmic locomotion with damaged limbs using methods of machine learning.
- Test the solution in the CoppeliaSim simulator and if possible, deploy it on a real robot.
- Compare the proposed method with one of the state-of-the-art approaches, such as [4], [5], or [6].

Bibliography / sources:

[1] Cully, Antoine, et al. "Robots that can adapt like animals." Nature 521.7553 (2015): 503-507.
[2] Plasberg, Carsten, et al. "Adaptive Walking Pattern Generation with Damaged Legs for the six-legged Robot LAURON V." 2021 20th International Conference on Advanced Robotics (ICAR). IEEE, (2021).
[3] Mailer, Christopher, Geoff Nitschke, and Leanne Raw. "Evolving gaits for damage control in a hexapod robot." Proceedings of the Genetic and Evolutionary Computation Conference. (2021).
[4] Miguel-Blanco, Aitor, and Poramate Manoonpong. "General distributed neural control and sensory adaptation for self-organized locomotion and fast adaptation to damage of walking robots." Frontiers in Neural Circuits 14 (2020): 46.
[5] Ngamkajornwiwat, Potiwat, Pitiwut Teerakittikul, and Poramate Manoonpong. "Online gait adaptation of a hexapod robot using an improved artificial hormone mechanism." International Conference on Simulation of Adaptive Behavior. Springer, Cham, (2018).
[6] Ren, Guanjiao, et al. "Multiple chaotic central pattern generators with learning for legged locomotion and malfunction compensation." Information Sciences 294 (2015): 666-682.

Name and workplace of master's thesis supervisor:

Ing. Rudolf Jakub Szadkowski Artificial Intelligence Center FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **31.01.2023** Deadline for master's thesis submission: **09.01.2024**

Assignment valid until: **22.09.2024**

Ing. Rudolf Jakub Szadkowski Supervisor's signature	Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

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

.	
Date of assignment receipt	Student's signature

Declaration

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

Prague, January 9, 2024

. . . . . .

Jan Feber

Acknowledgement

Writing this thesis would not be possible without the help of my supervisor Ing. Rudolf Jakub Szadkowski, who introduced me to the topic of locomotion, bio-inspired controllers, and many other concepts. He has guided me through the whole research process since I started it in the third year of my bachelor's studies, helped me whenever I did not know how to continue the work, and supported me the whole time.

I also wish to thank the entire team of the *Computational Robotics Laboratory* for creating a supportive environment, where the research experts were willing to advise me anytime I asked them. I am very grateful for the opportunity of being part of scientific reasearch and for the experience of participating at conferences.

Thanks should also go to the teachers who passed on their knowledge to me during the years of my education, without which I could have never written this thesis, especially the teachers who taught me the fundamentals of mathematics and physics on which I have built during my university studies. In particular Mgr. Jiří Musil; Mgr. Jana Čermochová; RNDr. Zdeněk Papež and RNDr. Jan Thomas.

I would like to thank the whole team responsible for the study program *Open Informatics*, provided by the *Faculty of Electrical Engineering* of the *Czech Technical University in Prague*. They all do great work to improve the program and maintain its quality at a high level. All the tutors and lecturers are trying their best to pass their knowledge on to future generations, which I am thankful for.

I am deeply indebted to my student colleague Ing. Jakub Brož, who helped me so many times during my university studies.

I cannot begin to express my thanks to my parents, Ing. Jiří Feber and Ing. Jitka Febrová, who nurtured me, always supported me throughout my life and encouraged me to study. I would also like to extend my deepest gratitude to my brother Ing. Jakub Feber for being my role model since my childhood and that he has always stood by me. I could not leave out my sister-in-law Ing. Nina Feber, who supports me, encourages me to make the needed step to the unknown, if it is necessary, recommended me to study the *Open Informatics*, and encouraged my career in science. I am also grateful to my uncle RNDr. Vladimír Hanák, who has always been open to mathematics-physics-philosophy-related discussions, which the other family members have not found amusing nor worth discussing, and to his wife and my aunt Ing. Lenka Hanáková. Both of them have been very close to me while I was growing up and have always been like second parents to me.

I would like to express my deepest appreciation to my best friends Bc. Šimon Sajner and Jiří Macek, who have never let me down, no matter how difficult their situation was, have always trusted in me, supported me, and whom I could have always trusted. I hope I will be as good friend to you as you have been to me.

I am incredibly grateful to prof. MUDr. Jan Janda, CSc., who cured me when I was little He treated me prudently, chose the right therapy, and rescued me from the danger of wrong body development.

Last but not least, I would like to express my deepest gratitude to our Lord and Savior Jesus Christ, for His affection and patience, for His sacrifice, and for standing by me, even though I have not always been faithful to Him.

Abstrakt

K poškození nohy šestinohého kráčejícího robotu, které zamezuje další lokomoci, může při robotizaci lidské práce v nebezpečném prostředí, jakým jsou doly nebo zborcené budovy, snadno dojít. Aby byli lidé vyvarováni nebezpečí, musí robot fungovat autonomě. Výměnou robotů za lidi se ale prostředí nestane o nic bezpečnější, takže se robot při vykonávání práce může snadno poškodit. V případě poničení musí být robot schopen pokračovat ve svém úkolu, vrátit se k lidskému operátorovi kvůli opravám nebo aspoň rychle opustit místo, kde došlo k poškození, protože tam může hrozit další nebezpečí. Žijící šestinohé organismy, které jsou předlohami pro šestinohé kráčející roboty, jsou celkem rychle schopny se vzniklé amputaci svojí chůzí přizpůsobit. V této práci používám z přírody odpozorovaná pravidla, určující vztahy mezi pohyby jednotlivých nohou. Tato pravidla upravují závislosti v dynamickém systému, který po amputaci mění rytmus pohybu robota, aby vykompenzoval chybějící nohu. Metoda adaptuje rytmus na základě vnitřního stavu dynamického systému, takže nepotřebuje žádný kontinuální senzorický vstup. Výsledná lokomoce je vyhodnocena navrženou ohodnocující funkcí, která upřednostňuje především stabilitu rychlosti a orientace robotu. Výsledná metoda je porovnána s konkurenční metodou, založenou na zpracování senzorického vstupu, a překonává ji kvalitou ve dvou ze tří testovaných scénářů amputace a konzistentností ve všech testovaných scénářích. Obě metody jsou také porovnány s navrhovaným přístupem optimalizace pomocí evolučního algoritmu, kde toto testování odhaluje společné trendy v adaptaci lokomoce po amputaci nohy u všech tří metod, navzdory jejich odlišným optimalizačním kritériím. Výsledky podporují myšlenku, že použitá z přírody odpozorovaná pravidla jsou vhodnou heuristikou pro adaptaci lokomoce a nalezené podobnosti mezi výsledky lokomoce a jejího ohodnocení u jednotlivých metod poukazují na možné vylepšení definice těchto pravidel.

Klíčová slova: Šestinohý kráčející robot, Amputace nohy, Lokomoce, Adaptace, Vzor chůze, Evoluční algoritmus, Ohodnocující funkce, Ohodnocení chůze, Adaptace rytmu, Kompenzace poškození, Centrální generátor vzorů, CGV, Dynamický systém

Abstract

The hexapod walking robot can easily come to leg damage, preventing the robot from locomotion while replacing humans on a mission in dangerous environments, like mines or collapsed buildings. To keep people away from the threats, the robot has to work autonomously. However, switching out the robots and people does not make the environment any less hostile, so the robot can be easily damaged during its task. In the case of damage, the robot has to continue in its task, get back to the human operator for repairs, or at least quickly leave the dangerous place where it came to harm because more damage can happen. The living six-legged organisms, which are templates for the hexapod walking robots, are able to adjust their locomotion swiftly after the leg amputation. In this work, I use nature-observed rules dictating relations among movement of individual legs. The rules provide dependencies within a dynamic system adjusting the robot's movement rhythm after the amputation to compensate for the missing leg. The method adapts based on the state of the system itself; hence, no continual sensory feedback is needed. The resulting locomotion is evaluated by the proposed reward function favoring mainly the stability of locomotion speed and stability of the robot's orientation. The proposed method is compared with competing sensory-based state-of-the-art method and overcomes it in terms of performance in two out of three tested amputation scenarios with more consistent results in all tested scenarios. Both the competing methods are also compared with proposed evolution algorithm optimization approach, revealing similar trends in locomotion adaptation across all the compared methods despite their different optimization criteria. The results support the nature-observed coordination rules as a suitable heuristic for locomotion adaptation and suggest the rules improvement given the observed relations between resulting locomotion and its gained reward.

Keywords: Hexapod walking robot, Leg amputation, Locomotion, Adaptation, Gait pattern, Evolution algorithm, Reward function, Gait evaluation, Rhythm adaptation, Damage compensation, Central pattern generator, CPG, Dynamic system

Contents

List of Figures

List of Tables

List of Abbreviations

GP	gait pattern
MC	motion cycle
LMC	leg motion cycle
CR	coordinatio rule
CCAR	consecutivity and contralaterality adjustment rules
CPG	central pattern generator
RBFN	radial basis function neuron
EA	evolution algorithm
RL	reinforcement learning
OSS	over-safe state
RVS	rules violation state
BC	body-coxa (referes to joint on the robot's leg)
CF	coxa-femur (referes to joint on the robot's leg)
FT	femur-tibia (referes to joint on the robot's leg)

List of Symbols

$\phi_i(t)$	phase of i-th leg motion cycle (LMC) start
$\Delta\phi$	duration of swing movement
$\phi^{\text{cpg}}(t)$	CPG's phase in time t
$\phi^{\text{MC}}(t)$	phase of the robot's motion cycle (MC)
$\phi_i^{\text{LMC}}(t)$	phase of the i-th leg's motion cycle (LMC)
$\mathbf{y}(t)$	CPG's state in time t
P	MC and LMC period length
$\mathbf{w}_i^{\text{s}}(t)$	RBFN centers activating swing start
$\mathbf{w}_i^{\text{m}}(t)$	RBFN centers activating swing middle
$\mathbf{w}_i^{\text{e}}(t)$	RBFN centers activating swing end
$s_i^{\text{s}}(t)$	activation signal produced by activation of i-th leg's RBFN center $\mathbf{w}_i^{\text{s}}(t)$
$s_i^{\text{m}}(t)$	activation signal produced by activation of i-th leg's RBFN center $\mathbf{w}_i^{\text{m}}(t)$
$s_i^{\text{e}}(t)$	activation signal produced by activation of i-th leg's RBFN center $\mathbf{w}_i^{\text{e}}(t)$
$\mathbf{s}^{\text{sw}}(t)$	vector of swing indicators $s_i^{\text{sw}}(t)$
$s_i^{\text{sw}}(t)$	binary variable indicating i-th leg's swing: 1 if performing swing
$\mathbf{s}^{\text{dmg}}(t)$	vector of damage indicators $s_i^{\text{dmg}}(t)$
$s_i^{\text{dmg}}(t)$	binary variable indicating i-th leg's damage: 1 if damaged
$\mathbf{d}(t)$	vector of dangerous swing indicators $d_i(t)$
$d_i(t)$	binary variable indicating whether leg i can perform swing: 1 if not
$a_i^{\text{s}}(t)$	binary variable indicating rules violation in the first half of the i-th leg's swing
$a_i^{\text{e}}(t)$	binary variable indicating rules violation at the end of the i-th leg's swing
$v_i^{\text{s}}(t)$	leaky integration of $a_i^{\text{s}}(t)$
$v_i^{\text{e}}(t)$	leaky integration of $a_i^{\text{e}}(t)$
c^{v}	leaky integrate & fire integrating rules violation detection
c^{o}	leaky integrate & fire integrating rules over-safe state detection
$\hat{u}_i^{\text{bc}}(t)$	i-th leg's body-coxa joint relative position ($\in [-1, 1]$)
$\hat{u}_i^{\text{cf}}(t)$	i-th leg's coxa-femur joint relative position ($\in [-1, 1]$)
$u_i^{\text{bc}}(t)$	i-th leg's body-coxa joint position
$u_i^{\text{cf}}(t)$	i-th leg's coxa-femur joint position
$u_i^{\text{ft}}(t)$	i-th leg's femur-tibia joint position
ϕ^{config}	individual in EA representing the GP
g_{count}	number of evolution generations
n_{min}	the smallest allowed number of individuals in one generation
$\mathbf{v}(t)$	robot's velocity vector (velocity in x, y, and z)
$a(t)$	robot's acceleration (norm of velocity vector change between measurements)
$\mathbf{o}(t)$	vector of robot's orientation
$\mathbf{r}(t)$	reward achieved in time step t
$r^{\text{c}}(t)$	cumulative sum of rewards $\mathbf{r}(t') : t' \leq t$
r^{GP}	reward used for GP evaluation (mean of $r^{\text{c}}(t)$ over last MC)

Chapter 1
Introduction

If a robot gets damaged, it needs a relatively fast adaptation mechanism to overcome the damage and leave the dangerous place as fast as possible to avoid further destruction and continue in its task, or at least to get to the closest repair point. The aimed use of robots does often consider missions in hostile environments, where human life could be endangered (e.g., inspections, search and rescue missions or mining in environments like industrial areas [1], mines, underground, caves, collapsed buildings [2, 3], extra-terrestrial objects and planets [4], etc.); thus, robot replacing the human in such tasks could save lives. However, if the environment is dangerous for humans, it is usually unsafe for the robots too, increasing the chances of malfunctions and failure. The significant advantage of the hexapod walking robot in comparison with quadrupeds and bipeds is its redundancy in the number of legs, which we would like the robot to utilize in the adaptation process in case of the robot's leg damage/amputation.

The design of legged robots is inspired by nature: bipeds are based on human anatomy, quadruped models originate in dogs, deers, horses, etc., and hexapod robot designs are inspired by stick insects, ants, cockroaches or beetles; hence, the robots' locomotion is also inspired by their animal templates. The inspiration comes from nature because the physique has already undergone millions of years of testing, inherently solving many problems by its design.

Despite the designs being inspired by known organisms, controlling their locomotion is not an easy task, mostly due to a large number of degrees of freedom. The six-legged robot usually consists of 18 joints, where each joint needs to be continuously fed by motion commands. Based on the used method and its goal, such parametrization can lead to thousands of degrees of freedom. Nevertheless, over the years, many relations and dependencies in locomotion have been described and used to make the search for suitable locomotion easier.

However, the locomotion with an amputated leg introduces another challenge because the locomotion of a fully functional robot does not transfer well to the problem of amputation. Although the researchers observe results of living-organisms locomotion adaptation after a leg injury, the adapting mechanism is not yet fully understood [5, 6].

Therefore, the robotics use a variety of methods tackling the damage compensation problem, like evolution optimization [7], reinforcement learning [8, 9], traditional control approaches [10], or they develop a dynamic system based on sensory feedback or other biomimetic mechanism, like neural oscillators [11], artificial hormones [12] or nature observed *Coordination Rules* (CRs) describing relations between the movement of individual legs [13].

The approaches often use sensory feedback (force, torques, robot orientation, etc.) as input for their dynamic system, adjusting the locomotion rhythm and sometimes also the amplitudes of the motor commands. However, the sensory feedback can produce noised or invalid data due to the damage that caused the leg amputation.

My method adjusts the locomotion rhythm based on nature-observed CRs extended for amputation scenarios with no sensory feedback, using only the information about which leg is damaged. The CRs put restrictions on the legs, forbidding or encouraging similarity between their movement. Hence, the proposed method does not rely on possibly unreliable continuous sensory feedback.

I compare a reactive approach relying on sensory feedback with no knowledge about particular damage with its "*orthogonal*" approach of my method, knowing only the information about damage having no other sensory feedback. The methods' scores are given by the proposed reward function,

designed to evaluate the robot's locomotion.

The compared methods adjust the timing of individual legs' forward and backward movement, given by parametrized motion primitives, within the repeating robot's movement. Thanks to the parametrization, the complex locomotion of the hexapod walking robot is reduced to a few tunable parameters.

In contrast to the compared online learning methods, I propose an evolution algorithm tuning the parameters. The evolution algorithm optimizes the proposed reward function value by tunning the locomotion parameters, hence giving an upper bound of what the competing online methods, tunning the same parameters, are able to achieve.

The gap between the upper bound, given by the computationally demanding evolution algorithm, and the online adapting methods is discussed, concluding the computational demands of evolution optimization are not worth the small reward gain it provides. However, a point of interest is that despite different optimization criteria, all three compared methods tend to adapt the locomotion after amputation towards similar results.

The results prove that, according to the proposed reward function, the proposed feedback-free method is more consistent and, in two out of three scenarios, better than the compared feedback-based method, thus justifying the CRs-based rhythm modification and importance of the damage origin knowledge.

The remainder of this thesis is organized as follows: In Chapter 2, I present the introduction to legged locomotion and gait patterns, together with the work related to the thesis topic. The problem statement can be found in Chapter 3. The proposed sensory-feedback-free method based on adjusted CRs is described in Chapter 4. Chapter 5 introduces the simulation setting, the proposed reward function for locomotion evaluation, adjustments made to the sensory-feedback-based competitive approach proposed by Miguel et al. [14], and the evolution algorithm providing the upper bound for the compared methods.

Results presentation and methods comparison are provided in Chapter 6, with a discussion following in Chapter 7, which also includes some insights into the relation between the coordination rules, expected reward values, and obtained reward values for achieved gait patterns. Another discussed topic is the comparison of the online learning methods, represented by the proposed method and Miguel's approach, with the evolution algorithm. The thesis concludes in the final Chapter 8.

Chapter 2
Background and Related Work

This Chapter is divided into two main sections. In the first Section 2.1, I introduce the concepts and basic principles of legged locomotion, together with the coordination rules on which the proposed method is based. The second Section 2.2 follows with an overview of different approaches and methods the robotics use to invoke the hexapod walking robot locomotion and how such concepts are (sometimes) extended to a damage adaptation.

2.1 Introduction to Legged Locomotion of Hexapods

The various approaches share a common foundation: utilizing the periodicity to simplify the complex problem of legged locomotion having to control many servo motors (usually eighteen in the case of hexapod walking robots). This section introduces the fundamentals of fully functional legged robots' locomotion and the coordination rules describing relations between individual legs movement crucial for the proposed method.

2.1.1 Locomotion of Fully Functional Walking Robot

The two fundamental branches of legged locomotion can be categorized as (i) precise motion planning for difficult terrain and (ii) the topic of this thesis, the use of *gait patterns* for regular terrain. The branch (i) is advantageous if the traversed terrain has many pitfalls and requires precise foot-placing to avoid an irreversible robot state (e.g., a fall into the water while crossing the river using protruding stones). Nevertheless, such approach is computationally very demanding and is situation-specific, requiring a reasonably good scan of the traversed terrain and the robot's model with weight distribution and body and leg dimensions to create a precise plan.

For traveling longer distances of regular terrain, there is no need to precisely plan the movement of every joint because minor irregularities occurring there do not usually put the robot at risk of irreversible state. Therefore, to decrease the demand for computational resources, which can be utilized for other tasks, it is reasonable to use the approach (ii) inspired by a nature-observed feature of living organisms' movement, the *Gait Patterns* (GPs) based on the movement repetitiveness.

The GP is a repetitive motion pattern prescribing the mapping between the phase of the motion and the movement of individual legs and their joints. GPs are widely recognized in the locomotion of all living organisms, whether it is trot, walk, gallop, or any other named or unnamed repetitive pattern of movement. The motion of each joint and each leg repeats periodically; hence, I use an expression *Leg's Motion Cycle* (LMC) denoting the periodic movement of an individual limb.

The LMCs of individual legs are mutually synchronized and ordered within the repeating movement of the whole body, here called the *Motion Cycle* (MC). The MC's phase, in combination with the phase in which the LMC starts within the MC, clearly determines the movement phase of the respective leg. The ordering of the LMCs starts within the MC is both determined by and determining the GP.

The LMC consists of two altering movements: the *swing* and the *stance*. The swing is the forward movement when a leg is lifted of the ground moving forward (i.e., it is making the step forward). The stance is the leg's backward movement when the leg is lying on the ground, supporting the body's weight and moving backward in relation to the body.

When no irregularities occur, the leg moves the same distance forward during the swing and backward during the stance. Therefore, the leg's altering forward and backward movement distance is the same, but the time spent in the swing can differ from the time spent in the stance; hence, the forward and backward leg speed can vary. The LMC consists of the swing followed by the stance (i.e., the combined duration of one swing and one stance gives the duration of one LMC), and the combined duration of one swing and one stance equals the length of MC's period.

An important detail is the *swing duration*, which is usually the same for all legs. Frequently mentioned in the other works is the *duty factor*, the fraction of the LMC period for which the leg performs stance. Because the LMC consists of swing followed by stance, both the swing duration and duty factor are defined by defining one of them. In this thesis, the swing duration is used.

Generally, the name GP encapsulates the whole motion pattern, not only the LMC starts and swing duration, but also the mapping from the LMC phase to the motor commands, i.e., the GP can be adjusted not only by the change of rhythm but also by change of servo motor movement amplitudes. To avoid confusion, in this thesis, the LMCs starts placement within the MC, together with the swing duration (or swing durations if we consider individual value for each leg), are called the *GP rhythm*.

It is believed that living organisms maintain the rhythm with the use of oscillators. The observations of living organisms revealed oscillators existing in their spinal cord (or in other subregions of the central nervous system in simpler organisms, like insects). Findings suggest that such oscillators in the hierarchy of animals' motor system (central neural system - neural oscillators - effector organs) have a role of generating a rhythm underlying the periodic movement [15,16]. Such oscillators consist of several coupled neurons, producing a recurrent neural network. The neural oscillators are called *Central Pattern Generators* (CPGs).

Some simplified CPG models are mathematically described and used in robotics. The CPG's key feature is its ability to generate rhythmical output without any rhythmical input, which is automatically restored if any outside influence perturbs the rhythm. This CPG feature is often used in the locomotion controllers design for automatic small perturbation handling caused by uneven terrain, for instance. Another useful and frequently exploited feature is the ability of multiple coupled CPGs to synchronize.

The CPG's ability to synchronize with other CPGs and to restore its rhythm shows itself while observing the CPG's state space. From all states in the state space, the CPG's state trajectory converges to a closed loop called the *limit cycle*. The CPG's state following the limit cycle trajectory determines the CPG's, and implicitly also the MC's or LMC's, phase.

Researchers in the community have proposed many different approaches, trying to use CPG/CPGs and the above-described features with various modifications and roles within the proposed locomotion controllers. Some of the approaches regarding the hexapod walking robots are described in Section 2.2.

2.1.2 Basic Gait Patterns of Hexapod Walking Robot

The morphology of a hexapod walking robot dictates its suitable GPs based on observations of cockroaches, stick insects, ants, and other six-legged organisms' locomotion. The usual design consists of three (front, middle, and hind) pairs of legs, where the legs within the pair are on contralateral/opposing sides of the robot's body. In other words, the robot design tends to copy the living organisms, see Figure 1. Therefore, it is not surprising that different optimization algorithms converged to GPs observed in living organisms, as proposed by millions of years of evolution.

The biomimetic GPs are the most frequently used GPs for the hexapod walking robots. The following paragraphs describe the three most occurring hexapod GPs in the robotics community.

The most common GP for a hexapod walking robot is the *tripod gait*. In general, a tripod gait is a name for any GP, fulfilling the condition of being supported by three legs in stance at every moment of the MC. Nevertheless, the tripod gait divides the legs into two triplets: (i) left front and hind legs combined with right middle leg, and (ii) right front and hind legs combined with left middle

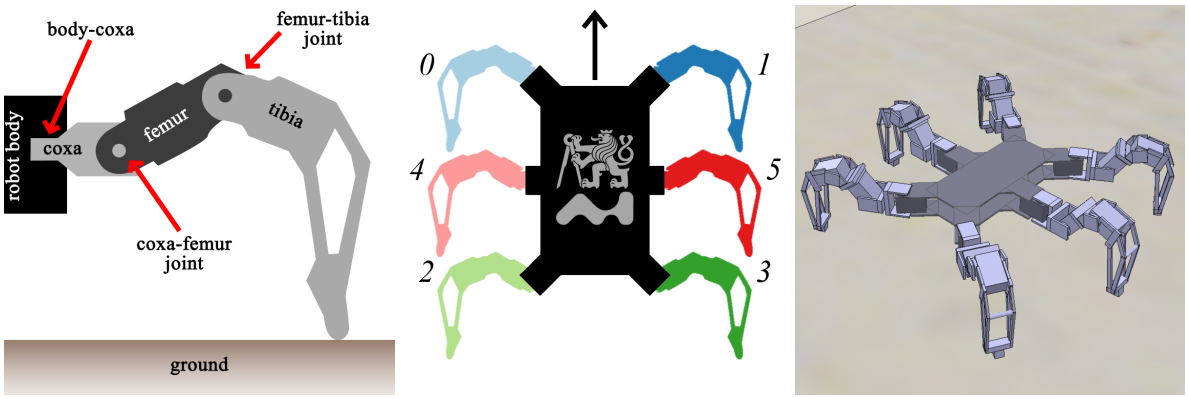

Figure 1: On the right is the screenshot of the robot from the simulator. The scheme of the robot with labels of individual legs is depicted in the middle. The color code of the legs holds in all other figures regarding legs in this thesis. The arrow shows the forward direction of the robot's walk. On the left is the schema of one leg with individual parts and the joins between them.

leg. Legs within those triplets move synchronously, performing the same actions simultaneously, i.e., their LMCs starts are equal. The triplets alter each other in swing and stance movements, resulting in three legs being in swing and three legs being in stance in each moment (i.e., the triplets perform their movement in antiphase), see Figure 2. In the tripod gait, the swing and stance durations are equal, i.e.,

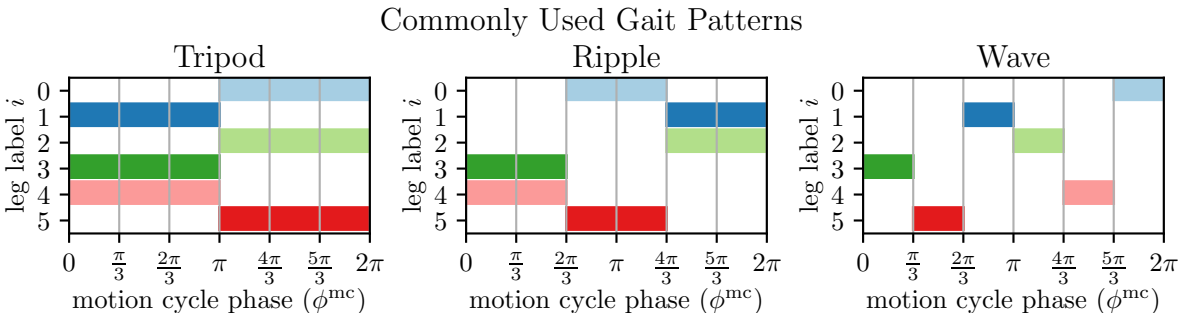

Figure 2: Showcase of the fundamental GPs. Three GPs are depicted: tripod gait, ripple gait, and wave gait, from left to right, respectively. The leg label i is on the y axis, while the MC phase ϕ^{mc} is depicted by the x axis. The color bar represents the respective leg's swing phase, with its left border representing the swing start $\phi_i(t)$ and length representing the swing duration. The individual leg colors and labels correspond to the schema in Figure 1.

half of the MC period, ensuring no overlap in the triplets' actions. Therefore, in each moment, the maximum number of legs are performing a step forward while the robot still maintains its stability due to the other three legs ensuring the robot's center of mass is above the polygon created by the legs' foot tips on the ground. Hence, the tripod gait is usually the best solution for maintaining the robot's stability while performing a relatively fast walk. The geometrical features of the hexapod morphology even suggest the tripod gait is very suitable for fast locomotion, maximizing the stride length, thus, maximizing the speed with a fixed number of stepping cycles [17].

Other very commonly used patterns are various types of *ripple gait*.[1] During the ripple gait, two legs always perform swing, and four perform stance at each moment. Some variants use separation into three pairs of legs: (i) right hind and left middle, (ii) right middle and left front, and (iii) right

[1]Ripple gait has many different names, and the terminology is inconsistent within the robotics community. The same names are often used for different GPs.

front and left hind, or the mirrored version of such division. Legs within those pairs undergo the same movement, i.e., their LMC starts are equal, and the LMC starts of the pairs within the MC are ordered as (i), then (ii), and (iii) at last, see Figure 2. Due to the requirement on the number of legs performing swing and stance at each moment, the swing duration has to be shorter than in the tripod gait case. The swing duration (equal to the phase offset between the LMC starts of pairs (i) and (ii), and (ii) and (iii)) is therefore equal to two-thirds of the MC period.

Other approaches using ripple gait do not join legs into pairs with equal LMC starts but start the individual LMCs one at a time, in order (i) right hind, (ii) left middle, (iii) right middle, (iv) left front, (v) right front, (vi) left hind, or a symmetrical version of such ordering. The phase offset of consecutive LMC starts (e.g., phase offset between right hind and left middle legs' LMCs starts) is then equal to one-sixth of the MC period, while the phase offset of two consecutive legs (e.g., left hind and left middle), is still the two-thirds of the MC period.

Finally, the third of the most frequently used GPs, the *wave gait*, which moves forward one leg at a time with swing duration equal to one-sixth of the MC period, maintaining five legs in stance in every moment of the MC, see Figure 2. The wave gait is slow, but the robot remains stable and can move more precisely if needed. Therefore, the wave gait is helpful in restricted spaces, where more careful maneuvering is needed, or the slower speed can lower slipping on slippery surfaces, like ice, for instance.

The tripod gait, ripple gait, and wave gait are well known within the community, but the approaches to invoke the gaits differ using different controller designs, which are described in detail in the following sections. The GPs hold nature-observed *coordination rules* introduced in the following section, serving in some form as a base for many bio-inspired hexapod locomotion controllers.

2.1.3 Gait Patterns in Context of the Coordination Rules

Observing the tripod, ripple, and wave gaits in living organisms and robots resulted in a formulation of rules in the work of Dürr et al. [13]. The rules were made while observing the locomotion of six-legged organisms with no leg trauma. Hence, the rules apply to fully functional six-legged robots. The stated rules include the relations of legs in the context of the GP rhythm, the load distribution, and leg placing. For purposes of this thesis, the rules about rhythm are relevant, i.e., the rules adjusting the coordination of the legs' movement; therefore, I call them the *Coordination Rules* (CRs), stated here as follows:

1. While a leg is lifted off, suppress the lift off of its consecutive leg.
2. If a leg touches the ground, initiate the lift off of its consecutive leg.
3. Do not lift off the contralateral legs at the same time.

The *consecutive leg* for the hind leg is the middle leg on the same robot's body side. Similarly, the consecutive leg for the middle leg is the front leg on the respective robot's body side. The *contralateral legs* are the legs on the same positions from the opposing robot's body sides (e.g., left and right hind legs). The consecutivity and contralaterality relations are visualized in the most left schema of Figure 3.

Taking a closer look at the presented GPs in the previous Section 2.1.2, all of them hold the presented CRs. In the tripod gait, the front and hind legs perform the swing together, while the middle leg (the rear leg's consecutive leg) is in stance; therefore, CR 1 holds. The end of the front and hind legs' swing initiates the swing of the middle leg, which is the consecutive leg of the hind leg, i.e., the CR 2 holds. Finally, the mentioned triplets of leg alter in the swing and stance phases, and in each triplet is exactly one leg of the pair of contralateral legs; hence, the CR 3 also holds. If the situation is in reverse (the middle leg is lifted and the front and hind legs are in stance), the middle leg is in

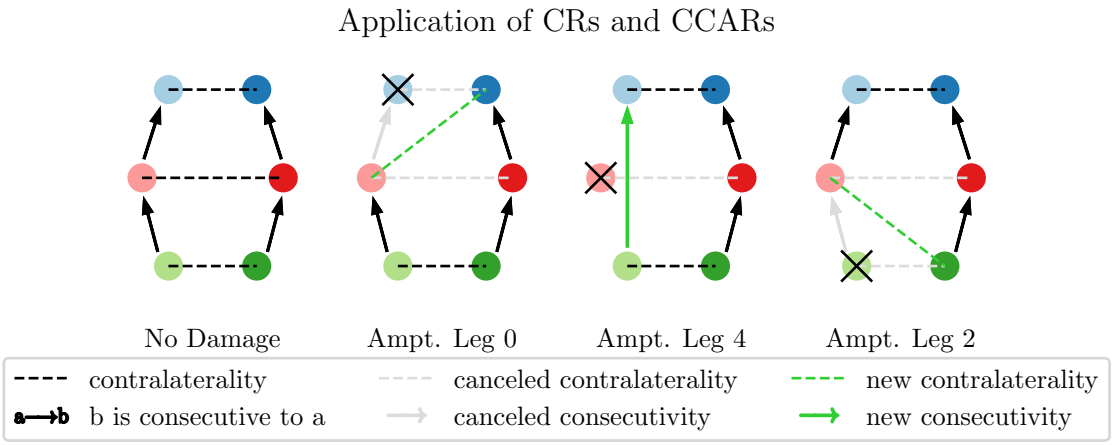

Figure 3: The Figure presents the relations between legs in different scenarios. The fully functional robot is presented on the left. The black dashed line represents the contralaterality relation between the legs. The black arrow from leg i to leg k means the leg k is consecutive to leg i. The other three schemas show the application of CCARs after the amputation, where the amputation of front left, middle left, and hind left legs is depicted in the schemas from left to right, respectively. The green dashed lines and arrows show the newly formed relations given the CCARs, while the grey dashed lines and arrows depict the no longer valid relations canceled by the CCARs in the respective amputation scenario. The individual leg colors correspond to the schema in Figure 1.

swing, while the lift off of its consecutive (front) leg is suppressed, i.e., the CR 1 holds. The end of the middle leg's swing initiates the swing of its consecutive (front) leg; hence, the CR 2 also holds.

In the ripple gait and wave gait, the legs on each side initiate their movement one after another (hind, middle, and front); therefore, CRs 1 and 2 hold automatically. In wave gait, each leg performs the swing by itself; hence, the CR 3 can not even be broken. The ripple gait always swings two legs simultaneously, but never the pair of contralateral legs (left middle - right hind, left front - right middle, left hind - right front, for instance, or a symmetrical version of the mentioned pairs). Therefore, the CR 3 also holds.

As mentioned above, the CRs are formulated for fully functional robots. However, they provide little to no information about rules in scenarios of leg damage. The following section presents an extension of the CRs for scenarios with leg amputation.

2.1.4 Coordination Rules Extension

The CRs, described in the previous Section 2.1.3, do not provide a clear guide on how the rules behave in case of leg amputation. Hence, in our previous work [18], we proposed extension of the CRs, which we call *Consecutivity and Contralaterality Adjustment Rules* (CCARs):

1. The most-front left (right) functional leg is a contralateral leg for the most-front right (left) functional leg.

2. The most-hind left (right) functional leg is a contralateral leg for the most-hind right (left) functional leg.

3. If the leg i is the contralateral leg of the leg j that becomes the contralateral leg for the leg k due to the proposed rule 1. or 2., then the contralateral relation between the legs i and j is no longer active.

4. If the leg j: (i) has consecutive leg k, and (ii) is the i-th leg's consecutive leg, and (iii) is not functional; then the leg k becomes the i-th leg's consecutive leg.

The reader of this text can realize that the CCARs are also applicable for the loss of more than just one leg. Experiments with two amputated legs using more complex controller architecture on real hexapod walking robot, lacking deeper comparison with competitive approaches, were presented in the already mentioned original work [18].

The visualization of the CCARs behavior in the leg amputation scenarios is depicted in Figure 3. The combination of CRs and their extension, the presented CCARs, provide guidelines for the dynamic system interactions, adapting the GP rhythm to amputation, proposed in this thesis.

2.2 Different Locomotion Approaches

The aim of this thesis is to propose a solution for GP rhythm adjustment for a robot with a damaged/amputated leg. The methods dedicated to locomotion adjustment after leg damage usually extend already existing methods designed for fully functional hexapod walking robots.

Most often, the damaged leg is considered amputated, elevated in such position to not conflict with the motion of other legs, and is no longer considered usable. The only branch of methods frequently handling many different kinds of damage while still using the damaged leg to some extent are the *Evolution Algorithms* (EAs) and *Reinforcement Learning* (RLs) methods because they simply try to optimize a reward function given the new sensory feedback. Hence, the methods are able to adapt to almost any possible damage, even replacing part of the broken leg with a piece of wood, for instance.

The ongoing research already suggested many different control approaches for fully functional, and also some for damaged, hexapod walking robot locomotion. For a deeper analysis of different development branches (of mostly fully functional hexapods), I strongly recommend reading the overview of the trends in hexapod robot control in the last ten years by Coelho et al. [19]. Here, I just briefly outline the main ideas behind different development trends in hexapod walking robot locomotion.

2.2.1 Traditional-Control Approaches

The traditional control approaches consider the legs as robotic manipulators. The kinematic-based techniques mostly use the robot model and different sensor types to calculate whether the robot's center of mass projection is within the robot's *support polygon* and to handle the environment irregularities [10,20–22]. Widely used is the *Static Stability Margin* method [23] for gait stability evaluation, computing the distance between the center of mass projection and the edge of the support polygon. The algorithms often rely on feedback from ground contact forces and many other sensors. The robots are even tested in environments with steps, ramps, and other similar types of obstacles that they climb or avoid [10, 24].

Dynamic-based control methods are also frequently based on the robot's state towards the support polygon. However, instead of assuming quasi-static circumstances, they consider the influence of the robot's motion. Introducing the dynamics improves the robot's abilities in obstacle avoidance and also enables the possibility of energy consumption evaluation [25–28].

The third group of traditional approaches can be considered real-time planning, using mostly *Rapidly Exploring Random Tree* [29, 30]. These methods, like many planning methods in other disciplines, sometimes implement other advanced mechanisms to take care of some parts of the planning, like *Artificial Neural Networks* and *Fuzzy Logic* [31]. Nevertheless, like other traditional control-based methods, a robot model with precise dimensions and described movement capabilities is needed.

In the scenario of damage, the traditional control methods usually hold on maintaining the center of mass within the support polygon [32, 33]. There are also innovative approaches considering adaptable robot's architecture sliding its legs along the side of the robot's body and improving the stability [34].

The requirement for a precise model in traditional approaches can be troublesome if the environment changes the robot, e.g., mud sticks on the robot's legs, which become heavier. The change in sensor readings can significantly influence the planning as it distorts the state of the robot. Given the planning constraints, the state change can even make the search unfeasible if the distortion is significant. On the other hand, rhythm-based controllers using the CPGs utilize the sensory input mostly to adjust the GP rhythm, possibly distorting the rhythm in extreme cases but not making the locomotion impossible.

Even in traditional control methods, the concept of GP is used to simplify the process of creating the complex motion of, most often, eighteen servo motors forming the hexapod. The locomotion decomposition using the CPG as rhythmic underly for the GP is used to even greater extent in the bio-inspired methods, described in the following section.

2.2.2 Bio-Inspired Controllers

Bio-inspired controllers are based on nature-observed principles functioning in living organisms. Some methods use the concepts purely as inspiration for their models, using only the idea of the principles. On the other end of the spectrum are researchers who try to take the specific nature-observed mechanisms and replicate them as truly as possible. Most GP-based controllers use the concept of CPGs: their coupling or their connection with sensory feedback.

This work considers the *Evolution Algorithms* (EAs) and *Reinforcement Learning* (RL) approaches as part of the bio-inspired controllers, specifically for two reasons: (i) they often use the CPGs to the same extent, as other bio-inspired methods, using the EA or RL for search of coupling weights or CPG parameters; (ii) the idea of EA and RL are themselves inspired in nature-observed mechanisms. Thus, I split the group of bio-inspired controllers into two parts: (i) *reward evaluating controllers*, the EA and RL controllers need to repeatedly evaluate the gait while learning, and (ii) *dynamic system controllers*, the CPG network with its learning mechanism and with or without sensory feedback connection and the CPG signal to motor commands mapping is usually one connected dynamic system handling all the problems online.

2.2.2.1 Evaluating Controllers

The RL and EA methods are good in search space exploration; hence, they usually give very good results in optimizing a given metric finding and exploiting promising solutions. However, there are three issues resulting in possible limitations.

Firstly, the common problem of RL and EA approaches for GP-learning is the high dimensionality. The RL and EA can optimize almost anything if given enough time and space, which can both be required in enormous amounts if optimizing too many parameters at once. Therefore, it is essential to parametrize the GP-determining features as compactly as possible. The choice of the parametrization is usually a challenging task by itself.

Secondly, the EA and RL methods need a lot of data to learn from. During the learning process, a lot of information has to be stored. We can only learn from what was already tried, and if we do not store the data, we need to try possibly already explored possibilities, which can be a dead end. There exist methods focusing on effective data storage for GP learning, e.g., *behavior-performance map* [8]. Nevertheless, a significant amount of data must be kept to use the methods' potential fully.

In the case of damage, the robot must leave the dangerous place as fast as possible, not having time to gather the data nor explore the enormous search.

Thirdly, which does not apply to all EA and LR methods, the learning uses a simulator, resulting in (i) a Sim-to-Real problem, which summarizes the differences between the simulator and the real-world (sometimes the algorithm even learns to exploit a simulator's quirkiness, which is non-existing in real world [7]), and (ii) the problem of running the simulator in the robot, or the need to connect to a server

providing simulator. Few methods do not need the simulator and test all on the real robot. However, such learning, especially in the EA case, can lead to dangerous robot behavior while exploring possibly damaging movements when the robot can irreversibly damage itself. Also, running the simulations is often time-consuming and computationally expensive.

The reward evaluating controller methods often implement advanced strategies improving the optimum search, e.g., the RL approaches use *Intelligent Trial and Error* for balancing exploration and exploitation of policies [8, 9, 35]. There are also experiments with incorporating other concepts into the learning procedures like *monte carlo* [36] or *neural networks* [37]. Attempts are also made with different sensory inputs, like gyroscopes [38], ultrasonic sensors [39], external camera [40], etc., to enhance the robot's capabilities (e.g., obstacle avoidance).

Overall, if given the correct setting, the EA and RL can solve GP optimization for almost any scenario, including damage or part of a leg replaced by a piece of wood. The methods considering damage usually try to take advantage of already gathered data from the fully functional state and use them to improve the learning in damaged state [8]. Nevertheless, the computational demands of the EA and RL methods do not always meet the possibilities the damaged robot deployed on a mission in a hostile environment has.

2.2.2.2 Dynamic System Controllers

Dynamic system controllers usually consist of one or more CPGs generating rhythm for swing and stance alternation, controlling a mechanism that transforms the CPG signal into motor commands for individual joints. The key feature of such controllers is the real-time approach to problem-solving. The controllers are designed to handle the emerged problems on the fly, differing significantly in comparison with previously introduced *reward evaluating controllers*.

The architectures differ in all possible aspects, for instance, the CPG count, used CPG model, coupling strategy, CPG signal to commands mapping, or sensory data fusion and its further usage. The variety of proposed architectures suggests that the problem is still ongoing, and none of the proposed solutions overcomes the others in all aspects.

Regarding the number of used CPGs, the most common is the use of six CPGs, one for each leg [11, 41], or in other words, each CPG generates rhythm for one LMC. More rarely, the architecture with eighteen CPGs, one for each joint with three joints per leg, is used [42]. There are also approaches using three CPGs per robot, one CPG per one pair of contralateral legs [43], or other various suggestions. Other works, including this thesis, use one CPG per robot generating rhythm for MC.

The CPG models are of various types, and many of them are named, like the Van der Pol oscillator used in [44], Hopf chosen by authors of [40] or Matsuoka oscillator deployed in [42]. The CPG models also differ in the number of dimensions their state space has and in the shape of their limit cycle.

CPG coupling strategy is usually an integral part of the proposed solution design in bio-inspired methods. The way CPGs are or are not coupled determines the system's behavior. Therefore, there is great variety among the coupling approaches. There are methods connecting every CPG to all other CPGs [42]; also, the concept of one master CPG innervating the other CPGs was proposed [11], etc. A few proposed solutions also change the CPG coupling based on desired GP [45]. Another curious approach is not connecting the CPGs directly at all but rely on the sensory feedback (ground contact force) providing enough inhibiting input for the CPG synchronization [14]. One of the well-known architectures is the WALKNET [13], which is developed continuously [46, 47].

Converting CPG signals to motion commands is a topic by itself from which different development branches emerge. A lot of proposed techniques define a continuous function that directly converts the CPG signal into signals setting positions of servo motors [44, 48]. Few approaches use the signal as a binary switch to recognize whether the leg should perform swing or stance [49]. In this thesis, we use CPG-RBFN architecture [42, 50], where RBFN stands for *Radial Basis Function Neuron*, which can be used as a binary switch or it can be extended to multi-phase switch.

In the CPG-RBFN architecture in this thesis, the RBFN is a neuron placed in the CPG's state space activated by the closeness (i.e., distance, hence "*radial basis*" function neurons) of the CPG's state. The activation triggers a pulse in the output signal, which can represent any desired phase within the CPG's cycle, corresponding to a particular phase in a particular LMC. In other words, the RBFN activation triggers specific leg action. The advantage of using the CPG-RBFN architecture is its flexibility. There can be as many RBFNs as we need, signaling whatever action we want within the phase. Opposed to the continuous function, converting the CPG signal needs a specific function for a specific CPG. Also, the CPG can change its output signal due to synchronization, perturbations, etc., changing the CPG output signal significantly and therefore requiring a change of the mapping function. In contrast, the RBFNs can be used for any CPG model in the exact same way. There can be issues to learn the neurons to be placed in close vicinity of the CPG's limit cycle, especially for CPGs with analytically inexpressible phase. For this issue, we proposed a solution in [51].

Regarding the bio-inspired dynamic system-based methods of handling leg damage, I also include two novel solutions proposing unique approaches: The first one is the use of artificial hormone (the artificial hormone mechanism was originally proposed in [52]), which automatically handles the problems caused by damage [12]. The Second one is the work of Owaki et al., where the *tegotae* principle of sensory feedback influencing the CPG's phase by detecting unwanted behavior is introduced [53].

A problematic area is the detection of malfunction. The malfunction is usually detected by comparing the expected and measured sensory feedback. It is a non-trivial task to find out whether the sensory feedback signaling damage is just a sensor malfunction, a momentary change caused by terrain (e.g., the foot stepped into a pit), a change caused by the environment (sticking mud increasing the needed force for motion, loose sand, etc.) or actual damage. Nevertheless, the problem of damage identification is out of the scope of this thesis.

There is a wide variety of approaches handling the hexapod walking robot locomotion. Traditional approaches need detailed robot model, which can be made obsolete due to damage or environmental influences, like sticking mud on the robot legs. Almost all the approaches rely on sensory feedback, whether it is for measuring the robot state in traditional approaches, evaluating the GP in reward evaluating methods, or adjusting it, like the dynamic system controllers do. However, in case of damage, the sensors can become unreliable and even make the adaptation method unfeasible. The difference is also in deployment criteria, with some methods being able to adapt the gait in real-time, such as dynamic system methods and real-time planning methods, while others require considerable time to adjust, particularly evolution algorithms, reinforcement learning, and motion planning, increasing the danger of the robot by remaining in the potentially risky place where the injury happened. In this work, I propose an online adapting solution adjusting solely based on the dynamic system relations with no need for continuous sensory feedback.

Chapter 3
Problem Statement

This thesis aims to adjust the GP rhythm after a leg amputation. The GP can be decomposed into two parts: (i) the rhythm control, defining when the actions take place, and (ii) the movement control, taking care of what the performed actions are. The two actions are swing and stance with their corresponding motion primitives. When the actions are performed and how long is their duration is given by the GP rhythm. The CPG model, described in the following paragraphs, drives the GP rhythm causing the RBFNs activations that trigger the motion primitives driving the servo motors.

The GP rhythm consists of LMC starts $\phi_i(t)$ for $i = 0, \ldots, 5$, each representing the swing movement start phase of the i-th leg within the MC, and the swing duration $\Delta\phi$. Ordering of the LMC starts $\phi_i(t)$ within the MC means assigning each $\phi_i(t)$ a value from $[0, P)$, where $P = 2\pi$ is the period length of the MC, and implicitly also the period length of the LMC. The swing duration $\Delta\phi$ is the same for all legs. Its lower bound corresponds to the wave gait swing duration (hence, lower values do not make sense because the wave gait already swings only one leg at a time), while the upper bound corresponds to the tripod gait swing duration (hence, greater values do not make sense if the stability should still be maintained, because, at some point within the MC, less than three legs would support the robot's body). Therefore, the values are bounded within interval $[\frac{P}{6}, \frac{P}{2}]$ corresponding to $[\frac{\pi}{3}, \frac{\pi}{2}]$ for $P = 2\pi$. For the scenario with an amputated leg, the formal description remains the same, except we do not care about the amputated leg j; thus $i \in \{0, 1, 2, 3, 4, 5\} \backslash \{j\}$.

The MC phase state is given by the CPG's phase state. In this thesis, I use a simplified CPG model [54], in which the phase is defined by a single constant, making the phase easy to track. The CPG model is defined by its differential with respect to time $\frac{\mathrm{d}\phi^{\mathrm{cpg}}(t)}{\mathrm{d}t}$ given by the following equation:

$$\frac{\mathrm{d}\phi^{\mathrm{cpg}}(t)}{\mathrm{d}t} = \omega + p(t) \sin\left(\phi^{\mathrm{cpg}}(t)\right), \tag{1}$$

where $\omega = 1$ is a positive constant ensuring constant phase growth, and $p(t)$ denotes the external input to which the CPG can synchronize. The MC phase $\phi^{\mathrm{MC}}(t)$ is given as the remainder after dividing the $\phi^{\mathrm{cpg}}(t)$ by the length of the CPG's (and also MC's and LMC's) period $P = 2\pi$, i.e., $\phi^{\mathrm{MC}}(t) = \phi^{\mathrm{cpg}}(t) \bmod P$.

I focus on amputation recovery without perturbations to the CPG in this work. Therefore, the perturbation input is set to zero, i.e., $p(t) = 0$. Hence, the model is simplified as follows:

$$\frac{\mathrm{d}\phi^{\mathrm{cpg}}(t)}{\mathrm{d}t} = \omega, \tag{2}$$

$$\phi^{\mathrm{MC}}(t) = \phi^{\mathrm{cpg}}(t) \bmod P, \tag{3}$$

and the individual LMC phases $\phi_i^{\mathrm{LMC}}(t)$ are defined by equation:

$$\phi_i^{\mathrm{LMC}}(t) = (\phi^{\mathrm{MC}}(t) - \phi_i(t)) \bmod P. \tag{4}$$

The CPG's state $\boldsymbol{y}(t)$ is defined by a vector $\binom{u}{v} \in \mathbb{R}^2$ as follows:

$$u(t) = \cos(\phi^{\mathrm{cpg}}(t)) = \cos\big(\phi^{\mathrm{MC}}(t)\big), \tag{5}$$

$$v(t) = \sin(\phi^{\text{cpg}}(t)) = \sin\big(\phi^{\text{MC}}(t)\big). \tag{6}$$

In the CPG-RBFN architecture, a specific movement action is triggered by the RBFN activation depending on the MC phase, i.e., RBFNs are activated by a specific CPG state defining the phase of the MC. The RBFNs are defined by their centers $\boldsymbol{w}_i^{\text{s}}(t), \boldsymbol{w}_i^{\text{m}}(t), \boldsymbol{w}_i^{\text{e}}(t) \in \mathbb{R}^2$ which if activated trigger the swing start, middle of the swing and end of the swing (start of the stance) actions for the i-th leg, respectively. The RBFNs placing within the CPGs' state space for correct activation is in this work defined as follows:

$$\boldsymbol{w}_i^{\text{s}}(t) = \begin{pmatrix} \cos(\phi_i(t)) \\ \sin(\phi_i(t)) \end{pmatrix}, \tag{7}$$

$$\boldsymbol{w}_i^{\text{m}}(t) = \begin{pmatrix} \cos\left(\phi_i(t) + \frac{\Delta\phi}{2}\right) \\ \sin\left(\phi_i(t) + \frac{\Delta\phi}{2}\right) \end{pmatrix}, \tag{8}$$

$$\boldsymbol{w}_i^{\text{e}}(t) = \begin{pmatrix} \cos(\phi_i(t) + \Delta\phi) \\ \sin(\phi_i(t) + \Delta\phi) \end{pmatrix}. \tag{9}$$

The activation signal transmitted by the RBFNs $s_i^{\text{s}}(t)$, $s_i^{\text{m}}(t)$, $s_i^{\text{e}}(t)$ depends on the centers $\boldsymbol{w}_i^{\text{s}}(t)$, $\boldsymbol{w}_i^{\text{m}}(t)$, $\boldsymbol{w}_i^{\text{e}}(t) \in \mathbb{R}^2$ and their distance from the CPG's state $\boldsymbol{y}(t)$ [55]:

$$s_i^{\text{s/m/e}}(t) = \exp\left(-\alpha \left\| \boldsymbol{y}(t) - \boldsymbol{w}_i^{\text{s/m/e}}(t) \right\|\right), \tag{10}$$

where $\|-\|$ is the euclidean norm and $\alpha = 12$ is the coeficient of the neuron's sensitivity. Therefore, the peak/activation of the RBFN means that corresponding $s_i^{\text{s/m/e}} \approx 1$.

For more complex CPG models, we proposed a solution in [18, 51], where the system estimates the CPG phase and the RBFN uses the estimation to stick to the CPG's limit cycle in place, where the CPG state's phase is estimated to correspond to the desired phase.

The information about amputation is stored in binary vector $\mathbf{s}^{\text{dmg}} \in \{0, 1\}^6$ indicating the amputated leg j by $s_j^{\text{dmg}} = 1$. This thesis assumes the vector is given.

The problem description as motion primitives timed by GP rhythm enables to achieve various GPs only by a change of seven parameters: the LMC starts $\phi_i(t)$ and swing duration $\Delta\phi(t)$. In the next Chapter, I propose the method arranging the LMC starts $\phi_i(t)$ within the MC and adjusting the swing duration $\Delta\phi(t)$ based on the CRs violation detection and subsequent CRs violation avoidance, adapting the GP rhythm after the leg amputation. The method is a modified version of the above-mentioned solution proposed in [18], here transformed for easier comparison with other approaches. An upper bound towards which the competing methods are compared is provided by the AE, introduced in Chapter 5, optimizing the reward function proposed in the same Chapter. The presented EA takes advantage of the problem specification and optimizes the reward function by assigning values to LMC starts $\phi_i(t)$, thus evolving the GP rhythm.

Chapter 4
Proposed Method

The proposed method adapts the GP rhythm after an amputation occurs to enable the robot to continue in its task. The method is based on a dynamic system detecting the violation of the CRs and their extension CCARs, introduced in Secs. 2.1.3 and 2.1.4, respectively, and actively adjusting itself to avoid the violations in the future. The rules violation causes the LMCs starts and the swing duration $\Delta\phi(t)$ to change given the proposed relations within the dynamic system. From the proposed interactions within the system emerges the amputation-adjusted GP rhythm configuration.

4.1 From Rules to Dynamics

The proposed method detects the rules violation to eliminate the violation occurrence afterward. The CRs are violated if certain pairs of legs perform swing simultaneously, which is detected using the RFBNs activation signals $s_i^{\mathrm{s}}(t)$, $s_i^{\mathrm{m}}(t)$, and $s_i^{\mathrm{e}}(t)$, introduced in Eq. 10, tracking the swing and stance timing.

The swing start (leg lifting off), middle (leg starts to descend) and end (leg steps on the ground and the stance starts) are signalized by the activations of the $s_i^{\mathrm{s}}(t)$, $s_i^{\mathrm{m}}(t)$, and $s_i^{\mathrm{e}}(t)$, respectively. The phase switches from stance to swing by activation of the $s_i^{\mathrm{s}}(t)$, i.e., $s_i^{\mathrm{s}}(t) \approx 1$, and it switches back from swing to stance if the $s_i^{\mathrm{e}}(t)$ activates, i.e., $s_i^{\mathrm{e}}(t) \approx 1$.

The switch between the swing and stance is tracked by the swing indicators $s_i^{\mathrm{sw}}(t)$ are introduced forming a vector $\mathbf{s}^{\mathrm{sw}}(t) = (s_0^{\mathrm{sw}}(t), \ldots, s_5^{\mathrm{sw}}(t))$, where $s_i^{\mathrm{sw}}(t) = 1$ and $s_i^{\mathrm{sw}}(t) = 0$ represents the i-th leg is performing swing and stance, respectively. The amputated leg j is considered to be in swing the whole MC.

The i-th leg swing indicator helps the robot to recognize two different system states balancing each other. The first is denoted as *Over-Safe State* (OSS), representing all legs (except the amputated one) being in stance, i.e., all functional legs are supporting the robot's body, and none of them is performing swing. The OSS is unwanted because its existence implies the robot does not use the time during the MC to its full potential for movement. The time spent in the OSS can be instead utilized to perform swing, thus moving the leg and the robot forward. Hence, the OSS slows the robot down.

The second state recognized using the swing indicator $s_i^{\mathrm{sw}}(t)$ is the *Rules Violation State* (RVS). The RVS is the state when the CRs extended by CCARs are violated, i.e., the leg i violates the CRs if it performs swing simultaneously with (i) its contralateral leg or (ii) leg to which the i-th leg is consecutive. The case "*its consecutive leg*" is not considered because it is already covered by the case (ii) applied on its consecutive leg. Another violation occurs if the lift off of the leg i would leave only two legs in stance in total (i.e., the robot would not have the minimal three points supporting the body).

The RVS and OSS oppose each other. The RVS signalizes not enough of the stance movement (or too much of the swing movement), while OSS indicates too much of stance (more of swing movement could be performed). The method tunes the parameters influencing the occurrence of OSS and RVS to balance them.

The following sections consist of steps leading to the dynamic detection of the RVS and OSS and their use in the online change of the GP rhythm in afford to avoid the RVS and OSS in the future.

4.1.1 Rules Violation Detection

The RVS occurs if the activations corresponding to swing of one leg occur within the interval of swing of its swing-conflicting leg (the conflicting leg pairs are given by the CRs and CCARs). Such occurrences are recognized and integrated by the system reacting to the occurrences by changing itself. The change is different based on whether the RVS was detected during or at the end of the swing, and the amount of change is proportional to "*how much*" the RVS was detected.

The swing indicators $\mathbf{s}^{\mathrm{sw}}(t)$ in combination with damage vector $\mathbf{s}^{\mathrm{dmg}} \in \{0,1\}^6$ indicating the damaged leg j by $s_j^{\mathrm{dmg}} = 1$ while undamaged leg i is indicated by $s_i^{\mathrm{dmg}} = 0$ are inputs of function $\mathbf{d}(\mathbf{s}^{\mathrm{sw}}, \mathbf{s}^{\mathrm{dmg}}) = (d_0(t), \ldots, d_5(t)) \in \{0,1\}^6$, with the following meaning: (i) $d_i(t) = 1$ if leg i would be performing swing in time t, it would violate the rules, and (ii) $d_i(t) = 0$ leg i performing swing in time t would not conflict with the rules.[2]

Having the information about swing performance $s_i^{\mathrm{sw}}(t)$ of individual legs and the logical function $d_i(t)$, the system distinguishes two possibilities of CRs violation, early lift (leg starts to lift off despite it should remain on the ground) and late support (leg is lifted while it should be already on the ground), represented by binary signals a_i^{s} and a_i^{e}, respectively. The $a_i^{\mathrm{s}}(t) = 1$ marks the violation during the first half of the swing movement (the beginning or the middle of the swing), and the $a_i^{\mathrm{e}}(t) = 1$ signalizes the violation occurring at the end of the i-th leg's swing:

$$a_i^{\mathrm{s}}(t) = d_i(t) s_i^{\mathrm{sw}}(t) [\![s_i^{\mathrm{s}}(t) > \gamma \vee s_i^{\mathrm{m}}(t) > \gamma]\!], \tag{11}$$

$$a_i^{\mathrm{e}}(t) = d_i(t) s_i^{\mathrm{sw}}(t) [\![s_i^{\mathrm{e}}(t) > \gamma]\!], \tag{12}$$

where the Iverson bracket $[\![-]\!]$ gives value one if the predicate in them is *true*; otherwise, it is equal to zero. The threshold hyperparameter $\gamma = 0.5$ marks the activation of the RBFN signal if passed. The signals are further smoothened by integration:

$$\frac{\mathrm{d}v_i^{\mathrm{s}}(t)}{\mathrm{d}t} = \left(a_i^{\mathrm{s}}(t) - v_i^{\mathrm{s}}(t)\right)\delta, \tag{13}$$

$$\frac{\mathrm{d}v_i^{\mathrm{e}}(t)}{\mathrm{d}t} = \left(a_i^{\mathrm{e}}(t) - v_i^{\mathrm{e}}(t)\right)\delta, \tag{14}$$

where hyperparameter δ is empirically set to 10.

4.1.2 Leg Motion Cycle Starts Dynamics

With the RVS detected, the system adjusts the LMCs starts, trying to avoid the RVS in the next MC. The signals $v_i^{\mathrm{s}}(t)$ and $v_i^{\mathrm{e}}(t)$ are used as the dynamic weights pushing the LMC start to avoid the rules violation, as shown in the following equations:

$$\frac{\mathrm{d}\phi_i(t)}{\mathrm{d}t} = g_{\epsilon_1}\left(v_i^{\mathrm{s}}(t) - v_i^{\mathrm{e}}(t)\right) + \mathrm{sgn}\left(\Phi\right)\left(\Phi\right)^2, \tag{15}$$

$$\Phi = g_{\epsilon_2}\left(\left(\phi_{f_i^{\mathrm{c}}(t)}(t) \oplus \frac{P}{2}\right) \ominus \phi_i(t)\right), \tag{16}$$

$$g_\epsilon(c_1) = c_1 [\![|c_1| \geq \epsilon]\!], \tag{17}$$

[2] There are more ways how to get the binary values for vector $\mathbf{d}(t)$ based on the binary vector $\mathbf{s}^{\mathrm{sw}}(t) = (s_0^{\mathrm{sw}}(t), \ldots, s_5^{\mathrm{sw}}(t))$ and binary vector $\mathbf{s}^{\mathrm{dmg}} \in \{0,1\}^6$. For instance, the methods of propositional logic or finite state automaton, etc. The technique I used is a bit tedious and not very interesting; therefore, it is presented in Appendix A.

$$\phi \oplus \phi' = (\phi + \phi') \bmod P, \tag{18}$$

$$\phi \ominus \phi' = \begin{cases} \phi - \phi' & \text{if } |\phi - \phi'| \leq \frac{P}{2}, \\ \phi - \phi' + P & \text{if } \phi - \phi' < -\frac{P}{2}, \\ \phi - \phi' - P & \text{if } \phi - \phi' > \frac{P}{2}, \end{cases} \tag{19}$$

where $\text{sgn}(-)$ is the sign function, $f_i^{\text{c}}(t)$ is a function returning a label of leg contralateral to leg i according to CCARs, and the term Φ is a regularization enforcing the antiphase motion of contralateral legs. Eq. 17 is a thresholding function providing some tolerance margin stabilizing the system (e.g., it is not necessary to enforce the contralateral legs to have a phase offset precisely equal to $\frac{P}{2}$). The constants are set as $\epsilon_1 = \epsilon_2 = 0.1$. Eqs. 18 and 19 ensure correct shifting within the interval $[0, P)$. Finally, the term inside the function $g_{\epsilon_2}(-)$ in Eq. 16 computes the shift the leg i needs to make to be in antiphase with its contralateral leg.

Important note: the LMC start $\phi_i(t)$ is also bounded by the interval $[0, P)$; hence, if the gradient moves the value outside the interval, the operation mod P is applied on the value $\phi_i(t)$. Nevertheless, the LMC start dynamics can not solve all the violation problems, as discussed in the previous section 4.1. Therefore, the following section uses the violation detection $a_i^{\text{s}}(t)$ and $a_i^{\text{e}}(t)$ for adjustment of the swing duration $\Delta\phi(t)$.

4.1.3 Swing Duration Dynamics

The swing duration $\Delta\phi(t)$ influences the violation detection significantly. To demonstrate the influence, consider the two extreme cases when the swing duration value is $\Delta\phi(t) = 0$ and $\Delta\phi(t) = P$, respectively. The first case leads to the OSS and causes the robot to stop moving. The second case leads to constant rules violation, i.e., constant RVS, because if every leg is always in swing, the conflicting pairs will perform the swing simultaneously, resulting in the rules violation. Therefore, I propose the system balancing the swing duration $\Delta\phi(t)$ to make it as long as possible to maintain the robot's movement, avoiding the OSS, while avoiding the rules violation and implicitly avoiding the RVS.

Change of the swing duration $\Delta\phi(t)$ can lead to new OSS or RVS detection. Therefore, the change can cause interactions in the system adjusting the LMC starts, compensating the detected rules violations, which takes some time to resolve. Hence, the $\Delta\phi(t)$ should not be lowered every time the CRs violation is detected, but only when "*too much*" of the CRs violation is detected. Following the same logic, prolonging the swing duration $\Delta\phi(t)$ should also occur after a certain amount of detected OSS.

Thus, I propose to use the bio-inspired mechanism of *Leaky Integrate&Fire* (LIF) neurons $c^{\text{v}}(t)$ and $c^{\text{o}}(t)$, which integrate the "*error*" increasing the neurons potential until it reaches a certain threshold, then fires causing the change in the $\Delta\phi(t)$. After firing, the neuron's potential decreases below zero and takes some time to reset before integrating again, providing the system time to adapt. If the threshold is not reached, the neuron's potential starts to reset to zero; hence "*leaky*". The LIF neurons act as short-term memory, forgetting the violation occurrence after a while. The LIF neurons are described in the following equation:

$$c^{\text{v}}(t + \Delta t) \begin{cases} -\theta^{\text{v}} & \text{if } c^{\text{v}}(t) \geq 1, \\ c^{\text{v}}(t) + \Delta t \left(\eta^{\text{v}} \sum_{i \in I} v_i^{\text{s}}(t) - c^{\text{v}}(t)\right) & \text{otherwise}, \end{cases} \tag{20}$$

$$c^{\text{o}}(t + \Delta t) \begin{cases} -\theta^{\text{o}} & \text{if } c^{\text{o}}(t) \geq 1, \\ c^{\text{v}}(t) + \Delta t \left(\eta^{\text{o}} [\![0 = \sum_{i \in I} s_i^{\text{sw}}(t)]\!] - c^{\text{o}}(t)\right) & \text{othrwise}, \end{cases} \tag{21}$$

where I is a set of labels corresponding to fully functioning legs, hyperparameters influencing the input weight $\eta^{\mathrm{v}} = 8, \eta^{\mathrm{o}} = 10$ and determining the leakage after firing $\theta^{\mathrm{v}} = \theta^{\mathrm{o}} = \frac{100\Delta t}{P}$ are set empirically with Δt being the integration step, and the expression $[\![0 = \sum_{i \in I} s_i^{\mathrm{sw}}(t)]\!]$ indicates the OSS.

The LIF neurons $c^{\mathrm{v}}(t)$ and $c^{\mathrm{o}}(t)$ fire, if they reach value of 1. The neurons' potential is way below zero after firing. The rapid decrease is detected and drives the change of the swing duration $\Delta\phi(t)$.

The RVS and OSS have opposing effects on the swing duration $\Delta\phi(t)$; hence, the OSS influences $\Delta\phi(t)$ only if the potential of the LIF neuron $c^{\mathrm{v}}(t)$ signalizing the RVS is close to zero. The swing duration $\Delta\phi(t)$ dynamics are driven by the following equation:

$$\frac{\mathrm{d}\Delta\phi(t)}{\mathrm{d}t} = \begin{cases} -\frac{\Delta\phi(t)}{\psi} & \text{if } c^{\mathrm{v}}(t) < -\theta^{\Delta\phi}, \\ \frac{\Delta\phi(t)}{\psi} & \text{if } c^{\mathrm{v}}(t) \approx 0 \wedge c^{\mathrm{o}}(t) < -\theta^{\Delta\phi}, \\ 0 & \text{else}, \end{cases} \tag{22}$$

where the hyperparameters $\psi = 8$ and $\theta^{\Delta\phi} = 0.8$ regulate the change and mark the activation threshold, respectively. The value $\Delta\phi(t)$ is bounded by interval $[\frac{P}{6}, \frac{P}{2}]$, as discussed in Chapter 3.

The introduced update rules for the LMC starts $\phi_i(t)$ and swing duration $\Delta\phi(t)$ modify the GP rhythm based on the violation of CRs giving relations among legs adjusted by proposed CCARs trying to achieve stable GP rhythm in which the altering swing and stance do not conflict with the given CRs. The LMC starts $\phi_i(t)$ dictate the positions of their respective RBFNs $\mathbf{w}_i^{\mathrm{s}}(t), \mathbf{w}_i^{\mathrm{m}}(t), \mathbf{w}_i^{\mathrm{e}}(t)$, as described in Chapter 3, whose activation triggers specific motion primitives controlling the servo motors movement. The only necessary sensory input is the binary vector $\mathbf{s}^{\mathrm{dmg}} = \{0, 1\}^6$ marking the j-th leg amputated by setting $s_i^{\mathrm{dmg}} = 1$.

In the following Chapter, the experiment setup is presented, in which the necessary modifications of the competitive method, originally proposed by Miguel et al. [14], are introduced, together with a simple evolution algorithm proposed to upper bound the possible reward for GP rhythm adjustment after leg amputation. The results are further compared using the reward function, proposed in section 5.4 and discussed in the following Chapters.

Chapter 5
Evaluation Methodology

This Chapter, consisting of five parts, provides the experiments setup and the methodology for the proposed method evaluation: (i) The first part 5.1 provides the mapping of the RBFNs activations to relative servo motor position. (ii) The second part 5.2 describes the competitive method implementation and a few changes that were made in contrast with the original authors' work [14] to make the methods comparable. (iii) Next part 5.3 introduces the evolution algorithm. (iv) The fourth part 5.4 proposes the reward function for GP rhythm evaluation, based on which the methods are compared. (v) Finally, the last part 5.5 introduces the settings of the simulator, the used robot model, and the experiment, together with mapping of the relative servo motor positions to the motor commands, common for all the compared methods.

For method comparison, a metric evaluating their performance is needed. A metric is provided in (iv) as a reward function. The reward considers the desirable locomotion features, such as speed and directional stability, and normalizes them. The normalized sensory inputs are then weighted to make the fundamental gait patterns (commonly used for fully functional hexapod walking robots, see Section 2.1.2) gain positive cumulative reward over the period of the MC.

There is possibly a limit to how much only the GP rhythm change can improve the GP after a leg amputation. To see how far from the limit the compared methods are, the EA optimizing the proposed reward function by adjusting the GP rhythm is provided in part (iii). In this part, a significant search space reduction is proposed using the GP rhythm parametrization. The parametrization enables encoding the complex locomotion after amputation into only four tunable parameters, leading to a large reduction of the search space and, thus, the parametrization decreases the needed computational resources for the evolution optimization method.

All the compared methods adjust the GP rhythm with servo motor amplitudes not being considered. Hence, the methods output is the servo motor relative position within its movement boundaries. The approach in (i) provides the proposed method with a mechanism mapping the RBFN activations to the relative servo motor position.

Converting the relative positions to the actual motor commands, common for all the compared methods, is provided in part (v), together with the introduction of the used simulator, robot model, and experiment setting.

5.1 RBFN signals to relative Amplitude

In this work, I use two steps for mapping the RBFN output signal $s_i^{\mathrm{s/m/e}}$ to motion commands. In this section, the RBFN activation switches between swing and stance, activating predefined functions determining the servo motor relative position. The second step maps the relative positions to real positions, which is introduced in Section 4.1.1.

The relative values given by the functions between which the RBFNs activations switch are in range $[-1, 1]$, where the borders represent the minimal and maximal amplitude of the regular leg movement, respectively;[3]. The RBFN signals are converted to swing indicating binary vector $\mathbf{s}^{\mathrm{sw}}$ used in the previous Chapter 4, Section 4.1.1, which are further used in the presented mapping definition in this section.

[3]The value can be a little out of the interval, e.g., if during the adaptation process, the swing is a bit longer than initially expected due to change of LMC starts between two consecutive MCs.

The *Body-Coxa* (BC) joint is used for forward/backward leg movement, and *Coxa-Femur* (CF) joint for up/down leg movement. The *Femur-Tibia* joint is fixed in a static position. Therefore, two different mappings from RBFN signals to relative amplitude $[-1, 1]$ are needed.

The BC joint has to perform movement across the whole range from the most posterior position, represented by relative value -1, to the most anterior position, represented by relative value 1, during the swing, and during the stance, it resets back from anterior 1 to posterior -1 position. I decided to parametrize the BC joint swing movement by part of the cosine curve and the backward movement by linear function influencing the relative BC joint dynamics $\hat{u}_i^{\text{bc}}(t)$ as follows:

$$\frac{\mathrm{d}\hat{u}_i^{\text{bc}}(t)}{\mathrm{d}t} = \begin{cases} \left(1 - \frac{\hat{u}^{\text{bc-last}}(t)+1}{2}\right) \frac{P}{2\Delta\phi(t)} \cos\left(\left(\phi_i^{\text{LMC}}(t)\right) \frac{P}{2\Delta\phi(t)}\right) & \text{if } s_i^{\text{sw}} = 1, \\ -\frac{2}{P-\Delta\phi(t)} & \text{if } s_i^{\text{sw}} = 0, \end{cases} \tag{23}$$

where $\phi_i^{\text{LMC}}(t)$ is the i-th leg motion phase introduced in Chapter 3, $\hat{u}^{\text{bc-last}}(t)$ is the last value of $\hat{u}_i^{\text{bc}}(t)$ before the swing began (i.e., usually ≈ -1) ensuring that $\hat{u}_i^{\text{bc}}(t) = 1$ at the end of the swing independently of the initial relative amplitude value $\hat{u}_i^{\text{bc}}(t)$ at the swing start.

The CF joint value $\hat{u}_i^{\text{cf}}(t)$ represents the up and down movement, where -1 represents the lowest point with the leg touching the ground and 1 stands for the highest elevation point, is represented by the curve of hyperbolic tangent's derivative, driving the CF joint relative amplitude by the following equation:

$$\hat{u}_i^{\text{cf}}(t) = \begin{cases} -\chi \tanh^2\left(\phi_i^{\text{LMC}}(t) - \frac{\chi'}{2}\right) + 1 & \text{if } s_i^{\text{sw}} = 1, \\ -1 & \text{if } s_i^{\text{sw}} = 0, \end{cases} \tag{24}$$

$$\chi' = \ln\left(\frac{\chi'' + \sqrt{(\chi'')^2 - 4}}{2}\right), \tag{25}$$

$$\chi'' = 2\frac{1 + \frac{2}{\chi}}{1 - \frac{2}{\chi}}, \tag{26}$$

where $\chi = 2.08$ is a hyperparameter regulating the smoothness of the swing start and end.

The mapping from interval $[-1, 1]$ to the servo motor position is common for all compared methods, and it is dependent on the defined range of the robot's servo motors. Therefore, the linear transformation is provided in the Section 5.5 with shared hyperparameters for all tested methods.

5.2 Competitive Method

This section is dedicated to a brief description of the method proposed by Miguel et al. [14], which I use for comparison with the method proposed in this work. [4] The authors use sensors for load sensing on each leg, with each leg having its own CPG, which are not coupled directly in the system but indirectly through the sensory feedback. Given the state of the CPG determining the LMC phase, the authors predict the sensory feedback and measure the error between expected and measured values. The error influences the slow and fast learners; the fast learner adapts quickly but also quickly forgets, while the slow learner remembers a longer time period but adapts slower. The learners affect the CPG phase, thus adjusting the GP rhythm.

The authors use one CPG for each leg, which is described by the following equation:

$$a_i(t+1) = \alpha S_i(t) + \sum_{j=1}^{2} w_{ij} o_j(t), \tag{27}$$

[4] For a smooth transition from the paper's notation, I decided to use the original authors' notations. Hence, consider variables in this section separate from all other variables in this work.

where $i \in \{1, 2\}$, t stands for simulation step/discrete time, w_{ij} is a fixed weight of a_j to a_i neuron connection.

The $S_i(t)$ is processed sensory feedback, evaluating the similarity of expected and sensed sensory input. The variable α is the dynamic weight of $S_i(t)$ influence on the neuron a_i. The α combines fast and slow learners; the fast learner adapts quickly but also quickly forgets, while the slow learner adapts slowly but remembers more extended time period. Those two learners run in parallel and are closely described in the original paper.

In the following paragraphs, I describe a few changes I made to the proposed system. The first is the use of variable $o_i(t) = \tanh(a_i(t))$ for $i = 1, 2$. The authors use an unspecified CPG post-processing unit for CPG signal-to-motor command mapping. I use the $o_i(t)$ feature, that the function hyperbolic tangens bounds the value $o_i(t)$ in interval $[-1, 1]$. Therefore, it works as an output signal of the CPG, providing the relative amplitude. $o_1(t)$ defines the movement of the BC joint and $o_2(t)$ prescribes the CF joint motion. However, I experienced twitching of the servo motors by sometimes step-wise progression of the values. Therefore, I decided to smoothen the output by averaging:

$$\hat{o}_i(t) = \frac{o_i(t) + o_i(t-1)}{2}. \tag{28}$$

Another issue for comparison was a different number of iterations per MC. The reward is computed as the cumulative sum over each iteration reward for five consecutive MCs; hence, a lower number of iterations with measures after different time steps could make the results incomparable. Therefore, I increased the number of iterations by linear interpolation of the motion commands between the commands computed by the method (between each two computed motion commands, there were 17 interpolated commands) to approximately match the number of iterations per MC.

The problem was also the difference in sensory feedback. The authors use load sensing feedback, while I use measured torque in CF joint. Nevertheless, the time progression of those two is similar, and a scaling factor of 3 approximately leveled up the values. The method uses the feedback to verify if progression is equal to expectation. Therefore, the difference should be negligible.

One more inconvenience to solve was the sensory feedback was sometimes similarly step-wise, as the mentioned CPG output $o_i(t)$. Hence, I also averaged the sensory feedback, where the sensory input for the method was averaged over the last three measured values:

$$F(t) = \frac{F_m(t) + F(t-1) + F(t-2)}{3}, \tag{29}$$

where $F_m(t)$ is the current sensory measurement. For computation of $F(0)$ the values for averaging were set to zeros: $F(-1) = F(-2) = 0$. The rest of the method implementation details are described in the original paper.

In comparison with the results of the original work, my implementation has a longer swing duration, which is probably caused by different CPG to motor commands mapping. Both the robot model and simulator I used are different from the original authors' experiments. The authors measure the performance by average robot speed and report its decrease after amputation. In my experiment, the adjusted implementation achieved similar results in the performance decline after the amputation.

5.3 Evolution Algorithm - Upper Bound

The competing methods adjust the GP rhythm without any knowledge of the reward function evaluating them. To see how much reward the methods can possibly achieve by adapting only the GP rhythm, I implement the EA optimizing the reward function by tuning the GP rhythm. The EA provides an upper bound revealing how well the competing methods optimize the GP rhythm.

While evolution algorithms can optimize anything given enough time/computational power, the need for high time and computational resources is their limitation. Therefore, I aimed to make the

search space as simple as possible with only a few tunable parameters, lowering the computational time and the needed number of generations to a minimum. The search space definition uses the GP rhythm description used throughout this thesis, making the search space relatively small.

Algorithm 1: evolution($g_{\text{count}}, n, n_{\text{min}}, b, j, P$)

Data: $0 < g_{\text{count}} \in \mathbb{N}; 0 < n \in \mathbb{N}; 0 < n_{\text{min}} \leq n, n_{\text{min}} \in \mathbb{N}$;
$b \in [0, 1]; j =$ label of amputated leg; $0 < P \in \mathbb{R}$;
Result: $\mathbf{g} = \left(\boldsymbol{\phi}_i^{\text{congif}}; i = 1, \ldots, \text{floor}\left(\max\left(\frac{g_{\text{count}}}{1.3}, n_{\text{min}}\right)\right)\right)$;
$\mathbf{r}^{\text{GP}} = \left(r_i^{\text{GP}}; i = 1, \ldots, \text{floor}\left(\max\left(\frac{g_{\text{count}}}{1.3}, n_{\text{min}}\right)\right)\right)$;

for $i = 1$ **to** m **do**
  $\boldsymbol{\phi}_i^{\text{congif}} \leftarrow \left(\phi_{i,k}^{\text{config}} = \text{random\_uniform}(0, P); k \in \{0, \ldots, 5\} \backslash \{3, j\}\right) \in [0, P)^4$
end
/* Initialize 1-st generation: */
$\mathbf{g} \leftarrow (\boldsymbol{\phi}_i^{\text{congif}}; i = 1, \ldots, n)$
$\mathbf{r}^{\text{GP}} \leftarrow \text{compute\_rewards}(\mathbf{g})$
$n_1 \leftarrow n$
/* Evolve generations 2 to g_{count}: */
for $i = 2$ **to** g_{count} **do**
  $\sigma \leftarrow \frac{8.8}{i}$
  $n_i \leftarrow \text{floor}\left(\max\left(\frac{n_{i-1}}{1.3}, n_{\text{min}}\right)\right)$
  $\mathbf{g} \leftarrow \text{breed}(\mathbf{g}, \mathbf{r}^{\text{GP}}, b, \sigma, n_i, j, P)$
  $\mathbf{r}^{\text{GP}} \leftarrow \text{compute\_rewards}(\mathbf{g})$
end

Figure 4: The function random_uniform(a, b) is a function randomly sampling from a uniform distribution. Function floor(a) rounds number $a \in \mathbb{R}$ to its closest integer $l \in \mathbb{N}$ lower than a: $l < a$. Function compute_rewards$(\mathbf{g})$ simulates five MCs of each GP configuration $\boldsymbol{\phi}_i^{\text{congif}}$ in in $\mathbf{g}$ and computes the configuration's reward r_i^{GP} according to Section 5.4. Function breed is described in Alg. 2.

5.3.1 Search Space Reduction

The EA needs to have a defined search space within which the mutations and crossovers can be performed exploring possible solutions. The straightforward approach for building the search space would be to find a motor command for each servo motor in each iteration of the MC, i.e., having a vector of $18 \cdot 222 = 3996$ elements, i.e., $(a, b)^{3996}$, where $a \in \mathbb{R}$ and $b \in \mathbb{R}$ are the limits of the servo motors movement.[5] Such massive search space takes a very long time to explore. Because of that, the straightforward approach is hardly ever used. I propose to use the definition of the GP rhythm from Chapter 3 to reduce the search space.

The aim is to find an optimal GP rhythm for the given amputated leg j, i.e., find the swing duration $\Delta\phi$ and the LMC starts $\phi_i(t)$ for $i = \{0, \ldots, 5\} \backslash \{j\}$, where $\phi_i(t) \in [0, P)$ and $P = 2\pi$. However, there is theoretically an infinite amount of symmetric $\phi_i(t)$ settings resulting in the same GP, which is only shifted in phase.

To avoid the symmetries, I fix the $\phi_3 = 0$, where label $i = 3$ stands for the hind right leg. With the hind right leg's LMC start fixed on zero, each gait will start by swing movement of the hind right leg, and all other equal gaits differing only by phase shift are eliminated from the optimization choice.

[5] In this work I use 222 iterations per MC.

Algorithm 2: breed($\mathbf{g}, \mathbf{r}^{\text{GP}}, b, \sigma, m, j, P$)

Data: $\mathbf{g} = (\boldsymbol{\phi}_1^{\text{congif}}, \ldots, \boldsymbol{\phi}_n^{\text{congif}})$; $\mathbf{r}^{\text{GP}} = (r_1^{\text{GP}}, \ldots, r_n^{\text{GP}})$;
$b \in [0, 1]$; $0 < \sigma \in \mathbb{R}$; $0 < m \in \mathbb{N}$; j = label of amputated leg; $0 < P \in \mathbb{R}$;
Result: $\hat{\mathbf{g}} = (\hat{\boldsymbol{\phi}}_1^{\text{congif}}, \ldots, \hat{\boldsymbol{\phi}}_m^{\text{congif}})$

$m_{\text{best}} \leftarrow \text{ceil}(bn)$
$\hat{m} \leftarrow m - m_{\text{best}}$
$\mathbf{p}^{\mathbf{g}} \leftarrow (r_i^{\text{GP}} - \min(\mathbf{r}^{\text{GP}}) + \varepsilon; i = 1, \ldots, n)$
$\mathbf{p}^{\mathbf{g}} \leftarrow (p_i^{\mathbf{g}} / \sum_{k=1}^{n} p_k^{\mathbf{g}}; i = 1, \ldots, n)$
/* Create $\hat{m}$ new individuals: */
for $i = 1$ **to** $\hat{m}$ **do**
  $\phi^{\text{par1}} \leftarrow \text{random\_choice}(\mathbf{g}, \mathbf{p}^{\mathbf{g}})$;
  $\phi^{\text{par2}} \leftarrow \text{random\_choice}(\mathbf{g}, \mathbf{p}^{\mathbf{g}})$;
  for $k \in \{0, \ldots, 5\} \backslash \{3, j\}$ **do**
    $d \leftarrow \max(|\phi_k^{\text{par1}} - \phi_k^{\text{par2}}|, \varepsilon)$;
    $\hat{\phi}_{i,k}^{\text{config}} \leftarrow \text{random\_uniform}\left(\min(\phi_k^{\text{par1}}, \phi_k^{\text{par2}}) - \alpha d, \max(\phi_k^{\text{par1}}, \phi_k^{\text{par2}}) + \alpha d\right)$;
    $\hat{\phi}_{i,k}^{\text{config}} \leftarrow \left(\hat{\phi}_{i,k}^{\text{config}} + \sigma \text{ random\_normal}(0, 1)\right) \bmod P$;
  end
  $\hat{\boldsymbol{\phi}}_i^{\text{congif}} \leftarrow (\hat{\phi}_{i,k}^{\text{config}}; k \in \{0, \ldots, 5\} \backslash \{3, j\})$;
end
/* Let m_{best} individuals survive to next generation: */
for $i = 1$ **to** m_{best} **do**
  $\hat{\boldsymbol{\phi}}_{\hat{m}+i}^{\text{congif}} \leftarrow \text{get\_ith\_best}(\mathbf{g}, i, \mathbf{r}^{\text{GP}})$;
end

Figure 5: The functions random_uniform(a, b) and random_normal(μ,std) are functions randomly sampling from uniform and normal distributions, respectively. The function random_choice$(\mathbf{a}, \mathbf{p})$ makes a random choice of elements a_i based on their respective probabilities p_i. Function ceil(a) rounds number $a \in \mathbb{R}$ to its closest integer $l \in \mathbb{N}$ greater than a: $l > a$. Function get_ith_best$(\mathbf{a}, i, \mathbf{b})$ returns i-th best solution from all a_k given their respective b_i, where highest b_i is the best. The hyperparameter $\varepsilon = \frac{1}{16}$ keeps the values further from zero. Hyperparameter $\alpha = \frac{1}{16}$ enlarges the possible children value range from convex space between parents to convex space with vicinity.

I also propose a parameter reduction for the tested scenarios with amputation of one leg by fixing the $\Delta\phi = \frac{P}{3}$. This value is the swing duration value for ripple gait when two legs are always lifted off while four legs support the body. After tripod gait, which has the swing duration $\Delta\phi = \frac{P}{2}$ not usable in the case of an amputated leg if three legs should support the body at any moment, the ripple gait is the second fastest out of the observed gaits in living six-legged organisms. Moreover, the swing duration $\Delta\phi = \frac{P}{3}$ is applicable (does not by definition conflict with the CRs) even for the scenario with an amputated leg. Therefore, I conclude that setting the swing duration to value $\frac{P}{3}$ optimizes this parameter automatically, and its optimization by EA is no longer needed.

Using the abstraction applied throughout this thesis (i.e., represent the gait by LMC starts and swing duration) combined with the above-explained parameter fixing, the evolution algorithm has to optimize the GP's reward r^{GP} by optimizing only a vector of four parameters:

$$\phi^{\text{config}} = (\phi_i \in [0, P); i \in \{0, \ldots, 5\} \backslash \{3, j\}), \tag{30}$$

where j is the label of the amputated leg, and the ϕ_i in the vector are ordered by their index from

lowest to largest. The proposed search space parametrization results in bounded search space $[0, P)^4$.

5.3.2 Evolution Algorithm Parametrization

The used EA, together with the algorithm taking care of generating a new generation from the old one, are both presented in Algs. 1 and 2, respectively, where the input parameters for Alg. 1 are following: $g_{\text{count}} = 50$, $n = 200$, $n_{\text{min}} = 20$, $b = 0.15$, $P = 2\pi$, and $j \in \{0, 2, 4\}$ differs for different amputation scenarios.

There are three mechanisms deployed in the EA. The σ and number of individuals n_i in generation i are dependent on the generation, where the first influences the mutation extensiveness, and the second reduces the number of newly formed individuals in each generation. The third mechanism is the mechanism of *elites* ensuring the new generation's optimal value is greater or equal to the previous generation.

The aim of the mutation extensiveness is to first search through different parts of the search space and then optimize the most promising areas of it. Hence, I propose to change the possible mutation range, denoted by σ, in each generation. In the first few epochs, the large σ enables extensive random mutations exploring possibly unknown areas of the search space. With ongoing evolution, the σ lowers, making the mutation more subtle, exploring close vicinity of the best-so-far-found individuals.

To decrease the needed computational time, the number of individuals in each progressing generation lowers. The algorithm starts with maximal generation size to sample the search space in the most uniform way possible. Then, each new generation is smaller than the previous one until the minimal number n_{min} of individuals per generation is reached, stopping the population size from lowering.

The elites mechanism is enforced by always adding $\approx bn_i = 0.15n_i$ best members of the generation i to the next generation $i + 1$, where n_i stand for i-th generation members number, see Alg. 2 lines 1 and 15-17.

The remainder of the new generation individuals is created by combining the individuals of the old generation; see Alg. 2 lines 5-14. The resulting new individual is then influenced by a mutation, as discussed before.

There are $n_{\text{min}} = 20$ individuals in the last generation, among which the best-found solution is present.[6] The fitness of the individuals is determined by the reward value r^{GP}, proposed in the following Section.

5.4 Reward Function

This section introduces a reward function $r(t)$ and the resulting reward value r^{GP} designed to compare the proposed method with the method presented by Miguel et al. The aim is to reward velocity and stability while suppressing changes in velocity (acceleration), orientation, and position in unwanted directions. The measurements used for the reward function are also selected partially based on the plausibility of living organisms being able to perceive such measurements and include them in their reward system while learning to walk. I use three measurements: (i) position $\mathbf{p}(t)$, (ii) orientation $\mathbf{o}(t)$ and (iii) torques of coxa-femur joint $\mathbf{b}(t) = (b_0(t), \ldots, b_5(t))$.

The simulation consists of discrete steps, with step Δt. Therefore, the velocity $\mathbf{v}(t)$ and acceleration $a(t)$ is computed from position differences in time:

$$\mathbf{v}(t) = \mathbf{p}(t + \Delta t) - \mathbf{p}(t), \tag{31}$$

$$a(t) = \|\mathbf{v}(t + \Delta t) - \mathbf{v}(t)\|. \tag{32}$$

[6] Thanks to the elites mechanism, the best individual is guaranteed to survive all the way to the last generation, even if formed in the first generation. Otherwise, the record of the best solutions would have to be kept.

The orientation is a unit vector oriented in the direction of the robot's forward direction. The reward function $r(t)$ is defined by following equation:

$$r(t) = \langle \mathbf{r}(t) | \alpha \rangle, \tag{33}$$

$$\mathbf{r}(t) = \begin{pmatrix} r_1(t) \\ r_2(t) \\ r_3(t) \\ r_4(t) \\ r_5(t) \end{pmatrix}; \quad \alpha = \begin{pmatrix} \alpha_1\beta_1 \\ \alpha_2\beta_2 \\ \alpha_3\beta_3 \\ \alpha_4\beta_4 \\ \alpha_5\beta_5 \end{pmatrix}, \tag{34}$$

where $\langle -|- \rangle$ stands for dot product, and the alphas and betas are constant weights of different elements of $\mathbf{r}(t)$. The first component $r_1(t)$ given as follows:

$$r_1(t) = \left\langle \mathbf{o}(t) \middle| \frac{\mathbf{v}(t)}{\|\mathbf{v}(t)\|} \right\rangle \|\mathbf{v}(t)\|, \tag{35}$$

rewards the velocity in the forward direction. The first term is largest if the direction of the velocity $\frac{\mathbf{v}(t)}{\|\mathbf{v}(t)\|}$ (position change direction) and the robot's orientation $\mathbf{o}(t)$ are equal. The reward can also be negative if the orientation and velocity direction differ. Hence, the first term weights the second term, the velocity, in a way that rewards large velocity in the correct direction and penalizes velocity in unwanted directions.

The $r_1(t)$ is the only positively weighted component of the $\mathbf{r}(t)$ vector. Therefore, its weight $\alpha = 5$ is relatively large in comparison with the following coefficients to outweigh the other components $(r_2(t), \ldots, r_5(t))$.

The second component, $r_2(t)$, penalizes acceleration because we want our robot to move at a consistent speed without slowing down or speeding up within the MC. Assuming the robot carries sensors like cameras, it is reasonable to maintain stable speed suppressing motion blur causing twitches. $r_2(t)$ is given by the following equation:

$$r_2(t) = \|a(t)\|. \tag{36}$$

The third component is meant to minimize the torques to minimize the servo motors' effort. Therefore, it is simply defined as the sum of absolute values of coxa-femur joints' torque:

$$r_3(t) = \sum_{i \in I} |b_i(t)|, \tag{37}$$

where I is a set of labels corresponding to functioning legs. The fourth and fifth terms penalize the orientation change and position change in unwanted directions (the forward), respectively:

$$r_4(t) = \|\mathbf{o}(t + \Delta t) - \mathbf{o}(t)\|, \tag{38}$$

$$r_5(t) = |p^{\mathrm{h}}(t)| + |p^{\mathrm{v}}(t)|, \tag{39}$$

where $p^{\mathrm{h}}(t)$ and $p^{\mathrm{v}}(t)$ are shifts of the robot's body in the horizontal and vertical robot's axis (i.e., sideways and up and down movement).

The two types of hyperparameters serve different purposes. The beta-coefficients are set to normalize the inputs into a similar scale, while the alpha-coefficients play the role of weights of the input contributions into the reward. The beta-coefficients $\beta_1, \ldots, \beta_5$ are set based on the sensory values to make them all of a similar scale, e.g., if $r_1(t) \approx 10^2$ and $r_2(t) \approx 10^{-4}$, then $\beta_1 = 10^{-2}$ and $\beta_2 = 10^4$. In this work for our model, the β-coeficients are set as $\beta_1 = 10^2, \beta_2 = 10, \beta_3 = 1, \beta_4 = 10, \beta_5 = 10^3$.

The remaining alpha-coefficients are empirically set to make the $r(t)$ values of the known tripod, ripple, and wave gaits positive: $\alpha_2 = -0.2, \alpha_3 = -0.5, \alpha_4 = -1, \alpha_5 = -0.5$, see upper plot in Figure 6.

The GP rhythm performance varies within the MC (see Figure 6). Therefore, to evaluate the GP rhythm, it is necessary to consider all the rewards it gained. Hence, in this work, I use the cumulative sum of the reward function results (see the lower plot in Fig 6):

$$r^{\mathrm{c}}(t) = \sum_{i=0}^{\lfloor t/\Delta t \rfloor} \mathbf{r}(i\Delta t). \tag{40}$$

I let the experiment run for five MCs and as the final GP score I take the mean of the last MC's cumulative sums:

$$r^{\mathrm{GP}} = \frac{1}{n_{\mathrm{P}}} \sum_{i=4n_{\mathrm{P}}}^{5n_{\mathrm{P}}-1} r^{\mathrm{c}}(i\Delta t), \tag{41}$$

$$n_{\mathrm{P}} = (2\pi)/\Delta t, \tag{42}$$

where the n_{P} is the number of discrete steps of length Δt in one period $P = 2\pi$. In the remainder of this thesis, the GPs are evaluated using the r^{GP} as a score of the GP.

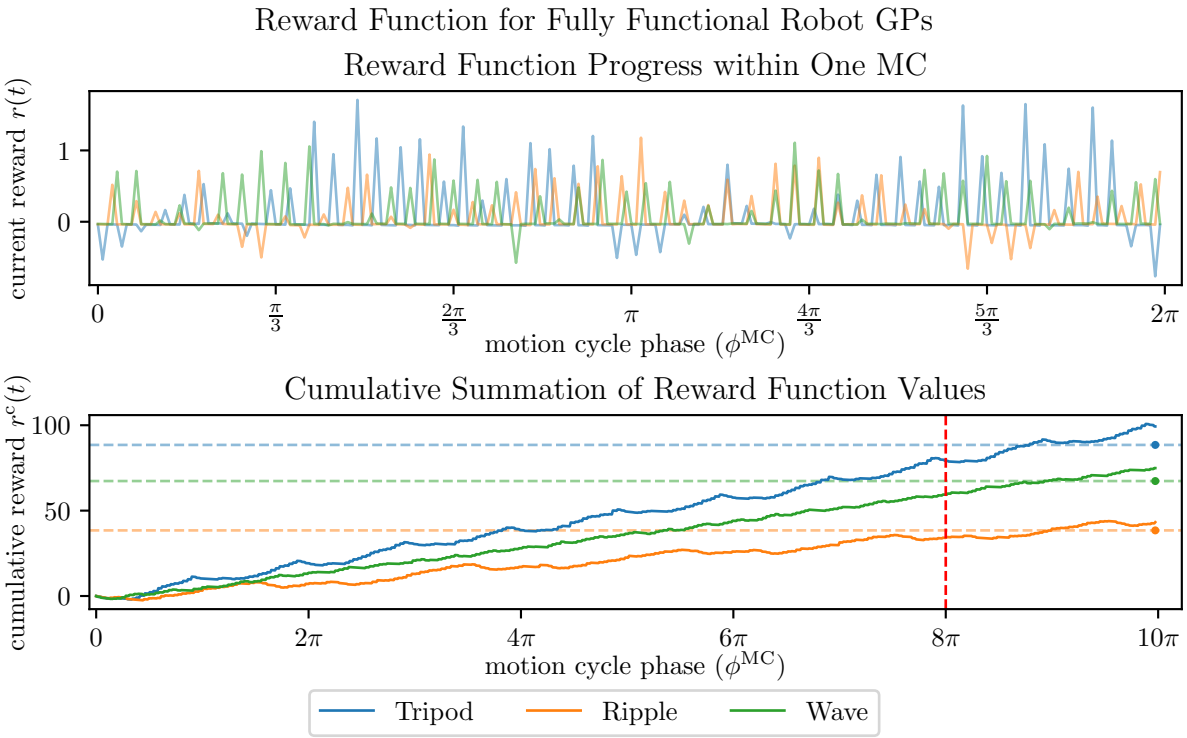

Figure 6: The Figure shows the proposed reward value behavior on traditional GPs. The reward r (on y axis) within one MC (MC phase ϕ^{MC} is on x axis) and its cumulative sum r^{c} (presented on the y axis) over five MCs (MC phase ϕ^{MC} depicted on x axis) are depicted in the top and bottom plots, respectively. The reward for the tripod, ripple, and wave gaits are presented by blue, orange, and green colored lines, respectively. In the bottom picture, the red dashed line shows the start of the fifth MC. The gaits are evaluated based on the cumulative sum values mean r^{GP} over the last (i.e., fifth) MC. The r^{GP} is shown by the dot at the end of the fifth MC or by the dashed line with color corresponding to its respective GP.

5.5 Experiment Setting

In this section, the experimental setup of the compared methods is presented. The used simulator and simulation model are described together with the parameter setting of the robot's default servo motor positions and movement range for individual joints. All the compared methods output relative motor values in range $[-1, 1]$, which are mapped to particular motor commands in this section.

The simulation is run in simulator *CoppeliaSim* [56] using a hexapod walking robot model *PhantomX-v3.1* [57] controlled by `python3` API walking on a straight surface.

The robot has eighteen position-controlled servo motors, three servo motors for each leg. The leg forward and backward movement is provided by rotation in *Body-Coxa* (BC) joint (see Figure 1), and up and down movement is ensured by *Coxa-Femur* (CF) joint. The *Femur-Tibia* (FT) joint was not used in this work and remained in a fixed position during the whole simulation. The leg damage/amputation was simulated by raising the amputated leg above the robot's body to a non-conflicting position with the other legs and with the ground.

The methods generate values from interval $[-1, 1]$ representing the relative servo motors positions. The robot's resting servo motors positions $u_i^{\text{bc-rest}}, u_i^{\text{cf-rest}}, u_i^{\text{ft-rest}}$ are set to values: $u_i^{\text{bc-rest}} = -0.25$ and $u_i^{\text{bc-rest}} = 0.25$ for front ($i \in \{0, 1\}$) and hind ($i \in \{2, 3\}$) legs' BC joins, respectively; $u_i^{\text{bc-rest}} = 0$ for middle ($i \in \{4, 5\}$) legs' BC joins; $u_i^{\text{cf-rest}} = 0.5$ for all legs' CF joins and $u_i^{\text{ft-rest}} \approx -1.37$ for all FT joins. The resulting servo motor positions $u_i^{\text{bc}}(t), u_i^{\text{cf}}(t), u_i^{\text{ft}}(t)$ are given by the following transformation:

$$u_i^{\text{bc}}(t) = u_i^{\text{bc-rest}} + (\xi_2^{\text{bc}} \hat{u}_i^{\text{bc}}(t) + \xi_1^{\text{bc}}), \tag{43}$$

$$\xi_1^{\text{bc}} = \frac{u_{\text{max}}^{\text{bc}} + u_{\text{min}}^{\text{bc}}}{2}, \tag{44}$$

$$\xi_2^{\text{bc}} = u_{\text{max}}^{\text{bc}} - \xi_1^{\text{bc}}, \tag{45}$$

where the $u_i^{\text{cf}}(t)$ and $u_i^{\text{ft}}(t)$ are computed analogously (with fixed FT joint implying $\hat{u}_i^{\text{ft}}(t) = 0$ for all $t > t_{\text{ampt}}$, where t_{ampt} represent the time of amputation) and movement range amplitudes $u_{\text{max}}^{\text{bc}} = 0.32, u_{\text{min}}^{\text{bc}} = -0.32, u_{\text{max}}^{\text{cf}} = 0.7, u_{\text{min}}^{\text{cf}} = 0.2, u_{\text{max}}^{\text{ft}} = 0.3, u_{\text{min}}^{\text{ft}} = -0.2$ are empirically set.

The three tested methods were run differently according to their specifics. I already discussed in Section 5.4 that the evaluation of the GP is computed based on five MCs. Given the GP rhythm, the five MCs can be run; hence, the evaluation of individuals in the evolution process runs for five MCs, i.e., $222 \cdot 5$ iterations, because the GP does not change during the run. On the other hand, Miguel's and my methods adapt online; thus, a few MCs are needed for adaptation before the GP stabilizes.

The Miguel method starts with the leg already amputated and with randomly initialized CPG states $a_i(0) = \text{random\_uniform}(-1, 1)$ for $i \in \{1, 2\}$ for each leg's CPG. The simulation runs for 25 MCs, during which the adapted gait changes only slightly. Based on observations, Miguel's method does not fully stabilize; hence, there is no point in waiting for the system to fully converge to unchanging GP. Nevertheless, I empirically confirmed the 25 MCs are enough for the system to adapt to the amputation.

In contrast, in most cases, my proposed method stabilizes after a while. At the beginning of the experiment, the robot locomotes using the tripod gait for 2 MCs. The amputation occurs during the third MC, with the amputation time selected based on random uniform distribution within the third MC, introducing randomization into the state in which the system starts to adapt. The adaptation process starts after the amputation.

There are two possible stopping criteria for the experiment. The simulation stops when: (i) the LMC starts $\phi_i(t)$ do not change for five consecutive MCs, or (ii) the 50-th MC ended. In most cases of stopping the simulation after 50 MCs, the GP rhythm is usually "*almost stabilzed*", i.e., the swing/stance timings do not change anymore, but their phase offset remains approximately the

same. In other words, slight numerical instability occurs, forcing the system to never fully stabilize, but for the naked eye, the GP seems stable.

Each method was tested for amputation of the front left ($i = 0$), middle left ($i = 4$), and hind left ($i = 2$) legs. Those three cases cover all possible amputations because amputating legs from the robot's right body side would cause the same but mirrored situation. The evolution generated 50 generations of individuals, from which the top 30 best-performing individuals for each leg amputation scenario are selected for comparison with the two other methods. The proposed and the Miguel's methods ran 30 times for each amputation scenario.

After running the experiments for each method, the reward is computed based on the performance in the last five MCs. The following Chapter introduces the achieved results.

Chapter 6
Results

The results presented in this section compare the reward values (computed with proposed reward function, see Section 5.4) of three different approaches: (i) Miguel's reactive method with no information about particular damage with (ii) proposed method relying strictly on the damage information and rhythm change with no sensory feedback, and with (iii) evolution algorithm approach with its thirty best-performing individuals in each scenario representing the upper bound achievable with the GP rhythm change[7].

The methods modify the GP rhythm to adapt to the leg amputation. Three scenarios were tested in the experiments: amputation of the front left, the middle left, and the hind left legs, with results depicted in Figure 7.

At the end of this Chapter, the overall results, giving an insight into the reward value dependency on the GP rhythm, are presented together with GPs resulting from the individual methods showing similarities despite different optimization criteria of respective approaches.

6.1 Simmilar Best Results and More Consistent Performance

A reasonable demand for the use of any solution to a problem is its consistency. If the aim is for the robot to get to the closest repair point after the damage occurs, one could prefer the robot gets there every time but possibly slower, over sometimes getting there fast and sometimes not getting there at all in the extreme case.

The results suggest the proposed sensory-free method is not only comparable with the reactive-based approach of Miguel et al. but also achieves better results in some scenarios given the introduced reward function. In scenarios with amputated front and hind legs, the introduced method performs better than Miguel's method. In the case of middle leg amputation, Miguel's method seems to offer better results than the proposed method.

However, in all scenarios, the standard deviation of the achieved reward of the introduced method is at least two times lower than the standard deviation of the achieved rewards by Miguel's method. The proposed method's worst result is better than Miguel's worst result in each scenario, with the best results achieved by both approaches scoring very similarly in all the amputation cases.

The statistics comparison of individual scenarios is presented in Table 1, with results distribution demonstrated by histograms in Figure 8. Therefore, the proposed method is more consistent in the expected outcome in all tested cases due to lower standard deviation and better worst-case scenarios.

6.2 More Consistent Traversed Trajectory

The robot navigation towards a goal relies on the robot's capability to maintain its movement direction. The goal in the trajectory is to keep the movement straight, parallel with the y axis, and, of course, with the maximum possible velocity. The leg amputation disbalances the power, pushing the robot on the opposing sides of its body, resulting in turning towards the weakened side. A less significant imbalance in the locomotion movement direction improves the quality of the adjusted locomotion.

[7] Recall the definition the GP rhythm form Section 2.1.1, where GP rhythm is given by the LMC starts $\phi_i(t)$ for each leg together with the swing duration $\Delta\phi$.

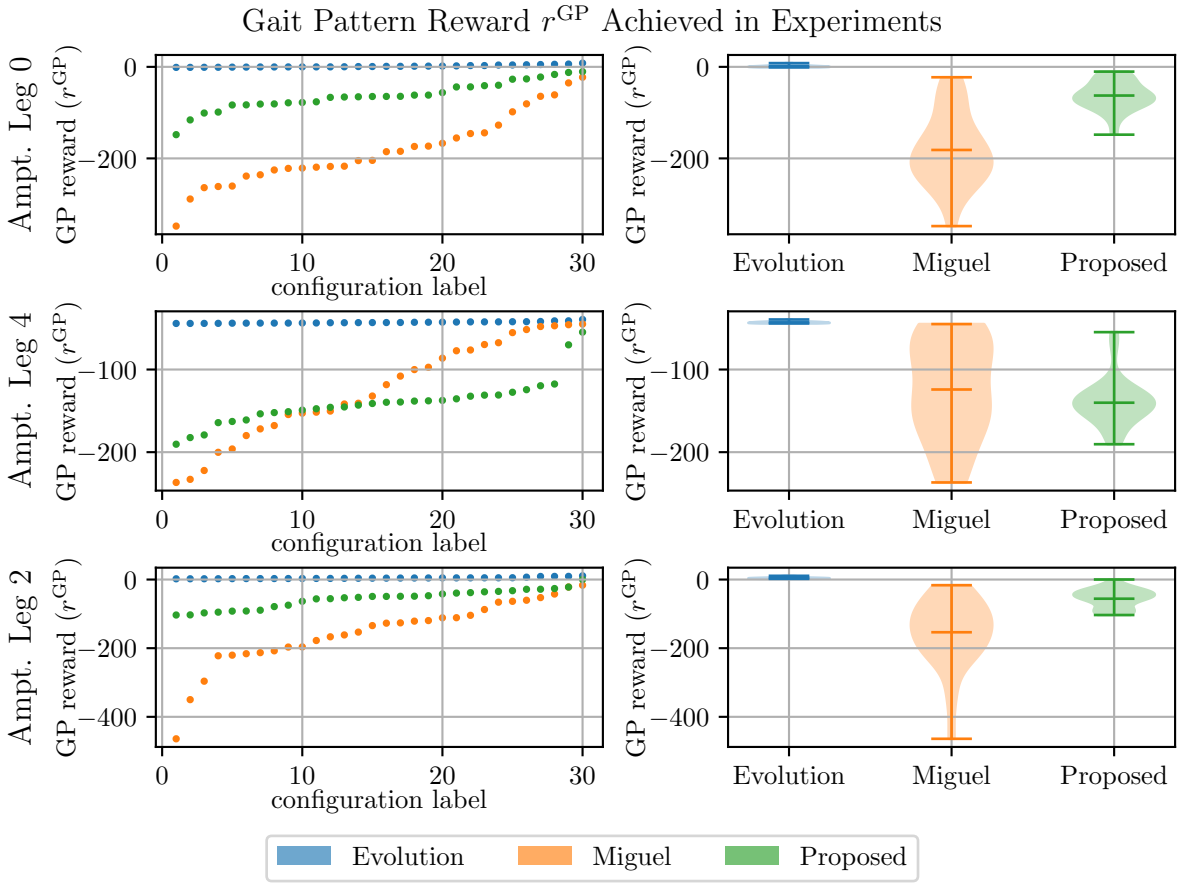

Figure 7: Achieved reward value r_{GP} by individual methods. On the left, the individual experiment runs, ordered from the worst to the best, color-coded for particular methods: Miguel's (orange), my (green), and evolution optimization (blue). On the right, the box plot of the rewards with the same color-code.

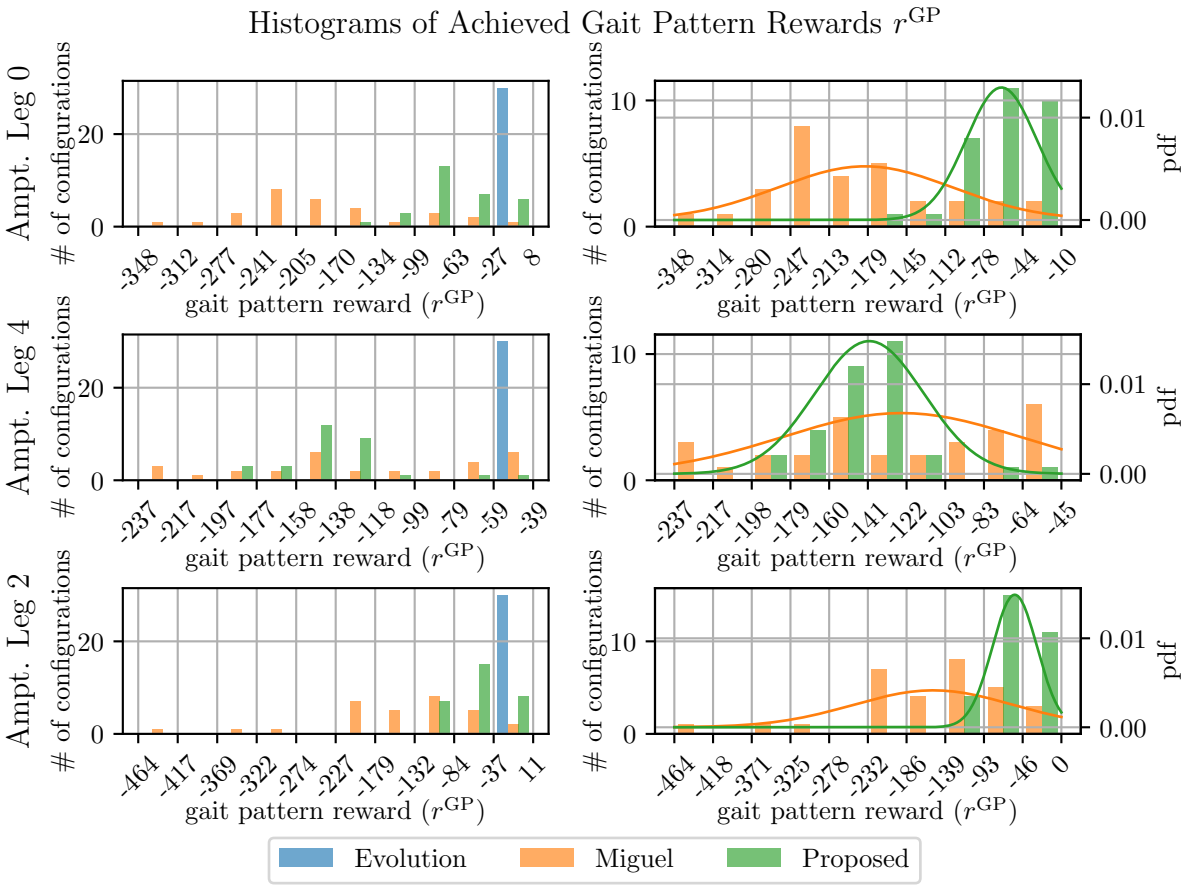

Figure 8: Histograms of achieved reward value r_{GP} by individual methods. On the left, a histogram for all methods. On the right, the detail of only the Miguel's and my method are shown with fitted Gaussian distributions based on the achieved reward value r_{GP} mean and standard deviation.

Table 1: Statistics of methods' reward value in experiments

Ampt. leg	method	min	max	median	mean	std
front left	Miguel	−347.9	−22.7	−194.8	−181.6	76.1
	Proposed	−148.1	−10.4	−64.7	−62.7	30.8
	Evolution	−1.2	8.2	0.9	1.6	2.4
middle left	Miguel	−236.6	−45.0	−125.2	−124.2	58.9
	Proposed	−190.4	−54.8	−140.4	−140.2	27.0
	Evolution	−44.3	−39.4	−43.2	−43.0	1.1
hind left	Miguel	−464.0	−16.5	−130.6	−153.6	96.1
	Proposed	−103.4	0.1	−49.2	−55.9	26.8
	Evolution	2.10	10.8	4.0	4.5	2.3

Miguel's method's greater deviation is noticeable even in the robot's movement trajectory; see Figure 9. All the upper bound solutions, given by the evolution algorithm, do not deviate much. The introduced method deviates more, but in comparison with Miguel's approach, the trajectory has fewer sideways deviations from the forward direction.

Another difference is the variation in the trajectories' direction. Especially in the scenario with the amputated front left leg (leg 0, see Figure 9), the resulting gaits generated by Miguel's method walk in very different directions. Some are deviated about 40 degrees to the left, while others are deviated about 45 degrees to the right from the desired initial direction. In comparison, my method is more consistent in the resulting adapted GP movement direction.

Noticeable is also the consistency difference in the walked distance between the proposed method and Miguel's method. Miguel's approach achieves considerably longer traveled distance in some cases, but simultaneously, the worst attempts manage to travel about half the distance.

The proposed method traversed path is more predictable in both criteria, its length and its direction, making the navigation process easier than using the Miguel's approach, decreasing the demands in navigation adjustments in case of leg damage.

6.3 Consistency is Independent on the MC State in Time of Amputation

The relations within the dynamic system can favor some states in which the adaptation gives better results than in other states. The results presented in this section show that this is not the case for the proposed method.

The amputation time within the MC was randomized in the proposed method (described in previous Chapter 5, Section 5.5). In the experiment, no correlation between the time of amputation within the phase and the resulting reward r^{GP} was observed, confirmed by the dependency visualization in Figure 10. There is no visible dependency between the adaptation time and the achieved score. There are both cases, with the longest adaptation achieving both the best and the worst score in the amputation of the hind left (leg 2) and the front or the middle left (0 or 4) legs, respectively. The method also shows no clear sign of dependency between the phase of amputation (i.e., the start of the adaptation process) and the length of adaptation. Hence, I conclude the method behaves consistently in terms of both adaptation time and resulting score independently on the phase in which the damage occurs.

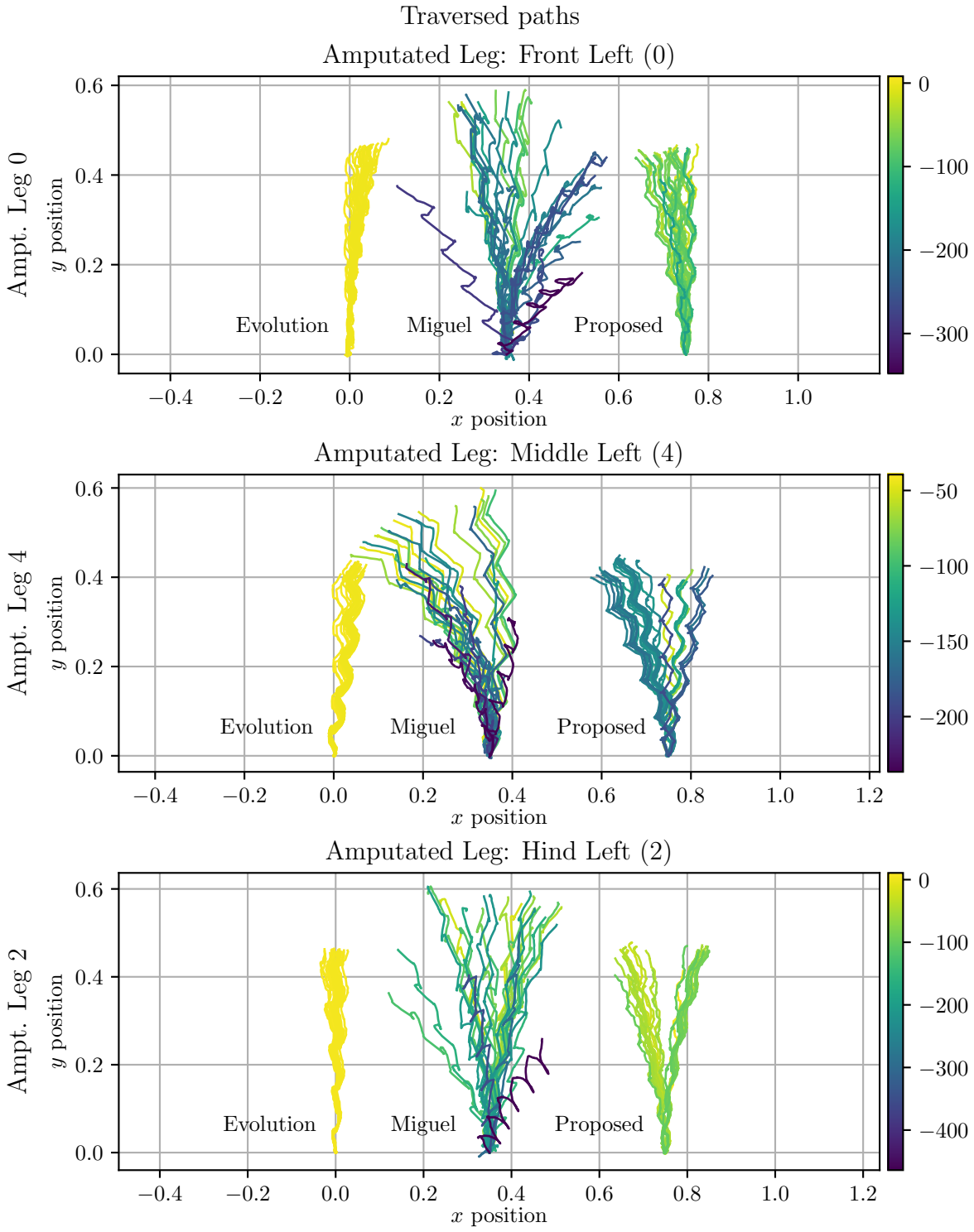

Figure 9: Top view, where each line represents the path the robot walked in 5 MCs during the experiment. The color represents the reward value r^{GP} achieved by the GP, which produced the path performing the locomotion. The colors are normalized for each plot individually, i.e., The brightest yellow (darkest blue] stands for the best (the worst) solution in each chart.

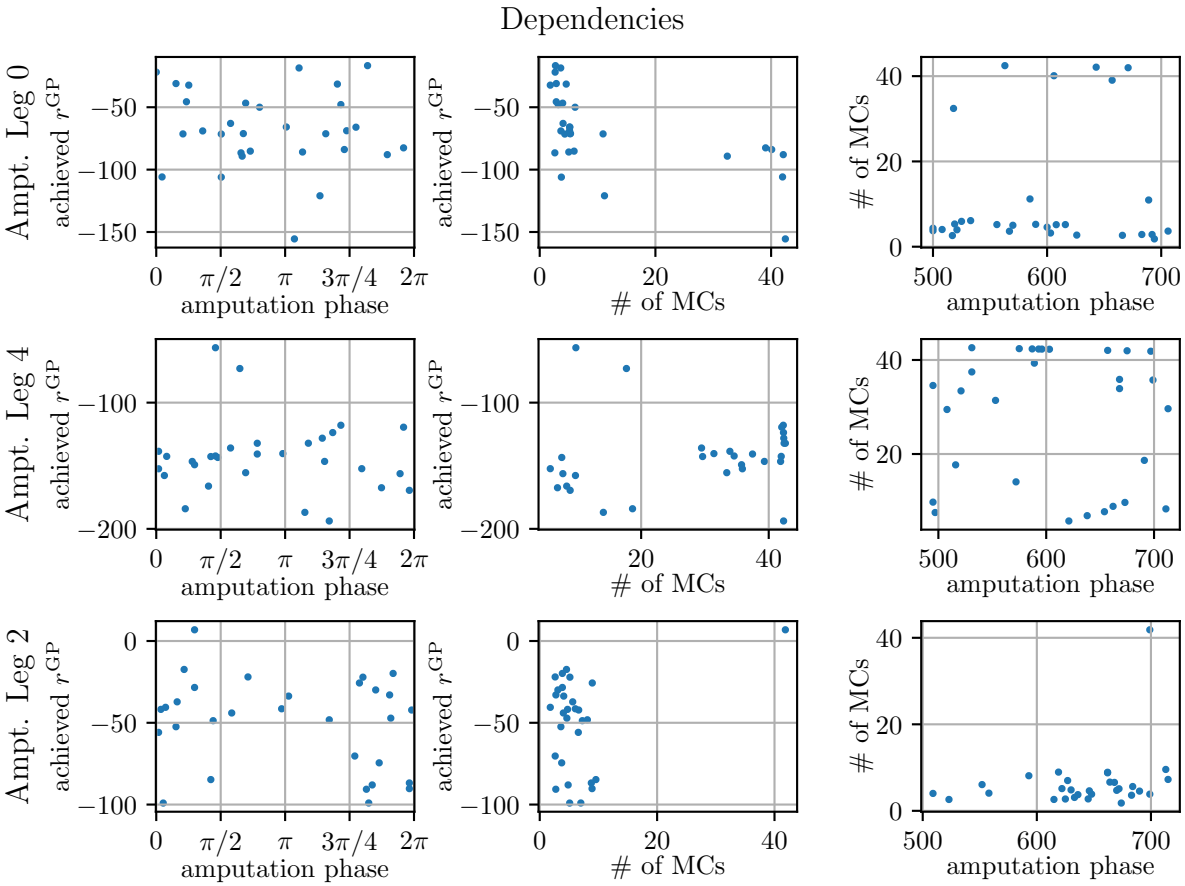

Figure 10: The figure shows plots of reward value r^{GP} dependency on the phase of amputation performance and length of adaptation (given in number of MCs) in the first and second column, respectively, and the dependency of the learning duration on the phase of amputation performance in the third column.

6.4 Reward Dependency on the GP

If the space of the GP evaluation dependency on the GP rhythm was known, classical optimization methods, like gradient descent, could be used to solve the problem of locomotion adaptation after leg amputation optimally. However, beforehand, the space is unknown. Nevertheless, the results gathered during the experiments give a hint of some parts of the space, which is introduced in this section and further discussed in the next Chapter.

The individual GPs with their measured reward generated by evolution and the tested methods offer an insight into the reward dependency on the GP. However, even with one leg amputated and one leg having a fixed swing start (hind right leg 3), the vector representing the GP has four elements. Therefore, I present the dependency in a series of plots, where each plot depicts swing starts $\phi_i(t)$ of two legs i and k on axis x and y, respectively, with reward function value for the given configuration depicted by color as presented in Figures 11, 13, and 15 for amputation of front left, middle left and hind left leg, respectively.

There is a visible tendency of my method results to cluster into lines in the GP subspace in the Figures 11, 13, and 15. The reason is the method's origin in the CRs, which enforces specific phase offsets of leg swing starts. All configurations on a line parallel to $\phi_i = \phi_k$ have equal differences, e.g., equal phase offset of the swing starts, which the CRs favor. The results show multiple correlations with the CRs, which are further discussed in the following Chapter.

6.5 The Resulting Gait Patterns and Their Similarities

The resulting output of the proposed and compared methods is the amputation-adapted GP. The reward-GP space visualized in Figures 11, 13, and 15 show only fragments of the adapted GPs, because each point gives only information about LMC starts of two legs, but not about the whole GP. Hence, a few picked GPs are presented in Figures 17, 18 and 19 for scenarios with amputated front left, middle left and hind left legs, respectively.

In scenario with amputated middle left leg (leg 4, see Figure 18), the diagonally oposing legs (i.e., pairs 1, 2 and 0, 3)tend to undergo the swing simultaneously. Such behavior makes sense from more angles of view: (i) The legs 0 and 1, and the legs 2 and 3 are contralateral; hence, their movement should be in antiphase, while legs 0 and 2 should also perform their LMCs in antiphase to each other because they are the only two remaining legs on the left side of the robot's body. The solution of synchronizing the diagonally opposing legs is the only solution in such scenario.

(ii) The robot in this scenario resembles quadruped with the additional leg on the right side of the body. Therefore, it seems natural the robot tends to use GP typical for quadrupeds[8], with the "*extra*" leg performing its movement somewhere within the MC.

A similar trend can be observed in the scenario with amputated hind left leg (2, see Figure 19) with the diagonally opposing pairs of legs 1, 4, and 0, 3, with the exception of Miguel's approach, where the middle left leg tends to synchronize with the middle right leg (5). However, in the scenario with the amputated front left leg (0, see Figure 17), the same logic does not seem to apply for diagonally opposing pair of legs 3, 4 for Miguel's and evolution method. The proposed method follows the previously mentioned trend. The cause of such difference will be a subject of future research.

Despite different adaptation approaches, the resulting GPs seem to have similar phase offsets of the LMCs for respective legs, or in other words, similar relations between legs seem to emerge from the different approaches. Such results suggest the fusion of the methods would not lead to large conflicts between them but could improve their performance.

[8]Many four-legged animals, including dogs, horses, etc., walk with such gait with front and rear legs from opposing sides performing swing and stance together.

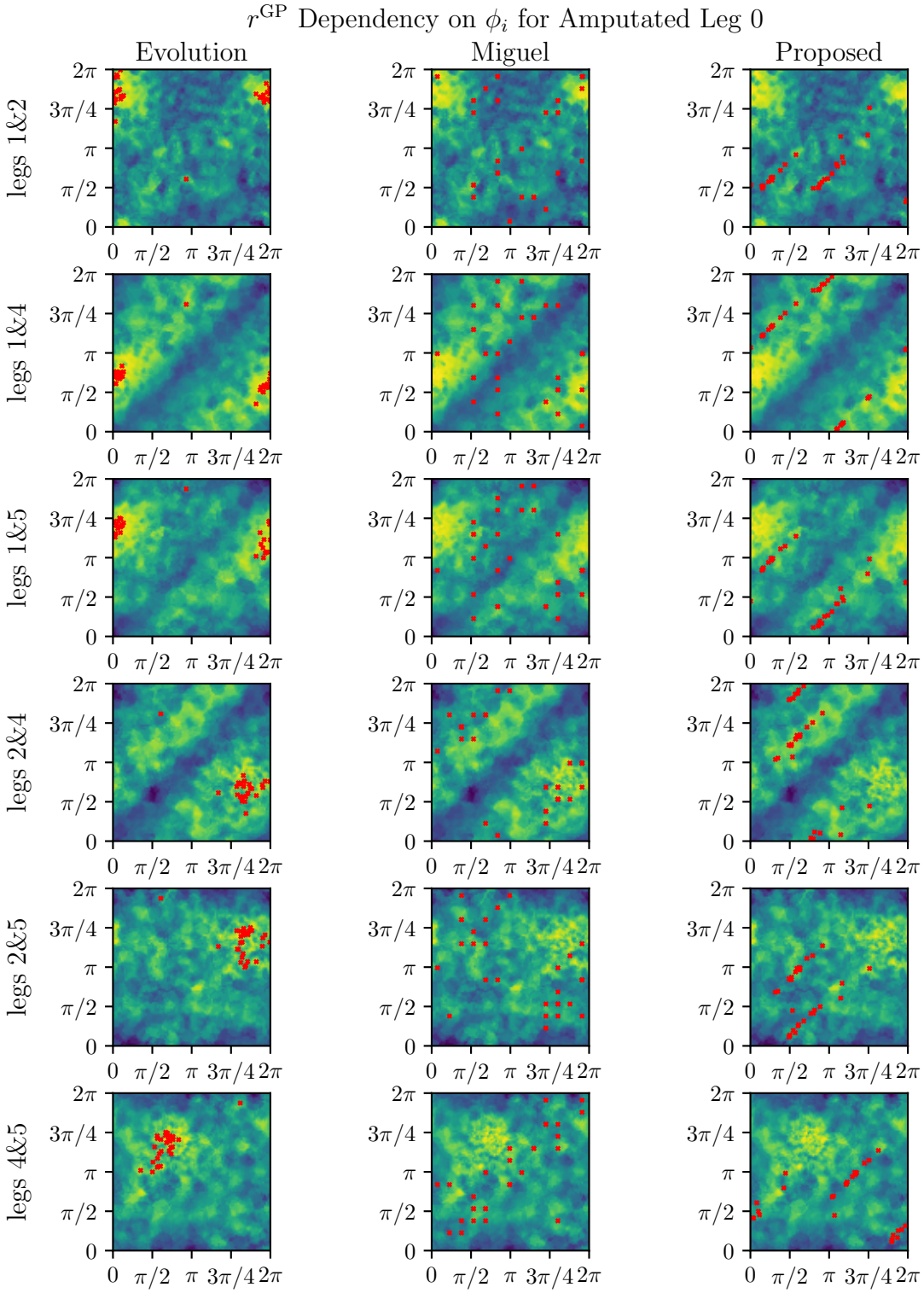

Figure 11: Approximation of reward value r^{GP} for amputation of front left leg given the swing starts ϕ_i and ϕ_k for legs i and k, respectively. Each row represents the values for legs $i\&k$, with legs i and k depicted on x and y axes, respectively. Each red cross represents one individual in evolution or one experiment resulting GP in Miguel's and my methods' in the first, second, and third columns, respectively. The achieved r^{GP} is depicted by the color, with maximum and minimum values represented by brightest yellow and darkest blue, respectively. Note the dependencies are, in fact, dependencies of three leg swing starts. The third leg is the hind right leg 3 with fixed swing start $\phi_3 = 0$.

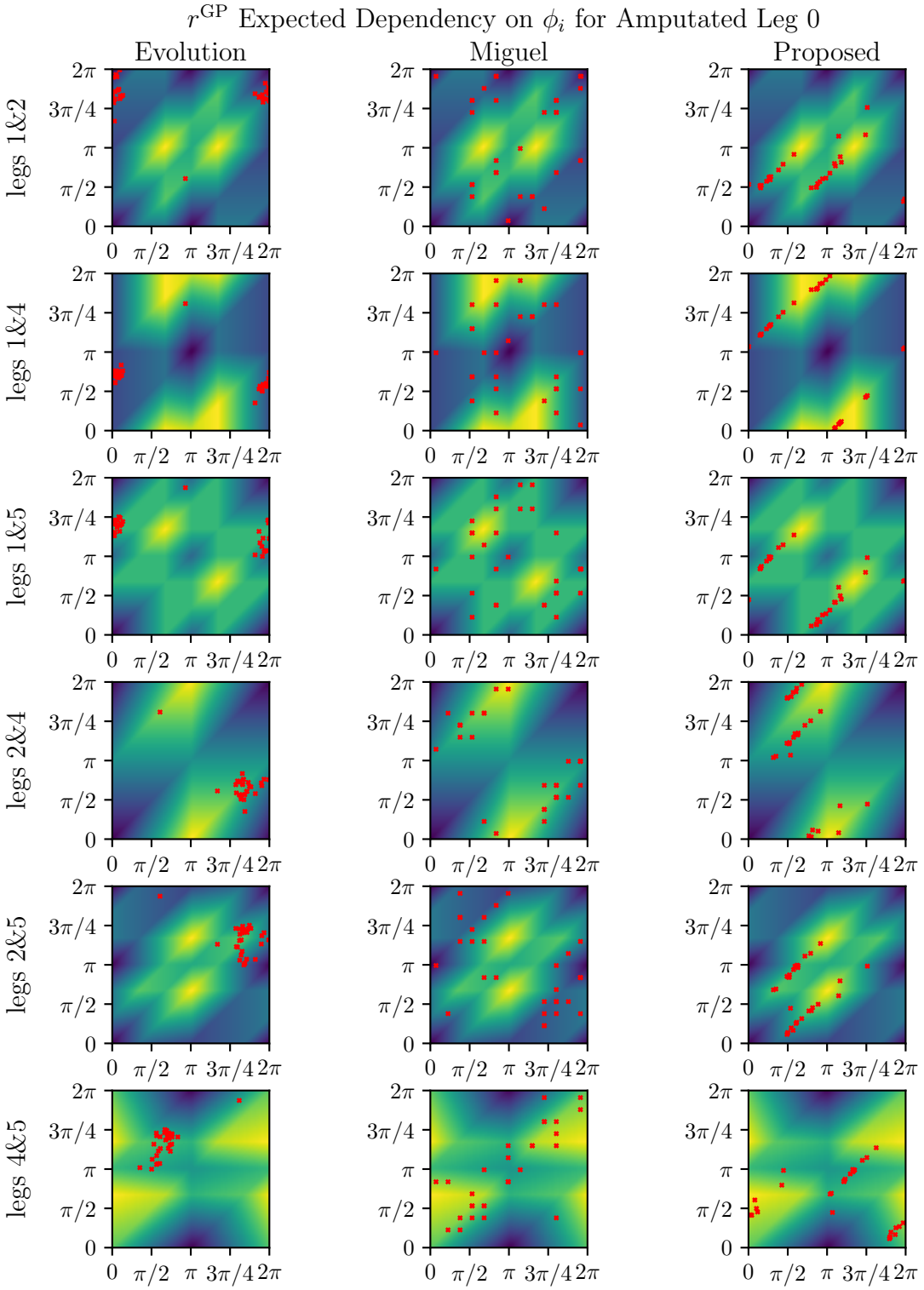

Figure 12: Approximation of expected reward value r^{GP} for amputation of front left leg given the swing starts ϕ_i and ϕ_k for legs i and k, respectively. Each row represents the expected values for legs $i\&k$, with legs i and k depicted on x and y axes, respectively. Each red cross represents one individual in evolution or one experiment resulting GP in Miguel's and my methods' in the first, second, and third columns, respectively. The expected r^{GP} for the given GP is depicted by the color, with maximum and minimum values represented by brightest yellow and darkest blue, respectively. Note the dependencies are, in fact, dependencies of three leg swing starts. The third leg is the hind right leg 3 with fixed swing start $\phi_3 = 0$.

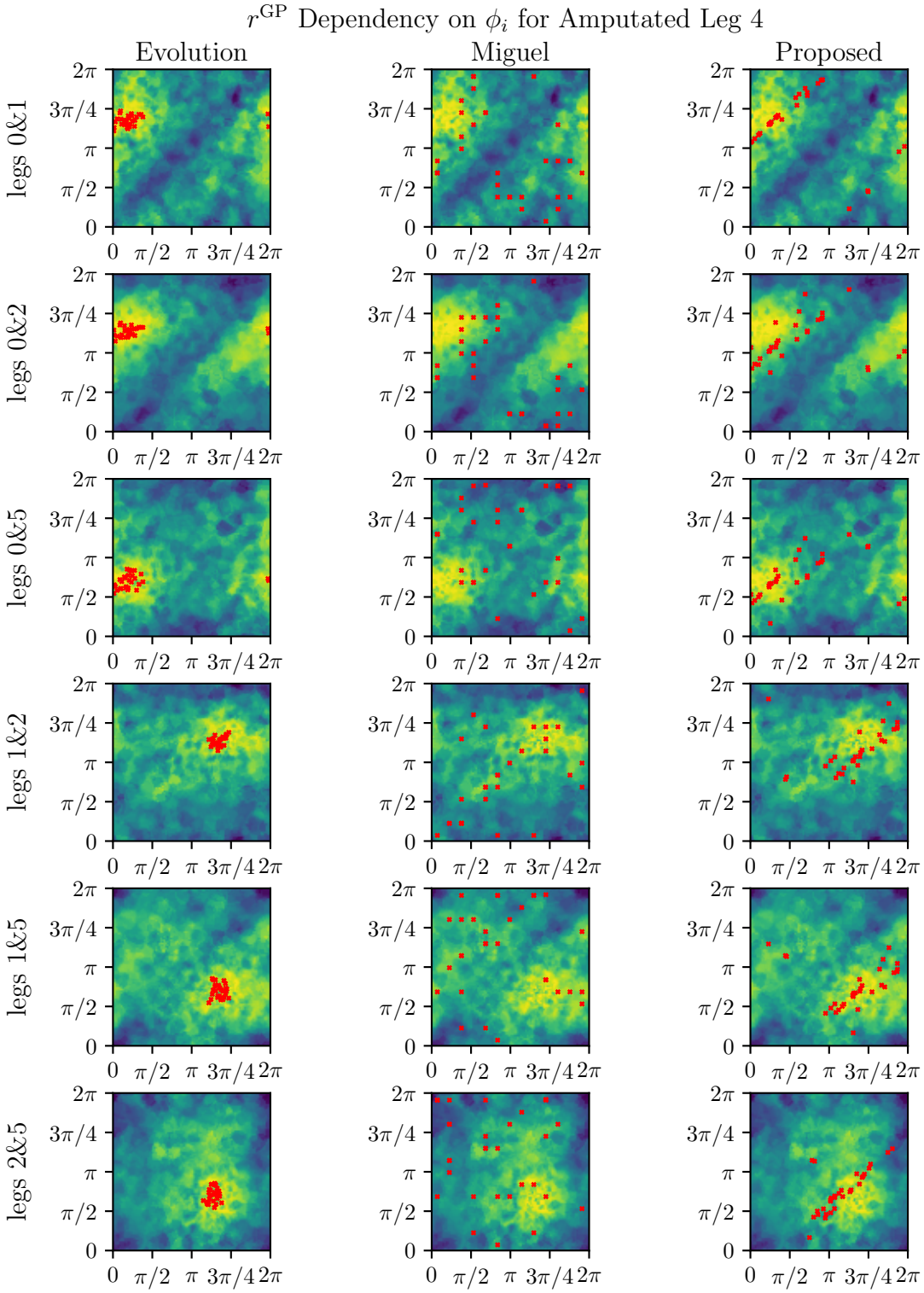

Figure 13: Approximation of reward value r^{GP} for amputation of middle left leg given the swing starts ϕ_i and ϕ_k for legs i and k, respectively. Each row represents the values for legs $i\&k$, with legs i and k depicted on x and y axes, respectively. Each red cross represents one individual in evolution or one experiment resulting GP in Miguel's and my methods' in the first, second, and third columns, respectively. The achieved r^{GP} is depicted by the color, with maximum and minimum values represented by brightest yellow and darkest blue, respectively. Note the dependencies are, in fact, dependencies of three leg swing starts. The third leg is the hind right leg 3 with fixed swing start $\phi_3 = 0$.

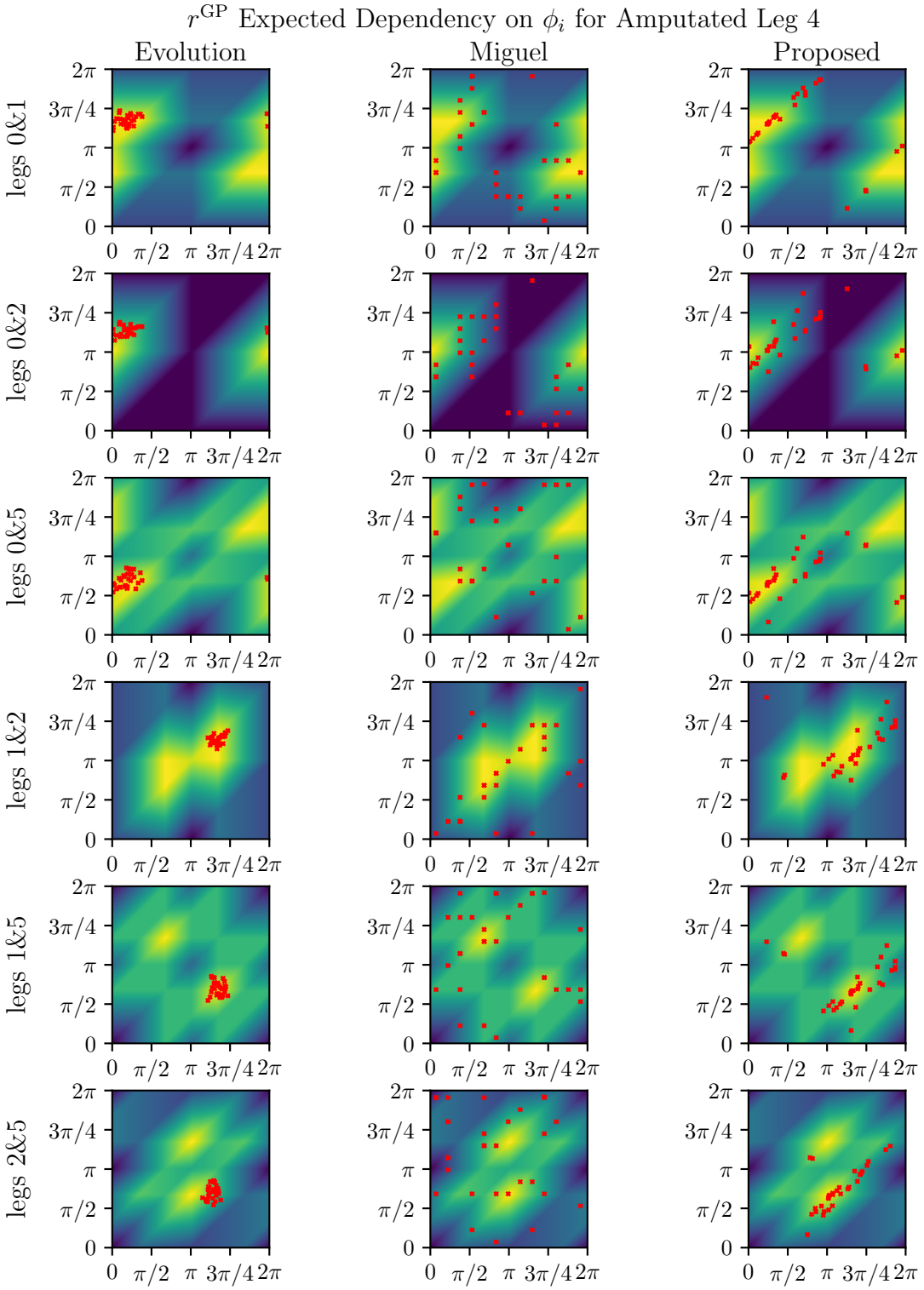

Figure 14: Approximation of expected reward value r^{GP} for amputation of middle left leg given the swing starts ϕ_i and ϕ_k for legs i and k, respectively. Each row represents the expected values for legs $i\&k$, with legs i and k depicted on x and y axes, respectively. Each red cross represents one individual in evolution or one experiment resulting GP in Miguel's and my methods' in the first, second, and third columns, respectively. The expected r^{GP} for the given GP is depicted by the color, with maximum and minimum values represented by brightest yellow and darkest blue, respectively. Note the dependencies are, in fact, dependencies of three leg swing starts. The third leg is the hind right leg 3 with fixed swing start $\phi_3 = 0$.

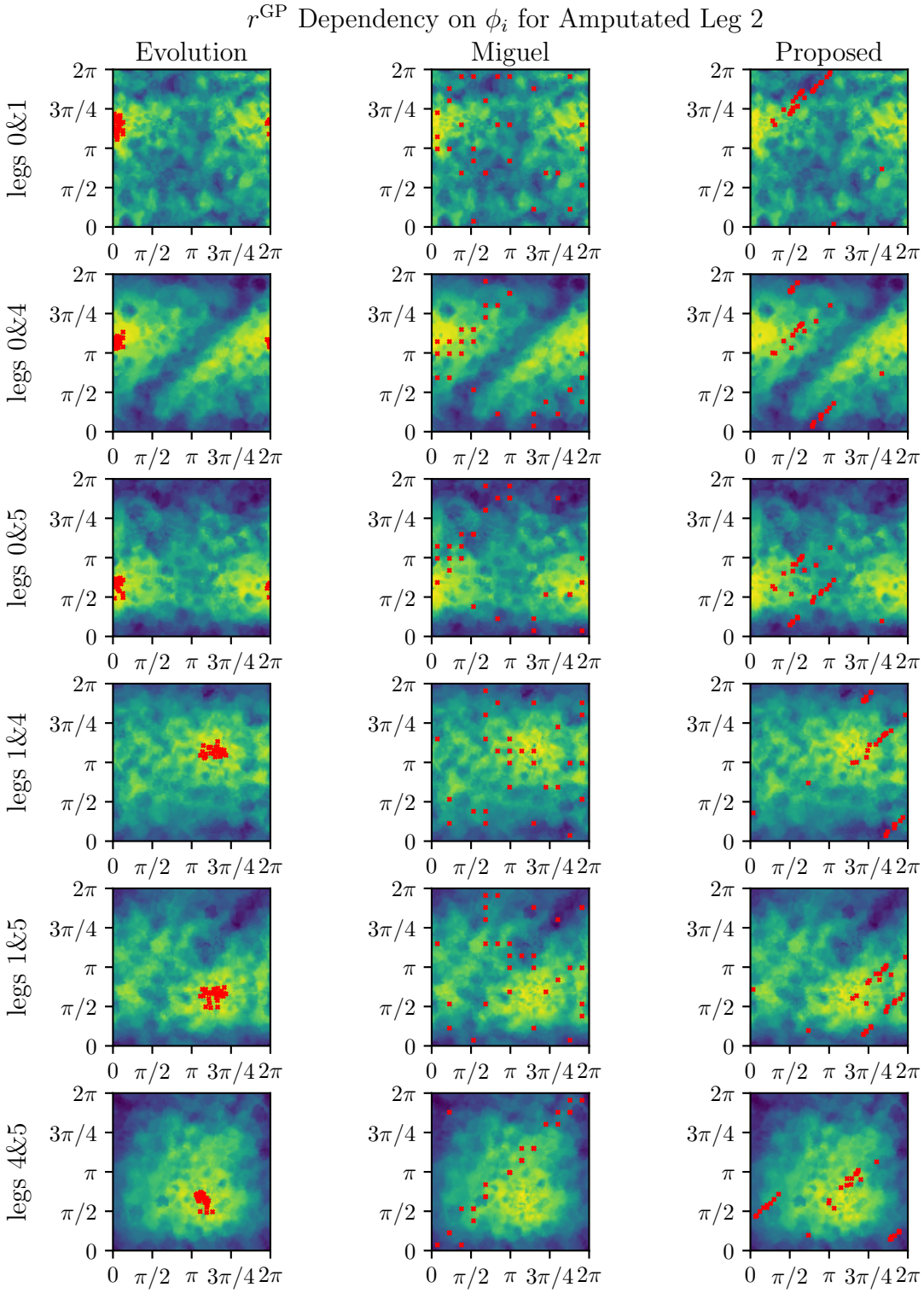

Figure 15: Approximation of reward value r^{GP} for amputation of hind left leg given the swing starts ϕ_i and ϕ_k for legs i and k, respectively. Each row represents the values for legs $i\&k$, with legs i and k depicted on x and y axes, respectively. Each red cross represents one individual in evolution or one experiment resulting GP in Miguel's and my methods' in the first, second, and third columns, respectively. The achieved r^{GP} is depicted by the color, with maximum and minimum values represented by brightest yellow and darkest blue, respectively. Note the dependencies are, in fact, dependencies of three leg swing starts. The third leg is the hind right leg 3 with fixed swing start $\phi_3 = 0$.

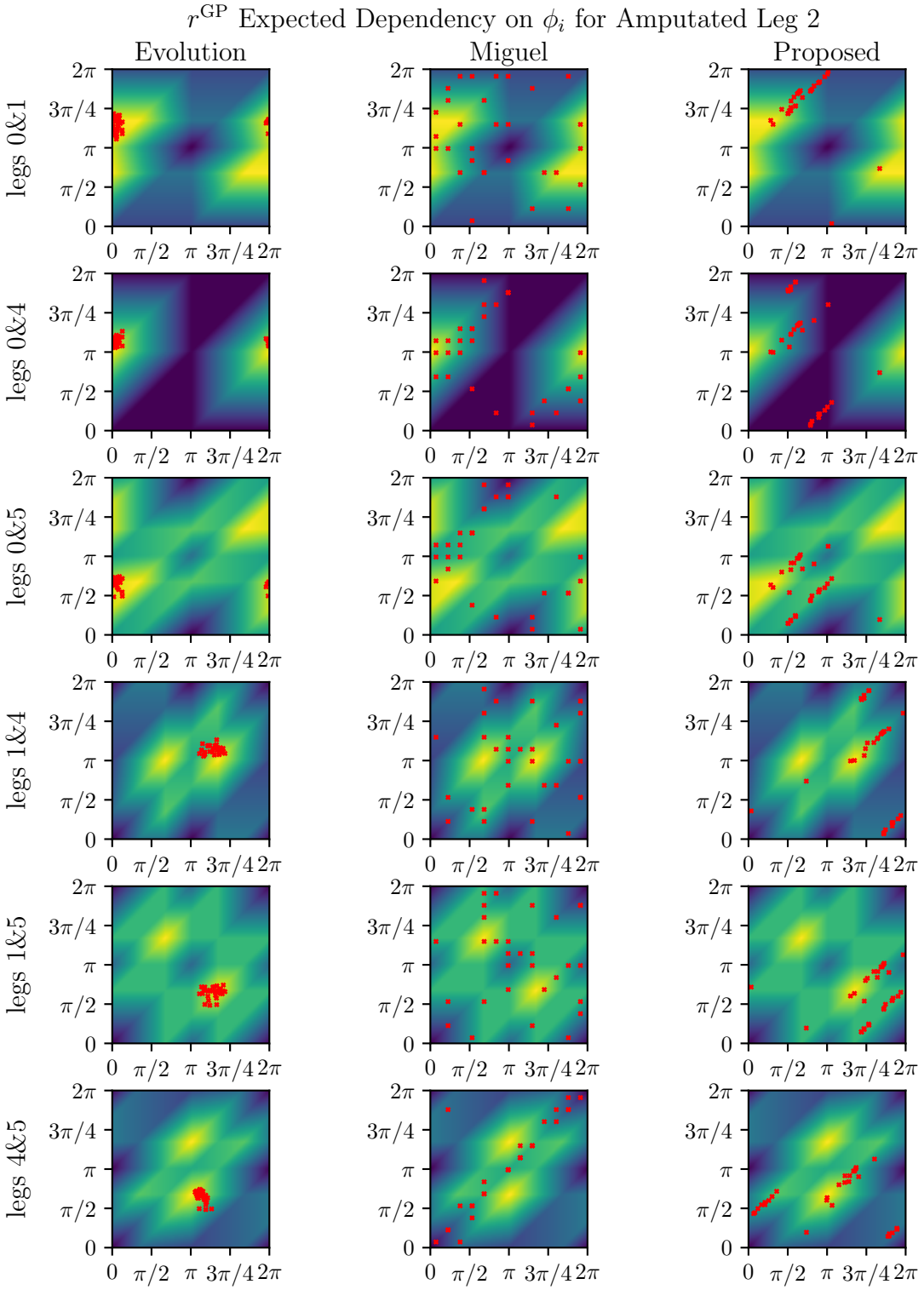

Figure 16: Approximation of expected reward value r^{GP} for amputation of hind left leg given the swing starts ϕ_i and ϕ_k for legs i and k, respectively. Each row represents the expected values for legs $i\&k$, with legs i and k depicted on x and y axes, respectively. Each red cross represents one individual in evolution or one experiment resulting GP in Miguel's and my methods' in the first, second, and third columns, respectively. The expected r^{GP} for the given GP is depicted by the color, with maximum and minimum values represented by brightest yellow and darkest blue, respectively. Note the dependencies are, in fact, dependencies of three leg swing starts. The third leg is the hind right leg 3 with fixed swing start $\phi_3 = 0$.

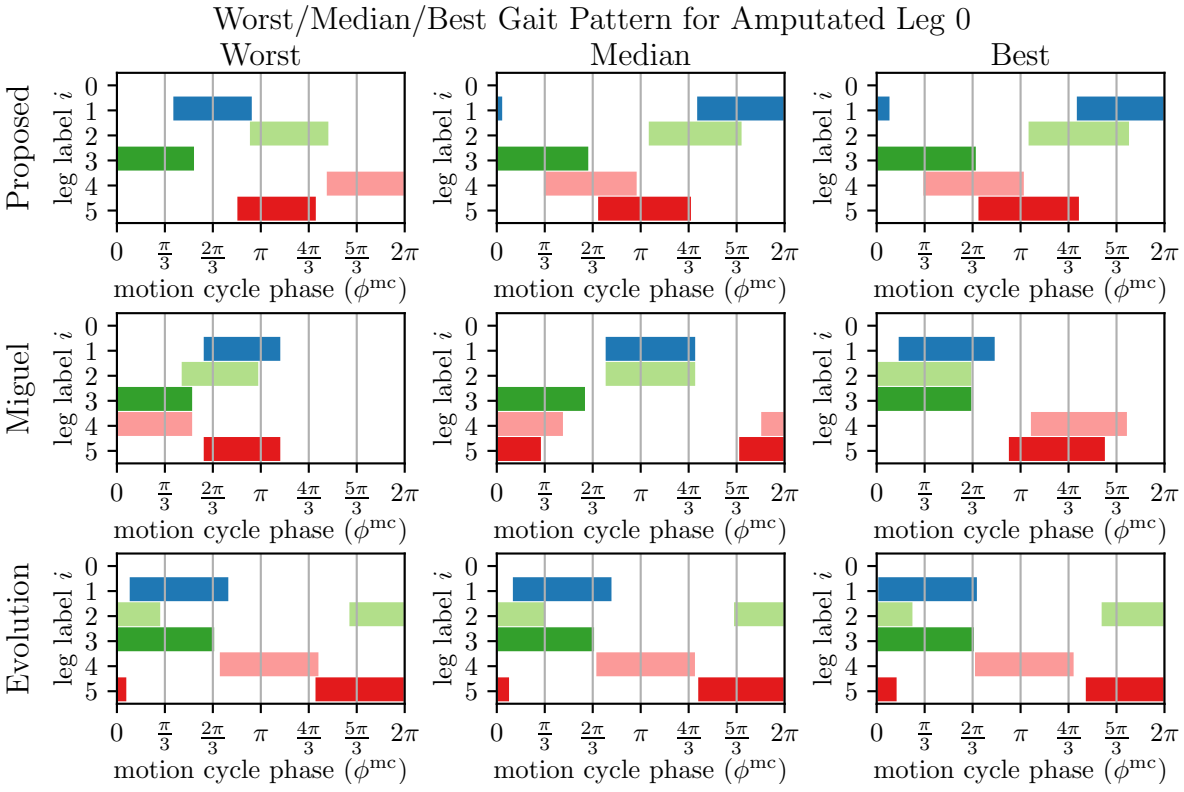

Figure 17: Showcase of GPs adapted to amputation of the front left leg by different methods. The individual rows belong to the proposed method, Miguel's approach, and the evolution algorithm, from top to down, respectively. Three GPs are depicted for each method: The worst scoring, the one with median reward, and the best achieved from left to right, respectively. (For the evolution, the worst is the worst out of the 30 best individuals, not the worst individual in the whole evolution process.) The leg label i is on the y axis, while the MC phase $\phi^{\mathrm{MC}}(t)$ is depicted by the x axis. The color bar represents the respective leg's swing phase, with its left border representing the swing start $\phi_i(t)$ and length representing the swing duration $\Delta\phi$. The individual leg colors and labels correspond to the schema in Figure 1.

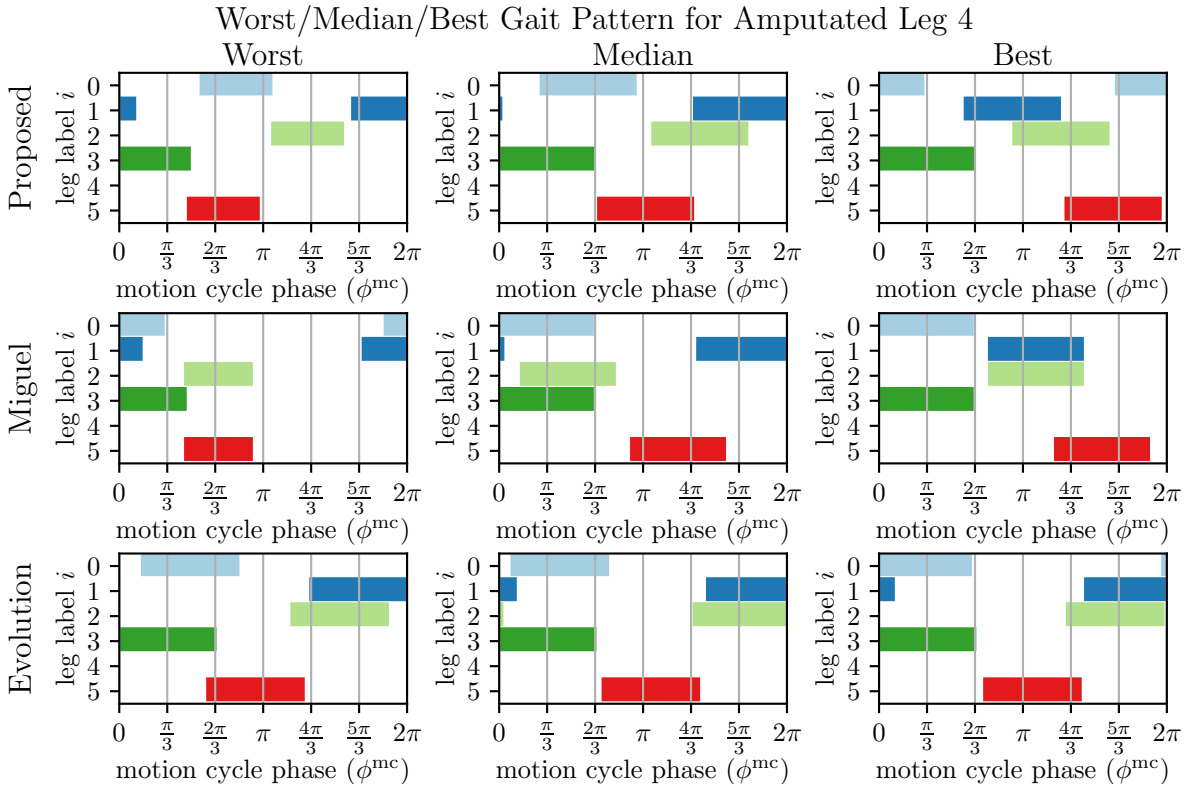

Figure 18: Showcase of GPs adapted to amputation of the middle left leg by different methods. The individual rows belong to the proposed method, Miguel's approach, and the evolution algorithm, from top to down, respectively. Three GPs are depicted for each method: The worst scoring, the one with median reward, and the best achieved from left to right, respectively. (For the evolution, the worst is the worst out of the 30 best individuals, not the worst individual in the whole evolution process.) The leg label i is on the y axis, while the MC phase $\phi^{\mathrm{MC}}(t)$ is depicted by the x axis. The color bar represents the respective leg's swing phase, with its left border representing the swing start $\phi_i(t)$ and length representing the swing duration $\Delta\phi$. The individual leg colors and labels correspond to the schema in Figure 1.

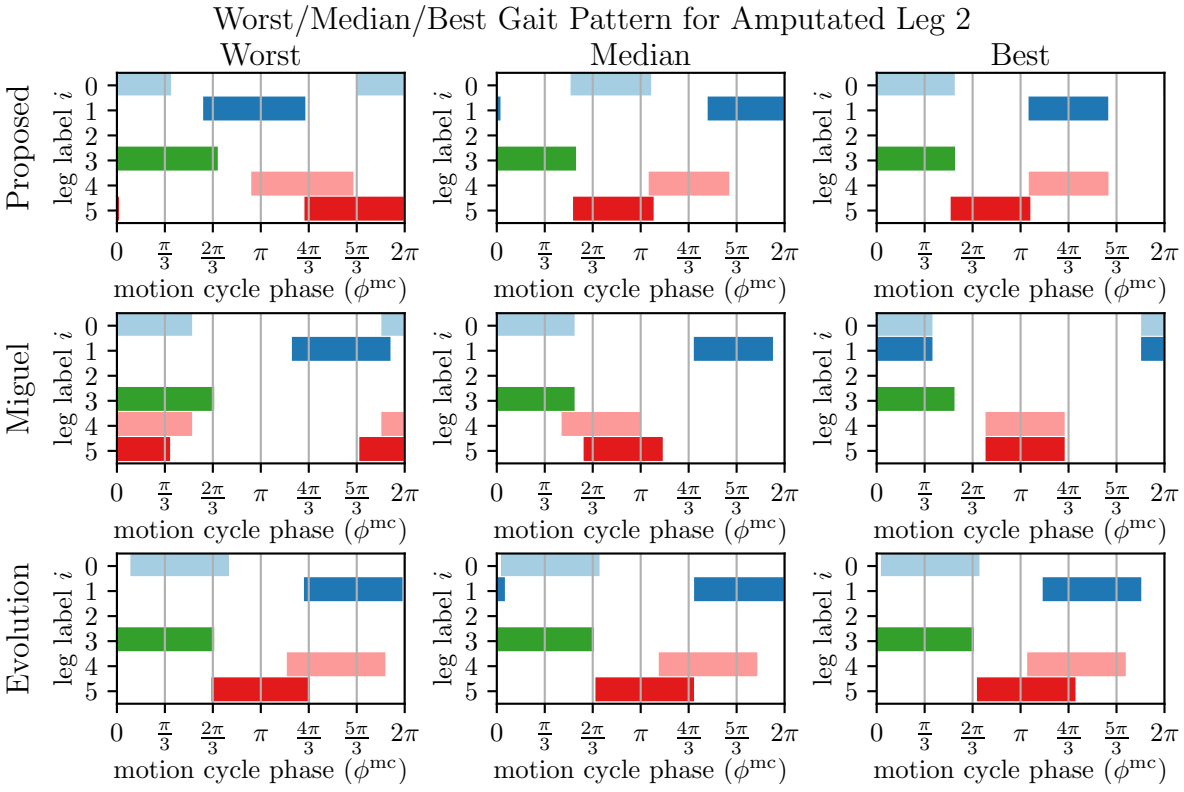

Figure 19: Showcase of GPs adapted to amputation of the hind left leg by different methods. The individual rows belong to the proposed method, Miguel's approach, and the evolution algorithm, from top to down, respectively. Three GPs are depicted for each method: The worst scoring, the one with median reward, and the best achieved from left to right, respectively. (For the evolution, the worst is the worst out of the 30 best individuals, not the worst individual in the whole evolution process.) The leg label i is on the y axis, while the MC phase $\phi^{\mathrm{MC}}(t)$ is depicted by the x axis. The color bar represents the respective leg's swing phase, with its left border representing the swing start $\phi_i(t)$ and length representing the swing duration $\Delta\phi$. The individual leg colors and labels correspond to the schema in Figure 1.

Chapter 7
Discussion

The proposed method deals with real-time gait adaptation after leg amputation to enable the robot to leave the place where the damage originated as fast as possible with no need for continual sensory input, which can also be unreliable due to the damage. The method is compared with the reactive method proposed by Miguel et al. [14] and with the offline-learning approach represented by the evolution algorithm, where the mutual methods comparison metric is the proposed reward function for GP evaluation. The proposed method proved itself to be overall more consistent in terms of traversed path and achieved utility given by the reward function, and in two out of three scenarios, it performed better than the competitive online method.

The previous Chapter 6, containing experiment results, introduces some interesting questions to discuss. Firstly, how do the online adaptation methods, represented by the proposed method and competitive Miguel's method, stand in comparison with the optimal solution given by evolution optimization? Secondly, how do the results obtained in the experiments connect to the original concept of coordination rules and the proposed adjustments? Last but not least, what are the implications of the GP rhythm adaptation and of the correlation of the expected and obtained reward value space? This Chapter aims to discuss and provide insight into those three topics, including a discussion about the shortcomings of the results and possible future work.

7.1 Online Adaptation Versus Evolution Optimization

The main difference between evolution optimization and the online adapting methods is the amount of computational time and computational resources needed for evolution to converge to an optimal solution. The online methods adapt within a few MCs, i.e., in a matter of seconds, while the evolution optimization took ≈ 9 hours to finish. If the robot is supposed to adapt using the evolution, it needs to perform a few MCs of each generated individual to find its reward value, or it needs to be able to run the simulator to test the solutions before use in the real world.

During such time, the robot needs to stop its locomotion and risk damage due to the evaluation of very bad individuals in the case of testing the individuals in the real world. In the case of in-simulation evaluation, the robot has limited computational power for other tasks due to the need to run the simulator, and the locomotion also has to stop until satisfyingly good parameters are found.

In this thesis, the evolution runs for fifty generations. However, one could argue that we do not need to run so many generations of evolution to outperform the best results of the online adapting methods. In fact, in the first generation, a very good solution, sometimes outperforming the best-achieved solution given by the online adapting methods, is already present, as shown in Figures 20, and 21.

The problem is that the best solution is one out of two hundred randomly generated individuals. Even if all the ten best randomly generated individuals would be considered satisfyingly good enough, we would still need to evaluate at least twenty randomly generated individuals to hope to have a good chance of finding one equivalently good solution. With the use of five MCs for evaluation of each, that would be one hundred MCs for evaluating locomotion. Still, we could be unlucky and end up with a very bad or even dangerous (servos-damaging) solution.

The online approaches usually need much fewer MCs to adapt (see Fig 10 for the adaptation length of the proposed method). Moreover, in the state of adapting, the robot usually does not perform such

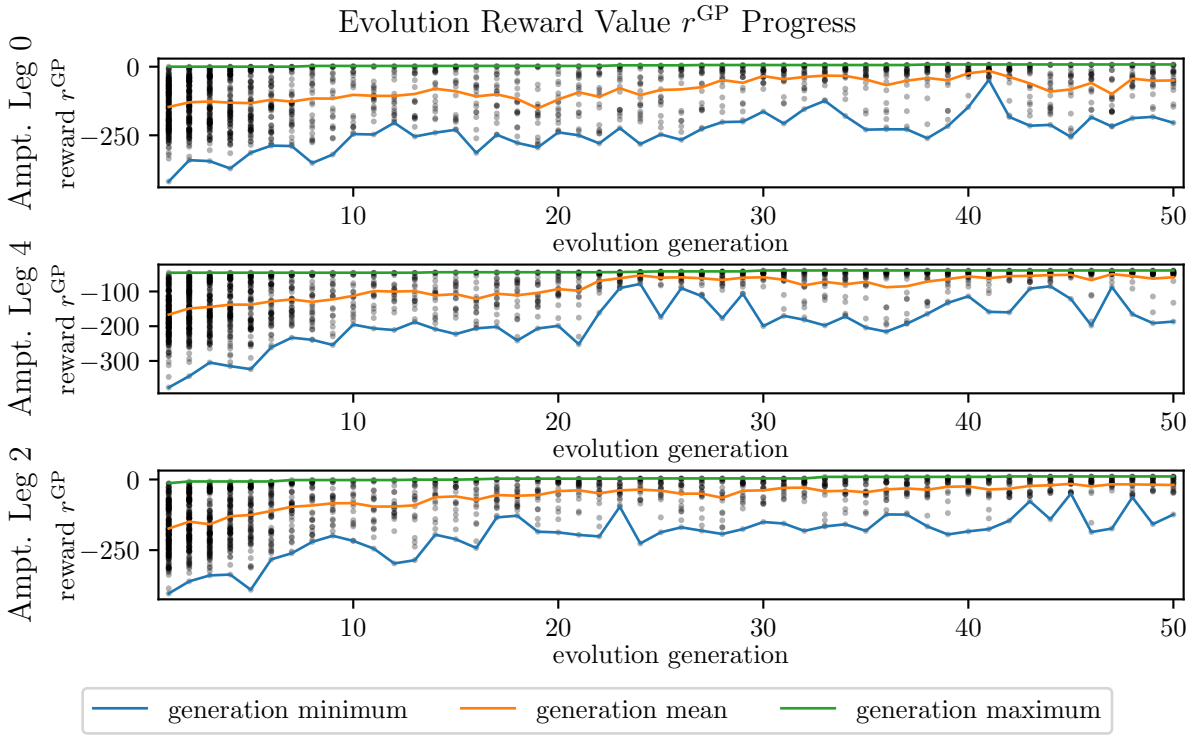

Figure 20: The figure shows the evolution algorithm progress for all three amputation scenarios. x axis depicts the generation number and y axis depicts the reward value r^{GP}. Each black dot is an individual in the given generation. The green, orange, and blue lines depict the maximum, mean, and minimum r^{GP} values in each generation, respectively.

self-damaging behavior as the randomly generated parameters could. In case to fully use the potential of evolution optimization, we would need to evaluate many more individuals, resulting in even longer adaptation time.

Given the hexapod walking robot has six legs, the adapted gaits could be precomputed and stored in the robot's memory before the mission even starts. In this case, such solution would beat the online approaches because it could be used instantly and would be better performing. However, the number of possible malfunctions and their intensity is too high to precompute every possible damage scenario. If future research expands the proposed rules, it could provide a dynamic system that is able to handle any kind of damage (e.g., implementing limping in case of lighter servo motor malfunction). In that case, the future scaling of the number of damage scenarios would favor the online-based approaches.

In terms of obtained reward, the gap between the tested online methods and their upper bound, given by the evolution's best solution, is relatively small, and the best online adapted individuals' resulting GPs have behavior trends similar to the best GPs achieved by the evolution optimization.

Given all the reasoning mentioned above, given the experiment conditions and the GP evaluation by the proposed reward value, I conclude the online adapting methods are more suitable for the locomotion adaptation after leg amputation than the evolution optimization approach.

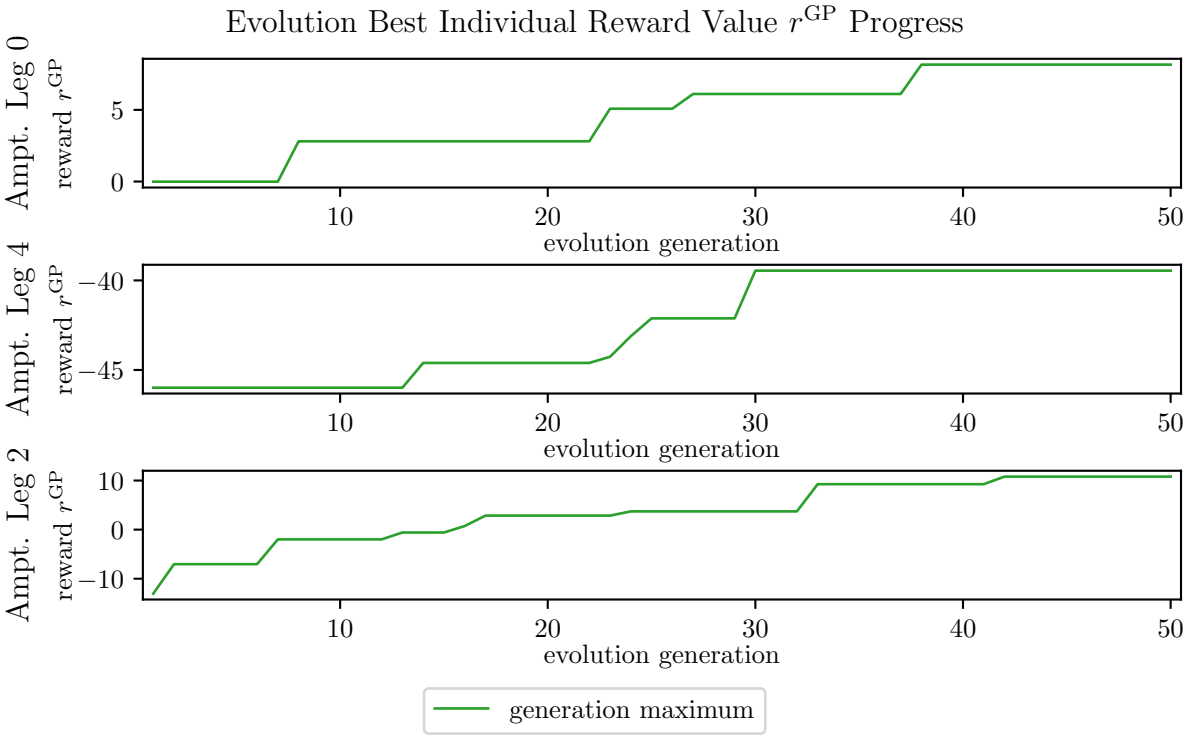

Figure 21: The figure shows the best-rated individual progress within the evolution algorithm for all three amputation scenarios. x axis depicts the generation number and y axis depicts the reward value r^{GP}. The green line depicts the maximum r^{GP} value reached in each generation. The trend is non-decreasing because the best 15% of individuals from the generation survive to the next generation; thus, the best reward in a new generation is at least as high as in the previous generation.

7.2 Relation Between Coordination Rules and Gait Patterns' Reward Value

While observing interpolated reward space, shown in Figures 11, 13, and 15, we can recognize some reappearing trends, e.g., a valley of low values around the line $x = y$, lower values in corners and around the edges of the given space, etc. The reward value r^{GP} trends for given pairs of $\phi_i(t)$s indicate which phase offsets of (i.e., the relation between) the respective LMCs (and by extension, which GPs) are more likely to achieve good results. However, shifting the legs swing starts $\phi_i(t)$, determining the LMCs offset, is exactly the point of the CRs and their adjustments. Therefore, one could expect we can extract the coordination rules from the reward value dependency on the swing starts, depicted in Figures 11, 13, and 15.

I decided to perform such comparison from the other end, and I created simplified plots of expected dependency of reward value r^{GP} on the leg swing starts $\phi_i(t)$ given the CRs, as presented in Figures 12, 14, and 16. In the following paragraphs, I provide the reasoning for generating the dependencies the way they are depicted in the mentioned Figures systematically from first Figure 12 to the last Figure 16 and from top to down in each Figure.

For each pair of the swing start $\phi_i(t)$ values, there are several corollaries based on the CRs and CCARs. The highest expected reward is in the area where all the corollaries are met. On the contrary, the lowest expected reward is in the areas where none of the corollaries is met. Each corollary contributes to the expected reward value as follows: Each corollary has a weight of $\frac{1}{n}$, where n is the number of corollaries for the given pair of values. How much are the individual corollaries met for a given pair of swing start values $\phi_i(t)$ is given by the difference of the by-corollary-given phase offset and the actual phase offset of the $\phi_i(t)$ values. The corollaries for the individual pairs of swing start values are presented in Appendix B.

All the corollaries result in the expected reward value dependency on the legs' swing start offsets, depicted in Figures 11, 13, and 15. I claim the proposed method obeys the CRs and CCARs on which the corollaries are based. Indeed, comparing the achieved configurations (the red crosses in the Figures), they occur near the expected maximums, proving the system is really designed with respect to the CRs and CCARs.

However, the important question remains: Do the CRs-and-CCARs-based corollaries match the optimal solutions given by the evolution algorithm?

Comparing the expected and obtained reward value dependencies side by side, we can observe similar trends. The most remarkable similarity is probably the valleys of minimums along the line $x = y$. Every time they occur in the measured results (leg pairs 1&4, 1&5 and 2&4 in the scenario with amputated leg 0, see Figure 11; leg pairs 0&1, 0&2 and 1&5 in the scenario with amputated leg 4, see Figure 13; and leg pairs 0&4 and 1&5 in the scenario with amputated leg 2, see Figure 15) they also appear in the expected results. Such valley means the values should not be similar to each other.

Another occurring resemblance is the lower reward values in the corners (leg pairs 1&4, 1&5, 2&4 and 2&5 in the scenario with amputated leg 0, see Figure 11; 0&1, 0&2, 1&2, 1&5 and 2&5 in the scenario with amputated leg 4, see Figure 13; and leg pairs 0&4, 1&4, 1&5, and 4&5 in the scenario with amputated leg 2, see Figure 15) and around the edges (leg pairs 1&2, 1&5, 2&5 and 4&5 in the scenario with amputated leg 0, see Figure 11; leg pairs 0&1, 0&2, 0&5, 1&2 and 2&5 in the scenario with amputated leg 4, see Figure 13; and leg pairs 0&4, 0&5, 1&4, 1&5 and 4&5 in the scenario with amputated leg 2, see Figure 15) of the shown space, which co-occur in most of the instances.

Few of the configurations also show similar maximum trends in the compared Figures (the resemblance stands out in leg pairs 1&5 and 2&4 for the scenario with amputated leg 0, see Figure 11 and all the leg pairs of the scenarios with amputated legs 4 and 2, see Figures 13 and 15, respectively). Nevertheless, the mentioned corollaries building the expected reward dependency trends are only approximate, which often leads to unnecessary symmetries, highlighting some specific values much more than the neighboring ones (even in cases of the corollary where "*the values should be*

more similar than different"), etc. Therefore, a deeper analysis of the CRs and CCARs and a more precise reward value expectation approach could arguably estimate the trends even better.

The above-mentioned similarities show the expected CRs-and-CCARs-based search space correlates with independently created reward-value-based search space.

7.3 Limitations and Future Work

This thesis worked mainly with the concept of leg damage, meaning the leg amputation. Nevertheless, The leg damage does not have to be binary, i.e., fully functional or in need of amputation, but the leg can be slightly damaged and still usable. Future research should explore the possibility of using the slightly damaged leg to help the locomotion, e.g., via limping.

Whatever many hours of the simulation were run for data gathering in this thesis, there was no time left for deploying all the methods on a real walking robot, partially due to the unpredictability of the randomly generated individuals in the evolution algorithm, which could damage the robot's servo motors. However, the deployment in the real world can always have very different results from the results gathered in simulations.

The proposed method was already tested on a real walking robot using a bit more complicated locomotion controller architecture in our work in [18].[9] However, the results of the adjusted Miguel's method and the EA approach have not yet been tested on a real hexapod walking robot. Such comparison will be the subject of future research.

A few preliminary experiments were run with real walking hexapod robot *Daisy* by HEBI robotics, which is of larger scale and has differently shaped legs and body than the model *PhantomX-v3.1* used in the simulation or the *SCARAB* robot used in the experiments in the previous work [18]. The observed locomotion behavior showed the proposed method does not solely solve the adaptation for the tested robot. Its different weight distribution restricted some of the legs from lifting off during the locomotion due to the additional load put on the front legs by the absence of the amputated leg. Therefore, the fusion of the proposed and competitive method, or some other sensory-based approach, will probably be necessary, if a universal solution should be achieved.

The movement range and the legs' default positions in relation to the body were all predetermined by motion primitives with given bounds, meaning the proposed methods had no control over the movement amplitudes. Therefore, the locomotion adaptation could probably be improved by fixing the rhythm given by the compared methods and optimizing the amplitudes.

In Section 5.3.1, I mentioned the straightforward approach to optimize the GP by tunning 3996 parameters (one servo motor position command for each servo motor for each iteration within the MC), which would reduce to $15 \cdot 222 = 3330$ parameters for case of one amputated leg, optimizing both, rhythm and amplitudes at the same time, which can be decoupled to handle separately the rhythm and the amplitudes. Even if no other reductions were introduced and a similarly straightforward approach would be used to optimize only the amplitudes after using the compared methods for rhythm optimization, the number of parameters would decrease significantly. The motion primitives, activated in the right moments given the GP rhythm, could be parametrized by only two numbers for each servo, representing the range of movement for the respective servo motor, i.e., $18 \cdot 2 = 36$ parameters, or $15 \cdot 2 = 30$ parameters for case of one amputated leg. Therefore, the introduced reduction yields in total 35 parameters[10] for locomotion after amputation of one leg, significantly reducing the search space. Future work could explore the relations between amplitudes of individual legs and even further reduce the search space, decreasing the needed computational resources.

[9] The work also provides results of amputation of not only one but two legs simultaneously. However, including such experiments for all three discussed methods would significantly increase the time and computational power demand for this thesis.

[10] 30 for amplitudes and 5 for LMC starts and swing duration. Recall that one LMC start is fixed to value 0 to avoid symmetries.

Therefore, future experiments with differing morphologies will also be targeted to answer the question of whether the relations between the evolution-algorithm-found optimals and the expected optimals based on the CRs and proposed CCARs still hold for different morphologies. Such robustness could be partially tested by changing the default robot posture in the simulation (making the robot stand on more straightened legs or with legs spreaded to a greater distance around the robot's body, etc.). These simulations can be run before experiments using different real-world robots to ensure the behavior would not damage them.

The similarities between the expected reward value search space, based on the implications of the CRs and CCARs, and the experimentally obtained reward value search space show that the CRs and CCARs could be used as heuristic for the optimal reward value search. The reward value is dependent on sensory data, including robot position, velocity, and rotation tracking. Such data can be difficult to obtain with high precision, especially on small robots with limited payload for sensors or in environments with low visibility and/or no GPS connection. The proposed method provides a way to improve the optimal locomotion search in conditions where sensors are not available or reliable.

The observed similarities support the connection between the rules and the GP utility, with this thesis aiming to implement the rules to achieve the optimal GP rhythm. With the achieved results, it is possible to invert the direction of the workflow in future work and formulate a new set of possibly more precise GP rhythm rules based on the obtained reward value space, which could further improve the proposed method ability to amputation adaptation.

All the compared methods tackled the adaptation after leg amputation by tunning the GP rhythm, i.e., changing when the legs are lifted off moving forward and when the legs support the robot's body moving backward. The methods achieved improvement in the locomotion after amputation. Hence, the experiments proved the correct rhythm change itself can truly improve the locomotion.

Chapter 8
Conclusion

In this thesis, the coordination-rules-based method adjusting the *Gait Pattern* (GP) rhythm to adapt the locomotion after leg amputation of the hexapod walking robot was proposed and compared with competitive sensory-based adaptation approach [14] and with introduced *Evolution Algorithm* (EA) given the proposed reward function evaluating the GP. The proposed method originates in bio-inspired *Coordination Rules* (CRs) further extended by *Consecutivity and Contralaterality Adjustment Rules* (CCARs) for the case of leg amputation. The rules determine the relations between the movement of individual legs, implying which legs can or can not move simultaneously. The proposed method dynamically detects a violation of the rules and actively adapts the timing of the legs' movement to avoid the rules violations in the future.

The introduced reward value favors GPs with high and stable velocity, stable orientation, and straightness of the traversed path. The weights for sensory inputs are balanced to make the nature-observed GPs gain positive rewards. The EA takes advantage of the locomotion decoupled parametrization to reduce the locomotion search space significantly. The locomotion parametrization after leg amputation is achieved with only four parameters, automatically eliminating an infinite number of duplicate GPs differing only by phase offset of their periodic movement. The EA utility is the proposed reward function value optimized by changing the GP rhythm.

The comparison of the two online-adapting approaches shows the proposed CR-based method is more consistent and, in two out of three scenarios, outperforms the competitive method proposed by Miguel et al. The proposed method is more consistent in the achieved results in individual runs, in the traversed path, and in the robot's orientation. Moreover, in contrast with the competing method, the proposed method does not depend on continual sensory feedback, which can be unreliable after damage occurs. All three compared methods, the proposed method, the Miguel's method, and the EA, converge to similar areas within the GP space and yield GP rhythms with similar trends, although each optimizes different criteria. Such similarities imply a possibility of future fusion of the methods, hopefully improving their performance.

From the method evaluation, the GP evaluation space emerged using the samples gathered during the experiments. The obtained GP evaluation space was compared with the expected GP evaluation given the bio-inspired CRs. The comparison revealed correlations between the observed and expected GP utility space, supporting the idea that CRs and CCARs are suitable heuristic for GP search.

The obtained GP evaluation space gives the opportunity to form new CRs given the evaluation space observation of dependencies within the GP. Such novel CRs could further improve the performance of the proposed method. More experiments with different morphologies could also provide additional data for the GP evaluation space and result in correlations leading to CRs independent of the particular morphology. Combining the proposed sensory-free method optimizing the GP rhythm, enhanced by the newly formed CRs based on the observed GP evaluation space, with sensory feedback could provide a universal solution for gait adaptation of hexapod walking robots after leg amputation with possible extension to any leg damage adaptation.

References

[1] C. Dario Bellicoso, Marko Bjelonic, Lorenz Wellhausen, Kai Holtmann, Fabian Günther, Marco Tranzatto, Péter Fankhauser, and Marco Hutter. Advances in real-world applications for legged robots. *Journal of Field Robotics*, 35(8):1311–1326, 2018.

[2] Tomáš Rouček, Martin Pecka, Petr Cížek, Tomáš Petříček, Jan Bayer, Vojtěch Šalanský, Teymur Azayev, Daniel Heřt, Matěj Petrlík, Tomáš Báča, Vojtěch Spurný, Vít Krátký, Pavel Petráček, Dominic Baril, Maxime Vaidis, Vladimír Kubelka, François Pomerleau, Jan Faigl, Karel Zimmermann, Martin Saska, Tomáš Svoboda, and Tomáš Krajník. System for multi-robotic exploration of underground environments CTU-CRAS-NORLAB in the DARPA subterranean challenge. *CoRR*, abs/2110.05911, 2021.

[3] Tomáš Rouček, Martin Pecka, Petr Čížek, Tomáš Petříček, Jan Bayer, Vojtěch Šalanský, Daniel Heřt, Matěj Petrlík, Tomáš Báča, Vojěch Spurný, François Pomerleau, Vladimír Kubelka, Jan Faigl, Karel Zimmermann, Martin Saska, Tomáš Svoboda, and Tomáš Krajník. Darpa subterranean challenge: Multi-robotic exploration of underground environments. In *Modelling and Simulation for Autonomous Systems*, pages 274–290, Cham, 2020. Springer International Publishing.

[4] Aravind Seeni, Bernd Schafer, Bernhard Rebele, and Nikolai Tolyarenko. Robot mobility concepts for extraterrestrial surface exploration. In *IEEE Aerospace Conference*, pages 1–14, 2008.

[5] Fred Delcomyn. Perturbation of the Motor System in Freely Walking Cockroaches I. Rear Leg Amputation and the Timing of Motor Activity in Leg Muscles. *Journal of Experimental Biology*, 156(1):483–502, 1991.

[6] Matthias Wittlinger and Harald Wolf. Homing distance in desert ants, cataglyphis fortis, remains unaffected by disturbance of walking behaviour and visual input. *Journal of Physiology-Paris*, 107(1):130–136, 2013. Neuroethology.

[7] Justin Kon and Ferat Sahin. Gait generation for damaged hexapods using a genetic algorithm. In *IEEE 15th International Conference of System of Systems Engineering (SoSE)*, pages 451–456, 2020.

[8] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

[9] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, and Jean-Baptiste Mouret. Reset-free trial-and-error learning for robot damage recovery. *Robotics and Autonomous Systems*, 100:236–250, 2018.

[10] Yaguang Zhu, Tong Guo, Qiong Liu, Qianwei Zhu, Bo Jin, and Xiangmo Zhao. Turning and radius deviation correction for a hexapod walking robot based on an ant-inspired sensory strategy. *Sensors*, 17(12), 2017.

[11] Guanjiao Ren, Weihai Chen, Sakyasingha Dasgupta, Christoph Kolodziejski, Florentin Wörgötter, and Poramate Manoonpong. Multiple chaotic central pattern generators with learning for legged locomotion and malfunction compensation. *Information Sciences*, 294:666–682, 2015. Innovative Applications of Artificial Neural Networks in Engineering.

[12] Potiwat Ngamkajornwiwat, Pitiwut Teerakittikul, and Poramate Manoonpong. Online gait adaptation of a hexapod robot using an improved artificial hormone mechanism. In *From Animals to Animats 15*, pages 212–222, Cham, 2018. Springer International Publishing.

[13] Volker Dürr, Josef Schmitz, and Holk Cruse. Behaviour-based modelling of hexapod locomotion: linking biology and technical application. *Arthropod Structure & Development*, 33(3):237–250, 2004. Arthropod Locomotion Systems: from Biological Materials and Systems to Robotics.

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

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

[16] Shinichi Kimura, Masafumi Yano, and Hiroshi Shimizu. A self-organizing model of walking patterns of insects. *Biological Cybernetics*, 69(3):183–193, 1993.

[17] Christoph P. E. Zollikofer. Stepping Patterns in Ants: I. Influence of Speed and Curvature. *Journal of Experimental Biology*, 192(1):95–106, 1994.

[18] Jan Feber, Rudolf Szadkowski, and Jan Faigl. Gait adaptation after leg amputation of hexapod walking robot without sensory feedback. In *Artificial Neural Networks and Machine Learning – ICANN*, pages 656–667, Cham, 2022. Springer Nature Switzerland.

[19] Joana Coelho, Fernando Ribeiro, Bruno Dias, Gil Lopes, and Paulo Flores. Trends in the control of hexapod robots: A survey. *Robotics*, 10(3), 2021.

[20] Yudi Isvara, Syawaludin Rachmatullah, Kusprasapta Mutijarsa, Dinara Enggar Prabakti, and Wiharsa Pragitatama. Terrain adaptation gait algorithm in a hexapod walking robot. In *International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 1735–1739, 2014.

[21] Genji Tanaka, Tetsuta Takamura, Yuki Shimura, Kazuhiro Motegi, and Yoichi Shiraishi. Development of simulator and analysis of walking for hexapod robots. In *Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pages 1594–1599, 2019.

[22] Addie Irawan and Kenzo Nonami. Force threshold-based omni-directional movement for hexapod robot walking on uneven terrain. In *International Conference on Computational Intelligence, Modelling and Simulation*, pages 127–132, 2012.

[23] Fusheng Zha, Chen Chen, Wei Guo, Penglong Zheng, and Junyi Shi. A free gait controller designed for a heavy load hexapod robot. *Advances in Mechanical Engineering*, 11(3):1687814019838369, 2019.

[24] Adam Rushworth, Salvador Cobos-Guzman, Dragos Axinte, and Mark Raffles. Pre-gait analysis using optimal parameters for a walking machine tool based on a free-leg hexapod structure. *Robotics and Autonomous Systems*, 70:36–51, 2015.

[25] Abhijit Mahapatra, Shibendu Shekhar Roy, and Dilip Kumar Pratihar. Study on feet forces' distributions, energy consumption and dynamic stability measure of hexapod robot during crab walking. *Applied Mathematical Modelling*, 65:717–744, 2019.

[26] Servet Soyguder and Hasan Alli. Kinematic and dynamic analysis of a hexapod walking–running–bounding gaits robot and control actions. *Computers & Electrical Engineering*, 38(2):444–458, 2012.

[27] Hua Deng, Guiyang Xin, Guoliang Zhong, and Michael Mistry. Gait and trajectory rolling planning and control of hexapod robots for disaster rescue applications. *Robotics and Autonomous Systems*, 95:13–24, 2017.

[28] Bo Jin, Cheng Chen, and Wei Li. Power consumption optimization for a hexapod walking robot. *Journal of Intelligent & Robotic Systems*, 71(2):195–209, 2013.

[29] G Heppner, A Roennau, J Oberländer, S Klemm, and R Dillmann. Laurope-six legged walking robot for planetary exploration participating in the spacebot cup. *WS on Advanced Space Technologies for Robotics and Automation*, 2(13):69–76, 2015.

[30] Muhammad Sarmad Khan, Asad Ali Awan, Fahad Islam, Yasar Ayaz, and Osman Hasan. Safe-radius based motion planning of hexapod using rrt-connect. In *IEEE International Conference on Information and Automation*, pages 415–418, 2015.

[31] Patrik Kutílek, Slávka Vítečková, Jan Hejda, Václav Křivánek, Radek Doskočil, and Alexand Štefek. Control of hexapod with static-stable walking using artificial intelligence. In *International Conference on Mechatronics - Mechatronika (ME)*, pages 1–6, 2016.

[32] C. Plasberg, D. Kogel, A. Roennau, and R. Dillmann. Adaptive walking pattern generation with damaged legs for the six-legged robot lauron v. In *International Conference on Advanced Robotics (ICAR)*, pages 422–427, 2021.

[33] Yiqun Liu, Xuanxia Fan, Liang Ding, Jianfeng Wang, Tao Liu, and Haibo Gao. Fault-tolerant tripod gait planning and verification of a hexapod robot. *Applied Sciences*, 10(8), 2020.

[34] Kazi Mostafa, Chong-Shuen Tsai, and Innchyn Her. Alternative gaits for multiped robots with leg failures to retain maneuverability. *International Journal of Advanced Robotic Systems*, 7(4):34, 2010.

[35] Christopher Mailer, Geoff Nitschke, and Leanne Raw. Evolving gaits for damage control in a hexapod robot. In *Genetic and Evolutionary Computation Conference*, page 146–153, New York, NY, USA, 2021. Association for Computing Machinery.

[36] Chenyu Liu, Zhijun Li, Chengyao Zhang, Yufei Yan, and Rui Zhang. Gait planning and control for a hexapod robot on uneven terrain based on markov decision process. In *IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 583–586, 2019.

[37] Shresth Verma, Haritha S. Nair, Gaurav Agarwal, Joydip Dhar, and Anupam Shukla. Deep reinforcement learning for single-shot diagnosis and adaptation in damaged robots. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, page 82–89, New York, NY, USA, 2020. Association for Computing Machinery.

[38] Ashwin Sanjay Lele, Yan Fang, Justin Ting, and Arijit Raychowdhury. Learning to walk: Spike based reinforcement learning for hexapod robot central pattern generation. In *IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 208–212, 2020.

[39] Jun Hong, Kaiqiang Tang, and Chunlin Chen. Obstacle avoidance of hexapod robots using fuzzy q-learning. In *IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–6, 2017.

[40] Wenjuan Ouyang, Haozhen Chi, Jiangnan Pang, Wenyu Liang, and Qinyuan Ren. Adaptive locomotion control of a hexapod robot via bio-inspired learning. *Frontiers in Neurorobotics*, 15, 2021.

[41] Weihai Chen, Guanjiao Ren, Jianbin Zhang, and Jianhua Wang. Smooth transition between different gaits of a hexapod robot via a central pattern generators algorithm. *Journal of Intelligent & Robotic Systems*, 67(3):255–270, 2012.

[42] Rudolf Szadkowski and Jan Faigl. Neurodynamic sensory-motor phase binding for multi-legged walking robots. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.

[43] Guanjiao Ren, Weihai Chen, Christoph Kolodziejski, Florentin Wörgötter, Sakyasingha Dasgupta, and Poramate Manoonpong. Multiple chaotic central pattern generators for locomotion generation and leg damage compensation in a hexapod robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2756–2761, 2012.

[44] Haitao Yu, Haibo Gao, Liang Ding, Mantian Li, Zongquan Deng, and Guangjun Liu. Gait generation with smooth transition using cpg-based locomotion control for hexapod walking robot. *IEEE Transactions on Industrial Electronics*, 63(9):5488–5500, 2016.

[45] Lizhou Xu, Weihua Liu, Zhiying Wang, and Wenfu Xu. Gait planning method of a hexapod robot based on the central pattern generators: Simulation and experiment. In *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 698–703, 2013.

[46] Malte Schilling, Holk Cruse, and Paolo Arena. Hexapod walking: an expansion to walknet dealing with leg amputations and force oscillations. *Biological Cybernetics*, 96(3):323–340, 2007.

[47] Axel Schneider, Jan Paskarbeit, Mattias Schaeffersmann, and Josef Schmitz. Hector, a new hexapod robot platform with increased mobility - control approach, design and communication. In *Advances in Autonomous Mini Robots*, pages 249–264, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[48] Mathias Thor and Poramate Manoonpong. A fast online frequency adaptation mechanism for cpg-based robot motion control. *IEEE Robotics and Automation Letters*, 4(4):3324–3331, 2019.

[49] Y. Fukuoka, H. Kimura, Y. Hada, and K. Takase. Adaptive dynamic walking of a quadruped robot 'tekken' on irregular terrain using a neural system model. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 2037–2042, 2003.

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

[51] Jan Feber, Rudolf Szadkowski, and Jan Faigl. Gait genesis through emergent ordering of rbf neurons on central pattern generator for hexapod walking robot. In *Conference Information Technologies - Applications and Theory (ITAT)*, pages 114–122, 2021.

[52] Jon Timmis, Mark Neal, and James Thorniley. An adaptive neuro-endocrine system for robotic systems. In *IEEE Workshop on Robotic Intelligence in Informationally Structured Space*, pages 129–136, 2009.

[53] Dai Owaki, Masashi Goda, Sakiko Miyazawa, and Akio Ishiguro. A minimal model describing hexapedal interlimb coordination: The tegotae-based approach. *Frontiers in Neurorobotics*, 11, 2017.

[54] A. Pikovsky, M. Rosenblum, and J. Kurths. *Synchronization: A Universal Concept in Nonlinear Sciences*, chapter 7 Synchronization of periodic oscillators by periodic external action. Cambridge Nonlinear Science Series. Cambridge University Press, 2001.

[55] Rudolf Szadkowski, Miloš Prágr, and Jan Faigl. Self-learning event mistiming detector based on central pattern generator. *Frontiers in Neurorobotics*, 15, 2021.

[56] E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1321–1326. IEEE, 2013.

[57] Minh Thao Nguyenová, Petr Čížek, and Jan Faigl. Modeling proprioceptive sensing for locomotion control of hexapod crawling robot in robotic simulator. In *2018 Modelling and Simulation for Autonomous Systems (MESAS)*, pages 215–225, 2019.

Appendix A
Getting the Swing Danger Vector

The input are the vectors $\boldsymbol{s}^{\text{dmg}} \in \{0,1\}^6$, with $s_j^{\text{dmg}} = 1$ means leg j is amputated and $\boldsymbol{s}^{\text{sw}}(t) = (s_0^{\text{sw}}(t), \ldots, s_5^{\text{sw}}(t))$, where $s_i^{\text{sw}} = 1$ means leg i is performing swing and $s_i^{\text{sw}} = 0$ means leg i is performing stance.

The swing danger vector $\boldsymbol{d}(t)$ indicates whether each leg can perform swing or not, according to CRs and CCARs. The first conflict is the possibility of swinging the leg in the same time its contralateral leg performs swing. Finding the leg's contralateral leg is addressed in the following Section A.1. The second possible conflict is with the leg's consecutive leg, which is introduced in the second Section A.2

The implementation is done in `python3` with `NumPy` library. Therefore, the implementation of getting the vector $\boldsymbol{d}(t)$ uses available vector and matrix operations.

A.1 Get Label of Contralateral Leg

Two support matrices are defined: (i) binary matrix $\mathbf{M}^{\text{r}}$ of dimensions 6×6, where row r represents leg label $0, \ldots, 5$ and the column c represents the legs role in the morphology: 0: hind left, 1: middle left, 2: front left, 3: hind right, 4: middle right, 5: front right. Hence, the elements $m_{r,c}^{\text{r}} = 1$ for $(r,c) \in \{(0,2),(1,5),(2,0),(3,3),(4,1),(5,4)\}$ and all others are equal to zero:

$$\mathbf{M}^{\text{r}} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \tag{46}$$

(ii) The second support matrix is the matrix of consecutiveness $\mathbf{M}^{\text{c}}$, where $m_{r,c}^{\text{c}} = 1$ and $m_{c,r}^{\text{c}} = -1$ if leg c is consecutive to leg r and $m_{r,c}^{\text{c}} = 0$ otherwise:

$$\mathbf{M}^{\text{c}} = \begin{pmatrix} 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \end{pmatrix}. \tag{47}$$

The vector $\mathbf{m}^{\text{sel}}$ is given as the j-th row of the matrix $\mathbf{M}^{\text{c}}$, where j is the label of amputated leg, i.e., $s_j^{\text{dmg}} = 1$, so $\mathbf{m}^{\text{sel}} = \mathbf{m}_j^{\text{c}}$.

The index $i_{\text{r}} = \text{argmax}_i\, m_{j,i}^{\text{r}}$, where $\mathbf{m}_j^{\text{r}} = (m_{j,0}^{\text{r}}, \ldots, m_{j,5}^{\text{r}})$ is the j-th row of the matrix $\mathbf{M}^{\text{r}}$, is the role of the damaged leg. The index $i_{\text{c}} = i_{\text{r}} + \sum_{i=0}^{5} m_i^{\text{sel}}$ is the role of the leg k, for which the amputated leg j is the consecutive leg.

The matrix of roles $\mathbf{M}^{\text{r}}$ is then modified to correspond to the CCARs: If the leg j is amputated and is consecutive leg of the leg k, the leg k takes the role of the amputated leg j, i.e., the i_{r}-th column

$\mathbf{m}^{\mathrm{r}}_{\_,i_{\mathrm{r}}}$ is copied to the i_{c}-th column $\mathbf{m}^{\mathrm{r}}_{\_,i_{\mathrm{c}}}$ in the matrix $\mathbf{M}^{\mathrm{r}} = (\mathbf{m}^{\mathrm{r}}_{\_,0}, \ldots, \mathbf{m}^{\mathrm{r}}_{\_,5})$, i.e., $\mathbf{m}^{\mathrm{r}}_{\_,i_{\mathrm{c}}} := \mathbf{m}^{\mathrm{r}}_{\_,i_{\mathrm{r}}}$. Because the leg j is amputated, it no longer has any of the roles, thus $m^{\mathrm{r}}_{j,i_{\mathrm{r}}} = 0$.

i-th element s^{cont}_i in vector $\mathbf{s}^{\mathrm{cont}} = (s^{\mathrm{cont}}_0, \ldots, s^{\mathrm{cont}}_5)$ represents the label of the i-th leg's contralateral leg, according to the CCARs (i.e., according to newly adjusted matrix $\mathbf{M}^{\mathrm{r}}$), as follows:

$$s^{\mathrm{cont}}_i = \begin{cases} \operatorname{argmax}_k m^{\mathrm{r}}_{i,k} + 3 & , \text{if } \operatorname{argmax}_k m^{\mathrm{r}}_{i,k} < 3 \\ \operatorname{argmax}_k m^{\mathrm{r}}_{i,k} - 3 & , \text{if } \operatorname{argmax}_k m^{\mathrm{r}}_{i,k} \geq 3 \end{cases} \tag{48}$$

Vector $\mathbf{s}^{\mathrm{r}} = (s^{\mathrm{r}}_0, \ldots, s^{\mathrm{r}}_5)$ consists of elements s^{r}_i, where i stands for role in the morphology (i.e., recall 0: hind left, 1: middle left, 2: front left, 3: hind right, 4: middle right, 5: front right), and the element s^{r}_i represents the label of the leg with the given role:

$$s^{\mathrm{r}}_i = \begin{cases} \operatorname{argmax}_k m^{\mathrm{r}}_{k,i} & \text{if } s^{\mathrm{dmg}}_i = 0 \\ -1 & \text{otherwise} \end{cases}. \tag{49}$$

Finally, vector $\mathbf{d}^{\mathrm{cont}} = (d^{\mathrm{cont}}, 0, \ldots, d^{\mathrm{cont}}_5)$ is introduced, where $d^{\mathrm{cont}}_i = 1$ means i-th leg's contralateral leg is performing swing, thus the leg i should not perform swing:

$$d^{\mathrm{cont}}_i = \begin{cases} s^{\mathrm{sw}}_l & \text{if } s^{\mathrm{dmg}}_i = 0 \\ 0 & \text{otherwise} \end{cases} \tag{50}$$

$$l = s^{\mathrm{r}}_{s^{\mathrm{cont}}_i} \tag{51}$$

A.2 Conflict with Consecutive Leg

Here another support binary matrix $\mathbf{M}^{\mathrm{s}}$ of dimensions 6×6 is introduced, with $m^{\mathrm{s}}_{r,c} = 1$ if the leg c is on the same side of the robot's body as leg r, otherwise $m^{\mathrm{s}}_{r,c} = 0$ and also if $c = r$, then $m^{\mathrm{s}}_{r,c} = 0$:

$$\mathbf{M}^{\mathrm{s}} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}. \tag{52}$$

For each leg i a value a_i, representing how many legs on the same side are in swing (amputated leg is considered as performing swing), is computed as follows:

$$a_i = \sum_{k=0}^{5} m^{\mathrm{s}}_{i,k} s^{\mathrm{sw}}_k. \tag{53}$$

In the case of contralaterality, the previous section introduced a mechanism changing the matrix according to CCARs. However, here, I skipped such process and substituted it with the summation for the change in consecutivity. The expressions d^{cons}_i, forming the vector $\mathbf{d}^{\mathrm{cons}}_i$ carry an information about whether leg i can perform swing in relation to its consecutive leg ($d^{\mathrm{cons}}_i = 0$) or not ($d^{\mathrm{cons}}_i = 1$):

$$d^{\mathrm{cons}}_i = \begin{cases} 1 & \text{if } (i \notin \{2, 3\} \wedge s^{\mathrm{dmg}}_l = 0 \wedge s^{\mathrm{sw}}_l = 1) \vee a_i = 0 \\ 0 & \text{otherwise} \end{cases} \tag{54}$$

$$l = s^{\mathrm{r}}_{s^{\mathrm{cons}}_i} \tag{55}$$

Finally, the vector $\mathbf{d} = (d_1, \ldots, d_5)$, is given as:

$$d_i = [\![d^{\mathrm{cont}}_i + d^{\mathrm{cons}}_i > 0]\!]. \tag{56}$$

Appendix B
CRs and CCARs Implications Generating Expected Reward Value Space

B.1 Amputated Leg: Front Left 0

Relation of: front right 1 & hind left 2: According to CRs and CCARs, leg 2 is contralateral to leg 3, which has its swing start ϕ_3 set to zero. Therefore, leg 2 should have swing start ϕ_2 somewhere around $\frac{P}{2} = \pi$ in order to move in antiphase with leg 3. Similarly, the consecutive legs move after each other: the hind leg, middle leg, and the front leg at last, followed by the hind leg again. Hence, the phase offset between leg 1 and 3 should be approximately $\Delta\phi$ equal to $\frac{P}{3} = \frac{2\pi}{3}$. Combining these two corollaries, the leg swing start of leg 2 $\phi_2 \approx \pi$ and swing start of leg 1 $\phi_1 \approx \frac{2\pi}{3}$, therefore the phase offset between ϕ_2 and ϕ_1 should be around $\frac{\pi}{3}$.

Relation of: front right 1 & middle left 4: Based on the CRs and CCARs, legs 1 and 4 are contralateral; thus, the phase offset of their swing starts should vary around value $\frac{P}{2} = \pi$. Leg 2 is contralateral to leg 3, which has ϕ_3 equal to zero. Therefore, the leg 2 should start its swing around π. Because legs 4 and 2 are only two legs on the left side of the robot's body, the legs should alter; therefore, the swing start ϕ_4 of leg 4 should be approximately equal to the ϕ_3 (i.e., $0 = \phi_3 \approx \phi_4$) because leg 3 should also alter the leg 2. Lastly, leg 1 should approximately start its swing $\frac{P}{3} = \frac{2\pi}{3}$ after the leg 0, with the same reasoning as in the pair of swing starts ϕ_1 and ϕ_2.

Relation of: front right 1 & middle right 5: Because the legs 3, 5, and 1 are consecutive in that order, the phase offsets between the consecutive pairs should correspond to the $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$, according to the CRs.

Relation of: hind left 2 & middle left 4: Legs 2 and 4 are the only legs remaining on the left side of the robot's body. Therefore, they should alter in the movement, with their phase offset being approximately $\frac{P}{2} = \pi$. The leg 2 is also contralateral to the leg 3. Hence, the phase offset of legs 3 and 2 should also approximately correspond to $\frac{P}{2} = \pi$, and legs 3 and 4 should perform the swing in similar phase (i.e., $\phi_4 \approx \phi_3 = 0$).

Relation of: hind left 2 & middle right 5: Based on the CRs, the consecutive legs 5 and 3 should have phase offset of their swing start somewhere around $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$. Leg 2 is contralateral to leg 3; therefore, again, the phase offset between them should be approximately $\frac{P}{2} = \pi$. Based on those two corollaries, the phase offset between legs 2 and 5 should move in proximity to value $\frac{\pi}{3}$.

Relation of: middle left 4 & middle right 5: Given the reasoning above for a pair of swing start values ϕ_1 and ϕ_4, leg 4 should start its swing around 0, i.e., $\phi_3 \approx \phi_4$. The leg 5 is leg's 3 consecutive leg; hence $\phi_5 \approx \Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$. The leg 3 with $\phi_3 = 0$ moves in antiphase of the leg 2. Thus, leg's 5 movement following the movement of leg 3 should perform its movement close to the movement of leg 4, i.e., $\phi_4 \approx \phi_5$.

B.2 Amputated Leg: Middle Left 4

Relation of: front left 0 & front right 1: The legs 0 and 1 are contralateral; therefore, I expect the phase offset between them to be approximately $\frac{P}{2} = \pi$. Leg 0 is also expected to have approximately the same swing start ϕ_0 as leg $\phi_3 = 0$, with the same reasoning as for leg 4 in the scenario with amputated leg 0 (i.e., legs 0 and 2 are the only two legs on the left side; hence they should alter each

other, with leg 2 being contralateral to leg 0). Also, the reasoning behind the expectancy of leg's 1 swing start being around $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$ is the same as in the case of the scenario with amputated leg 0.

Relation of: front left 0 & hind left 2: Using the logic from the previous paragraph, the legs 2 and 0 should alter each other; thus, the mutual phase offset is expected to be around value $\frac{P}{2} = \pi$. The previous paragraph also explains the corollary of value $\phi_0 \approx \frac{P}{2} = \pi$. Finally, legs 3 and 2 are contralateral; therefore, I expect the value $phi_2 \approx \pi$, given the fixed value $\phi_3 = 0$.

Relation of: front left 0 & middle right 5: Once again, the value phi_0 expectation is somewhere close to value $\phi_3 = 0$. Leg 5, being the consecutive leg of leg 3, results in the corollary the value $phi_5 \approx \Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$. Thus, the expected phase offset of legs' 0 and 5 swing starts ϕ_0 and ϕ_5 is about $\frac{\pi}{3}$.

Relation of: front right 1 & hind left 2: On the leg 1 the corollary of its swing start ϕ_1 having phase offset with leg 0 roughly $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$ applies again. Leg 2 is contralateral to leg 3, implying the phase offset $\frac{P}{2} = \pi$. The phase offset of legs' 1 and 2 swing starts ϕ_1 and ϕ_2 use the weak corollary of being "*more similar than different*", written as $\phi_1 \approx \phi_2$.

Relation of: front right 1 & middle right 5: Leg 1 is consecutive to leg 5; thus, I expect the phase offset between their respective swing starts ϕ_1 and phi_5 to be approximately $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$. Also, leg 5 is consecutive to leg 0, and leg 0 starts its movement after leg 1. In other words, all the legs 3, 5, and 1 are consecutive in that order; therefore, I expect the distances between consecutive leg's swing starts to be approximately around the value. Hence, with leg's 3 swing start ϕ_3 fixed to value 0, the phase offset from 0 should be somewhere close to value $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$.

Relation of: hind left 2 & middle right 5: The legs' 2 and 5 swing start relations to the fixed swing start $\phi_3 = 0$ of leg 3 were already described. Given the relation between 2 and 3, and 3 and 5, the expected phase offset between swing starts of legs 2 and 5 is about $\frac{\pi}{3}$.

B.3 Amputated Leg: Hind Left 2

Relation of: front left 0 & front right 1: The legs 0 and 1 are contralateral; thus, the phase offset of their swing starts is expected to be near the value of $\frac{P}{2} = \pi$. For the pair of legs 4 and 0, the logic of altering their actions as they are the only two remaining legs on the left side applies again. Leg 4 being contralateral to leg 3 (according to the CRs and CCARs) with fixed swing start $\phi_3 = 0$ suggests the leg's 0 swing start ϕ_0 should be similar to ϕ_3, i.e., $\phi_0 \approx \phi_3$. Legs 3, 5, and 1 are consecutive in that order; hence, the movement of leg 3 is assumed to start after the movement of leg 1. Thus, the leg's 1 swing start should have the phase offset around $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$.

Relation of: front left 0 & middle left 4: The legs 0 and 4 are consecutive, and they are the only remaining legs on the left side. Therefore, their expected phase offset is $\frac{P}{2} = \pi$ in order to alter each other. Leg 4 is the contralateral leg to leg 3, due to the CRs and CCARs, implying the expected swing start ϕ_4 value being approximately $\frac{P}{2} = \pi$ because the leg's 3 swing start is fixed to value $\phi_3 = 0$. Because the legs 4 and 3 should move in antiphase and legs 4 and 0 should move in antiphase, the legs 0 and 3 should perform their movement in similar phase, i.e., $\phi_0 \approx \phi_3$.

Relation of: front left 0 & middle right 5: Leg 5 is the consecutive leg to the leg 3, which has fixed swing start $\phi_3 = 0$; hence, the leg's 5 swing start ϕ_5 is expected to have a phase offset with swing start ϕ_3 around the value $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$. The expected swing stat value of leg 0 ($\phi_0 \approx \phi_3$) was already explained in the previous paragraphs. Combining the expected values of legs 0 and 5, their expected phase offset is approximately $\frac{\pi}{3}$.

Relation of: front right 1 & middle left 4: The expected phase offset between legs 1 and 3 ($\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$) is already explained in the previous paragraphs. Leg 4 is, once again, considered contralateral to leg 3 due to the CRs and CCARs, implying the leg's 4 swing start be near the value of $\frac{P}{2} = \pi$, as leg 3 has fixed swing start value $\phi_3 = 0$. Hence, the expected phase offset of legs' 1 and 4

swing starts is around $\frac{\pi}{3}$.

Relation of: front right 1 & middle right 5: Legs 3, 5, and 1 are consecutive legs, respectively. Thus, the phase offsets between swing starts of legs 3 and 5, 5 and 1, and also 1 and 3, is approximately $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$, where leg 3 has fixed swing start $\phi_3 = 0$.

Relation of: middle left 4 & middle right 5: Finally, leg 5 is the consecutive leg of leg 3, and based on the CRs and CCARs, leg 4 is contralateral to leg 3, which has a fixed swing start $\phi_3 = 0$. Therefore, the phase offset between swing starts of pairs of legs 3 and 5, and 3 and 4 are $\Delta\phi = \frac{P}{3} = \frac{2\pi}{3}$, and $\frac{P}{2} = \pi$, respectively. Such relation implies the phase offset of legs' 4 and 5 swing starts near the value of $\frac{\pi}{3}$.