THE UNIVERSITY
of EDINBURGH

Self-Organisation of Internal Models in Autonomous Robots

Simón C. Smith Bize

Doctor of Philosophy
Institute of Perception, Action and Behaviour
School of Informatics
University of Edinburgh
2015

Abstract

Internal Models (IMs) play a significant role in autonomous robotics. They are mechanisms able to represent the input-output characteristics of the sensorimotor loop. In developmental robotics, open-ended learning of skills and knowledge serves the purpose of reaction to unexpected inputs, to explore the environment and to acquire new behaviours. The development of the robot includes self-exploration of the state-action space and learning of the environmental dynamics.

In this dissertation, we explore the properties and benefits of the self-organisation of robot behaviour based on the homeokinetic learning paradigm. A homeokinetic robot explores the environment in a coherent way without prior knowledge of its configuration or the environment itself. First, we propose a novel approach to self-organisation of behaviour by artificial curiosity in the sensorimotor loop. Second, we study how different forward models settings alter the behaviour of both exploratory and goal-oriented robots. Diverse complexity, size and learning rules are compared to assess the importance in the robot's exploratory behaviour. We define the self-organised behaviour performance in terms of simultaneous environment coverage and best prediction of future sensori inputs. Among the findings, we have encountered that models with a fast response and a minimisation of the prediction error by local gradients achieve the best performance.

Third, we study how self-organisation of behaviour can be exploited to learn IMs for goal-oriented tasks. An IM acquires coherent self-organised behaviours that are then used to achieve high-level goals by reinforcement learning (RL). Our results demonstrate that learning of an inverse model in this context yields faster reward maximisation and a higher final reward. We show that an initial exploration of the environment in a goal-less yet coherent way improves learning.

In the same context, we analyse the self-organisation of central pattern generators (CPG) by reward maximisation. Our results show that CPGs can learn favourable reward behaviour on high-dimensional robots using the self-organised interaction between degrees of freedom. Finally, we examine an on-line dual control architecture where we combine an Actor-Critic RL and the homeokinetic controller. With this configuration, the probing signal is generated by the exertion of the embodied robot experience with the environment. This set-up solves the problem of designing task-dependant probing signals by the emergence of intrinsically motivated comprehensible behaviour. Faster improvement of the reward signal compared to classic RL is achievable with this configuration.

Acknowledgements

I would like to thank Michael Herrmann for his support trough out all the Ph.D. program. For giving me invaluable guidance and advice on the broad topic of artificial intelligence and how to become a scientist. I am grateful for the always stimulating conversations and his excellent disposition to ever find time to meet for short or long talks. I appreciate the conversations about everyday life, and the fact that I could benefit from his ability to constantly have the right reference with the right name on mind.

I appreciate the helpful comments and guidance from Daniel Polani at each year review and conversations that we had at different conferences. Also, I would like to thank Subramanian Ramamoorthy for his review on my work.

Thanks to Ralf Der for his invaluable comments on my projects and thesis, his excellent disposition to discuss subjects related to self-organisation at conferences and at the time that I spent at Leipzig.

I would like to thank Georg Martius for the advice and discussions on multiple topics, and for making it possible for me to spend some time at Leipzig working on self-organisation and the development of the simulator. The research on artificial curiosity presented in this dissertation was developed with the supervision of Georg at the Max Planck Institute for Mathematics in the Sciences in Leipzig, Germany.

Many thanks to my family, especially my parents for their unconditional support. To Rosie, for all her love and advice in the not always straight forward English language, who has influenced this thesis in more ways that she can imagine. To all my friends that have become more of a family during these years abroad.

This work has been funded by the National Commission for Scientific and Technological Research CONICYT, grant BecasChile, government of Chile.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Simón C. Smith Bize*)

Para Mila y Francisco,
y para Alicia y Gustavo.

Table of Contents

Chapter 1

Introduction

Autonomous robotics is an area of research that gathers a great amount of attention in artificial intelligence. On one hand, it promises to develop robots that will be able to perform in situations not suitable for humans, for example, in dangerous environments or repetitive tasks. On the other hand, it is a complex problem that requires the combination of techniques from several fields. These factors make it a popular and interesting challenge. Autonomous robots are required to adapt to an ever-changing environment and to learn from their interaction with a complex world (McFarland and Bösser, 1993). In the first part of this dissertation, we present a novel approach to self-organisation of behaviour based on artificial curiosity. The second part of this dissertation assess the dependency of internal representations in embodied emergent behaviour. Finally, in the third part, we propose a set of novel combinations of self-organised behaviour and goal-oriented set-ups in order to improve learning in autonomous robots.

1.1 Embodied Systems

Artificial Intelligence (AI) was originally thought to create hardware or software that emulates human-like general intelligence. Initially, in the decade of the 1950s, AI paradigm understood intelligence as symbolic systems where problems could be solved by crunching data (Nilsson, 2007). Since then, AI has expanded into various branches bringing some remarkable results in applied computer science and engineering. For example, the victory of IBM's *Deep Blue* chess-playing computer against Garry Kasparov in 1997, the current world champion of chess. Under well-defined and tractable environments, knowledge-base approaches have been successful. However, in more

broad and naturally observed intelligence, e.g., in humans and animals, the process cannot be described only by a local computational process. Such complexity requires considering the body-environment interactions (Pfeifer et al., 2001). Behaviour-based approach (Brooks et al., 1986; Brooks, 1991) started a new trend in AI. This approach is based on a tight coupling between sensations and actions without the need for complex internal representation. Taking ideas from cognitive sciences, embodied cognition or *embodiment*, was characterised by deep dependency upon features of the physical body of an agent (Wilson and Foglia, 2011). Embodiment proposes that intelligence requires a body (Varela et al., 1992; Chiel and Beer, 1997; Pfeifer et al., 2007). In this new paradigm, intelligent behaviour emerges from the interplay between the brain, the body and the world. The brain is thought as a controller for embodied activity and not anymore as a high-level reasoning agent (Clark, 1997). A higher level of intelligence is brought about by a lower level of understanding. Thus, action and cognitions are not expected to be fully built into the agent, rather it has to emerge by embodied experience and be part of a self-development (Der et al., 1999; Lungarella et al., 2003; Oudeyer et al., 2007). Following this idea, epigenetic robotics was conceived as a way of modelling the development of increasingly complex cognitive processes in natural and artificial systems, and to understand how such processes emerge through physical and social interaction (Lungarella, 2007). In nature, complex systems, e.g., a flock of birds, reaction-diffusion systems or the human brain, organise by following local rules (Camazine, 2003; McCulloch, 1965). In robotics, the idea of self-organisation is used to bring about comprehensible embodied behaviour (Prokopenko, 2013). In this work, we use self-organisation of robotics behaviour to drive development in learning autonomous robots.

1.2 Development in Autonomous Robotics

In epigenetic robotics, development should occur within the lifespan of the robot. Hence, on-line learning is well suited to drive such progress. The most used on-line learning goal-oriented method for autonomous behaviour is *reinforcement learning* (Sutton and Barto, 1998). Reinforcement learning (RL) is learning by the interaction with the environment. Every action of the agent results in a new state and an associated reward signal. An RL-agent chooses its action based on both exploitation of past experiences and exploration towards new situations. The agent seeks to maximise the accumulated reward over time (Wörgötter and Porr, 2008). The method is based

on classical conditioning and the finding of the effect of dopamine as an activator of *pleasure centres* in the brain (Pavlov, 1927; Wise, 1980). Most successful RL set-ups are based on temporal difference learning or TD learning. In this learning method, a bootstrap effect is achieved by prediction of future rewards. Some remarkable successes in RL are present in the domain of board games, e.g., checkers (Samuel, 1959) and backgammon (Tesauro, 1994). In autonomous robotics, some of the most notorious examples include gait control for fast quadrupedal locomotion (Kohl and Stone, 2004), inverted flight of a helicopter (Bagnell and Schneider, 2001), and humanoid control (Peters et al., 2003). An important observation is that these examples perform with some degree of discretisation at the action-state level. In real-world robotics applications, RL requires some degree of expert knowledge about the system to overcome the curse of dimensionality. One of the extensions into continuous time and space makes use of functions approximators (Doya, 2000). Still, RL cannot provide a full solution to developmental robotics by itself. Some of the drawbacks, especially concerning epigenetic robotics, include the inability to deal with non-stationary environments, the exploration-exploitation dilemma, and temporal credit assignment (Wörgötter and Porr, 2008).

Already at the dawn of modern AI, the suggestion was to create a machine that started from a basic form and then evolved into a more complex one: “Instead of trying to produce a program to simulate the adult mind, why not rather try to produce one which simulates the child’s” (Turing, 1950). This type of development is taken into account by epigenetic robotics. Epigenetic robotics is associated with intrinsically motivated behaviour towards self-actualisation. Developmental psychology proposes that humans act intrinsically motivated to experience a particular feeling favouring competence and self-determination (Deci and Ryan, 1985). In mammals, complex sensorimotor coordination appears in an early postnatal phase. Such coordination cannot be fully explained by the genetic code and is attributed to a learning process (Gottlieb, 2007). Especially interesting for developmental robotics is the seemingly goal-less playful behaviour in infant mammals. As an evolutionarily favourable skill, the animal enhances its motor skills and produces a self-model that are vital throughout its life (Fagen, 1981; Cameron et al., 2008). Still, the adaptive significance of play, especially in social areas, remains as an open question in ethology (Davies and Kemble, 1983; Sharpe, 2005). Nevertheless, in a robotics set-up, playful behaviours is relevant for sensory-motor coupling acquisition. Intrinsically motivated exploratory actions are able to maintain a working regimen avoiding saturation of the behavioural dynam-

ics. Such behaviour does not respond to perturbations or deficit of any non-nervous-system tissue (White, 1959). Intrinsically motivated behaviour is closely related to self-organisation. An agent is able to bring about behaviour following local and intrinsically defined rules.

Self-organisation is a paradigm that has been used to describe emergent structures or functions in complex systems (Haken, 1983). These structures emerge from local interactions, without any high-level orchestrator. The drive towards organisation comes from the system itself, usually by a few set of rules. There is not a unique formal definition for self-organisation. However, one interpretation can be found in (Haken, 2006): "a system is self-organising if it acquires a spatial, temporal or functional structure without specific interference from the outside". Self-organisation has been used in learning systems, e.g., self-organising maps (Kohonen, 1998) where the structure emerges without external direction. In robotics, e.g., swarm robotics optimisation (Brambilla et al., 2013; Beni, 2005), has taken inspiration from pattern formation in flocking birds (Reynolds, 1987). It has been also used as a behavioural driver with guidance for robot development (Martius et al., 2007). The *homeostasis* principle was first[1] introduced (Cannon, 1932) as a way to understanding self-organisation in a biological complex system, and later expanded into artificial systems (Ashby, 1960). Homeostasis is the property of a system to regulate internal variables in order to maintain internal conditions stable. Typical examples of homeostasis include regulation of temperature, blood pressure and balance between acidity and alkalinity in the human body. The *homeostat* accounts as one of the first devices capable of self-adaptation (Ashby, 1960). The machine was able to maintain homeostasis by changing internal parameters, stabilising the effects of disturbances introduced into the system. However, homeostasis is not able to account for the generation of coherent behaviours (White, 1959). Some approaches to self-organisation in robotics have been proposed as behaviour driver. For example, in *empowerment* (Klyubin et al., 2005), an agent decides his actions based on the maximisation of the perceived amount of influence or control that it has over the world. Another example is the intrinsically motivated behaviour to explore for unknown situations, or *artificial curiosity* (Oudeyer et al., 2007).

In (Der, 2001; Der and Liebscher, 2002), *homeokinesis* was proposed as a principle to maintain the dynamic range of an autonomous robot at the edge of chaos. The

[1]Claude Bernard proposed the principle *milieu intérieur* in his work between 1854 and 1878. The principle suggests that the stability of the internal environment is the condition for free and independent life (Bernard, 1974).

edge of chaos hypothesis state that biological systems operate near a phase transition between chaotic and ordered behaviour for survival (Mitchell et al., 1993). A homeokinetic robot has a playful behaviour that is suitable for goal-less exploration of the action-state space. Homeokinesis does not require further knowledge of the system and is able to find the working regime by self-adapting its internal parameters. Thus, we use it as a goal-less exploratory controller in autonomous robotics.

Internal models have been widely used in control theory and robotics (Nguyen-Tuong and Peters, 2011). Most of the intrinsically motivated approaches on robotics use internal representation as a way to drive development. Internal models are mechanisms that can mimic the input/output characteristics in a motor apparatus/command (Kawato, 1999). They have been theorised as being part of the human brain, seemingly located in the cerebellum (Ito, 1970; Thach, 1998; Doya, 1999). Internal models can be divided into *forward* and *inverse* models. A forward model can be used for state estimation or prediction. For example, when applying a self-generated force to an object that is hold on the hand, an efferent copy of the force can be used to calculate the required grip so the bottle does not slip. An inverse model produces a control signal based on the actual situation and a desired objective. For instance, the force to hit the bottle is calculated on the actual position of the hand and the distance to the bottle. Some examples of their uses are: objects dynamics acquisition (Haruno et al., 2001), behavioural primitives attainment (Martius et al., 2008), self-modelling of the own configuration (Bongard et al., 2006), and model-based reinforcement learning (Kuvayev and Sutton, 1996). In intrinsically motivated robotics behaviour, the internal models drive the action. For example, artificial curiosity (Herrmann et al., 2000; Schmidhuber, 2006), the agent is encouraged to take actions that maximise the learning of a forward model. In a homeostatic robot (Der and Martius, 2012), actions are chosen as to minimise the prediction of the forward model. In homeokinesis, the system tries to reach predictable situations while maintaining sensitivity. Thus, the range of possible behaviours is directly related to the model prediction capacity. This is not a trivial concept where a bigger memory guarantees a better exploration. Different type of internal models generate different error characteristics which have an impact on the behavioural adaptation. Models can differ in the degree of flexibility and expressivity related to structural complexity, the capacity to acquire non-linearities and learning speed. One part of this thesis studies how the complexity of the internal models affects the exploratory behaviour in a homeokinetic set-up.

1.3 Research Questions & Overview

The discussion in the previous sections leads to three main questions that are addressed in this thesis:

- How can self-organised behaviour emerge by artificial curiosity in the sensorimotor loop?
- How does the complexity of the internal models affects self-organised behaviour in autonomous robots?
- How can internal models be self-organised to improve learning in goal-oriented tasks?

Autonomous robots operate under an information closed-loop. This close loop is termed the *sensorimotor* loop and its dynamics are studied in Chapter 2. Using dynamical systems theory (Strogatz, 2001), we present the phase change, stability and bifurcations of the sensorimotor loop. Also, we present in more detail the concepts of self-organisation and internal models as to understand the implications that they have in autonomous robotics.

In Chapter 3, we present how self-organisation brings about coherent behaviour in an autonomous robot in the sensorimotor loop. Since we use homeokinesis as the main driver of exploratory behaviour, we study the dynamics of the homeokinetic learning rules by dynamical systems theory thoroughly. The second part of the chapter includes our approach to self-organisation of behaviour. Self-organisation can emerge from different rules. For example, in homeokinesis, those rules are sensitivity and predictability. In artificial curiosity, the intrinsic rule is the maximisation of learning gain in a forward model. Artificial curiosity has been already explored in robotics settings (Hermann et al., 2000; Schmidhuber, 2006). In this thesis, we promote it as a driver of emergent behaviour in the sensorimotor loop. Artificial curiosity yields a sound theoretical approach to exploration. A robot that focuses only on unknown situations would end up visiting the whole state space until it has acquired all the possible scenarios. In the environments that we want autonomous robots to perform, that is not possible due to their intractable sizes. If the robot avoids scenarios that cannot be attained by the model, e.g., chaotic or random states, the search space is purposefully reduced. Artificial curiosity imposes such restriction by focusing only on situations that produce an improvement of the model. Thus, coherent behaviour emerges based on embodiment

and the capacity of the model. We study artificial curiosity in the sensorimotor loop as a promising approach to self-organisation of behaviour.

In intrinsically motivated systems, behavioural dynamics depends on its internal models in different degrees. In general, the discrepancy between internal and external data provides the essential information for learning and behaviour. When considering homeokinesis, the relation between the forward model and the adaptation rules is of special interest. On one hand, there is a direct relation between the prediction error and the homeokinetic canonical learning rule. On the other hand, the response of the model is also part of the update rule. The model response represents the sensitivity of the model to the inputs and is configuration dependent. In this sense, the actions taken by the robot build upon the type of internal model used for prediction. In Chapter 4, we study how the complexity of the internal models tampers the homeokinetic behaviour. Linear and non-linear models with different learning rules and storage capacities are assessed for differences in the behaviour. We define heuristics to measure the exploratory capacity of a two-wheeled robot in a goal-free environment.

Autonomous robots need to self-actualise their internal representations to acquire the environment dynamics. Usually, the learning starts from *tabula rasa* and without a prior exploration of the interactions. Exploring the action space plays a significant role in finding optimal actions. In reinforcement learning, optimistic initial values of the function approximators have been proposed (Sutton and Barto, 1998). In such initialisation, all the states are set to the highest possible return. Even though this initialisation induces exploration, it does not make use of embodiment. The most common approach is to explore the search space by random actions. Adding a random factor to the behaviour is preferred over designed solution as they do not require knowledge about the system and its dynamics. Nevertheless, neither initialisation nor exploration are intelligent in the sense of embodied cognition. Even more, robots usually have short life spans and often no access to meaning or labels for the sensory data. On the contrary, self-organised behaviour brings about coherent exploration based on embodiment without expert knowledge of the system. We see this exploratory mode as an educated probing signal. As a first advantage, such exploration is not designed, so no expert knowledge of the system is required. Second, it is coherent with regards to the configuration of the robot and the environment as opposite to random actions. In this thesis, we are interested in how intrinsically motivated behaviour can improve learning when used as an exploratory signal. We take homeokinesis and combine it with goal-oriented learning methods. In Chapter 5, we propose various techniques that include

a priori search of the action-state space for parameter initialisation, and dual control with self-organised behaviour and task-oriented policy learning.

In Chapter 6, we present the conclusions of our experiments and discuss future lines of research.

1.4 Main Contributions

- Adaptation of the artificial curiosity learning rule as an intrinsically motivated behaviour driver in the sensorimotor loop (Section 3.3).
- Identification of the emergent behaviour provided by artificial curiosity in the sensorimotor loop as *homeostatic* (Section 3.3.2).
- Extension of the homeokinetic framework to general models with their respective response equations (Section 4.3).
- Different complexity forward models were assessed with various learning rules for self-organised behaviour, including baseline comparison. The main result is that linear time learning internal models yield a better space coverage in homeokinetic robots in the conducted experiments (Section 4.5).
- Initialisation of parameters based on self-organised behaviour increase reward gain in continuous time and space reinforcement learning (Section 5.3.2) (Smith and Herrmann, 2013).
- Gait patterns can emerge by the adaptation of CPGs by a rewarded weighted correlation of self-organised sensorimotor coupling in high-dimensional robots (Section 5.4) (Smith and Herrmann, 2013).
- Definition of a dual-control strategy with self-organised exploratory mode and reinforcement learning exploitation (Section 5.5.1) (Smith and Herrmann, 2012).
- Homeokinetic self-organised behaviour used as probing signal detriments performance in low-dimensional robots with fixed points as goal objective (Section 5.5.2.1).
- Homeokinesis enhances performance in high-dimensional robots where the goal is to optimise mobility (Section 5.5.2.2).

Chapter 2

Robotics Control Paradigm

Control of autonomous robots is based on the sensorimotor loop paradigm. This paradigm combines the sensory system and motor system through sensory-motor coupling. In the loop, the interdependency of the two quantities is defined by a feedback system. This view allows the integration of the physical configuration of the robot, its actions and the environment. To get a better understanding of the system over time, we use dynamical systems theory to study how the dynamics of the system vary trough time. In this chapter, we present a study of the dynamical properties of sensorimotor loop for a low-complexity system. The generated dynamics are non-trivial, and the system can undergo a plethora of states. Complex robotics high-dimensional systems, are also bounded to these dynamics. The study of the evolution on time of the dynamics of the systems allows us to understand the behaviour of the robots with the controllers implemented in this dissertation.

As part to explain the architecture of the robots, we present the basic configuration and its counterpart in a computer simulation. In this work, we use computer simulated robots for various practical reasons. We present the pros and cons of both the real and simulated approaches and our final consideration to choose simulations.

The main effort in our work is to exploit the benefits of self-organised behaviour to develop autonomous robots. First, we present the main characteristics of self-organisation and its goal-oriented extension, i.e., guided self-organisation. All of our approaches are based on the internal representations that the robots acquire through experience. In this chapter, we explain the biological motivation for internal models and its implementation in the sensorimotor loop.

2.1 Self-Organisation

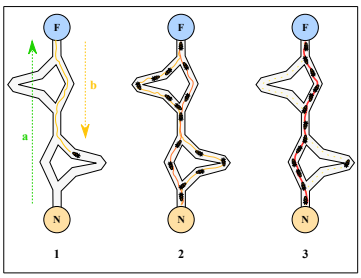
Self-organisation (SO) is a paradigm that has been used to describe phenomena in various fields such as biology, economics, computer science, physics, cybernetics, thermodynamics, information theory and chemistry (Haken, 1983; Gershenson, 2007). One way of characterising SO is "the spontaneous often seemingly purposeful formation of spatial, temporal, spatiotemporal structures or functions in systems composed of few or many components" (Haken, 2008). A system, initially in disorder, spontaneously develops order or coordination using local rules that apply to the elements of the system. There is no high-level control leading the process, yet a high-level coordination emerges. In synergetics, SO is described as the emergent "order" corresponding to the increase of negative *entropy* (Heylighen, 2001). A system requires a constant input of energy with low entropy, dissipating the internally generated entropy through the output of high-entropy energy, e.g., heat. Such energy input will produce *dissipative structures* that will maintain the system far from the thermodynamic equilibrium (Nicolis and Prigogine, 1977). Nevertheless, a full description in terms of energy may not be required when describing general dynamical systems.

A classic example of self-organisation is the light-source laser. The atoms within a glass tube (with mirrors on both ends), while excited by an electric current, emit random incoherent light. Above a critical value of the electric current, the properties of the emitted light change qualitatively. Due to the mirrors, only orthogonal light waves remain inside the tube and interact intensely with the light-emitting atoms. This effect leads to a *self-amplification* of the interactions driven by the external current. The transition from irregular light to coherent modes of light is an *emergent* quality in the system. The new temporal structure is self-organised, i.e., it is formed only by the action of the atoms without any extended organisation (Haken, 2008).

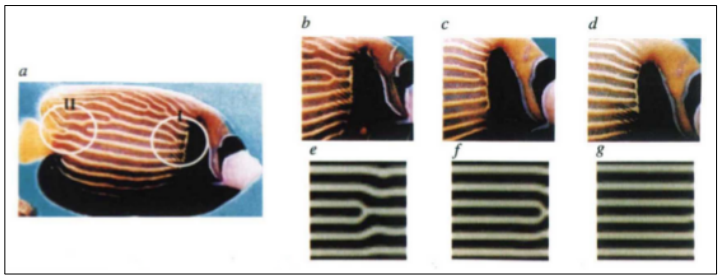
An example of SO in nature is appreciable in the Argentine ants, *Iridomyrmex humilis*, and their capacity to bring food to the nest efficiently. The ants are equipped only with a limited capacity of orientation. However, thanks to their ability to mark trails with pheromones they can find the shortest path between food and the nest (Goss et al., 1989). A trail of pheromones is left by each ant on its path to and from the nest. The decision that an ant takes while faced with more than one path is modified by the precedent ant's choice. This effect is a positive feedback that will achieve the *selection* of one branch. The shorter branch is usually chosen since it is marked by ants moving in both directions while longer branches are only marked by ants moving

in one direction. Initially, the decision point where an ant has to choose between more than one branch is unmarked; the first ant arrives, chooses randomly and travels through the chosen branch marking it in its way. After some time, the shortest branch will be marked not only by ants leaving the point but also from the ones that are arriving at the decision point from the shortest branch (Figure 2.1a). For a longer branch, the returning ants will only arrive later. As a result, the optimal path between the nest and food is found in a self-organised way.

Another example of SO is pattern formation in animal's coat or skin. The appearance of stripes and dots patterns are explained by reaction-diffusion systems, see Figure 2.1b. In these systems, chemical reactions account for the transformation of substances into each other. Diffusion spread out these substances along a surface in space. Alan Turing proposed the reaction-diffusion systems as part of the development of animal coats during morphogenesis (Turing, 1952).



(a)



(b)

Figure 2.1: In (a), the self-amplification trail of pheromones leads the ants to choose the shortest path from the nest (N) to the food (F) (Dréo, 2014). (b) shows different patterns formation on the skin of the fish *Pomacanthus imperator* at various life stages. Images (b.e-g) show computer simulations following reaction-diffusion models. Figure adapted from (Kondo and Asai, 1995).

An engineering point of view gives a definition more related to the robotic world: "a self-organising system would be one in which elements are designed to dynamically

and autonomously solve a problem or perform a function at the system level" (Bogg and Geyer, 2007). The system is not built to perform a function explicitly. Instead, it is built in such a way that their interactions and behaviour will lead to the system function. For example, a swarm of robots self-organises since the behaviour of each element of the swarm changes their behaviour based on the actual state of the environment (Dorigo et al., 2004). These artificial systems usually present both *adaptability*, by changing their state in order to fit to occurring changes (Holland, 1975), and *robustness*, i.e., they continue to function despite perturbations (Jen, 2005).

The SO phenomena can give rise to complex interactions and patterns in different systems. This increase in the order can be seen as if SO has some high-level intention, e.g., to solve a particular task. Nevertheless, SO does not have a goal on its own, it is only seemingly purposeful. When SO is driven to a target by an external force, it is called *Guided Self-Organisation*, a branch of study that extend the concept of SO. In this dissertation, we present different ways to guide the self-organised behaviour of robots (see Chapter 5). In the next section, the concept of Guided SO is presented.

2.2 Guided Self-organisation

Self-organisation can bring benefits to the whole system by an increment in the internal organisation. These benefits can be measured as increased resilience to external disturbances, adaptivity to novel tasks, and scalability with respect to new challenges. To design a system with such benefits is a difficult task: the intricate interactions within the system cannot follow a simple-minded blueprint and resists rough intervention. Thus, "the goal of *Guided Self-Organisation* (GSO) is to leverage the strengths of self-organisation while still being able to affect indirectly the outcome of the self-organising process" (Prokopenko, 2013). A self-organised system that is guided externally will profit from undirected self-organised behaviour while still finding a task-oriented bearing.

The example of SO of the shortest path in an ant colony to the food source (see Figure 2.1a) can be extended by the work in (Key and Baker, 1982). Such work presents a particular trail pheromone component in two ways to the ants: a wide and uniform swath of permeated air, and as a point source creating a time-averaged plume *downwind*. As a result, the ants travel significantly farther toward the pheromone source in wind than without the wind. The external pressure provided by the additional point source of the pheromone guide the ants in a particular way. This external guidance was

not provided as a direct control to each of the ants; there was no modification to the ants' neural circuit. The optimal path still emerged as a result of SO, but the path was affected or guided by the external modification of the distribution of the pheromones. Placing different point sources across the surface may guide the resulting optimal paths in various ways (Key and Baker, 1982).

In the robotics field, several approaches to GSO have been proposed in (Martius, 2010). A combination of a controller capable of generating coherent behaviour based on the embodiment of the robot, the environment and external guidance were tested. Robots were capable of following direct motors and sensors signals while still able to explore around the teaching signal. In another set-up, symmetries in the spatial configuration of the body of the robot are exploited. Thus, reducing the effective dimension of the sensory system, guiding the self-organisation towards a subspace with respect to the original one. Reward signals were also used to guide a spherical robot to maximise its linear and angular speed with respect to the perpendicular ground plane. In both cases, higher velocities were achieved, but in the latter case also a pirouette mode was engaged by the robot. By a combination of exploration driven by self-organisation, and exploitation directed by the reward signals, the robot was able to maximise reward.

Guided SO remains as an open field with increasing interest from the scientific community (Ay et al., 2012b). This dissertation presents various novels approaches towards GSO in autonomous robots. We present the methods in details, alongside with experiments and results in Chapter 5. Our point of interest is how internal representations of can be modified and shaped by SO processes in order to advance towards autonomous robots. In the next section, we present those internal representations.

2.3 Internal Models

Internal Models (IMs) are neural mechanisms or mathematical representations that can mimic the input/output characteristics, or their inverses, of the motor apparatus/command (Kawato, 1999). Usually, IMs are separated into two principal categories depending on its use and functionality within the system. The two main categories are *forward* and *inverse* model[1]. A forward model predicts future states of the system based on the actual state. An inverse model determines the action needed by the system to move from the actual state to the next one (Nguyen-Tuong and Peters, 2011).

[1]Other approaches to internal models include mixed model or multi-step forward model. They are covered in Section 2.3.2.

In biology, IMs have been theorised as a feature of the brain that provides support for complex behaviour like fine control movement. In (Ito, 1970), it was proposed that the cerebellum contains forward models capable of predicting future positions of the limbs. Evidence has been found supporting the idea that IMs can reside in the cerebellum of the human brain (Clark and Grush, 1999).

Fast and coordinated movements are hard to achieve only with the use of feedback loops due to slow response and small gains of sensors and actuators. The use of IMs to estimate future states provides uninterrupted sensory information in systems with relative slow feedback loop. This characteristic is especially true in biological systems. For example, in the visual feedback of arm movements the delay can variate from 150 to 200 ms. The concept of IMs has been applied effectively in control theory. Plants that rely on feedback control can optimise their response using the prediction of future states (Åström and Wittenmark, 2013). In robot control, they have been used as state estimator, sensory confirmation and cancellation, and context estimation (Wolpert and Ghahrami, 2000) among others. We present a detailed list of robotics IMs in Section 2.3.4.

In control theory, IMs have been regarded as indispensable both for good regulators (Conant and Ashby, 1970) and for delayed feedback control (Sontag, 2003; Volkinshtein and Meir, 2009). One way to achieve the model is by a designed analytical solution of the system. However, accurate design of models are hard to attain due to the complexity of the controlled agent, e.g., a high-dimensional robotic system in an intractable, unstructured and uncertain environment (Nakanishi et al., 2008; Nguyen-Tuong et al., 2009). As an alternative, IMs can learn the underlying mechanics of the world from the interaction of the agent with the environment. An IM approximate a usually unknown function based on experience. This approach allows nonlinearities to be taken into account, as opposite to standard physic-based modelling and designed models that often neglects them. For a system where model adaptation to time-dependent changes and generalisation of the model to a large state space is sought, on-line learning is necessary to implement such models.

2.3.1 Biological Internal Models

Feedback loop control cannot account by itself for fast and coordinated arm movements. This incapacity arises from the slow and small gain in biological feedback loops. The internal model theory proposes that the brain has to acquire an inverse dy-

namics model of the controlled object through motor learning (Kawato, 1999). After the learning has taken place, the motor control can be executed in a pure feedforward manner, just relying on the predictor. The first consideration of the concept of motor prediction was carried out by Hermann Helmholtz (Westheimer, 2008) regarding the localisation of visual objects. He proposed that, in order to locate an object relative to the head, it is necessary to take into account both the retinal location of the object and the gaze position of the eye within the eye orbit. The brain, rather than sensing directly the gaze position, predicts the gaze position based on a copy of the motor command acting on the eye muscle, termed *efferent copy*. A simple experiment was used by Helmholtz to demonstrate such idea: covering one eye, and moving the other one by pushing it trough the eyelid. The gaze position changes without the intervention of eye muscles, the retinal locations of visual objects change but there is no update on the predicted position of the eye. Thus, the false sensation that the world is moving arise (Wolpert and Flanagan, 2001).

The relations between the motor command sent to the eye and its consequence are relatively straightforward. However, when considering a motor command for a full body and its relations to behaviour, that relation is more complex due to the dynamics of multi-joint motion. Even more, the complexity of such relation increases if the interaction of the body with the environment are also taken into account. To be able to predict the effect of a motor command in such complex situation requires a system that can simulate the dynamical behaviour of the body and the environment. That system is termed internal forward model, as it is internal to the central nervous system (CNS). Such system models the behaviour of the body and is able to acquire the causal relationship between action and its consequence (Wolpert and Flanagan, 2001).

The existence of IMs in the brain was initially proposed by (Ito, 1970). The suggestion was that the cerebellum contains forward models of the limbs and other brain regions. Another computational approach allocated supervised learning in the cerebellum based on fMRI scanners (Doya, 1999; Thach, 1998). Theory suggests that IM acquires the necessary dynamics more efficiently through supervised learning. Thus, the cerebellum cortex seems to be the most appropriate location for them to be present. Neurophysiological data in favour of IMs in the cerebellum have been obtained by analysis of eye movements (Kawano et al., 1996; Kobayashi et al., 1998). The studies show that for some reflex eye movement, a motor command positional coordinates error signal is propagated to the Purkinje cells. Accordingly, their temporal waveforms can be reproduced by the inverse dynamics model of the eye. In addition, the

study of arm movement has shown evidence about error signal encoded by climbing fibre (Blakemore et al., 1998; Imamizu et al., 2000; Kitazawa et al., 1998). These error signals are theorised to have an active role in the adaptation of models neurons. IMs have been studied in other processes different to motor control, e.g., estimation of gravity and linear acceleration (Merfeld et al., 1999; Snyder, 1999). Also, finding evidence of its occurrence in the macaque cerebellum (Laurens et al., 2013) and in smaller animals (Dickinson, 2015). Other studies identify the IMs in different sections of the brain like the hippocampus (Berger et al., 1994; Buckner, 2010; Norman, 2010) and in the posterior parietal cortex of primates and humans (Grush, 2004; Rauschecker, 2015).

State Estimation. Among some well-understood examples that support the existence of a forward internal model in the brain is state estimation. Sensory input usually has a time delay that need to be taken into account for accurate motor control. The delay can be caused by receptor transduction, central processing and neural conduit. Additionally, sensors signal are often subject to random processes or noise. The main interpretation is that given an efferent copy of the motor command, the consequences of the action are predicted. Take, for example, a grip and load scenario. A subject is holding a bottle with his right hand; a force on the bottle can be performed by the same subject or by an external one (with no acknowledge from the original subject). When the push on the gripped object is executed by the same subject, he is able to adjust the grip at the same time that the load is taking place. A forward internal model predicts the pressure over the bottle based on the motor command efferent copy of the force applied to the bottle. If the force is executed from an external subject, only the sensory information is available to proportionate the grip. The delay in the sensory signal elicit a lag between the load and the grip, see Figure 2.2 (Wolpert and Flanagan, 2001).

Sensory Cancellation. Another example of IMs is the estimation of the input signal to determine the source of such signal. Depending on the origin (self-generated or external) different responses can be taken. For example, sensory information generated by self-motion can be cancelled out. The informed cancellation enhances the more relevant sensory information. Consider tactile stimulus: if it is externally generated it can be translated as a less intense tickle. Study of different time delay stimulus shows varying degrees of cancellation. With a large lag between the applied force and the received stimulus, the later is felt like a stronger tickle compared to others with shorter

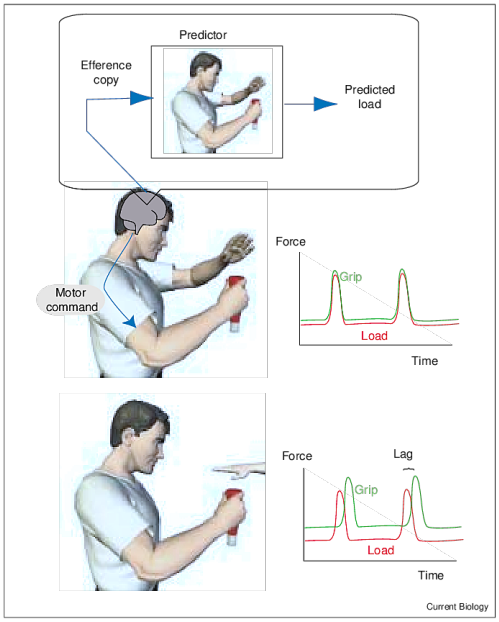


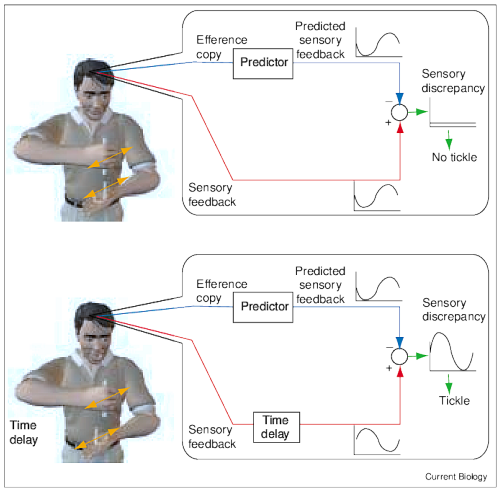
Figure 2.2: An internal forward model can be used to estimate the future state due to a self-generated force. The grip on the bottle can be adjusted based on the estimated load that the subject apply. The estimation of the state allows the immediate and accurate response of the right hand. The efferent copy sent to the predictor allows the response without lag. On the contrary, if the force is applied externally, the subject cannot predict the pressure. The only way to adapt the grip is based on the comparatively slow receptors signal. Thus, lagging behind the load. Image taken from (Wolpert and Flanagan, 2001).

delays, see Figure 2.3a (Wolpert and Flanagan, 2001). It has been proposed that the failure in this process may explain delusion of control in schizophrenia. Patients think that their body is moved by forces other than their own (Frith et al., 2000). Also, it has been studied the result of parietal damage. When this damage is present, there is an undermining of the capacity to determine whether viewed movements are ones own or not (Sirigu et al., 1996).

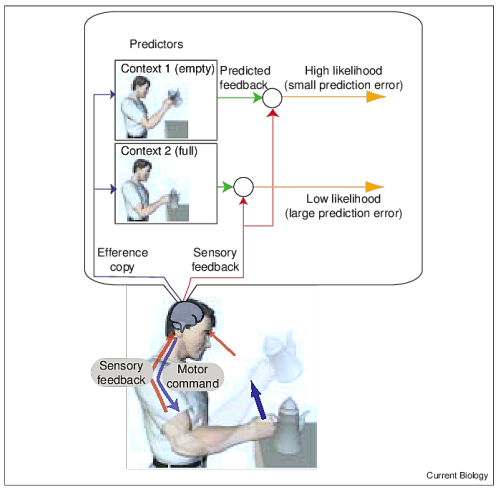
Context Estimation. It has been proposed that a set of predictors can be used to estimate the actual context. Initially, a prediction based on the efferent copy can be computed by each internal model. Then, the smallest prediction error between the prediction and the real sensory information can be used to select the appropriate controller, see Figure 2.3b (Wolpert and Flanagan, 2001). The Modular Selection and Identification Control (MOSAIC) architecture has been proposed as a solution for context

estimation (Haruno et al., 2001).

High-level cognitive functions can benefit as well from IMs. The prediction is also essential to action observation and understanding, mental practice or imitation, and social cognition. In these situations, IMs can predict the sensory outcome of an action without executing it. For example, mental practice can improve performance by tuning controllers or selecting between possible mental rehearsed actions (Wolpert and Flanagan, 2001).



(a) Sensory cancellation



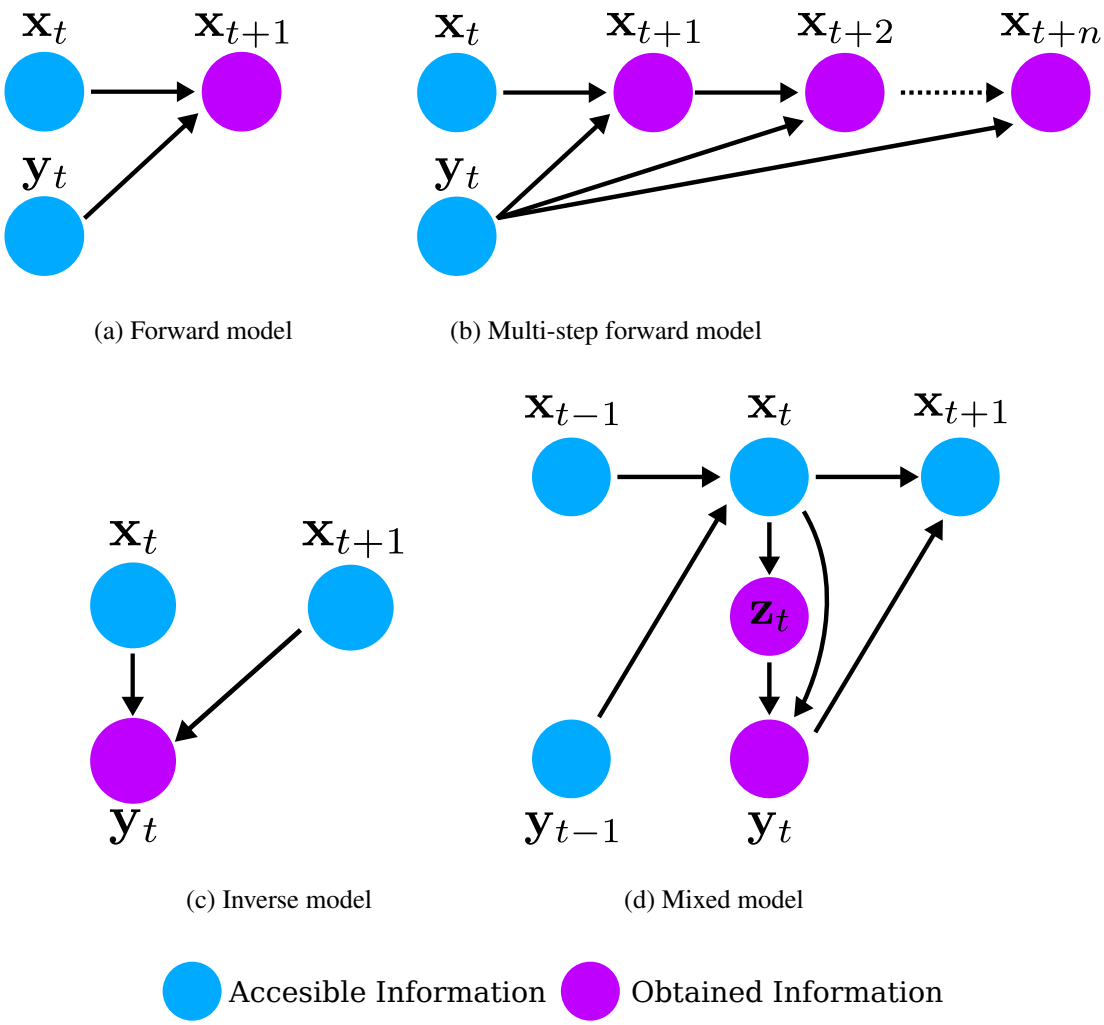
(b) Context estimation

Figure 2.3: In top (a), a stimulus is self-generated on the left hand. The efferent copy of the motor command is sent to the internal forward model. The predicted and the real sensory feedback cancel each other suppressing any tickle sensation. In bottom (a), the stimulus is delayed with respect to the motor command. The efferent copy is sent directly to the predictor. However, the real sensory feedback is delayed and cannot suppress the tickle sensation. In (b), the results of the motor command are unknown due to incompleteness of the context information. The subject does not know if the teapot is full or empty. The comparison between the feedback and the prediction is used to infer the context and to choose the most appropriated controller. All images from (Wolpert and Flanagan, 2001).

2.3.2 Internal Models for Robot Control

An accurate model of the system and the environment is essential for autonomous robotics control. In control theory, a forward model is defined as the predictor of the next state of the system $\mathbf{x}_{t+1}$ based on the actual state $\mathbf{x}_t$ and motor command $\mathbf{y}_t$. An

inverse model produces new motor commands $\mathbf{y}_{t+1}$ based on the actual state $\mathbf{x}_t$ and a desired next state $\mathbf{x}_{t+1}$. A special case of a forward internal model is the *multi-step* model used to predict further steps in the future. Another particular case is the *mixed* model, a combination of forward and inverse model able to disambiguate non-unique inverse mapping. Figure 2.4 shows the transition models for these four models. The purple circles represent the information that is obtained by the IMs, and the blue circles the information that is accessible.



(a) Forward model

(b) Multi-step forward model

(c) Inverse model

(d) Mixed model

Figure 2.4: (a) and (b) represent forward models, where future states $\mathbf{x}_{t+n}$ are predicted based on the actual state $\mathbf{x}_t$ and action $\mathbf{y}_t$. In (c), the inverse model is presented. The motor command is calculated by the IM based on the actual state and a desired new state. (d) shows a mixed model, where a hidden variable $\mathbf{z}_t$ is used to link the inverse and forward models. Arrows represent the flow of information.

2.3.2.1 Forward Model

A forward model represents a causal relationship between states and actions. Most of the time, the causal mapping can be learned directly with the use of standard regression techniques. One of the first forward models used for control was the *Smith predictor* (Smith, 1958). The predictor is used to compensate the effect of delay in the system. The delay can arise due to the physical distance between the sensors and the process or from slow response of the actuators. Usually, it is possible to reduce the gain of the controller in order to make it *wait* for the results of the actuator. However, the parameters of a close-loop with and without delay are very different in most of the cases. Another solution is to use a predictor of the process in order to estimate the effects of the controller without delay. This predictor does not eliminate the delay since it is intrinsic to the process, but it does compensate its effect. The control signal that is sent to the plant is also used as an input to a predictor, where the output signal is passed to the controller without delay.

We present in more details the forward model in a robotics close-loop in Section 2.4.1.

2.3.2.2 Inverse Model

In an inverse model strategy, the IM receives the desired and actual state as input, then it estimates the control signal that will move the system to the desired state. In comparison with the forward models, the inverse models map an anti-causal relationship. The inverse may not exist or could be not well-defined (Jordan and Rumelhart, 1992). In robot control, inverse models are used for computation of torque control (Craig, 2005). The inverse dynamics model is used to generate the necessary torques to drive a robot trough a desired joint space trajectory. Figure 2.5 shows a closed sensorimotor loop with an inverse model I. The model receives a reference signal ref_t, or desired trajectory, and the actual state $\mathbf{x}_t$. One possible control approach is to minimise the difference between the state and the reference by means of a PID controller (Rivera et al., 1986).

For example, in classic reinforcement learning (Sutton and Barto, 1998), the agent has to learn a policy that maximises the reward over time. The reward signal is measured from the environment as well as the state. A value function indicates the maximum future reward attainable at said state following the actual policy. The agent builds an inverse model with a unique policy that maximises the reward over time.

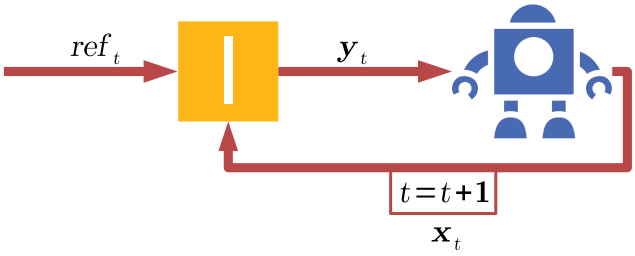


Figure 2.5: A closed sensory motor loop with an inverse model. The robot is controller by the motor commands $\mathbf{y}$. The inverse model calculate the actions based on the actual sensory input $\mathbf{x}$ and a reference signal. At the next time step, the state and reference signal are updated and the cycle continues.

2.3.3 Model Learning

A direct approach to generating models is to design them. Starting from an expert point of view, all the necessary information can be built into the models even before they start their operation cycle. Such standard is suitable only for a bounded set of systems. Systems that present low complexity, non-compliance, and interact with a highly predictable and well-defined environment. Still, for such systems, a design model can be too complicated to define or simply unknown. To cope with these restrictions, IMs can be built from experience, i.e., from the interaction of the controlled object with the environment. In this way, complex non-linear systems can be modelled with the absence of an expert.

Models need to capture the dynamics of the system. Usually, this learning is done based on the data obtained from experience. Depending on the problem to model, three different architectures can be used. The most common approach is *direct modelling*. It is a straightforward way to model a process. This learning paradigm uses the observed inputs and outputs. It is the most widely used model learning technique in robot control. Some examples include inverse dynamics control (Nguyen-Tuong and Peters, 2010) and vision-based control (Miller et al., 1987). A forward model can benefit from this architecture since it maps a causal relationship (see Section 3.2). When learning inverse models, if the problem is well-defined, this technique can be directly used. For example, the inverse dynamics model for computed torque (Craig, 2005). If the problem is ill-posed, where the mapping is multi-valued, direct modelling may not capture the system properly. Different solutions for the same goal will average on time. For such problems, *indirect modelling*, by means of feedback error model learning has been proposed (Kawato, 1990).

In feedback error model learning, an error signal is generated based on the output of a feedback controller. This error is used to update the IM representation. The total motor command provided as input to the robot is the sum of the feedback (from the feedback controller) and feedforward (from the IM) command. As the IM (an inverse model) improves, the feedback signal tends to zero. If the feedback error is non-zero, it corresponds to the error in the inverse model in the feedforward loop (Craig, 2005). The insight of the approach implies that by minimising the feedback errors, i.e., when the model converges, the feedback control term decrease. Hence, the feedback control part becomes irrelevant while the inverse model describes more precisely the inverse dynamics of the system. Feedback error learning, compared to direct learning, is goal-directed, resulting from the minimisation of the feedback error. It was biologically motivated by the cerebellar motor control (Kawato, 1999). In robotics control, it has been further developed with use of neural networks (Miyamoto et al., 1988; Shibata and Schaal, 2001).

The *distal teacher learning* approach can handle general inverse model learning, even from those who suffer the problem of ill-posedness (Jordan and Rumelhart, 1992). A combination of a forward and an inverse model allows to resolving the mapping non-uniqueness. The forward model acts as a distal teacher that guides the learning of the inverse model. On one hand, the forward model determines the error made by the inverse model during learning. On the other hand, the inverse model learns by minimising such error. This approach can be seen as the inverse model learning solution for a particular desired trajectory, whilst minimising the error between the output of the forward model and the input of the inverse model (Nguyen-Tuong and Peters, 2011). Hence, the inverse model will learn the solutions that are consistent with the unique forward model. Nevertheless, distal learning presents disadvantages when compared to the previous learning methods. Numerical and learning instability, and error accumulation are examples of disadvantages that impact the performance of the IM negatively. However, it has been successfully used in learning inverse kinematics in redundant robots (Jordan and Rumelhart, 1992) and robot control applications (D'Souza et al., 2001; Peters and Schaal, 2008; Ting et al., 2009).

Depending on the complexity of the system, different model implementations are preferred. Regression analysis like least square methods (Ljung, 1987) and linear models (Martius et al., 2007) are preferred when no long term memory is required. For complex systems, neural networks (Butz et al., 2007; Haykin and Network, 2004), fuzzy logic (Layne and Passino, 1996; Vempaty et al., 2009) or statistical approxi-

mation techniques (Lopes and Damas, 2007; Rasmussen, 2006; Schölkopf and Smola, 2002) are preferred. For inverse dynamics learning, non-local implementations, like ν-support vector regression (Schölkopf and Smola, 2002) and Gaussian processes regression (Rasmussen, 2006) present better overall performance compared to local predictors, e.g., locally weighted projection regression (Vijayakumar and Schaal, 2000) but at the cost of higher computational cost (Nguyen-Tuong et al., 2008).

2.3.4 Internal Models Robotic Applications

In robotics control, IMs are used in more ways than state prediction and inverse dynamics generator. Indirect measures, like the learning rate of change, have been studied in autonomous robots. Next, we present various applications of IMs in robotic control.

Predictive Model. In the homeokinesis principle (Der and Liebscher, 2002), an IM is used as a predictive model of the future input signals. The predicted input is compared with the actual input, giving rise to the *prediction error*. This error is then used to modify the behaviour of the robots as well as to update the forward model in order to decrease the error. This approach is widely used in this dissertation. For this reason, in Chapter 3, we present its dynamical properties in detail.

For the task of learning multiple goal-directed behaviours simultaneously (Tani et al., 2008), IMs are used as predictive models in conjunction with a continuous-time recurrent network (CTRNN). The main objective is to learn to regenerate sensory sequence patterns. Different modalities of sensation (vision-based object position and arm joint's proprioception) are taken as input. The CTRNN combines the inputs to generate a prediction of their time developments in the future. The network is trained to minimize the error between the teaching sequence pattern coming from the outside and the predicted sequence pattern generated by itself. In the next time step, the predicted sensory sequences are sent in a close-loop configuration back to the input, generating the behaviour of the robot given an initial context.

Auxiliary Signal Input. Robots develop an IM to rely completely on the reconstructed signal computed by the model. In this way, they can complete the assigned task when sensory stimulation is temporarily unavailable. In (Gigliotta et al., 2010), it has been shown that a simulated robot evolve to display navigation skills when the actual sensory input is deprived. The robot uses the learned IM for anticipating functional properties of the next sensory state rather than the exact state that sensors would

have assumed.

Intrinsic Motivated Behaviour. As a manner to generate intrinsically motivated behaviour, agents are encouraged to search for scenarios where they maximise the learning process. Instead of controlling the robot to minimise the prediction error, the maximisation of the learning rate is promoted. In this case, the robot is driven towards unknown interactions that are plausible to be learned by the model. In (Herrmann et al., 2000), to avoid that maximal predictability blocks any input in an autonomous agent, *artificial curiosity* was included in the loop. The agent actively avoids very small and very large error in favour of substantial decrease of the prediction error. Intelligent Adaptive Curiosity (Oudeyer et al., 2005) also pushes the robot towards maximisation of the learning progress. The robot focuses on situations that are neither too predictable nor too unpredictable. On time, the complexity of the robot activities autonomously increase, and a developmental sequence appears without being manually constructed. According to (Schmidhuber, 2010), curious agents are interested in unknown regularities that can be learned. A curiosity driven agent tends to get bored by both predictable and inherently unpredictable things. The mismatch between reality and expectations is translated into *curiosity rewards* for curious, creative and exploring agents. The agent is attracted to observe or create surprising aspects of the world in order to learn novel patterns. In these three approaches to artificial curiosity, the next input is predicted by a data compressor (a forward model) based on the history of actions and states. The controller is rewarded for actions that yield unknown states. To discourage the controller from focusing on truly unpredictable random inputs, the expected progress of the predictor is modelled.

Configuration Representation. Another use of IM is the self-recognition of the own configuration. In this approach, a robot can indirectly infer its morphology through self-directed exploration (Bongard et al., 2006). Based solely in the generated self-model, the robot can synthesize new behaviours by exploiting the own state prediction. In this case, the IM will predict the result of the actions without executing them. Moreover, if the robot's topology undergoes unexpected changes, the same process restructures its internal self-models. Updated knowledge of its configuration will lead to the generation of qualitatively different, compensatory behaviour.

Multiple Models. The MOSAIC architecture has been proposed as an answer of how the brain stores the dynamics of multiple objects (Haruno et al., 2001). Based on the most suitable model (minimum prediction error), it chooses the associated controller for each situation. In (Martius et al., 2008), a similar architecture has been proposed to learn behavioural primitives. Consistent behaviour carried out by a homeokinetic controlled robot is stored in a cluster of competing expert controllers. Then, each expert is chosen for learning based on its capacity to predict the current behaviour through its model. Furthermore, the experts can be exploited to reproduce the original behaviour in an inverse model fashion.

Hierarchical Modelling. In (Kawato et al., 1987), an IMs hierarchical architecture has been presented. In this approach, the goal of movement (desired trajectory) is translated into motor commands through several transformations. A hierarchical neural network model is used for such transformations. The model comprises three parts: i) a feedback loop, ii) a forward model for dynamics and iii) an inverse dynamics models. The motor command is generated based on the goal movement, the trajectory in task-oriented coordinates and the trajectory in body coordinates. The later one obtained indirectly from the task-oriented coordinates. As an example of its suitability, the hierarchical architecture was able to manipulate a robotic arm with three degrees of freedom successfully.

Model-Based Reinforcement Learning In model-based reinforcement learning (Kuvayev and Sutton, 1996; Doya et al., 2002), an agent builds a forward model of the environment as part of the process of policy optimisation. At the same time as exploring to optimise the policy, the agent maps the actual state and the action to the next state. The information gathered has two principal objectives. First, it is used to plan ahead the most suitable policy. Second, the learned model can be used for a different task. If the task changes, the policy is useless, but the model still predicts the same environment.

2.4 Sensorimotor Loop

Sensory system and motor system are integrated through sensory-motor coupling. Motor commands depend both on the stimulus and in the internal state of the system. In nature, synaptic processes, recurrent and feedback connections, and learning modify

the neural response at almost every stage of a sensorimotor pathway (Huston and Jayaraman, 2011). In robotics, a closed *sensorimotor* loop works as an abstraction of the information flow in a feedback system. In this loop, the controller receives input data from the sensors of the robots, and based on the these values and its internal parameters produces the new motor commands. After the execution of the motor commands, new sensor values are read and passed to the controller, closing the loop. In order to study the loop, we use the framework of dynamical systems. Within this theory, the evolution of a system on time is mathematically defined. This allows the understanding of the system from an analytical perspective. Dynamical systems theory has been successfully proposed for robot control (Khansari-Zadeh and Billard, 2012; Ijspeert et al., 2002; Martius et al., 2007; Ay et al., 2012a). The understanding of the flow of states is specially interesting for non-linear systems like the ones we are concerned in this work.

In a robotics sensorimotor loop, the update of sensor values and the generation of new motor commands occur at each time step of the loop. Each time step belongs to a discrete interval $t = 0, 1, 2, \ldots$. In practice, for real and computer simulated robots, the frequency of steps usually range from 10 to 100 Hz. This frequency depends on the speed of the information processing of the sensors, actuators and internal processes. The vector state $\mathbf{x}_t \in \mathbb{R}^n$ holds the values of the n sensors at time t. At each time step, the state is updated with new sensor values. For example, the sensory signals could be room temperature, joint position or angular speed of a wheel. Temperature and infra-red sensors are of the *exteroceptive* type since they provide information about the environment by itself or with relation to the robot. The position of a joint or the angular speed of a wheel represents *proprioceptive* sensors since they give information of the internal state of the robot. We present a more detailed list of sensors in Section 2.5.2.

The motor commands are represented by the vector $\mathbf{y}_t \in \mathbb{R}^m$, where each m-th element represents the commands of the m-th motor at time t. We can define the value of the motor vector in dependency of the sensory data:

$$\mathbf{y}_t = K(\mathbf{x}_t), \tag{2.1}$$

with $K : \mathbb{R}^n \to \mathbb{R}^m$. This is a minimal definition of the controller that can be extended to be dependent on an internal state $\mathbf{c}_t \in \mathbb{R}^k$. The internal states is also updated at each time step:

$$\mathbf{y}_t = K(\mathbf{x}_t; \mathbf{c}_t) \tag{2.2}$$

$$\mathbf{c}_{t+1} = O(\mathbf{x}_t, \mathbf{c}_t) \tag{2.3}$$

The function $O : \mathbb{R}^n \times \mathbb{R}^k \rightarrow \mathbb{R}^k$ defines the update rule of the internal state.

Figure 2.6 shows the information flow in the sensorimotor loop. At time t, the sensor values $\mathbf{x}_t$ are used as the input for the controller K. The controller produces the motor commands $\mathbf{y}_t$ depending on the sensory information and the internal values. The robot actuates the motor commands. At next time step $t+1$, new sensor values are fed into the controller and the loop repeats. A typical controller with some learning capacity will update the $\mathbf{c}$ parameter at each time step.

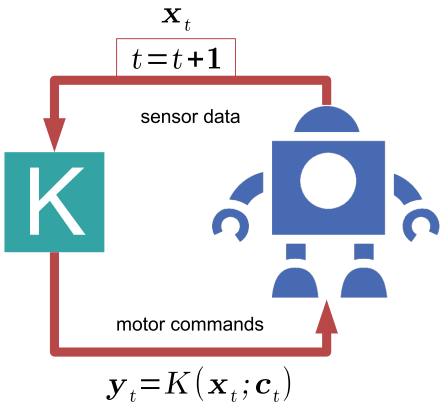


Figure 2.6: Sensorimotor loop. At time t the controller K uses the sensor values $\mathbf{x}_t$ and the internal state $\mathbf{c}_t$ to compute the motor commands $\mathbf{y}_t$. The robot acts these commands and delivers the new sensor values at time $t+1$. The controller also updates the internal state according to Equation. 2.3.

2.4.1 Forward Models in the Sensorimotor Loop

In the sensorimotor loop presented in the previous section, the robot acts based on its internal state and the actual sensory inputs. One way to add some cognition to the robot, minimalistic, is adding the capacity to learn from experiences in order to predict future outcomes. A forward model that can predict a future state is realised by,

$$\tilde{\mathbf{x}}_{t+1} = M(\mathbf{x}_t, \mathbf{y}_t), \tag{2.4}$$

where $M : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ maps the sensor values $\mathbf{x}$ and the motor commands $\mathbf{y}$ into new estimated sensor values $\tilde{\mathbf{x}} \in \mathbb{R}^n$.

If M defines a model that *knows* everything from a deterministic world, it will predict precisely every sensors states with the use of the actual states and the motors commands. In this case, the prediction is equal to the real sensor values. This

assumption is not feasible in real robots by various reasons: i) the complexity of usually non-linear interactions, ii) the intractable size of the behavioural space, iii) and stochasticity inherent in any physical system, e.g., noise both at sensory and motor level. Nevertheless, the mismatch between the real world and what the model is able to predict can be purposefully exploited, as we will present in Chapter 3.

We can define the term $\xi \in \mathbb{R}^n$ as a way to account for the non-deterministic part of the systems:

$$\mathbf{x}_{t+1} = M(\mathbf{x}_t, \mathbf{y}_t) + \xi_{t+1}, \tag{2.5}$$

ξ can be computed as the difference between real sensor values and the predicted ones:

$$\xi_t = \mathbf{x}_t - \tilde{\mathbf{x}}_t, \tag{2.6}$$

This quantity is accountable for the discrepancy between the estimated and real values generated both by the prediction capacity of M and also for the sensory noise present due to the physical limitations.

Now, the dynamics of the sensorimotor loop can be written in closed form in sensor space,

$$\mathbf{x}_{t+1} = M(\mathbf{x}_t, K(\mathbf{x}_t)) + \xi_{t+1} \tag{2.7}$$

and the prediction error can be defined as,

$$E_t^{pred} = \xi_t^\top \xi_t. \tag{2.8}$$

Let the model M be realised by a function with parameters $\mathbf{a} \in \mathbb{R}^p$. In order to improve the model, i.e., boost the prediction of the next time step state, the parameters of the models can be updated towards a minimisation of the prediction error. A gradient descent algorithm of the prediction error over the parameter yields the update rule:

$$a_{i,t+1} = a_{i,t} - \varepsilon_A \frac{\partial E_t^{pred}}{\partial a_i}, \tag{2.9}$$

for a parameter a_i with $i = 1, 2, \ldots, p$, and ε_A as the learning rate that tampers the speed of change of the parameter.

Figure 2.7 shows the flow of information in the sensorimotor loop with an integrated predictive forward model. The flow of information starts with the real sensor values $\mathbf{x}$ at time t from the sensors of the robot. The controller K calculates the motor commands $\mathbf{y}$ also at time t. Now, the model M based on this information predicts the next time step sensor values $\tilde{\mathbf{x}}_{t+1}$ based on an efferent copy of the motor commands. Likewise, the robot actuates the motor commands $\mathbf{y}$. At the next time step, the prediction E_{t+1}^{pred} is calculated based on the estimated and real sensor values. The update of the parameters of the model is carried out following Equation 2.9.

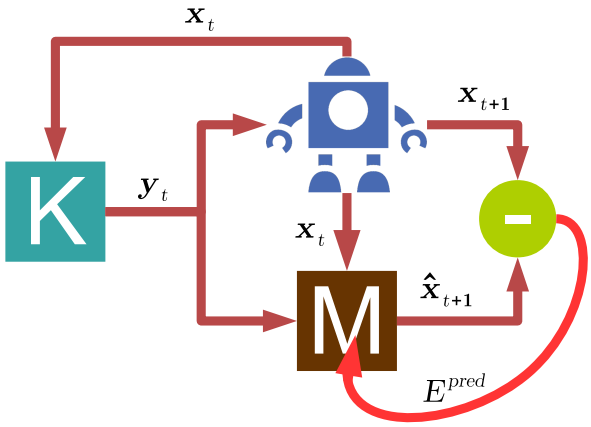


Figure 2.7: Sensorimotor loop with a predictive forward model. The controller receives the sensor values as input and calculate the motor commands. The commands are actuated by the robot and new sensor values are acquired. Meanwhile, the forward model predicts the next sensor values based on the actual sensor values and the motor commands. At next time step, the predicted and real sensor values are compared to obtain the prediction error. The error is used to update the parameters of the model in order to decrease the prediction error in subsequent steps.

2.4.2 Dynamics of the Sensorimotor Loop

In order to study the dynamics of a sensorimotor loop, a single input/output system without bias parameter will be considered. Such system only considers one sensors and one motor, i.e., $m = n = 1$, thus $x \in \mathbb{R}$ and $y \in \mathbb{R}$. Assuming a sensorimotor loop with Markovian property, i.e., the future states depends only on the present state and neglecting any prediction error for now, the system dynamics are:

$$x_{t+1} = M(x_t, y_t). \tag{2.10}$$

Using Equation 2.1 and 2.10 the system can be written in a close-loop form in sensor space:

$$x_{t+1} = M(x_t, K(x_t)). \tag{2.11}$$

For the purposes of this study, we define a pseudo-linear controller K with a sigmoidal activation function:

$$y_t = K(x_t; c) = g(cx_t), \tag{2.12}$$

with coefficient $c \in \mathbb{R}$ and sigmoidal function $g : \mathbb{R} \to \mathbb{R}$. The sigmoidal function deals with the non-linearities by maintaining the values in range. Since the values

remain within a working range, the function is especially fit when the sensor values are not in direct proportionality to the motor values. The function g can be seen as the activation function of a rate-based neuron that keeps the motor commands values confined to $(-1,1)$, as presented in Figure 2.8. Typical logistic function encompasses the logistic function, arctangent and hyperbolic tangent. We use the later function to define $g(\cdot) = \tanh(\cdot)$.

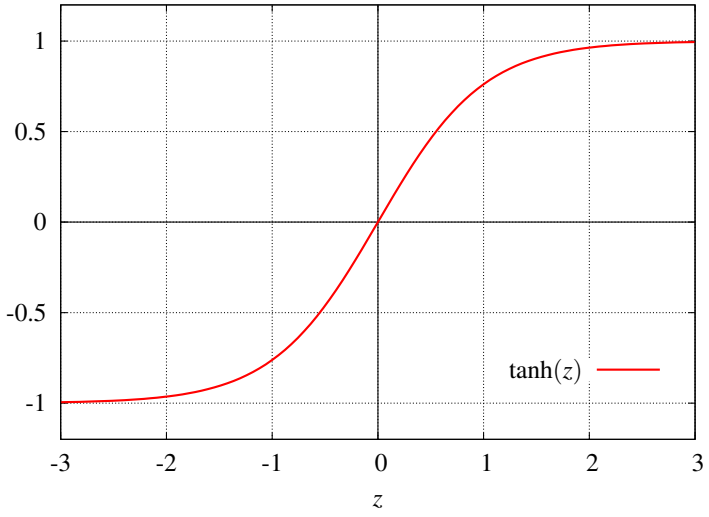


Figure 2.8: Hyperbolic tangent sigmoidal function with range $(-1,1)$. The neuron has a linear response when $z \approx 0$ with slope 1. For large absolute value of z, the neuron is in the saturation region with very low sensitivity to the input. In this dissertation, we use this definition as the activation function.

The controller can be seen as a one-layer neural network with a weight factor. We define the linear model as:

$$x_{t+1} = M(x_t, y_t) = \alpha y_t \tag{2.13}$$

where α is a hardware constant that is assumed to be known for the moment. The full system equation in a closed form now reads:

$$x_{t+1} = \alpha \tanh(c x_t). \tag{2.14}$$

Let $z_t = c x_t$ be the membrane potential of the neuron and the response $r = c\alpha$. Now, a convenient formulation to study the dynamic of the system is:

$$z_{t+1} = r \tanh(z_t). \tag{2.15}$$

The feedback strength of the sensorimotor loop is given by r. To understand its effect, an approximation of $\tanh(z) \approx z - \frac{z^3}{3}$ for small z is considered. If only the linear term

is used, then Equation 2.15 can be described as:

$$z_t = r^t z_0. \tag{2.16}$$

Such system tends to a rest state when $0 < r < 1$, hence $z_t \to 0$ as $t \to \infty$; and it will explode when $r > 1$ with z_t increasing exponentially. However, since $|\tanh(z)| < 1$ the sigmoidal function confines any further growth of z_t.

2.4.2.1 Fixed Points

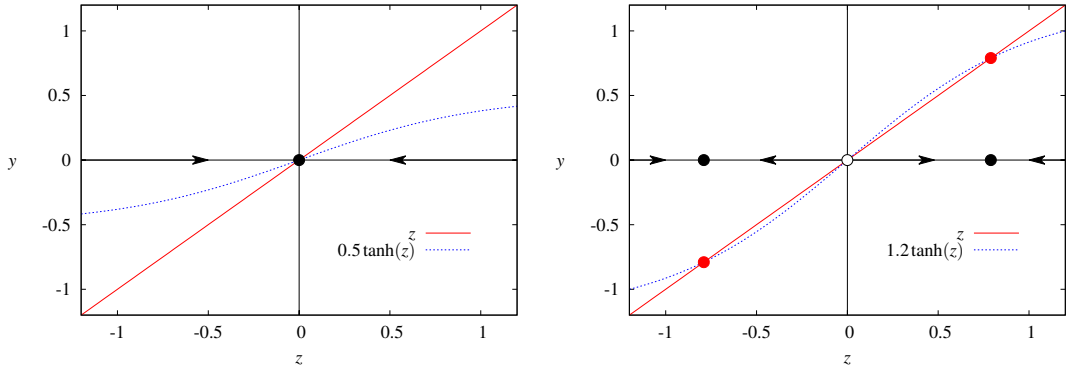
In order to proceed with a fixed points analysis, we write Equation 2.15 as a continuous differential equation:

$$\dot{z} = -z + r\tanh(z) \tag{2.17}$$

If $y = z$ and $y = r\tanh(z)$ are plotted (y as an auxiliary variable here), for $r \leq 1$ there is a single stable fixed point at $z^* = 0$. This fixed point can be seen in Figure 2.9a. When $r > 1$, this point turns into unstable and two stable fixed points appear where the curves collide, presented as red dot in Figure 2.9b. At $r = 1$ there is a supercritical pitchfork bifurcation with two emerging stable fixed points, shown in Figure 2.10. This bifurcation arise in systems that present *symmetry*. Equation 2.17 is invariant under the change of variable $z \to -z$. Thus, if we replace the variable and then we cancel the resulting minus signs on both sides of the equation we arrive at Equation 2.17 again. Replacing $z \to -z$, and using the equality: $\tanh(z) = -\tanh(-z)$, it can be shown that Equation 2.17 is *equivariant*:

$$f(-z) = -(-z) + r\tanh(-z) = -f(z).$$

This spatial left-right symmetry elicits the fixed points to appear and disappear in symmetrical pairs. To characterise in the single neuron approach, if the feedback strength is small enough, the membrane potential remains in a rest position. In this case, there is a stable fixed point corresponding to zero potential. However, if the feedback strength exceeds the resting threshold, the potential may go to a positive or negative value. The resting position has gone unstable, and two new symmetrical fixed points have been born.



(a) Single fixed point for $r = 0.5$.

(b) Two stable and one unstable fixed point for $r = 1.2$

Figure 2.9: Graphical solution for the fixed points of $\dot{z} = -z + r\tanh(z)$, Equation 2.17. The fixed points are located where z (red line) intersects $r\tanh(z)$ (dotted blue line). In (a), a single unstable fixed point is found at $z^* = 0$ when $0 < r \leq 1$. For $r > 1$ (b), two new fixed points emerge, in this case when $r = 1.2$ the fixed points are located at $|z^*| \approx 0.79$. The fixed point in the origin now is unstable.

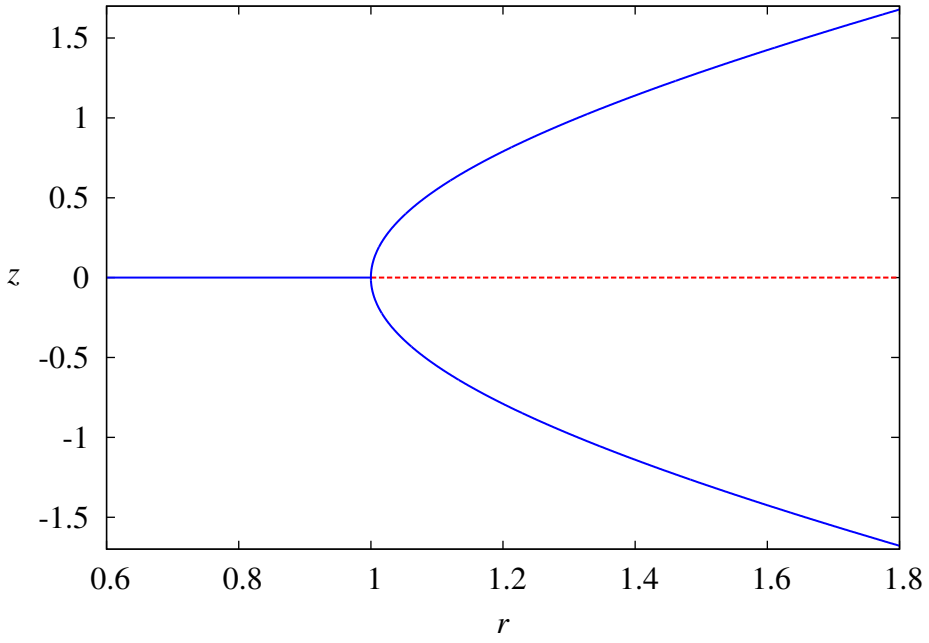


Figure 2.10: Supercritical bifurcation diagram. The solid line represents stable fixed points. The dotted line represents an unstable fixed point. The bifurcation point occurs at $r = 1$.

2.4.2.2 Extension of the Parameter Space

In the previous section, it was shown that the dynamics of the system appear from a simple configuration. Now, a bias is introduced to the definition: for a single neuron, a new term is added to the membrane potential. This expansion of the parameter space creates new dynamics. The effect of hysteresis can be appreciated under this new configuration.

The new term $h \in \mathbb{R}^m$, with $m = 1$, is included in the definition of the controller. This parameter is a bias capable of driving the system away from the symmetry. Now the controller is defined as:

$$K(x_t) = \tanh(cx_t + h).$$

In order to study the behaviour, the definition of the membrane potential is updated to $z = cx + h$, and now the system based on Equation 2.14 is:

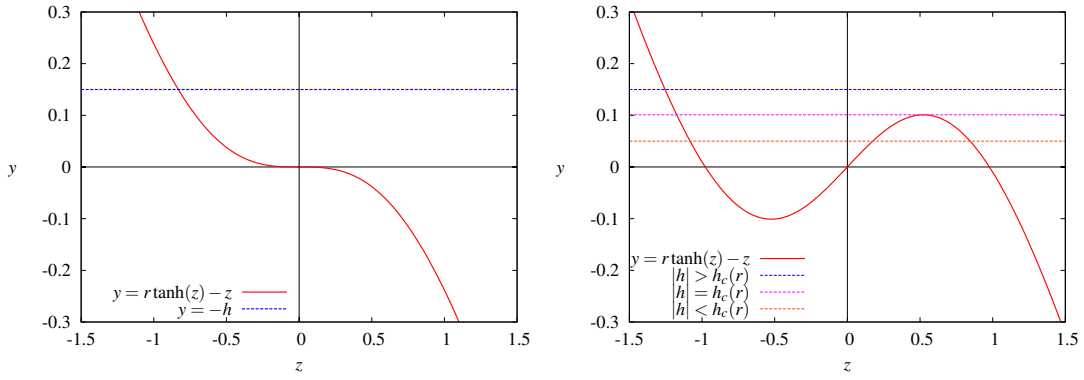
$$z_{t+1} = r\tanh(z_t) + h. \tag{2.18}$$

If $h = 0$ the system is the same as in Equation 2.17, the supercritical pitchfork bifurcation, and the symmetry between z and $-z$ remains (Equation 2.18). However, for $h \neq 0$, the symmetry is broken, that is why this parameter is known as the *imperfection parameter*.

The system now counts with two independent parameters (h and r). First, we analyse the system with a fixed and positive value for r. Negative values for r yield change of sign at every iteration, so they are kept out of consideration for a robotic controller. Such fast change in the sign is seen as undesired behaviour as it mainly provides shaking behaviour.

We use a graphical approach. Figure 2.11 shows a plot of $y = r\tanh(z)$ and $y = -h$ in the same axes, so the intersections indicate the fixed points. First, for $0 < r \leq 1$, the hyperbolic tangent is monotonically decreasing, the horizontal line $y = -h$ intersect in exactly one point (Figure 2.11a). For the case of $r > 1$, one, two or three intersections can arise, depending on the value of h (Figure 2.11b).

When the horizontal line is tangent either to the local maximum or minimum of the sigmoidal, there is a *saddle-node* bifurcation. This kind of bifurcation happens when a parameter is varied, and two fixed points move towards each other, collide, and mutually annihilate. The value for h when this bifurcation occurs can be found for the local maxima at $\frac{d}{dz} r\tanh z - z = 0$. To find the value, we use the approximation



(a) Single fixed point for $0 < r \leq 1$.

(b) One, two or three fixed points for $r > 1$.

Figure 2.11: Graphical solution for the fixed points of $\dot{z} = -z + r\tanh(z) + h$, differential version of Equation 2.18. The fixed points are located where h intersects $r\tanh(z) - z$. In (a), a single fixed point is found when $0 < r \leq 1$. For $r > 1$ in (b), one, two or three fixed points exists depending on the value of h.

$\tanh(z) \approx z - \frac{z^3}{3}$ again,

$$z_{max} = \sqrt{1 - \frac{1}{r}},$$

and the value at that local maxima is

$$h_c(r) = r(z_{max} - \frac{z_{max}^3}{3}) - z_{max}.$$

Analogously, the value at the minimum is the negative of this quantity. Accordingly, saddle-node bifurcation occurs when $h = \pm h_c(r)$. Equation 2.18 has three fixed points for $|h| < h_c(r)$, two when $|h| = h_c(r)$ and only one fixed point for $|h| > h_c(r)$. In Figure 2.12, the bifurcation curves $h = \pm h_c(r)$ are plotted. At $(r,h) = (1,0)$ a *cusp point* exists where the two curves meet tangentially. The labels indicate the regions with its respective number of fixed points. Saddle-node bifurcations occur all along the boundary of the regions.

In Figure 2.13 we present a bifurcation diagram of z^* vs. r, with a fixed value of h. When $h = 0$, the typical supercritical pitchfork diagrams emerges. However, for $h \neq 0$, the pitchfork disconnects into two parts. The lower part has only stable fixed points (solid blue line) while the upper part has both unstable (dotted red line) and stable fixed points. At $r = 1$, there is no longer an acute evolution, the fixed point just slides through the lower branch. Even more, the upper branch is accessible only using a relatively large disturbance.

Another view of the dynamics is through the plot of z^* vs. h, for a fixed r. When

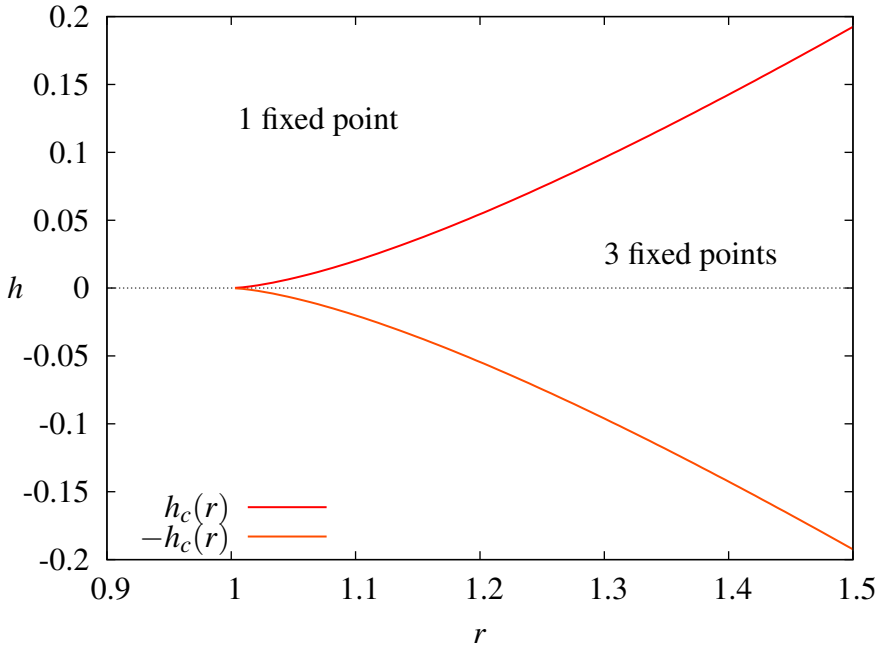
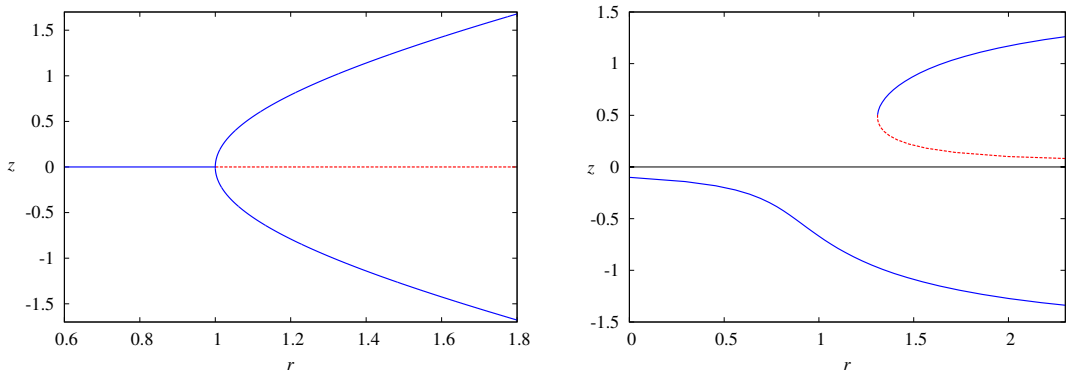


Figure 2.12: Stability diagram showing different types of behaviour while moving on the parameter space over the (r,h) plane. The lines represent saddle-node bifurcations and are boundaries for the two zones with one or three fixed points.

$r \leq 1$ there is only one stable fixed point for each h (as seen in Figure 2.11a). For $r > 1$, there are three fixed points when $|h| < h_c(r)$, and only one when $|h| > h_c(r)$. At the three fixed points region, the middle one is unstable (dotted red line), and both the upper and lower are stable, represented as a solid blue line in Figure 2.14. Note that this graph looks just like Figure 2.11b but rotated. The purple dotted lines and arrows show the transition by means of a *catastrophic bifurcation*. The system can undergo a drastic change when only small changes have taken place in the parameter space. In Figure 2.14, the pair (h,z) has been plotted for a supercritical value of $r = 1.2$. For a smaller h, if the system is in the lower branch, it will stay on that branch until the fixed point disappears. After the disappearance of the fixed point, the system will jump to the upper branch (through the purple dotted line). The system then will behave in the same way in the opposite direction following the change of h. This time-based dependence of the system is called the *hysteresis* effect. For one configuration of a parameter, the system can be in two possible states and h can be used to pass through this transition.

A three-dimensional plot of the system is shown in Figure 2.15. This representation contains all the previous plots as cross-sections or projections. It can be seen that



(a) $h = 0$ (b) $h \neq 0$

Figure 2.13: Bifurcation diagrams. In (a), the supercritical bifurcation pitchfork arises when $h = 0$. In (b), for $h \neq 0$ a catastrophic bifurcation emerge. The system can jump to relative distant state with small changes in the parameter space. Solid blue line identifies a stable fixed point while the dotted red line unstable fixed points.

the system forms a surface that folds into itself in certain places. Such surface is called a *cusp catastrophe*. The intersection of a horizontal plane at z- and r-axis for $h = 0$ can be seen in Figure 2.16a. The intersection reflects the same fixed points as in Figure 2.13a. If the plane is displaced to $h \neq 0$, then the Figure 2.16b is obtained, again, these fixed points correspond to the ones in Figure 2.13b. A cross section over the (r,h) plane (Figure 2.16c) shows the same fixed points and dynamics as in Figure 2.14. The projection of the folds of the surface onto the (r,h) plane, Figure 2.16d yields the saddle-node bifurcations as seen in Figure 2.12.

The term catastrophe references a change in the parameter space that can carry the state of the system over the edge of the fold of the surface in a discontinuous drop to an alternative region of the surface. Figure 2.14 and 2.17 show this drop.

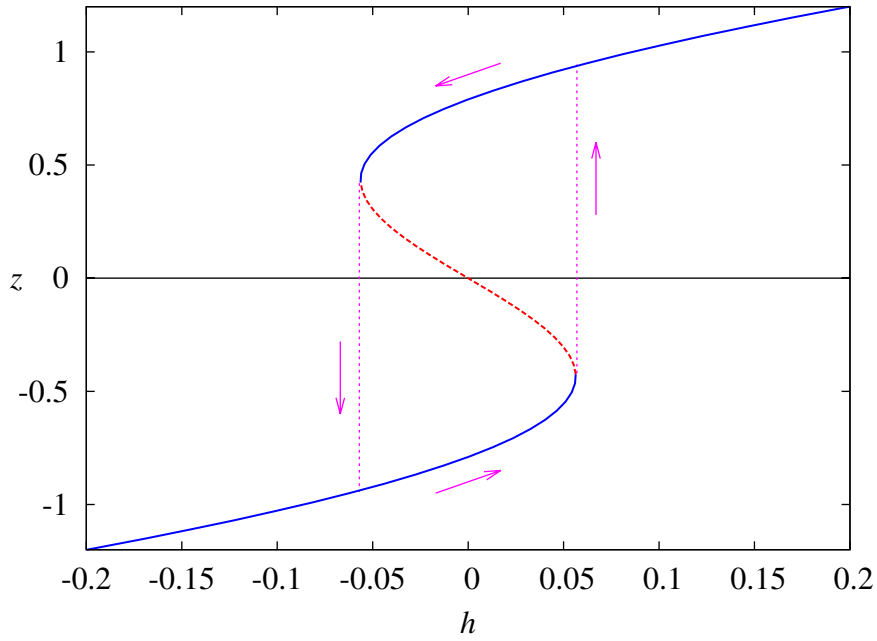


Figure 2.14: Bifurcation diagram for parameter h. Hysteresis effect for supercritical value for the parameter $r = 1.2$. Furthermore, it can be appreciated the catastrophic jump when varying h in one direction, as soon as the fixed point disappear (dotted red line) the system undergoes a large change of state.

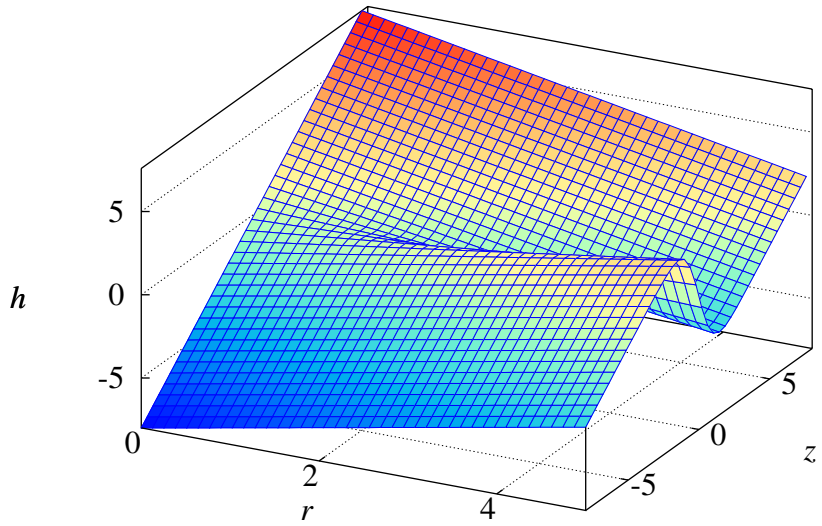
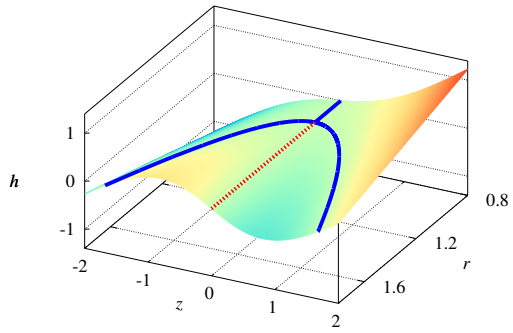
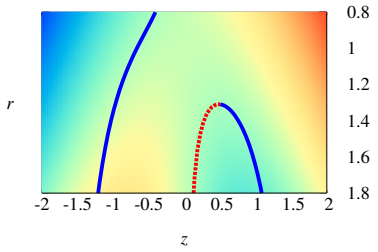


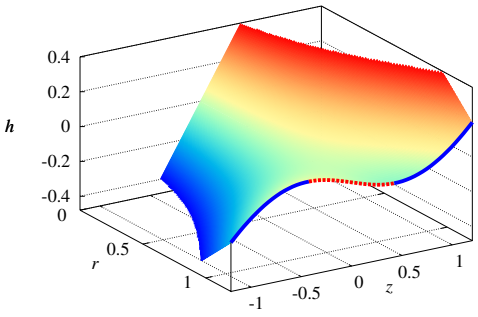
Figure 2.15: Surface of $z = r\tanh(z) + h$ that folds into itself forming a *cusp catastrophe* surface.



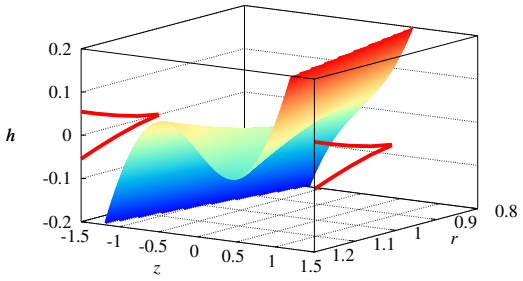
(a) Bifurcation curves for $h = 0$



(b) Bifurcation curves for fixed $h \neq 0$



(c) Bifurcation curves for supercritical fixed r



(d) Bifurcation curves projected into the (r,h) plane

Figure 2.16: Projection and cross sections of the surface shown Figure 2.15. In (a) and (b), the bifurcation curves for different fixed values of h can be seen. The differences between the surface shape and the curves are due to the effect of the approximation of the hyperbolic tangent. Nevertheless, the system holds the same dynamics properties. In (c), hysteresis and cusp catastrophe can be seen. The bifurcation curves of the edges of the folded surface are projected into the (r,h) plane in plot (d).

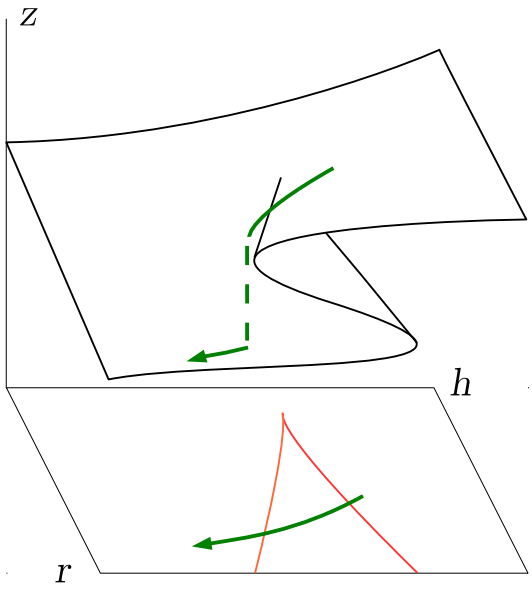


Figure 2.17: Cusp catastrophe surface. The red and orange lines represent the same bifurcation curves as in the stability digram Figure 2.12. The green line represents a trajectory of the system where it transits discontinuously from the upper region to the lower one. Image reproduced based on (Strogatz, 2001).

2.5 Robotics Architecture

Test of theories in robotics usually requires several trials to obtain enough statistics for clear understanding. In real robots, these trails have practical drawbacks. For example, computational power, state restarting or repositioning, battery charge, or collection of log data. In a computer simulated environment various of these difficulties can be overcome. In a simulation, there is no need for charging and replacement of batteries. Also, it is easy to implement the restart to initial conditions, simulation speed is not bounded by physical limitations, and it is possible to use integrated log and inspection of variables.

Computer simulations are a valid test-bed for theories and provide a good starting point for the development of robotic control (Goschin et al., 2007). Both, simulations and real robots, can be conceptualised as dynamical systems. The theory presented in this chapter is applicable to either methodology. Several developments in robotics started in conceptual levels and then tested in an artificial environment. For example, we have already mentioned reinforcement learning and intrinsically motivated behaviour approaches that have been successful in real robots. In this dissertation, we used the *LpzRobot* simulator (Martius, 2012) as it offers a realistic robotic simulation with a plethora of tools that aid in the study of our theories.

2.5.1 Computer Simulator Structure

The simulator consists of three main parts. First, the controller framework implements the robot controllers, neural networks, matrix library, introspection and support functions. Second, the physics simulator where the rigid body dynamics and graphical rendering is accomplished. Third, analysis tools used for data manipulation.

An *agent* is a set of components including the computer simulated robot and the wired controller. The data flow in the agent grants the execution of the sensorimotor loop, starting from reading the sensor values to the execution of the motor commands by the robot. A *wired controller* defines the information flow from robot sensor values to motor commands. As depicted in Figure 2.18, the wired controller has three modules: the *wiring* **W**, the *controller* **C** and the *logging and plotting* interface **L&P**. The wiring receives raw sensor values directly from the robot's sensor and process them for the controller. For example, the angular speed of a joint could be normalised by the nominal maximum speed or the position of a joint could be translated into another reference system. The controller receives the processed sensor values and produces raw

motor values that are used as input to the wiring. The processing of the motor commands could be the squashing or expansion of values to keep the robot in a working regime. At every level of information flow, within the wired controller, there are tools for logging and plotting. These tools are used to visualise data in an interactive plot or to save it on files for post-processing. In our simulations, the loop runs at a frequency of 100 Hz.

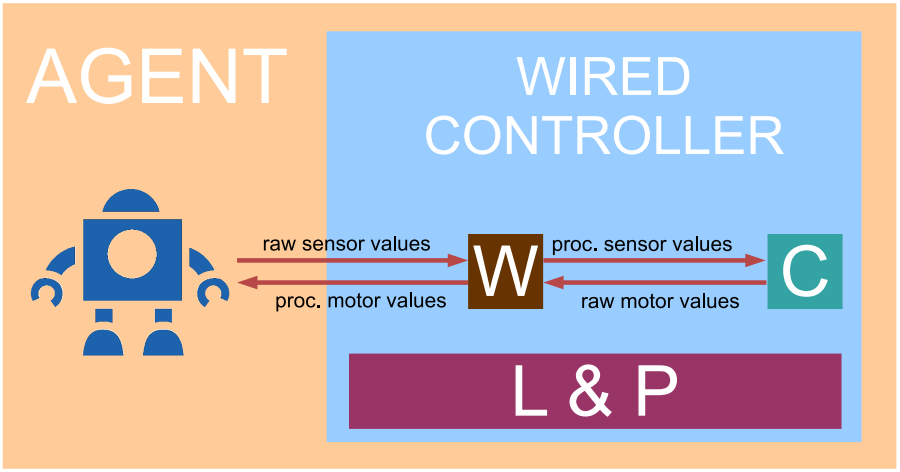


Figure 2.18: Architecture of a robotic agent in the LpzRobots simulator.

2.5.2 Sensors

Sensors are components of a robot that allow it to sense the environment and own stance. The proprioceptive sensors measure internal states of the robot, while the exteroceptive sensors measure external variables from environment. The combination of all the possible states of the robot's sensors is defined by the state space, a n-dimensional manifold. Sensors that are usually available in robots are:

- **Joint position**: These sensors supply the actual position of the joint (rotation or displacement) and its derivative gives the linear or angular speed. It can measure acceleration as well. This is the single most used sensor for the realisation of the self-organised robots. They are chosen because they can provide the full state of the robot stance and relative speed.

- **Orientation**: They provide the full orientation matrix (3×3) of an object, the vector of one internal axis or the projection of the internal axes onto the z-coordinate of the global coordinate system.

- **Infrared**: Sensor that issue the distance to an object. These sensors emit a light beam in the infra-red spectrum and measure the reflectance. The amount of reflectance is directly proportional to the distance of the origin of the beam and the reflected object. In the simulated infra-red sensor, for the maximal distance 0 is returned, and when the object is in the closest range a 1 is returned.

- **Microphone**: A simple microphone able to measure frequency, amplitude or direction of the sound source. In the simulator, this sensor does not take into account reflection or damping of sound due to other objects in the scene.

- **Camera**: This sensor renders a video stream. Usually, one picture is captured at each control time step. Values like resolution and field of view can be manipulated. Also, preprocessing of the image such as colour filters, edge detection, or others can be defined.

2.5.3 Motors

Motors are used to manipulate the stance of the robot and the environment. These actuators apply force or torque to the joints that connects two parts of the robot. The capacity of a robot to modify its surrounding environment is often pointed out as the difference with an agent. On fully actuated robots, each *degree of freedom* has a motor to modify its position. These changes can be dictated by a new position or velocity of the joint. Also, passive joints can have motors that only react to external forces, e.g., springs. The most common active motors in robotics are:

- **Angular motor**: This motor controls the angular velocity of the joint. The action parameter is the maximum torque that can be applied to the joint. In a single simulation time step the desired velocity is reached as long as the maximal torque is not exceed. This motor is simulated with a look-ahead algorithm in order to optimise its stability.

- **PID servo**: It is a Proportional, Integral and Derivative (PID) controller for a servo motor (Rivera et al., 1986). The motor values provide the nominal position of the joint and the force or torque f_t to be applied to reach the desired position:

$$f_t = P\left(e_t + D\dot{e}_t + I\sum_{\tau=0}^{t} e_\tau\right),$$

where e is the position error, i.e., nominal position minus current position, and $\dot{e}$ is the derivative of the error with respect to time. P, D and I are the gain factors. The implementation in the simulator is slightly different to the usual. In this case, P scale the entire force. Thus, the total power of the motor can be adjusted. The D factor scales the derivative term. This derivative term constrains the system in order to avoid overshooting. For such reason, it is also called the damping factor. The I factor accounts for the accumulation of errors, e.g., an offset due a constant external force.

- **Servo motor**: The motor values specify the nominal position of the joint, from which the velocity v_t is computed:

$$v_t = Ve_t, \tag{2.19}$$

where V is the maximal velocity of the servo and e_t is the position error. Proportionality between the error and the velocity avoids overshooting. The maximal force that the motor can apply to reach the velocity is:

$$f_t = F \tanh\left(1/C + |e_t|\right), \tag{2.20}$$

where F is the global maximal torque of a motor and $C > 0$ is a compliance term. If $C \ll 0$, then the output term does not depend on the error term. This leads to a stiff system without compliance. In this case, only little feedback from the physical system is used for computing the force. If $C > 1$, then the motor has little power at the target position, thus it behaves compliant. Forces acting on the joint will produce small deviations from target position that will be deflected by the action of the motor similar to an elastic system of muscle and tendons.

2.6 Discussion

In this chapter, we have presented self-organisation and guided self-organisation as a way to study the phenomena that robots undergo when their behaviour is intrinsically motivated. The development that a self-organised robot achieves exploiting embodiment is a desirable property. Our approaches to autonomous robots depend of self-organisation in different manners. As we present in Chapter 4, the resulting behaviour allows for a coherent exploratory mode and seemingly purposeful bearing. In the rest of this work, we utilise self-organised behaviour in combination with goal-oriented learning mechanism with successful results.

Our approaches towards autonomous robots are based on how the internal models affect the overall learning process. As we showed in this chapter, IMs cover several roles in robotics. IMs can be used as predictors, auxiliary signal generators, driver of artificial curiosity or for own configuration representation. Besides their functionalities, we are interested in how the complexity of the internal models regulates the overall performance of autonomous robots. Thus, in the next chapter, a study of various implementations of IMs are tested.

We have presented a wide variety of dynamics that the sensorimotor loop exhibit. Even in a simple configuration, with a singular dimension, the dynamics are complex and diverse. We have neglected noise, external disturbance and stochasticity on purpose as way to understand the basic dynamics. In general, noise acts as a small amplitude disturbance, allowing the system to start the working regime by escaping from the fixed point as a result of noise amplification. On the other hand, external disturbances are responsible for *catastrophic* state change. As well, when considering noise, the transitions between different dynamics become irregular. For example, following the bifurcation diagrams of Figure 2.13, the clear change between dynamics when r varies, becomes a less clear conjunction of fixed points when noise is present. Systems with such dynamics can remain in these areas without deciding for any of the paths of the bifurcation, as suggested in the noise-less dynamics. In (Der and Martius, 2012) a more detailed analysis is portrayed including these features. All these dynamics achieved in a simple set-up show the importance of fine tuning of the control parameters. Such tuning is non-trivial. In Chapter 3 we present two self-organisation approaches that adapt the control parameters towards intrinsic motivated behaviour.

The last section of the chapter introduced the robot simulator that we use to evaluate our robotics control approaches. Among a whole range of simulators available in the market, LpzRobots simulators is the most suitable for our experiments. The variety of tools present in the software allows us to understand the behaviour of the robots as well the evolution of their internal structures. The simulator presents a realistic implementation of the complete robotics architecture. This realistic implementation plus the use of dynamical systems as a way to understand both real and simulated interactions, make our experiments valid tests of our methods.

In the next chapter, we present two approaches to self-organisation of robotic behaviour. The first one is homeokinesis (Der, 2001), a control paradigm able to bring about coherent behaviour in high-dimensionality robots without expert knowledge. The second one is our approach to artificial curiosity in the sensorimotor loop.

Chapter 3

Self-Organisation of Behaviour

In this chapter, we present two approaches to self-organisation (SO) of behaviour. The central idea of this SO paradigm is that the actions taken by an autonomous robot obey local rules generating coherent behaviour. Such behaviour emerges spontaneously from intrinsic motivations. There is no high-level orchestrator controlling the degree of freedom (DoF) of the system. Rather than that, the local rules and experience from interaction with the environment bring about comprehensible behaviour. In the first approach, homeokinesis (Der, 2001), an agent tries to act both in a *predictable* and *sensitive* way. In the second approach, we propose the use of *artificial curiosity* into the sensorimotor loop for high-dimensional robots. We show the properties and results of both approaches and why we use homeokinesis as the main driver of exploration in our research.

3.1 Introduction

As mentioned in the Chapter 1, classical approaches to artificial intelligence (AI) based on knowledge base, were successful in specific and limited size domains. In extensive domains, only human operated robots have been successful (Sian et al., 2006; Tsui and Yanco, 2006). Controlling a robot to perform in an open and changing environment remains as a challenge to modern AI (Kemp et al., 2007).

In an effort to tackle these challenges, robots are no longer programmed to respond to every situation. On the contrary, they have to be able to change its internal configuration in order to adapt to the environment. Instead of having a full set of pre-defined commands, the behaviour emerges from a set of few local rules that shape the parameters of the robot based on its interaction with the environment. The spontaneous emer-

gence of some form of coordination or overall order from an initial disordered system is called *self-organisation*. Using this principle in the sensorimotor loop, robots can be controlled in such a way that they adapt to its environment; they can learn from its actions and respond effectively to unforeseen events. The *homeokinetic* controller (Der and Martius, 2012) produces coherent behaviour using exploitation of embodiment, i.e., the dynamics arise due to the interaction of the body and the environment. The emergent behaviour is a result of embodied self-organisation. A homeokinetic robot tends to maintain a dynamic regimen, probing different behaviours that are both predictable and sensitive. The resulting behaviour is coherent to the configuration of the robot and the response from the environment. The consistent probing of action and state space makes the result favourable for exploration. In this chapter, the exploratory characteristic is presented with several examples from low to high-dimensional systems. Also, in Chapter 4, we present how different internal model can shape the behaviour of autonomous homeokinetic robots. In Chapter 5, we utilise homeokinesis in various guided self-organisation tasks.

In this chapter, we present another method for emergent behaviour. This approach maximises the learning progress. Such rule has been used in dynamical systems before (Herrmann et al., 2000; Oudeyer et al., 2007; Schmidhuber, 2010), and termed *artificial curiosity*. A system that maximises the learning progress is driven to unknown yet able to learn states. The system presents *interest* toward situations that imply a gain of its internal representation. The system, after acquiring all the relevant information from such situation, loses interest and move to a new situation. We propose the use of this measure in the sensorimotor loop to produce emergent curious behaviour. In Section 3.3, we show how behaviour is promoted by the curiosity learning rule and its feasibility for developmental robots.

In the next section, we present the homeokinetic controller, its dynamics properties and how the learning rules defines the behaviour.

3.2 Homeokinetic Robot Control

Homeokinesis "propose a principle that accounts for the generation of playful and coordinated behaviour without specific goals" (Der and Martius, 2012). From a sensorimotor loop point of view, homeokinesis keeps the dynamics at the edge-of-chaos. The system fluctuates between inactivity and hyperactivity or between predictability and sensitivity. The work in (Der, 2001; Der and Liebscher, 2002), proposed the abstract

concept of homeokinesis. In that work, the energy function *time-loop error* is used to update the parameters of the controller. Later, a multidimensional version of the controller was presented in (Der and Martius, 2012). In the approach, the sensorimotor dynamics is regulated by the internal learning dynamics (Section 3.2.2). These rules allow symmetry breaking to set in, resulting in emergent self-driven dynamics. The robot behaves in a comprehensible way, including smooth movements that respond to irregularities in the environment. The embodiment nature of the controller makes it appropriate as a goal-less exploration.

3.2.1 Prediction and Time-Loop Error

In the close-loop homeokinetic control, the actions generated by the controller depend on the sensory information perceived by the robot. Figure 3.1 presents the flow of information within the loop. The controller K, in dependency of the actual state $\mathbf{x}$, produces the motor command $\mathbf{y}$ that will be actuated by the robot. The resulting state $\mathbf{x}$ are compared to the predicted ones $\hat{\mathbf{x}}$ and used to updated the values of K and the forward model M. To formalise the sensorimotor loop, the homeokinetic controller $K : \mathbb{R}^n \to \mathbb{R}^m$ is defined as a parametrised function,

$$\mathbf{y}_t = K(\mathbf{x}_t; C), \tag{3.1}$$

with $\mathbf{y} \in \mathbb{R}^m$ and $\mathbf{x} \in \mathbb{R}^n$, and C the parameters of the function. Motor commands can represent various quantities such as velocity or position of a joint. The sensors can represent proprioceptive signals, e.g., the angle of a joint or its derivative. We present a list of sensor and actuators in Section 2.5.2 and 2.5.3.

Assuming a static world W and Markovian settings, the trajectory of the sensor values can be represented as a simple dynamical system,

$$\mathbf{x}_{t+1} = W(\mathbf{x}_t, \mathbf{y}_t), \tag{3.2}$$

where the next time step sensory states depends on the actual commands and sensor state. Following Equation 3.1 and 3.2, the sensorimotor loop can be written in closed form as

$$\mathbf{x}_{t+1} = W(\mathbf{x}_t, K(\mathbf{x}_t; C)). \tag{3.3}$$

The W function represents the environment and all its interactions. In practice, this function is unknown to the system. Nevertheless, the system can build a representation

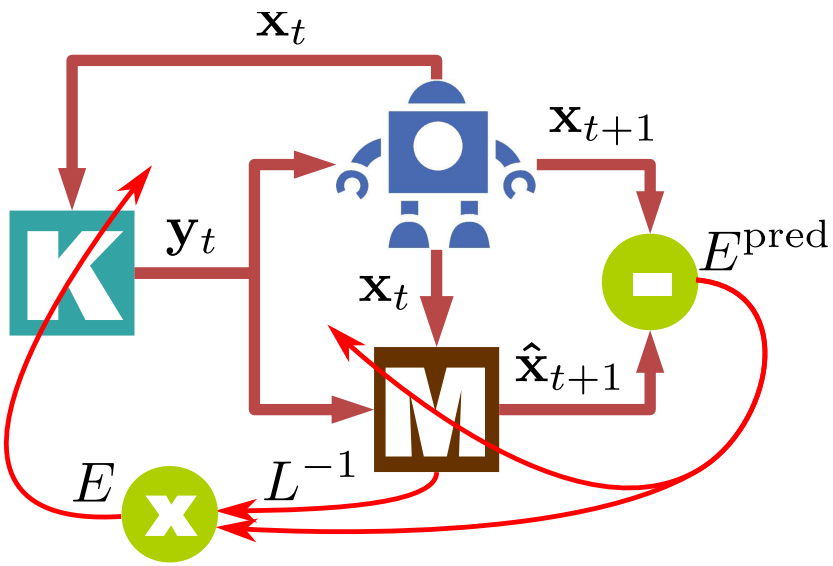


Figure 3.1: Information flow in the homeokinetic sensorimotor loop. The thick orange lines represent the flow of motor commands $\mathbf{y}$ and sensory information both real $\mathbf{x}$ and predicted $\hat{\mathbf{x}}$. Thin red lines represent the errors flow used to update the parameters of the model M and controller K.

of the world based on the experience of its actions and consequences. An internal forward model $M : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is defined as,

$$\hat{\mathbf{x}}_{t+1} = M(\mathbf{x}_t, \mathbf{y}_t; A), \tag{3.4}$$

with A the internal parameters of the models. This model produces a prediction of future time states $\hat{\mathbf{x}}_{t+1} \in \mathbb{R}^n$ for the actual sensory information $\mathbf{x}_t$ and actions $\mathbf{y}_t$.

The difference between actual and estimated states can be calculated as,

$$\boldsymbol{\xi}_{t+1} = \mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1}, \tag{3.5}$$

where $\boldsymbol{\xi}$ accounts for the stochastic component of the sensorimotor loop. The stochastic component can be related to inherent noise in the data acquired by the sensors, or to interactions not captured by the internal model. Now, the real sensor input can be described in terms of the model in sensor space:

$$\mathbf{x}_{t+1} = M(\mathbf{x}_t, K(\mathbf{x}_t; C); A) + \boldsymbol{\xi}_{t+1}. \tag{3.6}$$

The *prediction error* is defined as,

$$E_t^{\text{pred}} = \|\boldsymbol{\xi}_t\|^2 = \boldsymbol{\xi}_t^\top \boldsymbol{\xi}_t, \tag{3.7}$$

and is used to update the parameters of the model in order to minimise the mismatch in the prediction. A smaller prediction error implies a better prediction of the model. If the controller were updated with the premise of reducing the prediction error, then the robot behaviour will end up in highly predictable states. These states can be characterised in two ways. The first mode is an active yet repetitive behaviour. In this behaviour, the robot still performs some type of behaviour but repetitive. The second mode is a fixed position where the sensory input does not change over time. These behaviours correspond to homoeostasis, where all the variables are maintained at certain intervals (Ashby, 1960, p.91).

In (Der and Liebscher, 2002), autonomous robots are driven away from trivial behaviours by the inversion of time in the modelling process. Similarly, homeokinesis describe the *time-loop error* (TLE). The TLE is based on the amount of necessary change in the sensor input to compensate for the prediction error. By defining the internal representation of the sensorimotor loop in sensor space as $\psi(\mathbf{x}_t) = M(\mathbf{x}_t, K(\mathbf{x}_t))$, the input shift $\boldsymbol{\eta} \in \mathbb{R}^n$ is:

$$\psi(\mathbf{x}_t + \boldsymbol{\eta}_t) = \hat{\mathbf{x}}_{t+1} + \boldsymbol{\xi}_{t+1}. \tag{3.8}$$

Using a Taylor expansion, the LHS of Eq 3.8 can be written as,

$$\psi(\mathbf{x}_t + \boldsymbol{\eta}_t) = \psi(\mathbf{x}_t) + L_t\boldsymbol{\eta}_t + O(\boldsymbol{\eta}_t^2), \tag{3.9}$$

and following Equation 3.4 the prediction error can be expressed as,

$$\boldsymbol{\xi}_{t+1} = L_t\boldsymbol{\eta}_t, \tag{3.10}$$

where L is the Jacobian matrix of the system. L is defined as (with indices as matrix positions and time at t):

$$L_{ij} = \frac{\partial \psi_i}{\partial x_j}. \tag{3.11}$$

If the inverse of L exists, the input shift can be expressed as

$$\boldsymbol{\eta}_t = L_t^{-1}\boldsymbol{\xi}_{t+1}. \tag{3.12}$$

Now the TLE can be defined as,

$$E_t = \|\boldsymbol{\eta}_t\|^2 = \|L_t^{-1}\boldsymbol{\xi}_{t+1}\|^2 = \boldsymbol{\xi}_{t+1}^\top (L_t {L_t}^\top)^{-1}\boldsymbol{\xi}_{t+1}. \tag{3.13}$$

The homeokinetic controller updates the internal parameters in order to minimise E. By examining Equation 3.13 we can already have an idea of what type of behaviour is favoured by the minimisation of the TLE. First, the TLE is directly proportional to the

prediction error $\boldsymbol{\xi}$. Thus, highly predictable behaviours will be favoured by the minimisation of the TLE. With a large prediction error, the controller will tend to change the internal parameters, thus changing the actual behaviour of the robot. Second, the TLE is inversely proportional to the Jacobian. The matrix L can be explained in a more intuitive way as a measure of *sensitivity*. The term sensitivity describes how the changes in the sensory input affect the state of the system over time. Hence, it is expressed as the derivative of the loop function. In the sensorimotor loop, an increase in sensitivity leads to activity. In Figure 3.1, the red lines represent the TLE and E^{pred} updating the parameters of the controller and the forward model respectively.

Following the concept presented in Section 2.1, self-organisation can arise from the interaction of two opposing forces. In the case of the homeokinetic controller, the two interacting forces are *predictability* and *sensitivity*. The antagonistic nature of both forces drives the system to an active exploration at the edge of chaos. On one hand, predictability tends to dampen the system into a stationary state. On the other hand, sensitivity tends to lead to unpredictable behaviour due to the amplification of small perturbations. Since the controller is maximising both quantities, the resulting bearing of the robot is a compromise. The emerging behaviour of the robot produced by the controller results in an exploratory mode based on the embodiment.

So far, the controller has been defined in terms of the TLE. This definition is independent of the implementation. The implementation of the model can depend on any process that follows the learning rules. Still, different implementations can yield different behaviour characteristics. The study of such differences goes beyond the scope of this thesis. In the next section, we present in detail the canonical and explicit learning rules for homeokinetic standard settings.

3.2.2 Homeokinetic Canonical Learning Rule

In machine learning, the adaptation of parameters is often described by a gradient flow on an energy surface or *error function*. A gradient descent algorithm can be used to decrease the TLE. Writing $\Delta p_t = p_{t+1} - p_t$ for any parameter $p \in \mathbb{R}$, the gradient descent learning rule is defined as,

$$\Delta p_t = -\varepsilon \frac{\partial E_t}{\partial p_t}, \tag{3.14}$$

where ε is a learning rate. An iteration of the parameter adaptation will decrease the error to at least a local minimum, considering an appropriate small learning rate. Now,

defining the auxiliary vector

$$\boldsymbol{\chi}_t = (L_t^\top)^{-1}\boldsymbol{\eta}_t, \tag{3.15}$$

and following Equations 3.12 and 3.13, the TLE can be written as:

$$E_t = \boldsymbol{\chi}_t^\top \boldsymbol{\xi}_{t+1}. \tag{3.16}$$

The gradient step is presented in terms of the derivative of the Jacobian matrix w.r.t. any parameter p of the controller as (Der and Martius, 2012):

$$\Delta p_t = \varepsilon_C \boldsymbol{\chi}_t^\top \frac{\partial L_t}{\partial p_t}\boldsymbol{\eta}_t, \tag{3.17}$$

where ε_C is the controller's learning rate. Details on the derivation of this rule can be found in Appendix A.1. The Equation 3.17 is the central rule for the parameter dynamics. It is the responsible for generating the self-determined development of robot behaviours.

If the matrix L is non-singular, then the derivation of the rule is straightforward. Otherwise, the use of pseudoinverse or the addition of a small random matrix to the Jacobian L avoid singularities. Since the matrix L depends on the internal model, the controller also depends on the model. The parameters not only change by action of the prediction error E^{pred}, but also as on how sensitive the system is to sensory perturbations. The most common settings for homeokinesis is a linear controller and a linear model with gradient descent learning rules of the TLE and E^{pred} over the respective parameters. Nevertheless, in this thesis the standard settings are expanded to new forward models. The suitability to generate self-organised behaviour of linear and non-linear models is assessed in Chapter 4. Next, the update rules are derived from the standard settings.

3.2.3 Standard Settings Explicit Learning Rule

The standard homeokinetic settings comprise a linear controller and a linear model that only depends on motor signals. The linear controller $K : \mathbb{R}^n \to \mathbb{R}^m$ is defined as

$$\mathbf{y}_t = K(\mathbf{x}_t) = g(C\mathbf{x}_t + \mathbf{h}), \tag{3.18}$$

where the matrix $C \in \mathbb{R}^{m\times n}$ is part of the controller's parameters and the vector $\mathbf{h} \in \mathbb{R}^m$ act as bias. The sigmoidal function $g : \mathbb{R} \to \mathbb{R}$ is defined as $g(\cdot) = \tanh(\cdot)$ and is applied element-wise. The function deals with the non-linearities confining the results to the range $(-1, 1)$.

The linear model $M : \mathbb{R}^m \to \mathbb{R}^n$ defined in motor space is

$$\hat{\mathbf{x}}_{t+1} = M(\mathbf{y}_t) = A\mathbf{y}_t + \mathbf{b}, \tag{3.19}$$

where $A \in \mathbb{R}^{n\times m}$ and $\mathbf{b} \in \mathbb{R}^n$ represent the parameters and bias of the model. The dynamics model in sensor space loop can be written now as

$$\psi(\mathbf{x}_t) = Ag(C\mathbf{x}_t + \mathbf{h}) + \mathbf{b}. \tag{3.20}$$

Following Equation 3.11, L can be calculated based on Equation 3.20 explicitly as

$$L(\mathbf{x}_t) = AG'_tC, \tag{3.21}$$

where G'_t is the diagonal matrix:

$$G'_{ij} = \delta_{ij}g'(z_i) = \delta_{ij}g'_i(\mathbf{z}_t), \tag{3.22}$$

with δ_{ij} the Kronecker delta (indices i and j showing matrix and vector position) with

$$\mathbf{z}_t = C\mathbf{x}_t + \mathbf{h}.$$

The next step is to derive the explicit rule for the C_{ij} parameters using Equation 3.17. Consider

$$\boldsymbol{\chi}_t^\top \frac{\partial L_t}{\partial C_{ij}}\boldsymbol{\eta}_t = \boldsymbol{\chi}_t^\top AG'_t\frac{\partial C}{\partial C_{ij}}\boldsymbol{\eta}_t + \boldsymbol{\chi}_t^\top A\frac{\partial G'_t}{\partial C_{ij}}C\boldsymbol{\eta}_t.$$

Defining the auxiliary vector:

$$\boldsymbol{\mu}_t^\top = \boldsymbol{\chi}_t^\top AG'_t, \tag{3.23}$$

the first term on the RHS of the equation becomes

$$\boldsymbol{\mu}_t^\top \frac{\partial C}{\partial C_{ij}}\boldsymbol{\eta}_t = \sum_{kl}\mu_k\frac{\partial C_{kl}}{\partial C_{ij}}\eta_l = \mu_i\eta_j. \tag{3.24}$$

To evaluate the last term, G' has to be considered as a diagonal matrix:

$$G'_t = \mathrm{diag}[g'(z_1), \cdots, g'(z_m)] = \mathrm{diag}[g'_1, \cdots, g'_m],$$

obtaining

$$\frac{\partial z_k}{\partial C_{ij}} = \delta_{ik}x_j \;\text{ and }\; \frac{\partial g'_k}{\partial C_{ij}} = \delta_{ik}g''_i x_j,$$

then

$$\boldsymbol{\chi}_t^\top A\frac{\partial G'_t}{\partial C_{ij}}C\boldsymbol{\eta}_t = \left(\boldsymbol{\chi}_t^\top A\right)_i g''_i\,(C\boldsymbol{\eta}_t)_i x_j.$$

For the hyperbolic tangent activation function, we have $g_i'' = -2g_i g_i' = -2y_i g_i'$, now the previous expression can be written as:

$$-2\left(\boldsymbol{\chi}_t^\top A G_t'\right)_i (C\boldsymbol{\eta}_t)_i y_i x_j = -2\mu_i \zeta_i y_i x_j, \tag{3.25}$$

with

$$\boldsymbol{\zeta}_t = C\boldsymbol{\eta}_t r. \tag{3.26}$$

Finally, according to Equation 3.17, replacing it with Equation 3.24 and 3.25 the learning rule for the parameters in the matrix C is:

$$\Delta C_{ij} = \varepsilon_C \mu_i \eta_j - 2\varepsilon_C \mu_i \zeta_i y_i x_j, \tag{3.27}$$

with $i = 1,\ldots,m$ and $j = 1,\ldots,n$. When the learning rate ε_C is chosen small enough, the controller's parameters change on a time scale similar to the time scale change of the state. This similarity brings about behavioural flexibility and exploration of the behavioural space.

In the same way, the update rule for the bias parameters $\mathbf{h}$ can be expressed as

$$\Delta h_i = -2\varepsilon_C \mu_i \zeta_i y_i, \tag{3.28}$$

with $i = 1,\ldots,m$.

Simultaneously, the parameters of the linear model (Equation 3.19) are updated via gradient descent over the prediction error E^{pred},

$$\frac{\partial E_t^{\text{pred}}}{\partial A_{ij}} = \frac{\partial \boldsymbol{\xi}_t^\top \boldsymbol{\xi}_t}{\partial A_{ij}} = 2\boldsymbol{\xi}_t^\top \frac{\partial \boldsymbol{\xi}_t}{\partial A}_{ij}, \tag{3.29}$$

using $\frac{\partial}{\partial X}\mathbf{a}^\top\mathbf{a} = 2\mathbf{a}^\top\frac{\partial}{\partial X}\mathbf{a}$ for the differentiation of the square norm. Then

$$2\boldsymbol{\xi}_t^\top \frac{\partial \boldsymbol{\xi}_t}{\partial A_{ij}} = 2\sum_{kl}\xi_k \frac{\partial}{\partial A_{ij}}\left(x_k - (A_{kl}y_l + b_l)\right) = -2\xi_i y_j. \tag{3.30}$$

Following Equation 3.14 with the prediction error E^{pred} to update the parameters of the models, the learning rule is:

$$\Delta A_{ji} = \varepsilon_A \xi_j y_i, \tag{3.31}$$

with $i = 1,\ldots,m$ and $j = 1,\ldots,n$. The learning rate ε_A captures the factor 2, and all the times are in t except for the y values at time $t-1$. In a similar way, the update rule for $\mathbf{b}$ can be derived as

$$\Delta b_j = \varepsilon_A \xi_j, \tag{3.32}$$

with $j = 1,\ldots,n$.

3.2.4 Dynamics of the Homeokinetic Controller

To show the dynamics of the homeokinetic controller, we present a toy version of the sensorimotor loop with $n = 1$ and $m = 1$. The dynamical system is defined as $x_{t+1} = \alpha y_t + \varsigma_t$. Here, we assume $\alpha = 1$ as the hardware constant and ς as a zero-mean noise process. The parameters of the world model $M(y_t) = ay_t + b$ will approach $a = 1$ and $b = 0$ after successful learning.

The learning rule for the only controller parameter now reads

$$\Delta c = \tilde{\varepsilon}_t \left(1 - 2cyx\right), \tag{3.33}$$

where $\tilde{\varepsilon}_t = \varepsilon_C a \xi^2 L^{-3} G' > 0$ is a time-dependent learning rate (Martius, 2010). The fixed points can be found considering the dynamics of the sensorimotor loop in sensory space. Using the linear controller $K(x_t) = \tanh(cx_t + h)$, the dynamics can be expressed following Equation 2.14,

$$x_{t+1} = \alpha \tanh(cx_t + h). \tag{3.34}$$

In the same way as Section 2.4.2, we set $h = 0$ and solve the Equations 3.33 and 3.34 system for fixed points. The results, $c \approx 1.19$ and $x \approx \pm 0.648$, spans the membrane potential $z = cx + h \approx \pm 0.77$. There is a neglected solution for $c < 0$, as it corresponds to a dynamics with alternating sign of the motor values at each time step. According to Equation 2.15, the system response is $r = c\alpha = c$. In this case, r is moderately above the bifurcation point. A system with such characteristic, presents shifts between stable fixed points by both small external perturbations (Figure 2.13a) or small changes in h (see Figure 2.13b and 2.14).

Under these same assumptions the learning rule for the bias term is

$$\Delta h = \tilde{\varepsilon}_t \left(-2cy\right). \tag{3.35}$$

The dynamics of h acts in the opposite direction of the membrane potential z because the terms c, and $\tilde{\varepsilon}$ are positive and $y = g(z) = tanh(cx + h)$ has the same sign as z. Hence, averaged over noise, the relation between signs holds:

$$\operatorname{sgn}(\Delta h) = -\operatorname{sgn}(z). \tag{3.36}$$

The membrane potential z has essentially the same dynamics than x, since $c \approx 1.2$ and $h \ll 1$ in this configuration. Consequently, the dynamics of x will follow the dynamics of the membrane potential as it has been described in Figure 2.14 along the arrows. A computer simulation is shown in Figure 3.2 where the hysteresis effect on x can be appreciated.

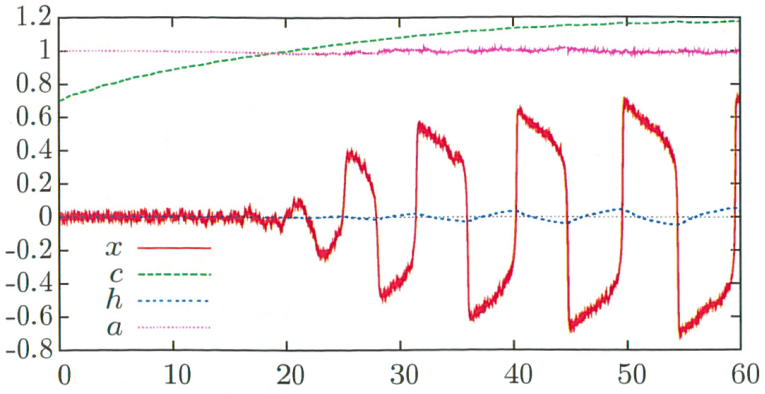


Figure 3.2: State and parameter dynamics in the one-dimensional system. The parameter c grows until reaching a value close to 1.19; the bias parameter h oscillates around 0 causing x to jump between fixed points (see Figure 2.14). The error is drawn from white noise with $\varsigma \in (-0.02, 0.02)$, and learning rate is set to $\varepsilon_C = 1$. Image taken from (Martius, 2010)

.

3.2.5 Homeokinesis in Robotics Systems

In this section, we present examples of low and high dimensionality robots. First, we show a low dimensionality system where the dynamic properties of homeokinesis can be observed by examining the changes in the parameters and signals at the sensory level. After that, we study examples of high-dimensionality systems. In such systems, the emergence of coordinated behaviour among the several degrees of freedom is the main characteristic.

3.2.5.1 Low-Dimensionality Dynamical System

The pendulum has been largely used as an example in various control task domains. Such system includes a simple set-up and has well-known motion equations. These characteristics allow a complete understanding of the dynamics of the system. Nevertheless, it exhibits a rich range of possible behaviours and non-trivial dynamics. In reinforcement learning, the swing-up pendulum has been used as a base example because its dynamics (Doya, 2000). It has also been used in other self-organising approaches as proof of concept (Salge et al., 2013). The fair complexity and the low dimensionality of the pendulum make it a well-suited example in order to visualise results of the homeokinetic controller.

A pendulum is an object of mass m attached trough an arm of size l to a pivot so

it can swing freely. On the pendulum, two main forces act on it. First, a torque T that correspond to the motor commands y, and second, the force of gravity $F = -mg$ that corresponds to the only external force (the environment) acting upon the pendulum. In this case, the torque is applied by an angular motor located at the pivot. The motor command is the desired angle with maximum applicable force bounded to $T^{\max} < mg$ (see Section 2.5.3 for details on servo motors). Thus, in order to get to the highest position, the system has to gain momentum. Such constraint makes the robot sensitive to the environment and its configuration. Note that the motor has friction μ built in, which will take the system to the rest position in case that no torque is applied. The state of the system is measured by sensors in the pivot joint with angle θ and the angular velocity ω (see Figure 3.3). To avoid the singularity at $\theta = \pm\pi$, θ is transformed into Cartesian coordinates θ_x and θ_y for the homeokinetic controller.

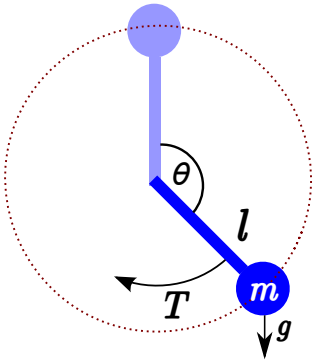


Figure 3.3: A pendulum, mass attached to a pivot allowing it to swing freely. Values for the simulations are mass $m = 1$, arm length $l = 1$, pivot friction $\mu = 0.01$, gravity $g = 9.8$, maximum torque $T^{\max} = 5$, $\varepsilon_C = \varepsilon_A = 0.1$.

Initially, the system is idle, the controller learning rate is set to $\varepsilon = 0$ and the pendulum is positioned in the resting state at maximum extension ($\theta = \pm\pi$). Before time $t = 7$ (marked with the arrow in Fig 3.4) the controller parameters remain at $C = 0$ and some noise can be seen in the measure of ω (θ_x and θ_y not shown). At the time marked by the arrow, the learning rate is changed to $\varepsilon_C = 0.1$. Immediate changes in the controller parameters can be observed, and an amplification of the noise is transformed into an increase in ω. The first effect to notice is that the system is starting to move. The controller captures small disturbances that arise from noise (at $t = 0$ the motor command is $y = 0$, i.e., resting position). Homeokinesis amplifies these weak signals (Equation 3.18). A bootstrap effect occurs, small disturbances yield limited motor command and in turn this motor command produce increasing change in the

sensor state.

Initially, only sub-critical values for the feedback strength of the sensorimotor loop exists. The influence of noise is damped, and only small fluctuations in ω are observed. The learning dynamics increase the values of the controller parameter C and the pendulum movement becomes stronger. Eventually the bifurcation point (as in Figure 2.10) is reached, and some irregular oscillatory motion appear. After a while, the pendulum has been able to accumulate momentum and can reach the highest position ($\theta = 0$). The embodiment takes control and some interesting behaviours can be distinguished (Figure 3.5). The emergence of these behaviours is attributed to the sensitisation paradigm; based on the current sensor values, these modes maximise the change in the sensor values over the time step. These modes are preferred by the physical system since they oscillate at the eigenfrequencies. The system has a better prediction by being more stable to perturbations, and also it behaves in a high active mode. In Figure 3.5a, the pendulum is swinging in a back and forth mode. Even at some points of higher velocity it can make a full turn (after time $23:15$, $23:20$, and so on). In Figure 3.5b, another behaviour can be observed. The pendulum performs several turns in the same direction following the bias parameter h (scaled in the figure). This mode of behaviour, where the pendulum accelerates and decelerates systematically, was first presented in (Hamed, 2007) and then verified by (Martius et al., 2007).

The pendulum alternate between these modes, exploring the behavioural and state space. The actual explored state space is presented in Figure 3.6. The radius of the polar plot represents the angular velocity and the angle is the position of the pendulum. Only clockwise, i.e., positive angular velocity, values are shown. Counter-clockwise angular velocity renders a vertically mirrored figure. The initial exploration occurs at $\theta = \pm\pi$ and $\omega \approx 0$. Later on, the self-amplification effect allows the system to "escape" from the resting position entering an oscillatory mode. The inscribed circle in the bottom half of Figure 3.6) shows this initial exploration. The exploration continues and the full range of angles is visited, including full turns at maximum speed. Such behaviour is represented by the external circumference of the same figure. The pendulum has an unstable fixed point at $\omega = \theta = 0$. Remarkably, this point and its vicinity are visited even regarding its repelling nature. Note that the controller has not been instructed to perform any particular task but to minimise the prediction and TLE errors. Behavioural space has been probed by the pendulum even when the controller has not been designed to explore such space directly.

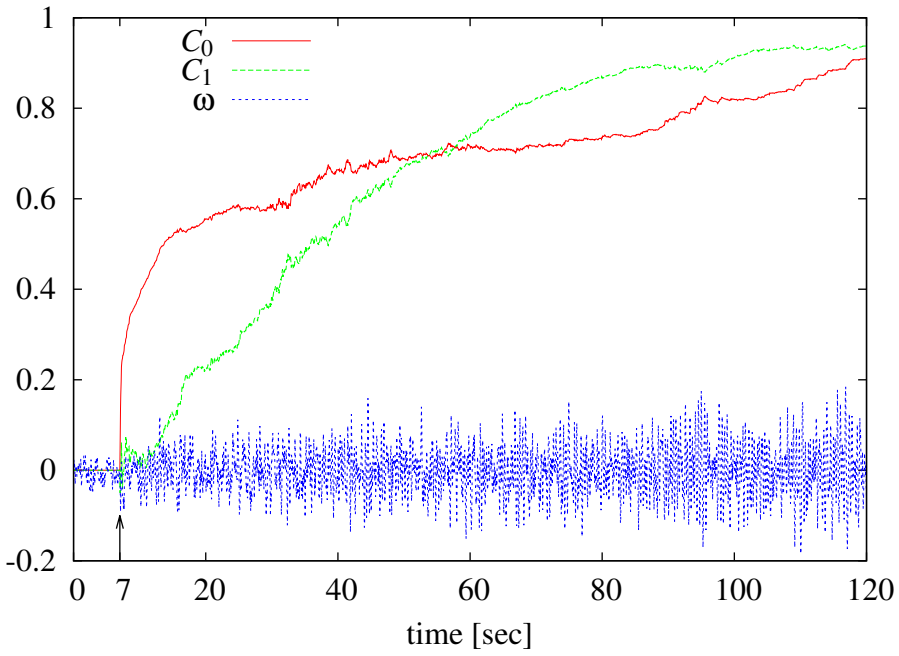
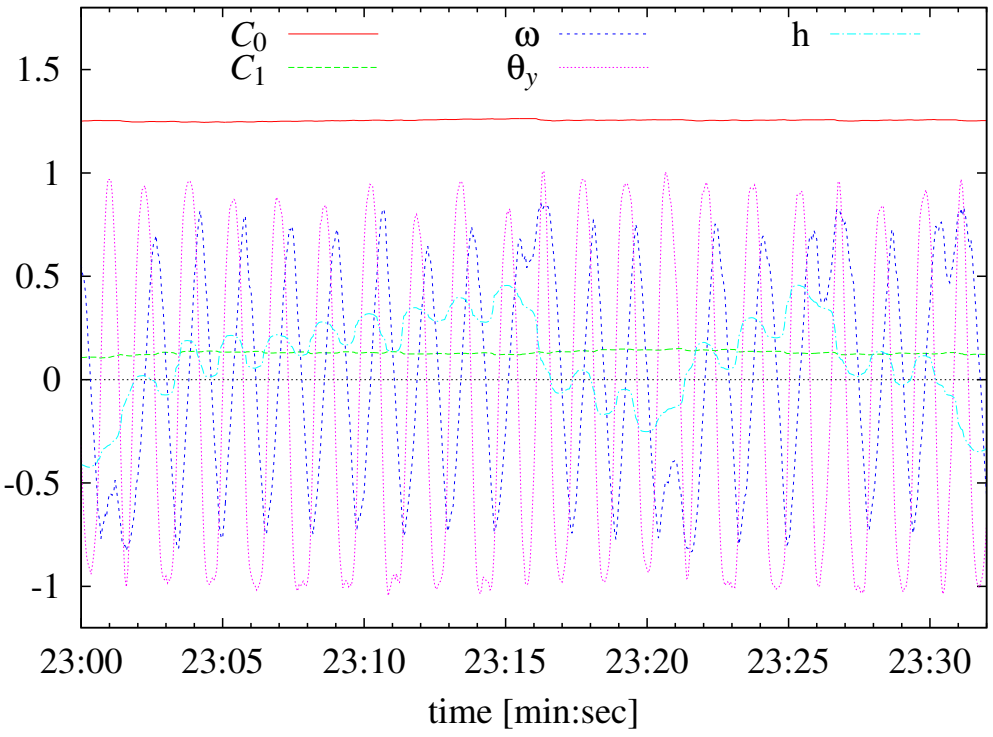
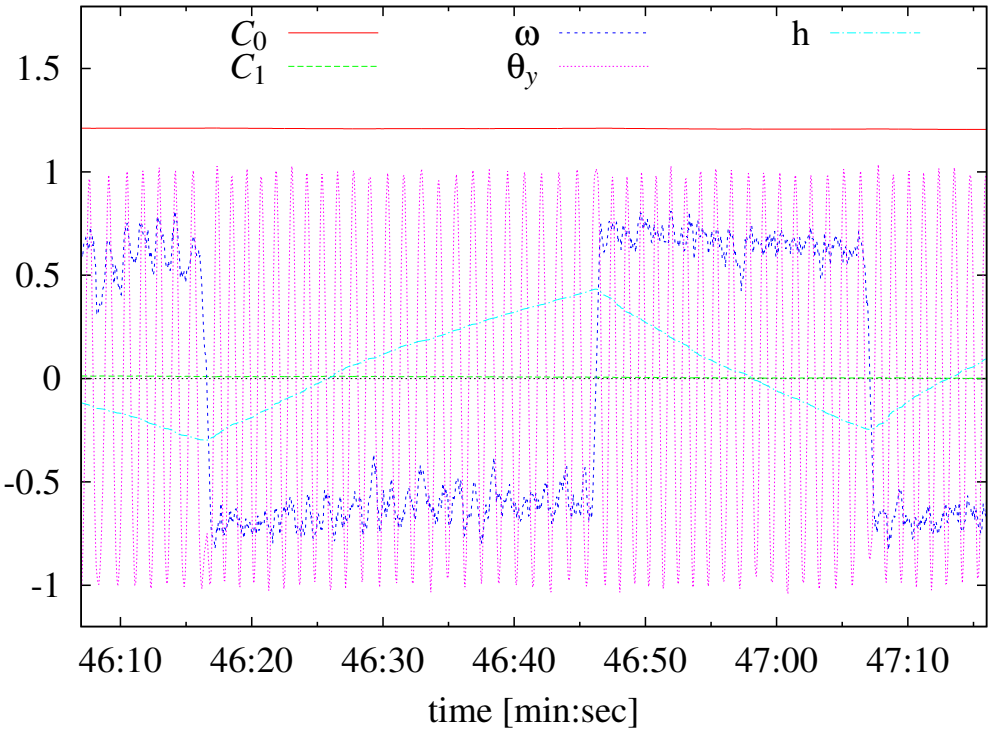


Figure 3.4: Before the arrow, the learning is turned off. Noise is present in the measure of the angular speed ω. At the arrow, the learning rate is changed to $\varepsilon_C = 0.1$, the controller parameter C changes and the pendulum starts to move back and forth (increase in absolute value of ω).



(a)



(b)

Figure 3.5: Two different behaviours for the pendulum. The embodiment has taken control and oscillatory modes emerge. In (a) the pendulum is swinging but able to make full turns only sporadically. In (b) the pendulum turns around the pivot several times until changing direction, the angular velocity follows the bias parameter h (scaled for visualisation in both figures).

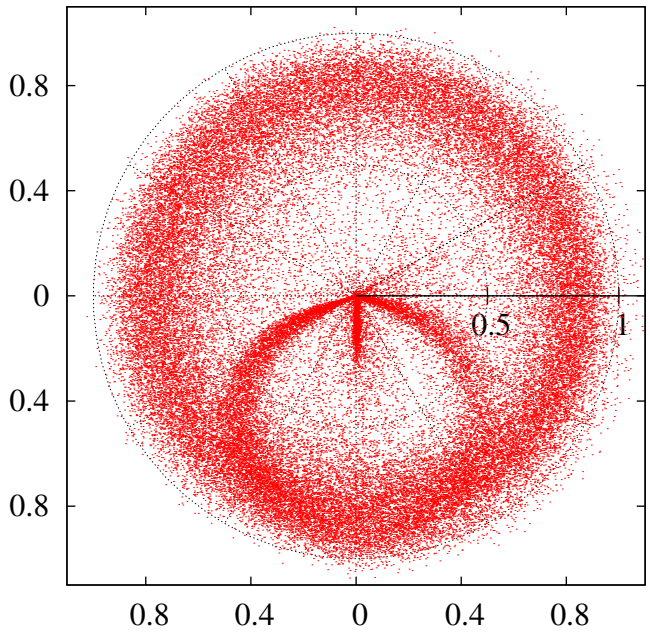


Figure 3.6: Visited state space (positive angular velocity $\omega \geq 0$ in the radial coordinate, θ in the angular coordinate) for the pendulum. The initial exploration close to the resting position ($\omega = 0$ and $\theta = \pm\pi$) is highly visited, as well as the forth and back movement before reaching full turns. Note the slight shift to the right due to gravity.

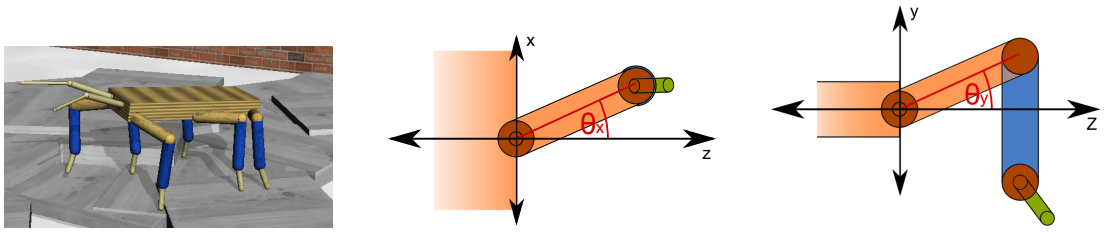
3.2.5.2 High-Dimensional Systems

One of the characteristics of homeokinesis is that it can control high-dimensionality complex dynamical systems. Homeokinesis is able to deal with numerous degrees of freedom and a vast action space. In this section, high-dimensional robots controlled by homeokinesis are presented to show how robots can enter into functional resonance with the environment making use of their specific physical properties.

Compelling behaviours in high-dimensional robots have been described in detail in (Der and Martius, 2012). For example, a chain of two-wheeled cars independently controlled by homeokinesis and only joined by the front bumper to the rear bumper of the next car shows spontaneous cooperation. The chain of cars navigate through a maze without any proximity sensor and only reacting to collisions. As another example, a couple of humanoid robots, also independently controlled by homeokinesis, are locked inside a small room. In such small environment, the robots are very likely to interfere with each other. The final interactions have been described as wrestling like behaviour. Note that in this set-up, the robots only feel each other by the changes in their sensors. In the same work, homeokinesis has been proposed as an emergency mode, e.g., to explore a whole range of motor commands that could rescue the robot from a stuck position. A trapped robot (by the action of another controller) can be set free from its confinement by the exploration of new possible scenarios.

As an example, we consider control of walking behaviour on a hexapod (Figure 3.7). The hexapod robot has six legs and 12 DoF. Each leg has two independent motors that enable the movement in parallel with respect to the torso and vertical lift of the leg. Each joint position is measured in two dimensions according to its two degrees of freedom. Compliant actuators, i.e., actuators that allow deviations from its equilibrium position, are used in each leg. When the simulation is started, the weight of the trunk of the hexapod pushes downward not allowing the actuators to react to the environment. Thus, the robot is in a non-sensitive state. The controller escapes from this situation by changing the values on the controller towards sensitivity. Slowly, the legs start to drive the system out of equilibrium. Amplitude increases and eventually coordinated movement between legs emerges. This coordination with both in-phase and anti-phase correlations between legs are necessary for different gaits. The emergent coordination is non-trivial: there is no direct exchange of information at the physical level nor between motor neurons; the correlations are established by the emergence of cross-channel couplings in the C matrix of the controller. Such coordination is possi-

ble even in high-dimensional systems in homeokinesis. Each motor neuron becomes sensitive to every input captured by the sensors. A handful of behaviours can bee seen in the hexapod with coordinated walking gaits. Among these behaviours, there is a tendency to move in the direction of the antennae based on body asymmetry. On an interesting behaviour, if a small barrier surrounds the hexapod, it will be capable of surmounting it. Note that the hexapod has not been guided to any of these behaviours or locomotions. The robot constantly spans from one behaviour to another, e.g., when in contact with the barrier it may linger around until ensuingly trespass it.



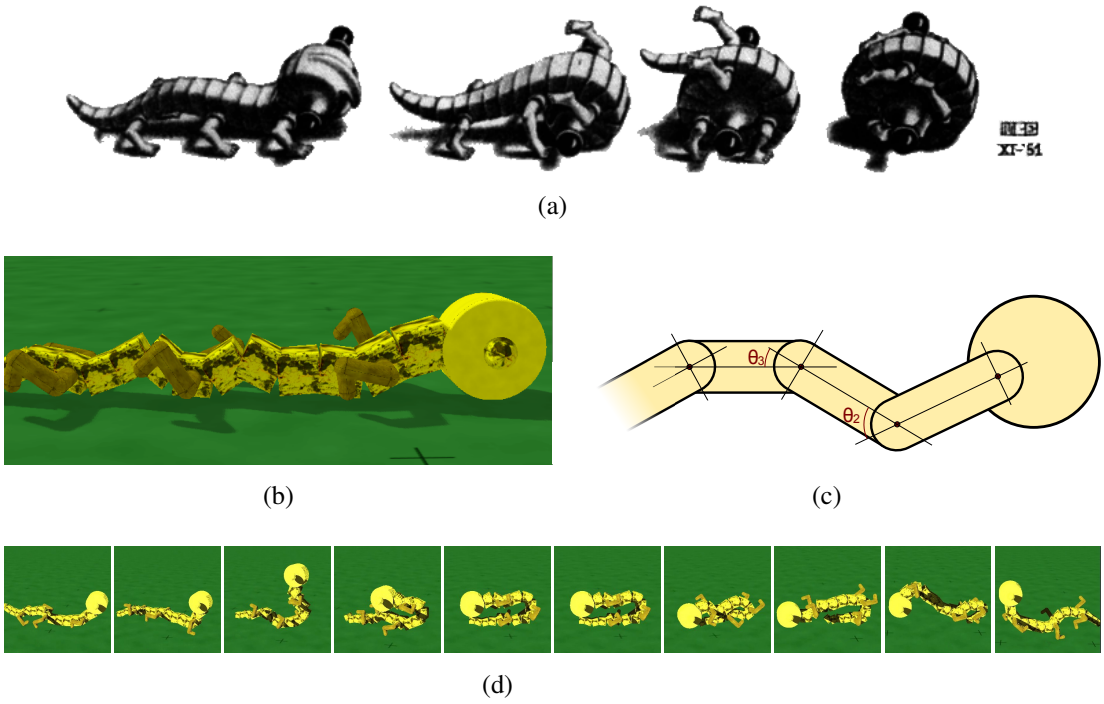
(a) LpzRobots simulator (b) Top view (c) Front view

Figure 3.7: Hexapod robot. A 12×12-dimensional system. The simulated version (a) adapted from (Martius, 2012). Figs. (b) and (c) show the angles measured at each joint. A passive spring is attached to each tarsus.

We control another high-dimensional robot with homeokinesis. The wentelteefje, which is based on the lithography by M. C. Escher, also called *Curl-up* (Figure 3.8). This imaginary creature has a round shaped head with a long trunk and tail able to fold into itself to engage a rotating mode. Six legs span from the body and they can rotate at the hips pivot [1], or in a secondary configuration, they are controlled in the same way as the hexapod. The body, including the tail and head, has 12 DoF. Combining the legs, the state space has as much as 18 dimensions. As a way to favour the learning process in high-dimensional systems, an external perturbation free approach has been proposed in (Der and Martius, 2012). Before the robot can enter into a more operative behaviour, i.e., in resonance with physical oscillations, it is hanged it in the air. There is no other interaction than the one among the DoF. In such case, the robot actualises the internal parameters in unperturbed legs and spine, much in the spirit of a scaffolding of the development process. Another way to proceed with high-dimensional systems is to initialise the C and A matrices to a scaled identity matrix. In this case, there are no initial cross-correlations between channels, and the feedback strength is close to a

[1] The precise mode of translation has not been described in the original work by M. C. Escher.

critical point, so the amplification of noise is already relevant. Again some interesting behaviour appear, and only by embodiment the robot is able to curl-up backwards. The full movement begins by lifting up his head and then moving it in the direction of the tail until reaching and surpassing it. The horizontal symmetry (excluding the feet, Figure 3.8d) allows him to deploy all his body again and continue with another behaviour, e.g., a walking gait.



(a) (b) (c) (d)

Figure 3.8: (a) The wentelteefje, adaptation from the original lithography © M. C. Escher Foundation. (b) Computer simulated high-dimensional wentelteefje robot. (c) Schematic view of the robot; each joint of the trunk and tail measure the relative angle with respect to the next component. (d) Emergent behaviour of the robot controlled only by homeokinesis.

In the next section, we present our approach to SO of behaviour based on artificial curiosity.

3.3 Self-Organisation of Curious Behaviour

In the previous section, we presented an approach to self-organisation of behaviour based on the maximisation of predictability and sensitivity. In this section, we propose another approach, based on the maximisation of learning. The idea is to drive the

robot to unknown situations that can be learned by an internal model. After learning, i.e., when no more improvement of the model is possible, the process is repeated in a new situation. Such behavioural rule has been termed *curiosity*, i.e., a drive towards novel and interesting stimuli. In (Herrmann et al., 2000), a representation of the environment is made by maximisation of the information gain by an agent. Such rule is preferred over a prediction increase in order to avoid trivial behaviours. In (Oudeyer et al., 2007), curiosity has been used to promote learning as intrinsically motivated behaviour. In a reinforcement learning set-up, the reward is proportional to the decrease in the prediction error. In the review (Schmidhuber, 2010), the author goes even beyond at formalising creativity, fun and intrinsic motivation with a similar approach. In such formalisation, an individual pursues situations that are unknown but familiar to previously visited ones. The individual drive itself towards unforeseen situations that yield high probabilities to be acquired by a compressor (a forward model). The last two mentioned approaches base the emergence of behaviour in agents by the *intrinsic reward* in a reinforcement learning controller. In the sensorimotor loop, a behaviour can be learned if the state prediction error is minimal. If the robot is driven only to unknown states, i.e., maximisation of the prediction error, it could be trapped in states that cannot be learned. For example, random signals or dynamically chaotic states that are hard to grasp by the internal models. The forward IM can help to avoid states that cannot be learned by driving the controller towards states that minimise the prediction error. We propose to use curiosity in the sensorimotor loop to foster coherent behaviour in autonomous robots. Our approach updates the parameters of the controller in order to drive the robot to states that maximise the learning rate of the IM. Here, the learning rate is the difference between the actual and previous prediction error.

The idea of curiosity and spontaneous exploration has been studied in developmental psychology. *Intrinsic motivation* pushes animals to explore the environment by the development of curiosity (Ryan and Deci, 2000). Such motivation has been argued as an essential part of cognitive growth and organisation in animals, and has been widely studied (Berlyne, 1960; Csikszentmihalyi and Csikzentmihaly, 1991; Deci and Ryan, 1985; White, 1959). Intrinsic motivation has been adapted for developmental robotics (Barto et al., 2004; Oudeyer and Kaplan, 2007), but not within the sensorimotor loop for real-time autonomous robots. In the next section, we present our approach to curiosity learning in the sensorimotor loop.

3.3.1 Curiosity Measurement

We introduce the implementation of artificial curiosity in terms of on-line learning in the sensorimotor loop. The implementation follows the same architecture presented in Section 2.4.1, Figure 2.7. The architecture is based on a closed-loop with a forward model that predicts future states. Following Equation 3.7, we define the learning gain as

$$D_t = E_{t-\theta}^{\text{pred}} - E_t^{\text{pred}}. \tag{3.37}$$

Note that $E^{\text{pred}} > 0$. The actual prediction error E_t^{pred} is compared to previous time $t-\theta$ prediction error. We want the robot to be driven to behaviours where D is maximal. A positive value of D comprises a decrease in the E^{pred}, i.e., a learning gain. Thus, we use a gradient ascending over the linear controller parameters (Equation 3.18):

$$\Delta C_{ij} = \varepsilon_C \frac{\partial D_t}{\partial C_{ij}},$$
$$\Delta h_i = \varepsilon_C \frac{\partial D_t}{\partial h_i},$$

with ε_C as the learning rates. To derive the learning rules for the C parameters, we separate the first derivative:

$$\frac{\partial D_t}{\partial C_{ij}} = \frac{\partial}{\partial C_{ij}} \left(E_{t-\theta}^{\text{pred}} - E_t^{\text{pred}} \right). \tag{3.38}$$

Now we focus on the derivative of the actual time, using the chain rule with matrix notation:

$$\frac{\partial E_t^{\text{pred}}}{\partial C} = \frac{\partial E_t^{\text{pred}}}{\partial \boldsymbol{\xi}_t} \frac{\partial \boldsymbol{\xi}_t}{\partial \hat{\mathbf{x}}_t} \frac{\partial \hat{\mathbf{x}}_t}{\partial \mathbf{y}_{t-1}} \frac{\partial \mathbf{y}_{t-1}}{\partial C}. \tag{3.39}$$

The firs term of the RHS of the equation, following Equation 3.7, is $2\boldsymbol{\xi}_t$. The second term is the unity. The third and fourth terms are obtained following Equations 3.20 and 3.21. Thus, the derivative of Equation 3.39 is:

$$\frac{\partial E_t^{\text{pred}}}{\partial C} = 2\boldsymbol{\xi}_t A G' x_{t-1}. \tag{3.40}$$

Finally, defining the auxiliary vector $\boldsymbol{\mu}^\top = \boldsymbol{\xi}^\top A G'$, the learning rule for the C parameters is,

$$\Delta C_{ij} = \varepsilon_C \left(\mu_{i,t-\theta} x_{j,t-\theta-1} - \mu_{i,t} x_{j,t-1} \right), \tag{3.41}$$

with $i = 1, \ldots, m$, $j = 1, \ldots, n$ and absorbing the scalar into ε_C. In a similar way, the update rule for the bias parameters is:

$$\Delta h_i = \varepsilon_C \left(\mu_{i,t-\theta} - \mu_{i,t} \right), \tag{3.42}$$

with $i = 1, \ldots, m$.

The update rules for the parameters of the forward model remains the same as in homeokinesis (Equations 3.31 and 3.32). The prediction error is minimised at every time step . Both rules decreases the prediction error: the minimisation is direct in the forward model update rules and indirect in the controller rules; the maximisation of the learning gain implies a reduction of the prediction error, as shown in Figure 3.9. If the only goal of the controller is to reduce the prediction error, a possible outcome is a stationary behaviour. To avoid this situation, we include an exploratory factor $f = \frac{1}{|D|}$ in the learning rule. Now, the learning rules are:

$$\Delta C_{ij} = f\varepsilon_C \left(\mu_{i,t-\theta} x_{j,t-\theta-1} - \mu_{i,t} x_{j,t-1}\right), \tag{3.43}$$

$$\Delta h_i = f\varepsilon_C \left(\mu_{i,t-\theta} - \mu_{i,t}\right). \tag{3.44}$$

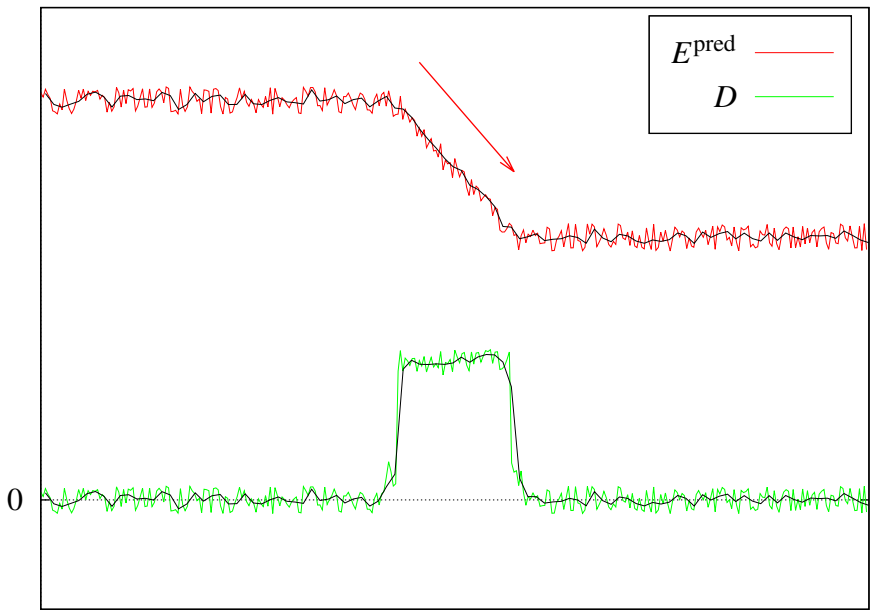
The f factor will promote exploration when the learning gain is minimal. The controller has a term that allows to escape from regular working regime like static or constant behaviour.

Another consideration is taken into account. The noisy nature of the sensory information could mislead the calculation of the learning gain. Especially when the learning gain is small compared to the noise of the signal. Also, the actions taken by the robot does not have an immediate effect on the sensory signal. This delay is related to the physical restrictions of actuators and sensors, and also by limits of the system such as physical resonance or momentum gain. Hence, we modify the calculation of learning gain to include the average of the prediction error,

$$D_t = \langle E^{\text{pred}}_{t-\theta} \rangle - \langle E^{\text{pred}}_{t} \rangle, \tag{3.45}$$

where $\langle \cdot \rangle$ denotes a sliding temporal average with time constant τ. In Figure 3.9, the black lines represent an average of both errors.

Independent of the size of θ, a large value of τ will average over too many time steps. The results of such average is a flat learning gain as dissimilar behaviours are taken into account. A small value of τ with a large θ will also average over distinct behaviours. A combination of small τ and small θ favour the measure of the learning gain. The small θ accounts for the measure of related values and the small τ averages out the noise without over generalising behaviours. The size of the values are dependent on the interaction level of the robot with the environment. In our test, a homeokinetic robot stay in the same behaviour from a couple of seconds to several minutes depending on the learning rates and the environment. In our experiments, we



(a)

Figure 3.9: Prediction error E^{pred} and learning gain D. The controller drives the robot towards a learning gain maximisation, depicted with the red arrow. The maximisation implies a reduction of the error. In order to avoid stagnation, the exploration factor f has been included. The black lines represent a smooth version of the quantities due to average.

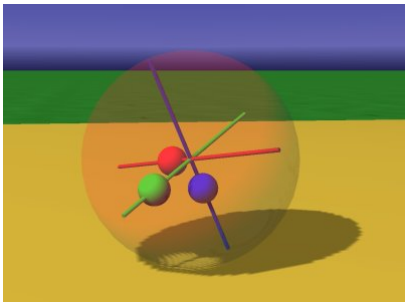
average between 10 to a 100 measures that are taken between every time step to every 10, i.e., $\tau \in [10:100]$, $\theta \in [1:10]$ with time step $\in [0.01:0.05]$.

To test the approach, we compare the results with two different forward models implementations. Also, the outcome is compared to homeokinesis in terms of variability of the emergent behaviour.

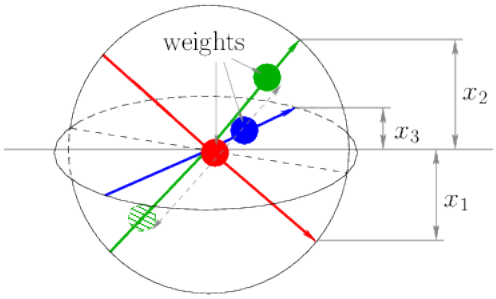
3.3.2 Curiosity in the Sensorimotor Loop

In order to test the controller, we use a spherical robot (Figure 3.10a). It has a ball-shaped body and is equipped with three internal masses. The positions of each mass are controlled by a servo motor ($m = 3$). The actuators value define the nominal position of the mass along the axis. A value of zero stands for a centred position and -1 and 1 correspond to the outer positions. In our configuration, each mass has the same weight as the hull itself. Sensors measure the axes-orientation ($n = 3$, Figure 3.10b). For each axis, the projection of its direction onto the z-component of the world coordinate

system is measured. To account for the different behaviours, we measure the rotation of each mass axis. Note that the controller only *senses* through the axes-orientation sensors, while the rotational sensors are only used for behaviour extraction.



(a) Screenshot from the LpzRobots simulator. The red, green and blue masses are moved along the axes by the actuators.



(b) Schematic view of the robot with axis-oriented sensors x_i.

Figure 3.10: The spherical robot with axis oriented sensors x_1, x_2 and x_3. Rotational sensors are included in each mass axis (not present in the image). Images taken from (Martius, 2010).

We tested our approach with two internal forward models. The first one, a linear model (Equation 3.18), same as standard homeokinesis configuration. The second implementation of the internal model is realised by an *Echo-State Network* (ESN) (Jaeger, 2001). The ESN is a *recurrent neural network* with a dynamic repository that allows the inclusion of non-linearities. In this network, only the output weights are updated. Thus, only linear time is needed for learning. This network plays a significant role in Chapter 4, as a fast and recurrent model for homeokinesis. A more detail view of ESN is presented in Section 4.2.2.1.

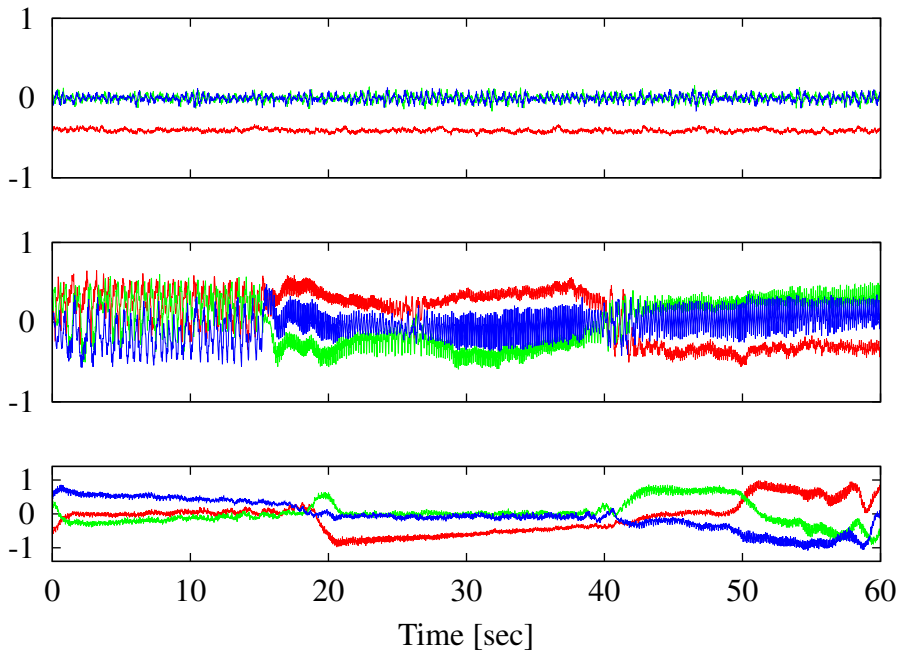
First, we tried our approach without the exploratory factor f. The robot quickly finds a rotational behaviour over one of the axes. Such bearing can be seen in the top image in Figure 3.11. Only one of the axis (red) has a large rotational speed while the other two do not rotate over the internal axis, only in relation with the sphere. This behaviour is a result of the learning rules, no modification has been made to the robot. Note that the masses still change positions with respect to their axes but this movement does not produce rotation on their respective axes. The overall behaviour of the robot is a constant displacement, rotating only over one axis. This result shows that the learning rule is not capable by itself to escape from local optima. When the exploratory factor is introduced, the behaviour changes. This modification of the behaviour can be seen in

the middle image in Figure 3.11. As the time progress, the internal models produce a smaller prediction error until the controller is not able to escape from a static position. With both internal models, the activity dies out after a couple of minutes. However, when the ESN is used there are more behaviours explored and the robot remains active for a longer period. The ESN allows for a smoother prediction error minimisation, inducing a larger change in the parameters of the controller.

We compare the curiosity approach with homeokinesis. The bottom image of Figure 3.11 presents the behaviour of the robot when controlled by homeokinesis. Behaviours with more regular angular rotation emerge. The robot changes behaviour even when the prediction error is minimal. Sensitivity drives the robot away from highly predictable states, i.e., static ones. The most important difference from homeokinesis with respect to curiosity is that it can maintain the critical dynamics in the long run.

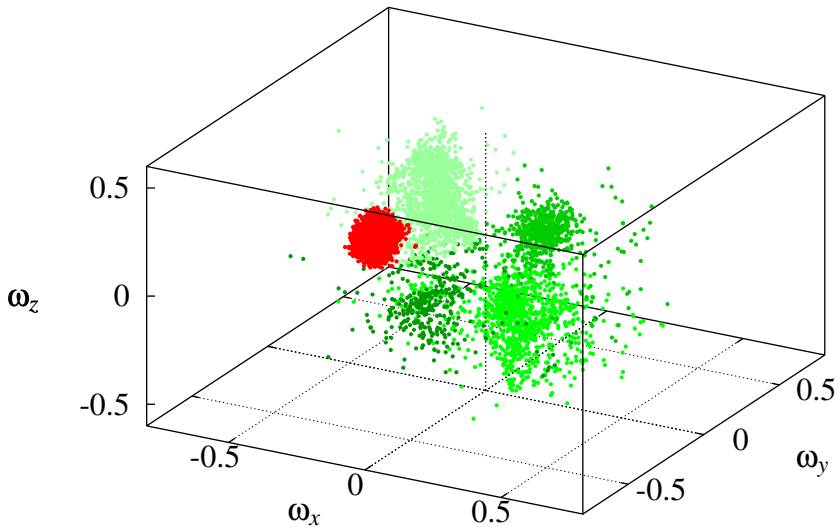
To characterise the behaviours even further, we plot the rotational speed that yield low E^{pred} in Figs. 3.12 and 3.13. In the first figure, the clusters correspond to the curiosity controller. The red cluster represents the behaviour of the robot without exploratory factor. The green clusters have been obtained with exploration. The plotted data correspond to low prediction error behaviour. In the second figure, the clusters correspond to the homeokinetic controller. For the same simulation time, more low prediction error behaviours are found when compared to curiosity. Also, a better use of the state space is achieved by homeokinesis. The curiosity exploration leads the robot away from low learning gain without further objectives. In contrast, homeokinesis drives the robot to behaviours with high sensitivity. This type of exploration gives better results for coherent emergent behaviour. Because of this characteristic, homeokinesis is better suited to bring about self-organised behaviour, and we use it as an exploratory paradigm for the rest of the work presented in this thesis.

This approach requires better policies to escape stagnation. The exploratory factor f, proved to be not suitable for this task. At small D and E^{pred}, the changes on the parameters are insignificant. The study of other exploration modes, e.g., noise or a modified version of sensitivity, remains as part of the future research of artificial curiosity in the sensorimotor loop.



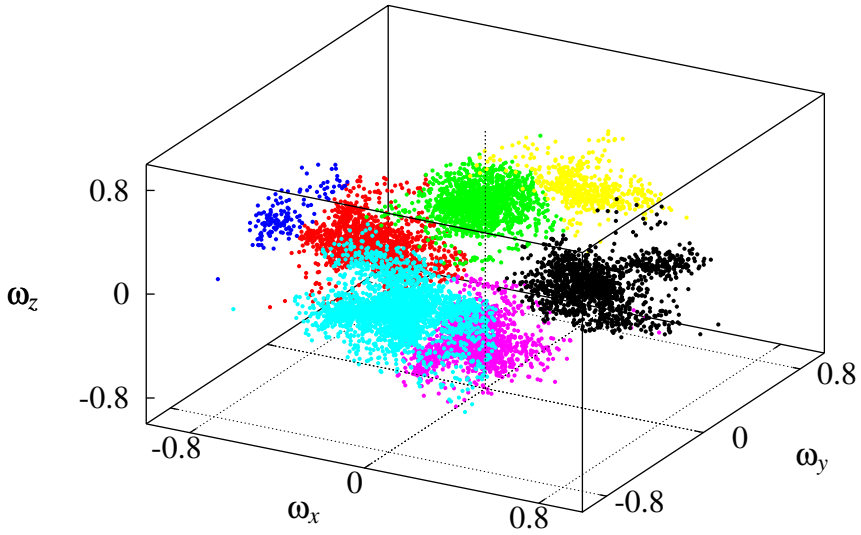
(a)

Figure 3.11: The plots represent the rotational speed of each axis of the spherical robot. The top image is the behaviour of the curiosity controller without exploratory factor. The robot remains in the same mode as he is unable to change the parameters of the controller. The middle image represents the behaviour of the ESN model including the exploratory factor. Such behaviours are only maintained temporarily before stagnation. The bottom image represents homeokinetic behaviour. The simulation parameters were $\varepsilon_C = \varepsilon_A = 0.01$, $\theta = 1$, $\tau = 10$, ESN reservoir size = 50.



(a)

Figure 3.12: Behavioural clusters generated by curiosity learning rules. The red cluster shows a controller without exploratory factor. The green clusters represent behaviours found when exploration is active. The limited variety of behaviours, and the stagnation of the system makes the curiosity driven controller not suitable for autonomous robotic exploration.



(a)

Figure 3.13: Clusters of behaviour obtained with homeokinesis. When compared to the clusters obtained by the curiosity controller (Figure 3.12), a better use of the state space is made and a wide variety is obtained within the same time. Also, the capacity of homeokinesis to avoid stagnation makes it a better exploratory controller for autonomous robots.

3.4 Discussion

In this chapter, we have presented two different approaches for self-organisation of behaviour. However, an underlying principle for SO of behaviour remains as an open question. Emergent coordination (spatial, temporal or spatiotemporal) respond to different causes depending on the system and the environment. The effect of two opposing forces is present transversely: an antagonistic reaction counteracts the self-intensification. In (Prokopenko, 2014), intelligent behaviour is expected to be predictable and stable, but sensitive over change. At the model presented in (Gierer and Meinhardt, 1972), several types of SO like patterns formation and morphogenesis emerge. In the approach, an autocatalytic substance or activator is responsible for increasing the activator concentration. In turn, the production of the activator is slowed down by the effect of an inhibitor. The *free energy principle* (Friston, 2010), tries to explain how biological systems maintain their order by restricting themselves to a limited number of states. According to this theory, self-organised systems maximise model evidence (minimise prediction error) by choosing actions that minimise free energy. That is, at the same time the system reduces energy while maximising entropy. In homeokinesis, it is also present a mechanism where two contrary forces interact. Sensitivity and predictability drive the system towards coherent behaviour. Sensitivity intensifies the noise in the sensory channel or the effect of external disturbances resulting in movement. When the robot is in motion, sensitivity aids in the emergence of behaviour that resonates with its physical oscillations. The unbounded increase in sensitivity would drive the system to a chaotic behaviour. However, the antagonistic force, i.e., predictability, confines the behaviour to those that can be internalised by the model. Since the prediction of chaotic behaviour is not trivial for an internal model, the robot is driven away from such situations. In the same way, the complexity of the internal model, characterised here as its capacity to predict a wider range of behaviour, also determine the exploration of the behavioural space. In Chapter 4, a set of different internal models are implemented and tested in an exploratory setting to assess how they interfere behaviour.

What is interesting within the scope of this thesis is the capability of homeokinesis to generate exploratory behaviour. The resulting behaviour is directly related to the physical configuration of the robot and its interaction with the environment. Part of this thesis aims to show that an educated exploration, like homeokinesis, is more effective than a random one (Chapter 5). Nevertheless, there are other approaches to

emergent, intrinsically motivated, self-organised behaviour in robotics. Evolutionary robotics (Nolfi and Floreano, 2000), can generate coordinated emergent behaviour using genetic algorithms. These behaviours respond to some specific and designed fitness function specialised in solving one task, narrowing the range of possible behaviours. Also, the nature of genetic algorithms is not compatible with real-time developmental robots. A reinforcement learning scenario with reward as an intrinsic motivation towards movement has been tested in robotics (Marshall et al., 2004; Chentanez et al., 2004). The results have longer learning times compared to homeokinesis and suffer from the curse of dimensionality in complex physical systems. Another approach that generate coordinated behaviour with similar time response as homeokinesis is based on the maximisation of predictive information in the sensory channel (Martius et al., 2013). Another approach with similar characteristic and from the same authors is based on normalised Hebbian learning in the sensorimotor loop (Der and Martius, 2015).

In our approach to artificial curiosity in the sensorimotor loop, the autonomous robot maximises its learning progress. The presented results show a less dynamic behaviour when compared to homeokinesis (Section 3.3). The robot is not able to maintain a continuous change in the behavioural space. The maximisation of the learning progress includes a minimisation of the prediction error. Thus, the behaviour is driven to stagnation. Artificial curiosity has been used successfully in RL settings (Oudeyer et al., 2005), where the system is restarted, inducing exploration. In our approach, we favour a behaviour that is continuous during the life of the robot. Homeokinesis delivers a more comprehensible control over complex systems. The system operates in a regime that is somewhere between the fully deployed complexity and a more or less ordered state (Der, 2003). Due to this ability to control complex dynamical system, we use it as an exploratory controller to assess how internal models can influence behaviour.

The homeokinetic adaptation rules include the minimisation of the prediction error. Thus, the model plays a significant role in the control of the agents. The model restrain the robot to enter chaotic behaviour by tampering sensitivity. It is of interest to understand how different internal model complexity can mould the spectrum of behaviours in the sensorimotor loop. In the next chapter, a set of experiments is run with different implementations of internal models.

Chapter 4

Self-organisation of Internal Models

In the previous chapter, it was shown how the homeokinetic controller can produce coherent behaviour based on the embodiment. This controller relies on a forward model of the environment to shape the behaviour of the robot. For the model, independently of the higher level task, i.e., exploration, the main goal is to predict the next state of the sensory input. A minimisation of the prediction error tends to maintain the system in a predictable state. Such states depend on the prediction capability of the model. Following this view, a model should be able to drive the system to a more complex behaviour if it can predict it. In this chapter, we assess the relation between the homeokinetic controller performance and the complexity of the internal model (IM).

4.1 Introduction

Internal models can differ in various ways. For example, different memory capacity, learning algorithms or use of non-linearities. These characteristics determine the complexity of the model. On one hand, we can have a model that comprises a large set of artificial neurons with recurrent connections. On the other hand, we can utilise a simple linear model with a few artificial neurons. The first model has an increased computational complexity and representational power than the second one. The recurrent network has a larger size and memory capacity and is also able to handle a plethora of scenarios, e.g., non-linear ones. The optimal representational power of a model, i.e., the number of computational elements and number of parameters, lays between maximal *expressivity* and minimal *overfitting*. Expressivity accounts for the breadth of states that can be represented by the model. The more expressive a model, the greater the variety and quantity of behaviours that can be represented. Models are supposed to

extract general information, not just snapshots of the environment. Overfitting occurs when a model describes random error or noise instead of the underlying relationship. This mismatch can arise when a model is excessively complex, such as having too many parameters relative to the number of observations. Thus, representational power should only increase as long as it provides better expressivity and no overfitting. More differences arise when the updating rule of each model is taken into account. Two models, with the same configuration, can vary their performance depending on a more or less suitable learning algorithm.

We are interested in assessing how the internal model can shape the behaviour of homeokinetic robots. The maximisation of the coverage of the state and action space, under consistent behaviour, is a desirable characteristic for developmental robots. Consider a robot that can explore its environment thoroughly and to come up with more complex but still coherent behaviour. The robot can benefit from those behaviours to attain higher order goals. The information gathered while exploring can be used before or on-line with respect to the search of the main objective. These ideas on control are studied in Chapter 5.

One of the points we address is the relation between the complexity of the model and the complexity of the environment. It is expected that a simple environment favour a simple model. A simple environment produces less variability in the state space than a complex one. Thus, a model with smaller representational power can provide enough computational complexity to acquire the necessary range of behaviours. Nevertheless, for non-trivial environments, the complexity of the domain is not easily measurable, so the optimal complexity of a model has to be measured indirectly. Already in (Der and Martius, 2012), a pseudo-linear model is preferred for self-organised behaviour. We expand the linear model with a recurrent neural network (ESN, Section 4.2.2.1), expecting the broad configuration to bring better results in more complex environments. The extended configuration should account for non-linearities that the linear model is unable to tackle, but maintaining the fast adaptation due to the characteristic linear learning rule of the ESN.

There are certain characteristics that are shared by all models. Every IM needs to adapt some internal parameters to learn the required mapping. For example, a linear model adapts its regression coefficients. Likewise, a neural network adapts the weight of the connections between neurons. Parameter adaptation is compulsory to deal with unforeseen events on developmental robots. Even a prolifically crafted statistical model cannot cope with complex system interacting with an intractable environ-

ment. Another mandatory characteristic is on-line learning. For an on-line controller as homeokinesis, the learning occurs with each new sensory input. To cope with a changing environment, the received information has to be processed as promptly as possible. An autonomous robot should be able to adapt its behaviour based on the actual situation. Moreover, the update has to occur in a convenient speed for its adaptation to remain pertinent. For example, if the model takes a relative large number of time steps to learn the actual state, then by the time it has already attained it, the system may be in an entirely different state. In this case, the learning is never completed, and the behaviour is poorer compared to faster models. Not only complexity matters for the controller but also the speed at which it can process information. Processing speed is especially important for robots with limited computational capacity dealing with realistic environments. Details for all the models are presented in the next sections. After that, the proposal of performance measure and the experiments are specified. Finally, we discuss the results where the main outcome is that for homeokinetic control a better exploration is accomplished by models with fast linear update rules. Faster adaptation plays a more important role compared to the rest of characteristics that we evaluated.

4.2 Internal Models Implementations

In this section, in order to enable an appropriate comparison IMs are divided into two main categories. The first category includes linear models. These models are characterised by a linear combination of the inputs to generate the output. The second category includes non-linear models. These models can represent an output that is not directly proportional to the input. All the linear models presented share the same structure but differ in the learning algorithm. The first uses a gradient descent algorithm to minimise the prediction error. The other two use linear regression with *least square estimation* for global or local data. Non-linear models include an Echo-State Network (ESN), a Real Time Recurrent Learning Network (RTRL) and a radial basis function network (RBFN) as artificial neural networks.

4.2.1 Linear Model

Regression methods are popular in statistics for the derivation of rules that describe a set of observed data. The general approach involves the estimation of a plane in the training data space that describes the data best. The estimation of future outputs is

derived based on the fitted plane.

The main reason for selecting regression methods is that the mapping performed by the internal model is linear or nearly linear in most cases. Also, regression methods are characterized by simplicity.

In our experiments, we use three linear models. The difference between them is the learning algorithm. As mentioned in Section 3.2.2, gradient descent is the default optimisation algorithms in homeokinesis. As well, two estimators for regression algorithms are tested. First, recurrent least square regression, which is a modification for ordinary least squares (OLS). Second, locally weighted regression which is based on OLS with the main difference that the inputs are weighted according to the actual input value.

All three models share the same configuration. The input signal $\mathbf{y}_t$ is linearly combined to produce the desired output:

$$\hat{\mathbf{x}}_{t+1} = A\mathbf{y}_t + \mathbf{b}, \tag{4.1}$$

with A the coefficient parameters matrix and $\mathbf{b}$ a bias vector.

The first linear model updates its parameter by a gradient descent algorithm over the prediction error (Equation 3.29 and 3.32). Note that we continue with the notation that was introduced for forward models in the sensorimotor loop (Equation 3.19). As opposite to the common nomenclature where the input is x and the output is y.

4.2.1.1 Recurrent Least Squares Regression

For the second model, recursive least squares regression is used to fit the parameters. This method allows for on-line learning in comparison to OLS. The update of the parameters matrix A (including the bias parameters $\mathbf{b}$), is updated at each time step as,

$$A_{t+1} = A_t + P_{t+1}\mathbf{y}\boldsymbol{\xi}^\top,$$
$$P_{t+1} = \left(P_t - \frac{P_t\mathbf{y}\mathbf{y}^\top P_t}{1+\mathbf{y}^\top P_t\mathbf{x}}\right).$$

Here, $\boldsymbol{\xi}$ is the prediction error of the model (Equation 3.5), and the matrix P is initially a diagonal matrix with big values. The derivation of the recursive rule is based on the Woodbury matrix identity (Hager, 1989) for updating of the inverse of a matrix.

4.2.1.2 Locally Weighted Regression

The third model uses locally weighted regression (LWR) Cleveland and Devlin (1988), for error minimisation. This algorithm assumes an ordinary least square estimation

adding a weight value. Said value accounts for the distance between the input and all the data stored until that moment. The motor commands and sensory data vectors are collected for ω time steps. They are stored as rows of the matrices Y and X respectively. The diagonal matrix W accounts for the distance between the actual motor command to all the stored data,

$$W_{ij} = \begin{cases} e^{-\frac{\|Y_{ij}-\mathbf{y}_i\|}{2\sigma}} & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases}$$

for $i = 1,\ldots,m$ and $j = 1,\ldots,n$, with σ as the standard deviation of the Gaussian function. The parameters value are obtained as,

$$A = (Y^\top WY)^{-1}Y^\top WX. \tag{4.2}$$

In the sensorimotor loop, it is impossible to store all the data from the sensor and motor values during the life cycle of a robot. That is why only the last $\omega = 2000$ values are stored, which account for the last 20 seconds of simulation.

4.2.2 Non-linear Models

Non-linear models use a non-linear combination of its parameters and the independent variables. They can include a variety of transformations of the parameter or the variables. For example, quadratic or other higher order power, trigonometric functions or a network with recurrent connections. In this section, three different configurations of networks are presented. First, two artificial neural network (NN), both with a reservoir of hidden recurrent neurons but different configuration and learning techniques. Last, a network of Gaussian radial basis functions is detailed. In the next section, the formalisation of NNs and the details on the implementations are presented.

4.2.2.1 Echo-State Network

Artificial Neural Networks (NN) are mathematical models inspired by biological neural networks. They consist of layers of artificial neurons connected to process some input. The first models for artificial neurons were proposed in (McCulloch and Pitts, 1943). NNs configuration consist of an input layer, one (or more) hidden layer(s) and an output layer, where each layer consists of a set of neurons. Neurons within or inter layers are connected with some individual weight representing the connection strength between them. If the connections do not form a directed cycle between neurons, the

NNs are termed feedforward NNs (FNN). One of the first successful FNN was the perceptron (Rosenblatt, 1958). If there are recursive connections within the hidden layer (or layers), then all the neurons in the hidden layer are grouped as a reservoir. This kind of networks are called recurrent NNs (RNNs). A reservoir is a way to provide resonance to the system (Smith and Zipser, 1989), where actual values depend on the previous state of the network and not only on the real input. This layer also provides memory to the model, allowing it to learn a wider set of scenarios. Connections, both between and within layers are possible in all combinations (except connections to the input layer since its value is set externally to the NN).

An NN adapts (learn) by changing the values of the connections (weights), according to some learning criteria. Usually, the non-input neurons have a non-linear activation function. In this way, the NN can compute non-trivial problems. These characteristics make NNs a more powerful and complex model than the linear ones.

We use NNs as state estimator or function approximator depending on the studied problem. In this chapter, the NNs are used as forward models, and in the Chapter 5 they are also used as policies approximation in reinforcement learning.

The first NN that we use as a forward model is the Echo-State network (ESN) (Jaeger, 2001). In an ESN, the neurons are randomly connected, and the learning process occurs only by adjusting the output weights (green arrows in Figure 4.1). The reservoir inner connections, input connections and feedback from the output layer are not subject to adaptation. Only some initial adjustment of the reservoir's eigenvalues to a near-critical value is carried out (red arrows in Figure 4.1).

We use ESN as a state estimator or forward model. The output is a function of the actual sensory input and motor commands:

$$M(\mathbf{x}_t, \mathbf{y}_t) = \hat{\mathbf{x}}_{t+1} = W^{xo}\mathbf{x}_t + W^{yo}\mathbf{y}_t + W^{so}\mathbf{s}_t \tag{4.3}$$

where $\hat{\mathbf{x}}_{t+1}$ is the estimation of the next time step state, $\mathbf{y}_t$ is the actual motor command and $\mathbf{x}_t$ is the sensory input as in the homeokinetic learning rule (see Chapter 3). As a reminder, the dimension of the sensory vectors is n and for the motors command vector is m. The vector $\mathbf{s}_t \in \mathbb{R}^p$ represents the state of the reservoir of p neurons at time t. The matrix $W^{xo} \in \mathbb{R}^{n\times n}$ represents the weights from the sensors input to the output. The matrix $W^{yo} \in \mathbb{R}^{n\times m}$ accounts for the weights from the motor commands to the output. Finally, the matrix $W^{so} \in \mathbb{R}^{n\times p}$ represents the weights from the reservoir neurons to the output. Note that all these weights are fixed, and they do not change during learning.

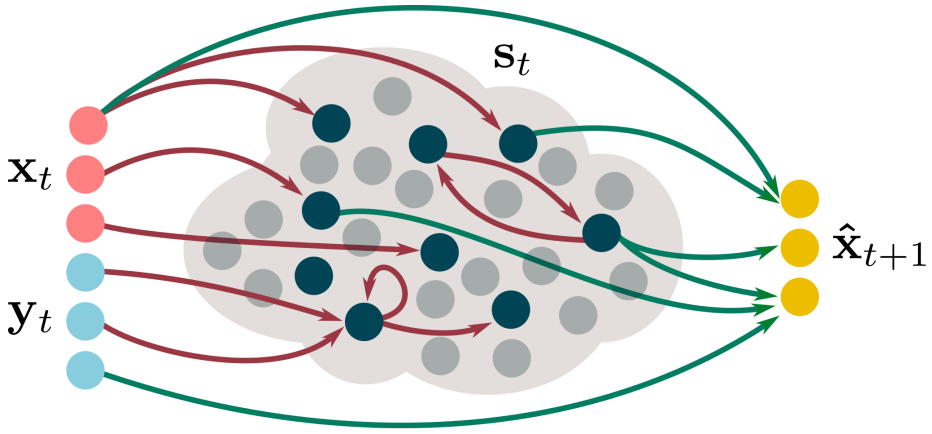


Figure 4.1: Echo-State Network. The input of the network correspond to the actual state $\mathbf{x}$, and motor commands $\mathbf{y}$. Connection between the input layer and the reservoir is random, and the weights are fixed. The reservoir consists of a random recurrent network with state $\mathbf{s}$, also with fixed weights. All the fixed weights are represented with red arrows. The output layer, correspond to the predicted next state $\hat{\mathbf{x}}$. The connections to the output layer (green arrows) are adapted during learning. Feedback connections can be added from the output layer to the reservoir (not depicted in the image).

The internal state of the reservoir's neurons is updated at each time step as:

$$\mathbf{s}_{t+1} = d(W^{xs}\mathbf{x}_{t+1} + W^{ys}\mathbf{y}_{t+1} + W^{ss}\mathbf{s}_t), \tag{4.4}$$

where the matrix $W^{xs} \in \mathbb{R}^{p\times n}$ represents the weights from the sensors state to the reservoir. The matrix $W^{ys} \in \mathbb{R}^{p\times m}$ represents the weights from the motor commands to the reservoir, and the matrix $W^{ss} \in \mathbb{R}^{p\times p}$ represents the weights within the reservoir. The function $d : \mathbb{R} \to \mathbb{R}$ is a sigmoid function applied element-wise.

In our experiments, the NNs are trained on-line. A gradient descent algorithm is used to minimise the prediction error of the models. Let $W = W^{xo} \oplus W^{yo} \oplus W^{so}$, the union of the weights matrices. The update rule for each weight is:

$$\Delta W_{ij} = -\varepsilon_A \frac{\partial \langle \xi_i^2 \rangle}{\partial W_{ij}}, \tag{4.5}$$

with ε_A as the model learning rate and ξ_i as the prediction error of signal i presented in Equation 3.5.

The main idea of ESN is shared with *Liquid State Machines* (Maass et al., 2002). Usually, these NNs and *Backpropagation Decorrelation* learning rule (Schiller and Steil, 2005) are categorised as *Reservoir Computing*. They are based on the idea that traditional learning algorithms for RNNs, where all weights are updated, tend to modify dominantly the output weights (Schiller and Steil, 2005). Typical all-to-all RNNs

have slow learning algorithms and are prone to disrupt the learning by bifurcations. In contrast to feed-forward networks, the output of a recurrent network can change drastically with an infinitesimal change in the network parameter when it passes through a bifurcation point (Doya, 1992) (see Section 2.4). Thus, convergence is not grant on recurrent networks when all the weights (input, recurrent and output) are updated. On the other side, ESN learning is fast compared to the most notable algorithms for supervised training of RNNs, i.e., *real-time recurrent learning* (Williams and Zipser, 1989) and *backpropagation through time* (Werbos, 1990). In ESN the learning occurs only on the output weights layer. The linear learning of the ESN acts on a continuous function of weights as each unit has a smooth output function. Thus, ESNs do not suffer from bifurcations and are easy to implement (Jaeger, 2007).

An ESN has to hold the *echo state property* to work. The property relates the dynamics of the reservoir with the input signal. It states that the reservoir will asymptotically wash out any information from initial conditions (Jaeger, 2007). To guarantee the property in the reservoir with sigmoid neurons, a sufficient condition is to keep the spectral radius smaller than unity. Thus,

$$\rho(W^{ss}) = \max\{|\lambda_1|, \cdots, |\lambda_s|\},$$

where $|\lambda_1|, \cdots, |\lambda_s|$ are the eigenvalues of the matrix W^{ss}. The weights of the matrix are scaled to yield spectral radius $\rho(W^{ss})$ close to one, ensuring the echo state property with fast convergence.

4.2.2.2 Real-Time Recurrent Learning Neural Network

The second RNN used as an internal model is the *Real-Time Recurrent Learning Network* (RTRL). It was introduced by (Williams and Zipser, 1989) as a way to efficiently train recurrent NNs. RTRL computes the exact error gradient at every time step and uses a gradient-descent method to update the connections. Such network is suitable for on-line learning algorithm where the internal weights are updated continuously with the new arrival of data. The learning rule yields low computational cost compared to other algorithms like *back-propagation through time* (Werbos, 1990). RTRL has been successfully applied in numerous applications including stream-flow forecasting (Chang et al., 2002) and recognition of finite-state grammars (Smith and Zipser, 1989).

In an RTRL, the network is fully connected. All the neurons are connected to each other with the except of the input nodes that do not receive the synaptic con-

nection. Following the homeokinetic notation (Chapter 3), an RTRL network has as input the actual motor commands $I = \{y_k(t),\ 0 < k < m\}$ and a set of units $U = \{s_k(t),\ 0 < k < n+q\}$ including q hidden and n output units. A unit of the network is defined as

$$z_k(t) = \begin{cases} y_k(t) & \text{if } k \in I \\ s_k(t) & \text{if } k \in U \end{cases}$$

Let W be a weight matrix for the network with a unique weight between each unit except to the input units. Thus,

$$s_k(t) = \sum_{l \in I \cup U} W_{kl} z_l(t) \tag{4.6}$$

denotes the state of a reservoir neuron k at time t, with l a unit of the network. The output of the network is calculated as,

$$\hat{x}_k(t+1) = g(s_k(t)) \tag{4.7}$$

with g a sigmoidal function applied element-wise. Let $T = \{d_k(t),\ 0 < k < n\}$ denote a set of units with a specified target. An error function can be defined for each output unit,

$$e_k(t) = \begin{cases} d_k(t) - \hat{x}_k(t) & \text{if } k \in T \\ 0 & \text{otherwise.} \end{cases} \tag{4.8}$$

Now, the prediction error is defined analogously to the homeokinetic prediction error (Equation 3.5) as,

$$\xi(t) = \frac{1}{2} \sum_{k \in U} (e_k(t))^2. \tag{4.9}$$

To update the connection weights, a gradient descent over the prediction error is defined as

$$\Delta W_{ij} = -\varepsilon_A \frac{\partial \langle \xi^2 \rangle}{\partial W_{ij}} = \varepsilon_A \sum_{k \in U} e_k(t) p_{ij}^k(t),$$

$$p_{ij}^k(t+1) = g'(s_k(t)) \left[\sum_{l \in U - T} W_{kl} p_{ij}^l(t) + \delta_{ik} z_j(t) \right],$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, with $p_{ij}^k(t_0) = 0$. The term $p_{ij}^k(t)$ represents the sensitivity of the output nodes to a small change of the weights. Note that the weight W_{ij} is not necessarily connected to the output $\hat{x}_i$. Thus, this learning rule is non-local. The sensitivity of the output node depends on changes that could have occurred in a different place in the topology of the network.

The calculation of the gradient error requires solving a n-dimensional system for *each* of the weights W_{kl}. It requires $O(r^3)$ memories and $O(r^4)$ computations, with $r = m+n+q$ being the total number of neurons. Such computational order represent a drawback for large systems (Doya, 1995).

4.2.2.3 Radial Basis Function Neural Networks

In Chapter 5, a reinforcement learning set-up with a self-organisation approach will be presented. For an agent to learn the best policy and value function, a *Radial Basis Function Neural Network* (RBFNN) is used as forward and inverse models. Note that the notation is standard here, $\mathbf{x}$ for inputs and $\mathbf{y}$ for outputs. An RBFNN is a feedforward neural network with one hidden layer. It was initially proposed by (Broomhead and Lowe, 1988). The configuration of the network includes a full connection between the input layer and the hidden layer, and between the hidden layer and the output layer (see Figure 4.2).

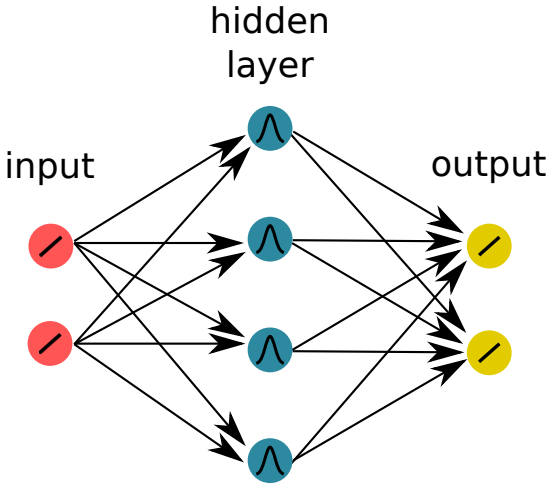


Figure 4.2: An RBFNN has one hidden layer of neurons with radial basis function activation. For local activations, e.g., Gaussian, values far from the centre of the unit do not activate the neuron.

A radial basis function is a real-valued function whose value depends on the distance from the origin. Let $\phi(\mathbf{x}) = \phi(\|\mathbf{x}\|)$, or also depending on the distance from point $\mathbf{c}$ called *centre*, so that $\phi(\mathbf{x},\mathbf{c}) = \phi(\|\mathbf{x}-\mathbf{c}\|)$. Any function ϕ that satisfies the property $\phi(\mathbf{x}) = \phi(\|\mathbf{x}\|)$ is a radial function. The norm used in this work is *Euclidean distance*. Nevertheless, other distance functions are possible as long as they satisfy said property.

A Gaussian radial basis is defined as:

$$\phi_k(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{c}_k\|^2}{2\sigma_k^2}\right), \quad k = 1,2,\cdots,h, \tag{4.10}$$

where σ_k is the width the kth basis function, and h is the number of hidden neurons in the network. A normalised version over all the basis functions of the network of Equation 4.10 can be built as:

$$\beta_k(\mathbf{x}) = \frac{\phi_k(\mathbf{x})}{\sum_{l=1}^{h}\phi_l(\mathbf{x})}. \tag{4.11}$$

The normalised Gaussian RBFNN is the linear combination of the basis function unit in the hidden layer. The value of each output neuron can be calculated as:

$$y_i(\mathbf{x}) = \sum_{j=1} W_{ji}\beta_i(\mathbf{x}), \tag{4.12}$$

for $i = 1,\ldots,m$ and $j = 1,\ldots,n$, where the function $y_i(\mathbf{x})$ is the output of the ith neuron in the output layer. W_{ji} represents the weight from the jth neuron of the hidden layer to the ith neuron of the output layer, and n the number of output neurons.

RBFNNs have been used as function approximator in reinforcement learning (Doya, 2000). The centre of the units are usually task dependent: they can be set by design or learn based on input data. In on-line settings, the centres of the basis function can be distributed uniformly across the normalised space and then modified at each time step. The network can be trained on-line with gradient descent algorithm. In Section 5.2, we show the learning rules and training methods for the internal models on a reinforcement learning setting.

4.3 RNN Models Response

In section 3.2.3, the homeokinetic rule was derived. Those learning rule are called *standard* since a linear model is used. Due to the dependency of the learning rules to the model, in this section the rules for homeokinesis with an RNN model are derived. Recalling the homeokinetic canonical learning rule (Equation 3.17):

$$\Delta c = \varepsilon_C \boldsymbol{\chi}^\top \frac{\partial L}{\partial c}\boldsymbol{\eta}, \tag{4.13}$$

where c is a parameter of the controller. The Jacobian L represent the response of the model to the input. To derive the learning rules, first $L = \frac{\partial M}{\partial \mathbf{x}}$ and then $\frac{\partial L}{\partial c}$ have to be derived.

We define an RNN as an extended model with dependency on the sensory input as well as on the motor commands, no feedback and identify output function as (following Equation 4.3):

$$M(\mathbf{x}_t, \mathbf{y}_t) = \hat{\mathbf{x}}_{t+1} = W^{xo}\mathbf{x}_t + W^{yo}\mathbf{y}_t + W^{so}\mathbf{s}_t,$$

with internal state update as (following Equation 4.4):

$$\mathbf{s}_{t+1} = d(W^{xs}\mathbf{x}_{t+1} + W^{ys}\mathbf{y}_{t+1} + W^{ss}\mathbf{s}_t).$$

Now, L can be expressed as in sensor space as:

$$L = \frac{\partial M(\mathbf{x}_t, K(\mathbf{x}_t))}{\partial \mathbf{x}} = W^{xo} + W^{yo}G'C + W^{so}\frac{\partial \mathbf{s}_t}{\partial \mathbf{x}}, \tag{4.14}$$

where G' follows Equation 3.21. The derivative also depends on the network reservoir sensitivity with respect to the input:

$$\frac{\partial \mathbf{s}_t}{\partial \mathbf{x}} = D'(W^{xs} + W^{ys}G'C + W^{ss}\frac{\partial \mathbf{s}_{t-1}}{\partial \mathbf{x}})$$

where, analogously to G', D' is the diagonal matrix:

$$D'_{ij} = \delta_{ij}d'_i(\mathbf{w}),$$
$$\mathbf{w} = W^{xs}\mathbf{x}_t + W^{ys}\mathbf{y}_t + W^{ss}\mathbf{s}_{t-1}.$$

Assuming $\frac{\partial s_{t-1}}{\partial x} = 0$, the response matrix is:

$$L = W^{xo} + W^{yo}G'C + W^{so}D'(W^{xs} + W^{ys}G'C). \tag{4.15}$$

Now the derivative of the Jacobian with respect to the matrix parameters C:

$$\frac{\partial L}{\partial C} = W^{yo}\frac{\partial G'}{\partial C}C + W^{yo}G' + W^{so}D'W^{ys}\frac{\partial G'}{\partial C}C + W^{so}D'W^{ys}G' \tag{4.16}$$

The activation function defined as $g(\cdot) = \tanh(\cdot)$ holds that $g'' = -2gg' = -2\mathbf{y}g'$, thus:

$$\frac{\partial G'}{\partial C} = -2\mathbf{y}G'\mathbf{x}.$$

Now, defining a new auxiliary vector $\boldsymbol{\upsilon}$, redefining the original auxiliary vectors $\boldsymbol{\mu}$ (Equation 3.23) and recalling $\boldsymbol{\zeta}$ (Equation 3.26):

$$\boldsymbol{\mu} = \boldsymbol{\chi}^\top W^{yo}G', \quad \boldsymbol{\upsilon} = \boldsymbol{\chi}^\top W^{so}D'W^{ys}G', \quad \boldsymbol{\zeta} = C\boldsymbol{\eta}, \tag{4.17}$$

replacing term in Equation 4.13 using Equation 4.16 and 4.17 the learning rule for the parameters of the controllers is:

$$\frac{1}{\varepsilon_C}\Delta C_{ij} = \mu_i\left(\eta_j - 2\zeta_i y_i x_j\right) + \upsilon_i\left(\eta_j - 2\zeta_i y_i x_j\right), \tag{4.18}$$

and analogously for the bias parameter h in the controller, the learning rule is:

$$\frac{1}{\varepsilon_C}\Delta h_i = -2\mu_i\zeta_i y_i - 2\upsilon_i\zeta_i y_i. \tag{4.19}$$

The first term on the RHS of Equations 4.18 and 4.19 represents the influence of the motor commands to the output, similar to the linear model (Equations 3.27 and 3.28). The second term adds the influence of the reservoir with respect to the motor command to the output.

The response model for the RTRL network is presented in the Appendix A.2.

4.4 Noise for Control

We also considered a trivial case as a baseline for comparison. Coloured noise is directly used as the value of the parameters of the controller. A time averaging value τ is used to shape the colour of the noise

$$\tau c_{t+1} = \sqrt{\tau} n_t - c_t \tag{4.20}$$

where c is a parameter of the controller and n is a uniform distributed noise; for $\tau \ll 1$ the produced noise is close to white coloured noise. For bigger values of τ, strong colour noise is produced. A model is not needed in this case, but the correlation parameter is fixed.

4.5 Experiments and Evaluation

The homeokinetic controller produces exploratory behaviour over the controlled robot. Exploration can be seen as a thorough examination of the environment. This broad definition cannot count as an objective measure of performance. Because of this absence of a performance indicator in homeokinesis, it is required to design a quality measure. One of the expected scenarios for an exploratory controller is to deal with an irregular environment. The controller should be able to adapt to new conditions imposed by the environment to maintain an exploratory mode. For a homeokinetic robot, placed on an irregular terrain, the total explored area indicates how well it can traverse the environment. A larger coverage indicates a better adaptation.

4.5.1 Experiment Set-Up

To measure the performance of the controller with different internal models, we have chosen a simple robot in an enclosed yet variable environment (Figure 4.3a). The measure is the percentage of coverage of the planar layout of the environment (Figures 4.3c-4.3e). A low-dimensional robot is suitable to measure this quantity, compared to a complex robot. A complex robot may need to obey other control principles, e.g., avoidance of self-blocking configuration or maintenance of a stand-up position. These extra constraints may lead to different solutions not directly related to the exploratory problem[1].

Planar coverage provides a summary measurement of the robot's ability to deal with an irregular environment. Moreover, it shows the ability of the controller to cross between various obstacles, indirectly indicating a degree of flexibility. Additionally, the measure of the total travelled distance is quantified (path length).

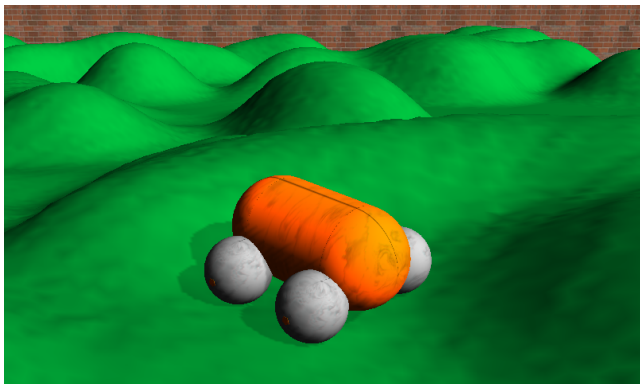
The coverage rate is calculated by counting the number of visited cells in a square grid over the planar environment (Figure 4.3b). In such grid, the cells spacing correspond to the double of the diameter of the robot. A single trial last 20 minutes of simulation with some minutes unaccounted for internal initialisation. The parameter space of the homeokinetic controller includes both the learning rate for the controller ε_C and the internal model ε_A. Moreover, an environment difficulty variable $\alpha \in [0;1]$, represents the steepness of the terrain. A value of $\alpha = 0$ defines a flat terrain. Increasing values add different steep to regular bumps on the terrain. Even at the maximum value $\alpha = 1$, the traverse of each cell is guaranteed. The robot is physically able to cross all the terrain at any α value.

Additionally, a natural measure of the performance of a forward model is the average prediction error. A smaller prediction error shows a better capacity of the model to capture the environment. On one hand, the reduction of prediction error tends to reduce the size of the explored region. On the other hand, large errors tend to produce large effects on the controller, i.e., less persistent behaviour can be expected.

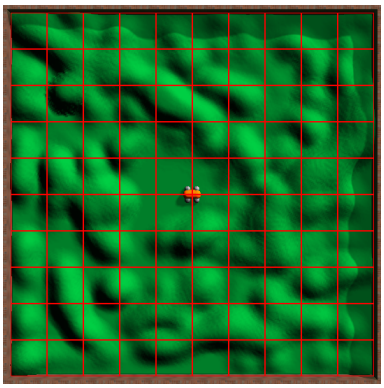
Note that, as with any homeokinetic robot, the controlled vehicle does not know any parameters about the environment. The robot only feels the interaction with the environment by the changes in its proprioceptive sensors. In the case of the vehicle, the robot has four proprioceptive angular velocity sensors. The forward internal model, encodes the motor commands of the robot, but not the state of the robot. Such con-

[1] The behavioural space of complex homeokinetic robots has been explored in (Martius, 2012).

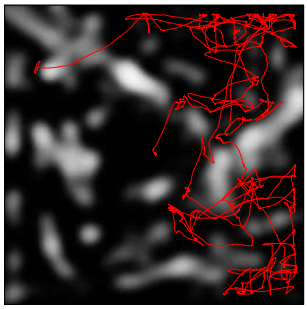
figuration is proposed as an extended model. The extended model includes the motor commands and the state of the robot (only proprioceptive) to calculate the next time step state. The extended configuration has been studied in (Der and Martius, 2012) presenting a sound theoretical understanding of the prediction process, but yielding similar behaviour for the robots.



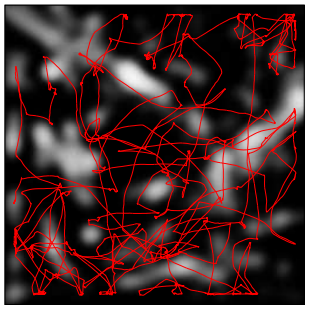
(a) Four wheeled robot



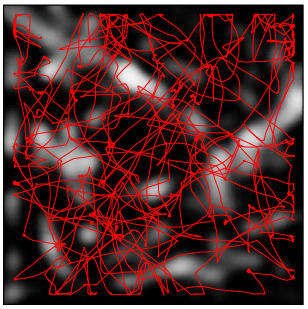
(b) Top view of the gridded environment



(c) 40% of coverage



(d) 85% of coverage



(e) 95% of coverage

Figure 4.3: (a) A four-wheeled robot simulated with LpzRobot (Martius, 2012). The robot has proprioceptive angular velocity sensors. Motor commands set the angular velocity of each wheel independently ($n = m = 4$). Figure (b) shows the cell delimitation of the environment. Figures (c) to (e) represents three different coverage performances. The highest points of the environment are represented by white colour. The robot is capable to traverse the whole area, but the exploration is focused on more accessible parts of the environment.

4.5.2 Results

The models have been tested with various different learning rates to find the best performance for each one. In Figure 4.4, the average prediction errors for all the models —except noise controller— are plotted depending on the difficulty of the terrain. Each data point represents an average over a series of 30 trials for a set of specific learning rates. For all the models, the prediction error tends to decrease with a larger difficulty of the terrain. On a more difficult terrain, the robot presents a less dynamic behaviour. The variety of scenarios that the robot faces is less diverse. It is confined to a smaller area of exploration. This confinement is confirmed by the smaller coverage reached under higher terrain difficulty (see Figure 4.5). The best performances in lower difficulty ($\alpha < 0.5$), are achieved by the ESN, linear model and RTRL. When difficulty grows ($\alpha > 0.5$), RTRL does not improve the prediction as much as the other two models. The last two models, linear regression and LWR, despite being linear, their learning rules makes result in poorer prediction performance.

In Figure 4.5, coverage for different models —including noise controller— are presented for several learning rates. At lower terrain difficulty, i.e., a flat ground, almost full coverage is obtained by most of the models. The noise controller maximum coverage is only around 60%. At higher difficulty, an expected lower coverage is achieved by all the models. At higher terrain difficulty, $\alpha > 0.5$, ESN and the linear model present a better coverage compared to the other models at same terrain difficulty.

Coloured noise with different τ value still shows a poorer performance compared to the rest of models. Even though, certain noise frequency can produce some exploration. Such exploration is limited due the lack of adaptability to the environment. For a successful exploratory controller, there has to be a correlation between the adaptation of parameters and the interaction of the robot with the environment.

Another way to measure the performance of the controller is the path length travelled by the robot (Figure 4.6). This value can tell in overall how active was the robot, independently of the number of traversed cells. A robot could cross a large path but only cross a few different cells. The path length tends to decrease linearly when the difficulty grows. There is a more noticeable difference between ESN and the linear model for less difficult terrains comparing to coverage with same terrain. The linear and the ESN models have the larger path length in most of the different ground difficulties. Noise has the lowest performance again.

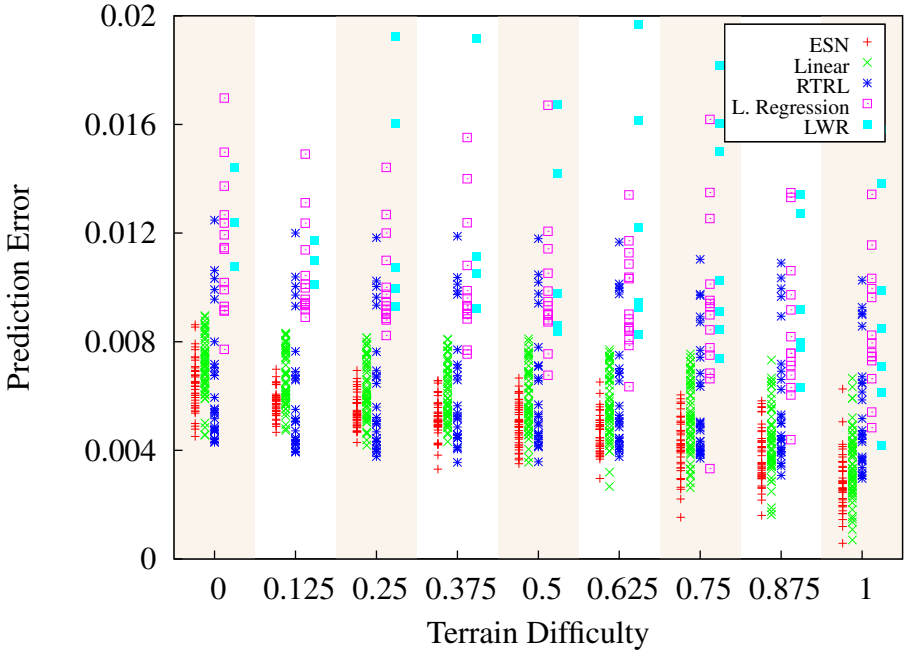


Figure 4.4: Prediction error for each model at different terrain difficulty. Each point represents the average of a particular learning rate tested over 30 trials. It can be seen that the error decreases with the difficulty of the terrain. There is a smaller state space explored, so a better prediction is achievable. ESN and the linear model have the best prediction at highest terrain difficulty. The relationship is close to linear as the tendency lines show. There is a saturation of the prediction error and the coverage that tends to be linear for the optimal pairs of learning rates.

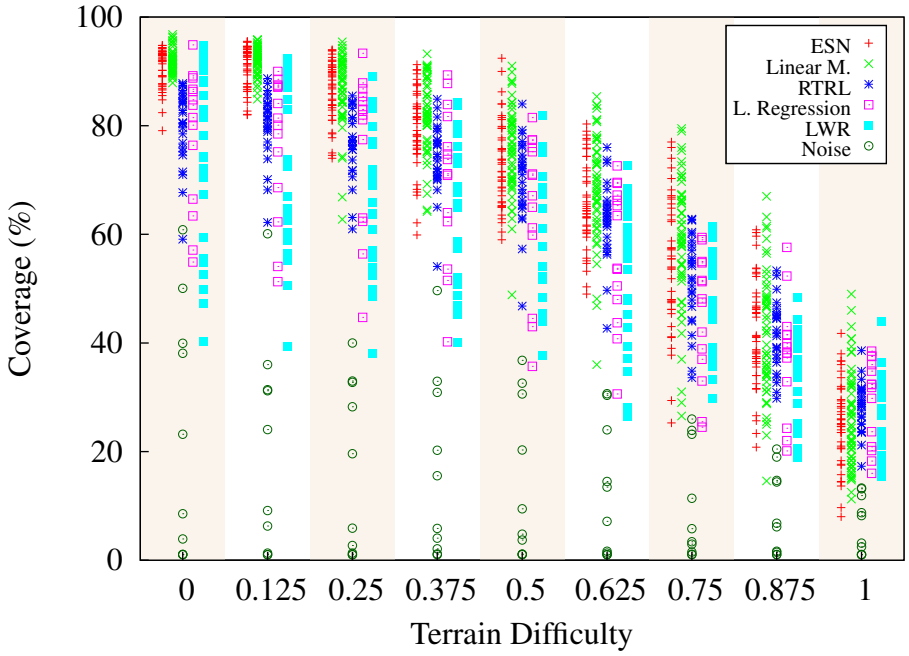


Figure 4.5: Coverage percentage for different terrain difficulty. Several learning rates for the models and controller. At lower terrain difficulty, virtually full coverage is attained. At $\alpha = 0.75$, ESN and linear models have the best performance. The performance of the other models is similar under different terrain difficulty. Coloured noise achieves poor performance for all different colours (τ parameter).

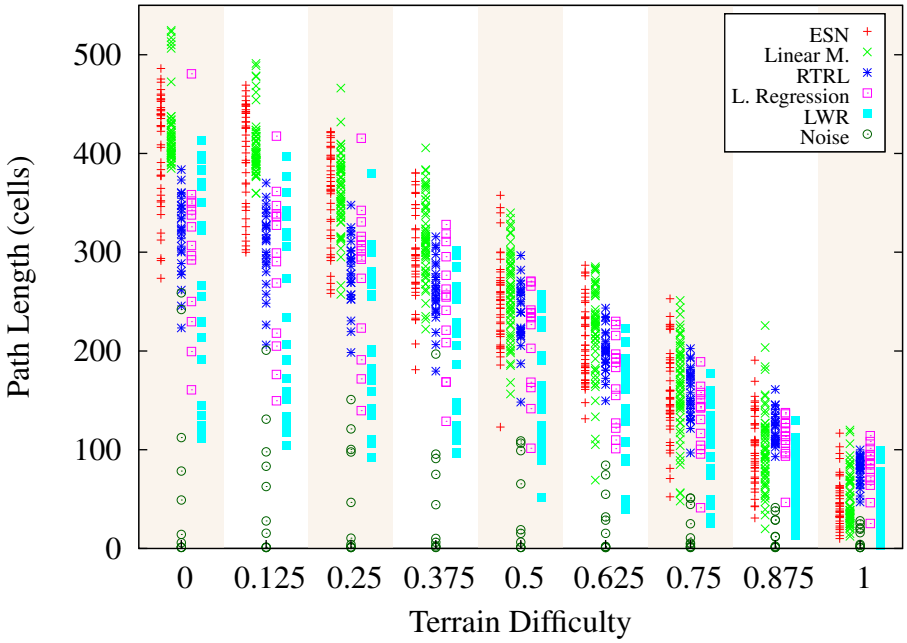


Figure 4.6: The total path length covered by the robot in different terrain difficulty. The linear and ESN models present a better performance compared to the other models. Coloured noise has the poorest performance consistently among the parameter space.

The best-combined performance of the controller is when the coverage (or path length) has maximum performance, at the same time a minimum prediction error. In Figure 4.7, the coverage is plotted against the prediction error of each model for different learning rates and terrain difficulty. Given the nature of the two optimised values, while making one individually better yields the worst performance in the other one. This behaviour can be explained as, that small prediction error produce minor changes in the controller parameters. Thus, providing lesser exploration. Instead of having one absolute optimal combination of learning rates, the best results are Pareto efficient. Consider that only the values on the Pareto frontier are shown in the plot. The best results lie where the coverage is maximised (upper side of the plot) and the prediction error is minimised (left side of the plot). ESN, linear model and RTRL present the most efficient combination of prediction error and coverage. Where the prediction is maximal for the three models, the prediction error is comparable. A difference appears for lower coverage, where ESN and linear models decrease the error, but RTRL presents similar values for all coverage percentages. Linear regression and the LWR presents favourable coverage but always at the expense of larger prediction error.

Similarly, path length versus prediction error are plotted in Figure 4.8. Correspondingly to the previous plot, ESN, RTRL and the linear model present the highest path length with lower prediction error. Again, at smaller path length, RTRL remains with similar prediction error. The linear regression and LWR methods have comparable path length but at a higher prediction error. In the last two plots (Figs. 4.7 and 4.8), the behaviour obtained by noise has not been plotted. Noise control does not produce a prediction error.

Analysing the results in the last two figures, the ESN and the linear models are the best-suited models for homeokinetic control. RTRL has good performance as well but is not able to reduce the prediction error as much as the other two models. One difference between the first two and RTRL is the order of learning rules. Both, ESN and linear model can update their weights in linear time. On the contrary, RTRL has much slower learning rule. For RTRL, a small reservoir of only 15 neurons was able to be tested on the same machine, maintaining acceptable runtime performance. Another disadvantage in RTRL are the output feedback connections. The output feedback determines the extent to which an NN has an autonomous pattern generation component (Jaeger, 2007). The controller already provides to the model the real state input. In this sense, the predicted values add unnecessary complexity to the network when are fed back. To obtain comparable results, a modification had to be applied to RTRL.

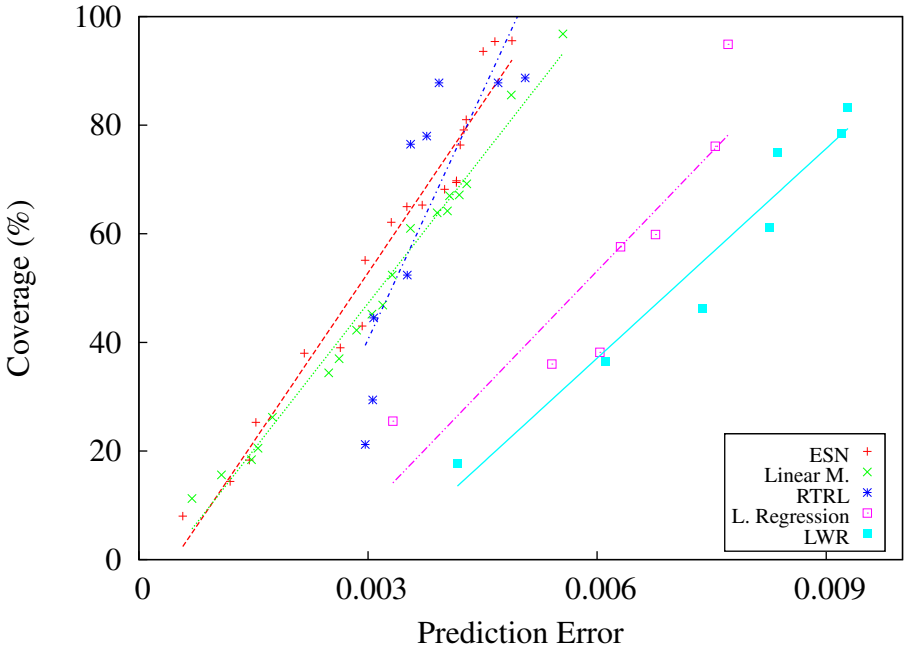


Figure 4.7: Pareto frontier of the optimal combination maximum coverage versus minimum prediction error. Each point in the graph represents a Pareto efficient point for a pair of model and controller learning rates. ESN responds with higher coverage and lower prediction error for several learning rates. The linear model has a comparable prediction error for similar coverage. RTRL has comparable, and better performance for high coverage. The linear regression and LWR models achieve good coverage but at expenses of a larger prediction error.

Removing the sigmoidal function from the output layer increased the prediction performance. Since sensors are bounded to a range of $[-1;1]$, the sigmoidal function require high values to produce output values close to ± 1. Such values require large connection weights in the output layer that were not achieved by RTRL.

Homeokinesis has also been tested in high dimensional robots. Both, linear controller and linear model can cope with high dimensional systems. In the presented experiments, only a low dimensionality robot has been used (4×4 dimensions). The linear model and the ESN can cope with higher dimensional systems. They will still respond within viable speed for a real-time robot. However, RTRL will stagnate on higher dimensionality. It is still interesting to compare the performance of all the models under settings that can be handled by all of them.

The linear model, linear regression and LWR share a similar architecture. However, the learning rules are different. This difference produces variable results in terms of

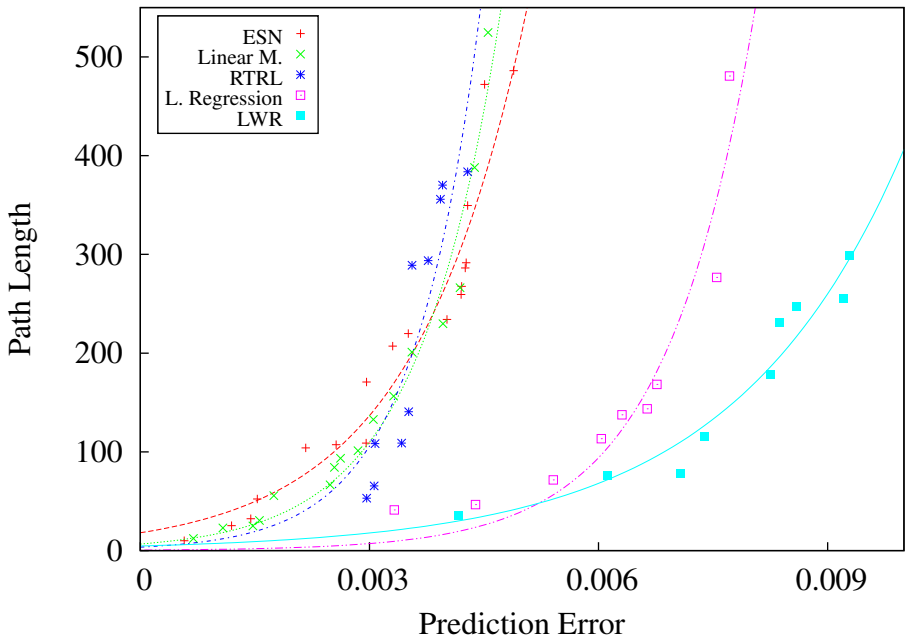


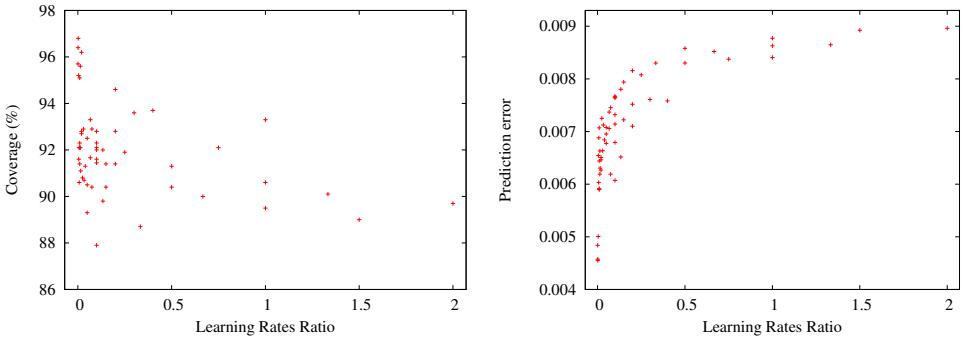
Figure 4.8: Path length of the exploration versus the prediction error. To the left the two best performances achieved by the ESN and the linear model. The linear regression and LWR models achieve similar path lengths but at higher prediction error. RTRL present the poorest performance. The curve fitting are an exponential regression, except for RTRL.

coverage and prediction error. The linear model update its parameters by gradient descent in the direction of minimisation of the error. Meanwhile, the other two methods rely on the updating of an inverse matrix. Furthermore, LWR is adapted to work online with a window of past inputs to be fitted.

4.5.2.1 Learning Rates Ratio

In the previous results, the robot behaviour has been produced with several different learning rates for the controller and the model. Plots between the ratio of the two learning rates ($r = \varepsilon_C/\varepsilon_A$) versus coverage and prediction error are presented in Figure 4.9a and Figure 4.9b respectively. The results have been taken from several trials of the robot with a linear model in a low difficulty environment. In the figures, a low ratio represents a controller learning rate slower than the model, and vice versa for high ratios, i.e., faster learning for the controller. A ratio $r = 1$ indicates same learning rate for both the model and the controller. At small ratio, higher coverage can be achieved (Figure 4.9a). For large ratios, the coverage tends to decrease. A more noticeable ten-

dency can be seen when the ratio is compared to the prediction error (Figure 4.9b). The lowest values for the prediction error are achieved for the smaller ratio, increasing logarithmically with the ratio. This results can be seen as that the model has to be faster than the model to achieve the best performance. Alternatively, from the controller point of view, it has to *give time* to the model to learn the actual scenario before moving to a new one. A ratio $r < 1$, is the usual configuration for homeokinetic control with a linear model.



(a) (b)

Figure 4.9: Coverage and prediction error compared to the ratio of the learning rates ($r = \varepsilon_C/\varepsilon_A$). Both best results, high coverage and low prediction error, are achieved by a small ratio, i.e., a smaller controller learning rate compared to the model learning rate.

4.5.2.2 Noise Control

Different noise correlation (Equation 4.20) have been tested to compare the homeokinetic approach to a noisy signal. The best coverage for noise control is achieved at highly correlated noise (large τ, Figure 4.10). At these values, the noisy signal can maintain the parameters of the controller changing consistently both for the robot and the environment. Such coordination is useful only locally. However, as soon as the environment change, new controller parameters are needed to maintain a consistent behaviour. For small τ values, almost all exploration is lost: the robot only *shakes* in its position without producing any coherent movement. In this case, the parameters of the controller are rapidly changed. Such fast adjustment of the parameters does not allow the controller to produce interesting motor commands for the robot. Such commands are not capable of performing any movement related to its physical configuration. Figures 4.5 and 4.6 show that the exploration achieved by the coloured noise signal has poor performance when compared with any other model. These results enforce the

idea that a predictive and, at the same time, sensitive controller —like homeokinetic— is better suited for exploration than a noisy signal, even a highly coherent one.

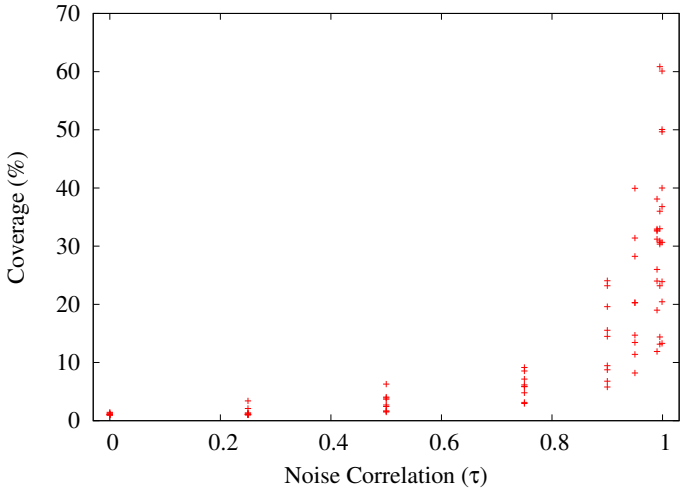


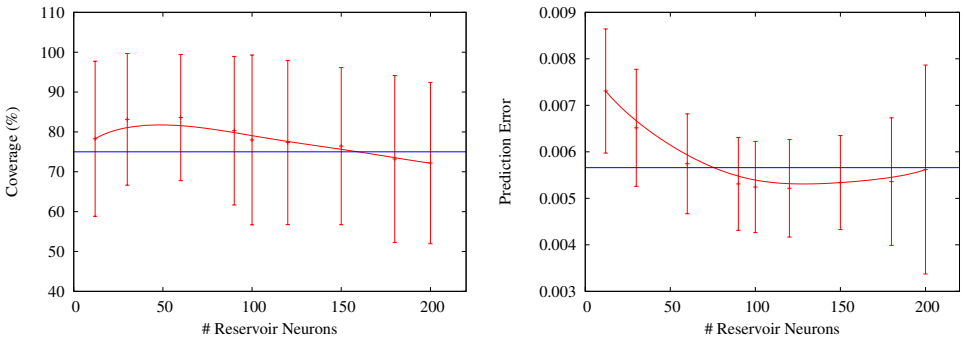
Figure 4.10: Coverage percentage respect to noise correlation. For values of τ close to a 0, the noise is uniformly distributed. For larger values, more coverage is achieved due to a more correlated noise. Still, it doest not perform as good as the homeokinetic controller.

4.5.2.3 ESN Representational Power

The ESN model has been tested with the fixed number of $p = 100$ neurons in the reservoir. Experiments with different reservoir size have been carried on. The idea is to assess the relationship between the size of the reservoir and the behaviour of the robot. Reservoir size in the range of $[12, 200]$ have been tested. In Figure 4.11a, coverage is plotted against the number of neurons in the reservoir. Coverage is not significantly varied by the size of the reservoir. The blue line is the average result of the linear model for the same terrain difficulty. Slightly better coverage than the linear model, in average, is obtained for 150 neurons or fewer, but still inside the standard deviation.

A clearer relationship can be seen between the reservoir size and the prediction error (Figure 4.11b). For values of p below 90, the prediction error is larger than the average. A small reservoir adds instability to the system. The signals generated by the reservoir are not complex enough to capture the environment, but they still modify the overall output. For larger values, there is a better prediction performance of the model. For a larger number of neurons, the advantage of the reservoir is washed-out, and the

standard deviation grows even bigger. The optimal size of the reservoir is around 100 neurons for the prediction error. Lesser neurons increase the error as the network does not have enough representational power to produce the most accurate state prediction. When the reservoir grows over the optimal size, the reservoirs describes noise instead of the desired relationship. Again, the blue line represents the average prediction error of the linear model under similar settings.



(a) (b)

Figure 4.11: Coverage (a) and prediction error (b) in dependency to the number of reservoir neurons in the ESN model (values in red). The blue line, represent average values for the linear model. Prediction error decrease for values around 100 neurons. More neurons do not seem to improve the prediction further. Coverage is on average better than linear for values below 100 neurons. At these configurations, the error is larger, thus producing a faster change in the parameters of the controller. All trials were performed at same terrain difficulty.

4.6 Discussion

In this chapter, it has been shown a set of IMs for homeokinetic control. For each model, the response and learning rule (Equation 4.13) have been derived. It is interesting to compare the difference between the linear models and the NNs. Recalling the learning rules for the **h** parameters of the controller (for simplicity compared to C), both for linear and NN models:

$$\Delta h_i = -2\varepsilon_C \mu_i \zeta_i y_i$$
$$\Delta h_i = -2\varepsilon_C \mu_i \zeta_i y_i - 2\varepsilon_C \upsilon_i \zeta_i y_i,$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\zeta}$ represent the intervention of the motor commands to the output and the parameters of the controller respectively. Additionally, for the non-linear controller, $\boldsymbol{\upsilon}$ represents the tampering of the reservoir or internal layers. This last expression includes the weights from the reservoir to the output layer. This extra layer of process improves the prediction performance of the ESN model compared to the linear model for certain sizes (Figure 4.11b). On the one hand, the reservoir adds more computational power to the model. The model has better chances to incorporate more intricate behaviours. On the other hand, the reservoirs provide a longer term memory compared to the linear model. Using recurrent connections, signals can resonate in the internal state of the model. Events that occurred further back in time still shape the output of the model. Still, a small reservoir only seems to perturb negatively the model. In the same way, a larger one does not improve the performance further. This disruption was also the case for RTRL. Only few hidden units were able to be implemented. Due to the nature of the learning algorithm for RTRL, numerous neurons abate the processing speed. With 20 or more hidden neurons, the model could not respond in real time on the same computational power where the other models run smoothly. An $O(r^4)$ computational order (r the total number of neurons) limits the use of the network on-line.

The memory that the reservoir or hidden layer provides to the model here is interesting for exploration reasons. If the model can predict more complex behaviour, it is plausible for the robot to enact them. A longer term memory, i.e., memory that stores *behaviour primitives* is out of the reach of this thesis. Nevertheless, work in such direction can be revised in (Martius, 2010). In said work, a network of expert controllers extract repetitive behaviour for a posterior use.

The results of the experiments show comparable performance between the linear model, the ESN and RTRL models. Linear regression, LWR and noise control have the worst performance overall. The first three models, all share a version of gradient descent learning. Such learning rule has fast computation time and is widely used in machine learning. Still, such an algorithm can be stuck in local minima or, depending the architecture of the model, it can be computationally expensive (as in RTRL). Among the best models, RTRL loses performance on more difficult terrain. Following the plots with the Pareto frontier (Figure 4.7 and Figure 4.8), the linear model and the ESN model remain the best two models. Both can minimise the prediction error even in difficult terrain. ESN presents a slightly better prediction performance. In practical terms, the linear model and the ESN are both suitable for exploration maximisation in homeokinesis. Both can be used on-line, in real time and have an easy implementation.

An interesting task would be to measure the complexity of the behaviour. In a robot with higher dimensionality, more intricate, yet predictable behaviour could arise when an ESN is used instead of a linear model.

Another interesting point is the comparison of learning rates. Usually learning rates are set *small enough* by trial and error. With the values presented in Figure 4.9, a preferred combination can be found. The best performance has been achieved when the model has a larger learning rate than the controller. The logic behind these results is that an action has an immediate response from the environment, on the contrary, the model needs to process (sometimes more than once) the gathered data. That is why, to achieve a good prediction, the controller has to *allow* the model to learn. Nevertheless, a too big difference between the learning rates is not beneficial. The controller will not change its internal parameters too fast to cope with the environment. The controller loses sensitivity, which will lead to unresponsive robots.

In the case without model, i.e., the noise control, the performance was inferior to all the models. Only for highly correlated noise there is some comparable coverage (see Figure 4.10), but still inferior. A controller without an internal model cannot predict future states (from a developmental robots point of view). Thus, its behaviour is not predictable. Also, if a homeokinetic controller would count with a perfect model, i.e., a model that predict future states precisely, it would not change the parameters controllers. The prediction error would be 0, cancelling any learning. Again, it would be an insensitive and unresponsive robot. In theory, if we had such perfect model, there would not be a need to explore. The robot would already know *everything* from its environment. Exploiting the model would be a more energy and time efficient way to find a way to reach a goal. In practice, such model is impossible to attain. Memory capacity, noisy signals, and intractable environment and interactions are insurmountable problems for limited capacity robots. Thus, the homeokinetic controller remains a useful practical tool for embodiment exploration in high dimensional robots.

The capability of the homeokinetic controller depends both on the complexity of the environment and on the type of forward model. There is a direct relation between the terrain difficulty and homeokinetic performance (Figure 4.5 and 4.6). The quantities of coverage and path lengths depend directly on the controller, as the actual driver of behaviour, and only indirectly on the model. The homeokinetic framework does not specify the type of model. Nevertheless, the update of the parameters of the controller depends on the model response and in the prediction error. Therefore, the homeokinetic capability is a result of the relation between the model and the environment.

In the next chapter, the combination of homeokinesis as an exploratory signal in reinforcement learning is studied. It is shown how homeokinesis can be used as probing signal in comparison to usual random noise. Also, how homeokinesis can be used as a policy that explore the environment assessing the reward signal.

Chapter 5

Self-Organisation Guided by Reinforcement Learning

In this chapter, we propose the use of the homeokinetic controller as an advanced exploration signal to improve the performance in goal-oriented tasks. We take reinforcement learning (Sutton and Barto, 1998) as goal-oriented learning paradigm. Within three strategies, the self-organised exploration assumes different roles as part of the learning process. First, homeokinetic exploration and greedy RL exploitation are combined in a dual control task. Second, an initialisation of the RL function approximators based on homeokinesis is used as a way of simplifying the search space and for guiding RL. Third, the self-organisation of *Central Pattern Generators* (CPG) via a rewarded correlation average is applied in high-dimensional robots. We compare results with classic actor-critic RL with random walk exploration. In most cases, the SO exploratory strategy increases the learning task performance.

5.1 Introduction

Reinforcement learning (RL) aims at the acquisition of optimal policies to maximise a numerical reward signal (Sutton and Barto, 1998). The learning agent, in our case a developmental robot, tries different actions and receives a reward after each action is completed. Successful applications of RL in robotics domains include gait control (Kohl and Stone, 2004), control over dynamically complex and fast changing state space (Bentivegna and Atkeson, 2001; Ng et al., 2006; Bagnell and Schneider, 2001), as well as humanoid robots learning of template trajectories by natural gradient actor-critic (Peters et al., 2003). Notwithstanding, RL in high-dimensional and continuous

state, action and time systems remains an unsolved problem. The challenges that a classical RL agent in discrete space and low dimensionality faces also extend to high-dimensional robots. Challenges as the curse of dimensionality, (temporal) credit assignment, partial observability, state-action space titling, non-stationary environment, credit structuring and exploration-exploitation dilemma (Sutton and Barto, 1998) are recurrent in robotics RL. Nevertheless, the particularity of RL to find optimal solutions with little knowledge of the system makes it an appropriate candidate for developmental robotics.

One of the key components in RL is exploration. Exploration of the state and action space allows to find optimal solutions by driving the agent away from visited states avoiding local optima. Exploration can be divided in *undirected* and *directed* exploration. Undirected exploration relies only on knowledge related to utility estimates and does not utilise any specific knowledge about the learning process itself (Thrun, 1992). Often, a random walk is used for undirected exploration technique as baseline comparison. The modification of the probability distribution for action selection has been studied in RL, e.g., Boltzmann distributions (Sutton, 1990; Claus and Boutilier, 1998) and semi-uniform distributions (Whitehead and Ballard, 1991; Mahadevan and Connell, 1992). However, these interesting approaches cannot be generalised to continuous domains easily. Directed exploration uses knowledge of the learning process (Sutton, 1990; Schmidhuber, 1991a; Thrun and Möller, 1992). In the case of directed exploration, the agent "directly" explores the states that promise to maximise the learning gain. As an example, error-based exploration depends on the prediction error of an internal model. The RL tends to visit states with high prediction error, in order to try to maximise the knowledge gain. Another type of direct exploration is *recency-based* exploration. In this technique, less recently visited states are provoked by the controller (Sutton, 1990). The assumption is that the control in a non-recently visited state gets worse over time. Nevertheless, neither undirected and directed techniques make use of embodiment. In a scenario where a complex robot interacts with the environment, it is ideal to exploit the interactions between them for learning purpose. The self-organised homeokinetic controller (Chapter 3), already present suitable exploration characteristics (Chapter 4). The generated behaviour is coherent with the robot configuration and the environment response. In comparison with other direct explorations, the self-organised actions are meaningful from an exploration point of view.

The ambivalence of training and self-organisation reflects an important principle in

biological learning. Although the external event distribution is sufficient in many cases to drive learning successfully, there are often intrinsic mechanisms available as a more or less equally successful fall-back option. The organism can rely on such options when the environment deviates from the evolutionarily anticipated standard. However, we will not discuss how organisms deal with this case. Instead, we will focus on the potential benefits of an embodied exploration applied at different learning stages.

The first presented approach promotes homeokinesis as preparation before environmental reward signals are available or while they are not yet critical such as in play in a protected environment (Section 5.3). In robotic applications, the prior-learning scheme can add naturalness to the movements and simplify the search space when the purposeful movements are to be learned subsequently. Only the best actions are propagated from the exploratory mode to reinforcement learning, based on the on-policy value function. The second strategy includes a self-organised closed-loop controller (Section 5.4). This controller, similar to a central pattern generator (CPG), exploits coordination factors that are acquired following the instantaneous reward. Since the control is based on correlations rather than a full action-state space, this approach is able to escalate to high-dimensional systems. The third approach presented in this chapter (Section 5.5) is a dual control scheme. In a classical actor-critic RL setting, actions are promoted by an exploratory signal. In this strategy, the usual undirected exploratory signal is replaced by homeokinetic behaviour. The sensitivity of homeokinesis to the environment response allows an embodied exploration at the same time that the reinforced learning is carried on.

We expect homeokinesis, as unsupervised learning, to yield a discrimination of the search space that can be effectively harnessed by RL. Guiding the exploration with smooth, responsive actions and narrowing the explorable space to where embodied comprehensible behaviour is viable.

We present a comparison of our approaches with a standard version of continuous reinforcement learning (Doya, 2000) in low-dimensionality. In high-dimensionality, we use the direct reward to propagate the correlation between the degrees of freedom. In the next section, we present the continuous time and space actor-critic RL method, following with the details of the proposed approaches and experimental results.

5.2 Reinforcement Learning in Continuous Domains

Most of the interesting robotics control tasks require smooth and continuous actions. Usually in response to high-dimensional, real-valued sensory input. The work in (Doya, 2000) presents an RL framework for continuous time dynamics systems without a priori discretisation of time, state and action. The framework tackles coarse space and time discretisation avoiding non-smooth control policies. Also, avoids the use of fine discretisation thus large memory and many learning trials are not needed. Furthermore, tries to prevent the elaboration of state partitioning that requires prior knowledge. All these attributes are desirable for autonomous developmental robots where control solutions are expected to emerge from experience rather than design.

There are several RL algorithms that focus on continuous domains. For example, (Bradtke, 1993) presents a dynamic-programming based Q-learning algorithm for continuous state, discrete-time systems with linear dynamics. Another approach (Duff and Bradtke Michael, 1995), derive a TD(λ) algorithm for discrete-state and continuous-time for semi-Markov Decision Problems (a continuous time generalisation of discrete time MDPs). The work in (Baird III, 1993) proposes *advantage update* as an extension of Q-learning into continuous time. Other studies extends hierarchical RL (Barto and Mahadevan, 2003) to construct high-level skill hierarchies in continuous domain (Konidaris and Barreto, 2009). An RL framework for continuous time dynamical systems without a priori discretisation of time, state and action is presented in (Doya, 2000). We follow this last framework to self-organise RL in two different approaches (Sections 5.3, and 5.5).

We will use an RL algorithm for non-linear dynamical systems that is based on the Hamilton-Jacobi-Bellman (HJB) equation for infinite-horizon, discounted-reward problems. Consider the continuous-time deterministic system,

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \tag{5.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ represents the state (sensory input), $\mathbf{u} \in \mathbb{R}^m$ is the action (motor command) and $f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ is the function dynamics. We define the reward in time as

$$r(t) = r(\mathbf{x}(t), \mathbf{u}(t))), \tag{5.2}$$

with $r : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}$. The policy,

$$\mathbf{u}(t) = \mu(\mathbf{x}(t)), \tag{5.3}$$

where $\mu : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is the policy function that maps sensory input into motor commands. The prediction of the expected future reward is stored in the *value function* of the state $\mathbf{x}$,

$$V^{\mu}(\mathbf{x}(t)) = \int_t^{\infty} e^{-\frac{s-t}{\tau}} r(\mathbf{x}(s), \mathbf{u}(s)) ds, \tag{5.4}$$

with $V^{\mu} : \mathbb{R}^n \rightarrow \mathbb{R}$ for any initial state $\mathbf{x}(t)$, and with τ the time constant for discounting future rewards. The superscript μ indicates that the value function learns on-policy. The discounted reward guarantee that the value function converges even when the state is attracted to non-zero reward state.

According to the *principle of optimality* (Bellman, 1966), the optimal value function at time t for an optimal policy μ^* (Equation 5.3) is

$$\frac{1}{\tau} V^*(\mathbf{x}(t)) = \max_{\mathbf{u}(t)} \left[r(\mathbf{x}(t), \mathbf{u}(t)) + \frac{\partial V^*(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}(t), \mathbf{u}(t)) \right]. \tag{5.5}$$

Equation 5.5 is a discounted version of the HJB equation. At the same time, the optimal policy is defined as,

$$\mathbf{u}(t) = \mu^*(\mathbf{x}(t)) = \operatorname*{argmax}_{\mathbf{u}} \left[r(\mathbf{x}(t), \mathbf{u}) + \frac{\partial V^*(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}(t), \mathbf{u}) \right]. \tag{5.6}$$

Reinforcement learning defines the processes to bring the current policy μ and value function V closer to their optimal values. In (Doya, 2000) these goals are achieved following two guidelines. First, estimating the value function V based on the actual policy μ. Second, improving the policy μ by choosing greedy actions following the estimate of the value function V.

5.2.1 Value Function Optimisation

For learning the value function in continuous domains, a function approximator has to be considered. We define the estimate of the current value function as

$$V^{\mu}(\mathbf{x}(t)) \simeq V(\mathbf{x}(t); \mathbf{w}), \tag{5.7}$$

where $\mathbf{w} \in \mathbb{R}^p$ is the vector parameter of the function approximator. From now on, we will denote the approximator simply by $V(t)$. Following the TD learning framework, the update of the value function by a self-consistent condition that is local in time and space. The update of the value function can be found, for any policy, by differentiating Equation 5.4 by t as

$$\dot{V}^{\mu}(\mathbf{x}(t)) = \frac{1}{\tau} V^{\mu}(\mathbf{x}(t)) - r(t). \tag{5.8}$$

When the estimate of V is perfect, the consistency condition $\dot{V}(t) = \frac{1}{\tau}V(t) - r(t)$ is satisfied. Otherwise, the estimate is updated to decrease the TD error:

$$\delta(t) \equiv r(t) - \frac{1}{\tau}V(t) + \dot{V}(t). \tag{5.9}$$

Equation 5.9 is the continuous-time counterpart of discrete TD error (Barto et al., 1983; Sutton, 1988). We can define the objective function,

$$E(t) = \frac{1}{2}|\delta(t)|^2, \tag{5.10}$$

which gives rise to an update rule of the weights $\mathbf{w}$ by a gradient descent algorithm. A TD(λ) algorithm is achieved by the incorporation of *eligibility traces* e_i for the parameter w_i. Finally, the update rules for the weights that we consider are,

$$\dot{w}_i = -\varepsilon_V \frac{\partial E}{\partial w_i} = \varepsilon_V \delta(t) e_i(t), \tag{5.11}$$

$$\dot{e}_i(t) = -\frac{1}{\kappa}e_i(t) + \frac{\partial V(\mathbf{x}(t), \mathbf{w})}{\partial w_i}, \tag{5.12}$$

where $0 < \kappa \leq \tau$ is the time constant of the eligibility trace.

5.2.2 Policy Optimisation

In order to improve the policy $\mathbf{u}(t) = \mu(\mathbf{x}(t))$, the associated value function $V(\mathbf{x})$ is considered. In an actor-critic method (Barto et al., 1983) the policy is improved stochastically. In this method, the TD error (Equation 5.9) is used as the effective reinforcement signal.

Consider the policy implemented by the actor as

$$\mathbf{u}(t) = s\left(A(\mathbf{x}(t); \mathbf{w}^A) + \sigma \mathbf{n}(t)\right), \tag{5.13}$$

where $A(\mathbf{x}(t); \mathbf{w}^A) \in \mathbb{R}^m$ is a function approximator with parameters $\mathbf{w}^A$, $\mathbf{n}(t) \in \mathbb{R}^m$ is an exploratory signal, σ is an exploration ratio, and $s()$ is a monotonically increasing output function. Following (Gullapalli, 1990), the parameters can be updated by a stochastic real-valued (SRV) unit as

$$\dot{w}_i^A = \varepsilon_A \delta(t) \mathbf{n}(t) \frac{\partial A(\mathbf{x}(t); \mathbf{w}^A)}{\partial w_i^A}. \tag{5.14}$$

The TD error $\delta(t)$ and the exploratory term $\mathbf{n}(t)$ indicates what direction the update follow. The greedy policy (Equation 5.13) is tampered by the probing signal. If the probing signal has caused the unit to receive a reward signal that is *more* than the

expected reward, then the values should be updated in the direction of the noise. If the probing signal is positive, the weight should be updated so the value increases. Conversely, if the noise is negative, the weight should be updated so that the mean value decreases. In addition, if the reinforcement received is *less* than the expected reinforcement, then the weights should be updated in the direction *opposite* of that of the probing signal (Gullapalli, 1990).

5.3 Self-Organisation of Generic Policies

The initialisation of the parameters of function approximators becomes non-trivial when —as in developmental robotics tasks— an optimal learning speed is required. Ideally, the initialisation should be such that the learning trajectory can follow the gradient without being trapped in undesired optima. Usually, the approximator is initialised with small random values, which in some cases aids the exploration of the state and action space. A random initialisation suffices in environments where the actions exhibit some degree of symmetry so that random walks eventually reach every region of the state space (Jong et al., 2008). Other approaches assign optimistic value to every position of the state space which also provides an initial incentive to explore until values in a more realistic range are found (Sutton and Barto, 1998). However, in most cases these value will not be close to the true expected future rewards. The optimistic values decay linearly while sufficiently exploration takes usually longer. Thus, flexible exploration strategies are worth being considered at least in more complex problems.

We propose another strategy for the use of the homeokinetic controller as an exploratory operator in autonomous robots. In this new strategy, an autonomous learning stage is operated before the reward signals are directly used. The behaviour is utilised to pre-shape the parametric representation of the policy in an actor-critic RL scheme. Part of the strategy includes the use of the SO exploration method to calculate an on-policy estimate of the value function. At this stage, the value function becomes a good indicator of how well the robot will perform a specific task if only the promising actions of the policy are used. Furthermore, the exploration will introduce a bias that can reduce the complexity of the problem by using the information that was inexpensively obtained earlier. In our robotic application, this means that the robot is preferentially guided to regions in the state space where controllability and predictability of the dynamics are high.

5.3.1 Direct Learning of the Actor

Initially, the agent is controlled by the homeokinetic controller (recalling from Chapter 3, Equation 3.18):

$$\mathbf{y}(t) = K(\mathbf{x}(t)) = g(C\mathbf{x}(t) + \mathbf{h}), \tag{5.15}$$

passing the motor signal $\mathbf{y}_t$ to the agent. The motor signals are used to shape the parameters of the RL actor. We implement the actor with a function approximator $A(\mathbf{x}(t); \mathbf{w}^A) \in \mathbb{R}^m$ as in Section 5.2.2. We define the actor's error E^A between the motor commands and the policy,

$$\boldsymbol{e}^A(t) = \boldsymbol{y}(t) - A(\boldsymbol{x}(t); \boldsymbol{w}^A) \tag{5.16}$$

$$E^A(t) = \frac{1}{2} \parallel \boldsymbol{e}^A(t) \parallel^2 . \tag{5.17}$$

The weights $\boldsymbol{w}^A$ of the actor's approximator are updated with a gradient descent algorithm,

$$\dot{w}_i^A = -\varepsilon_H \frac{\partial E^A}{\partial w_i} \tag{5.18}$$

with ε_H the learning rate.

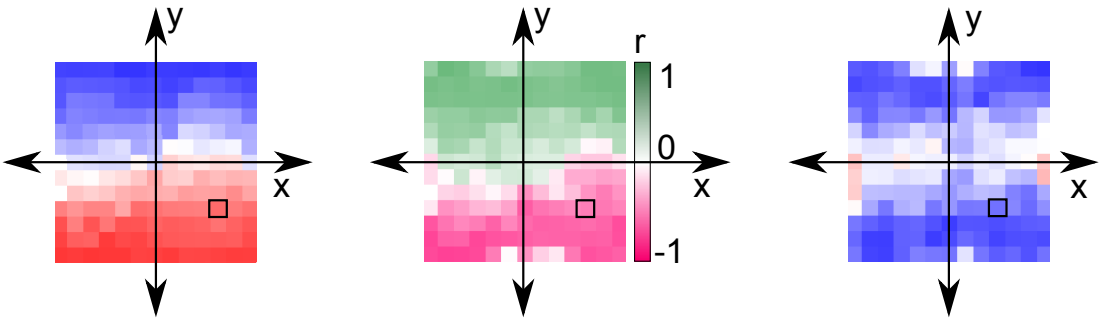
After adaptation, when the function approximator has stabilised, the RL algorithm is activated. The policy is then calculated from the actor's function approximator (Equation 5.13). The RL actor explores with random noise (Equation 5.14).

5.3.2 Self-Organisation for Parameters Initialisation

In order to test the described approach, we will study a control problem for a simulated six-legged *wentelteefje* robot, see Figure 3.8. Initially, the robot is controlled by the exploratory mode. Switching from this mode to RL is triggered after a set amount of time. Other ideas for switching include amplitude and the rate of change of E^A. These and other procedures will be discussed later in this chapter. After exploration is done, we modify the policy based on the value function. When the value function is positive, the policy already showed the suitability to perform the task. Thus, the self-organised policy is directly propagated to the RL policy. For negative value function, we have to decide what values to propagate. A negative value function only informs that the actual policy does not yield high accumulated reward starting in the actual state. It does not indicate in what direction the policy should be improved. One strategy is to propagate a random action different from the exploratory one. Another strategy is to exploit the symmetry of the action space. Instead of random actions, the inverse of the

policy with negative value functions is propagated. We assume that the opposite of the actual command is a better guess than a random action. This strategy will carry the coherence in the behaviour found by the exploration but in the opposite direction of the actions. We illustrate the approach in the next experiment.

To explain the last point, we present an example where the reward is directly related to the y-axis position of one leg of a hexapod. The state space corresponds to the coordinates of the x- and y-axis position of the foot. The policy for any state is the x- and y-axis desired position of the leg. Now consider the motor command only for the y-axis dimension. Figure 5.1a presents the policy for the y-dimension for a discretisation of the state space. The blue colour shows a higher desired position, whilst the red colour is a lower position compared to the origin. The figure presents the exploration policy learned in the exploration mode. Figure 5.1b represent the on-policy value function. The reward r in this example is directly related to the y-axis position. Figure 5.1c shows the propagated values from the exploration signal *filtered* by the value function. The upper half of the value function indicates the favourable performance of the exploratory policy. Thus, this part of the policy is directly propagated to the RL initial policy. The bottom half of the value function presents a poor performance for the exploratory policy. In this case, we promote the inverse of the exploratory policy where the value function is negative.

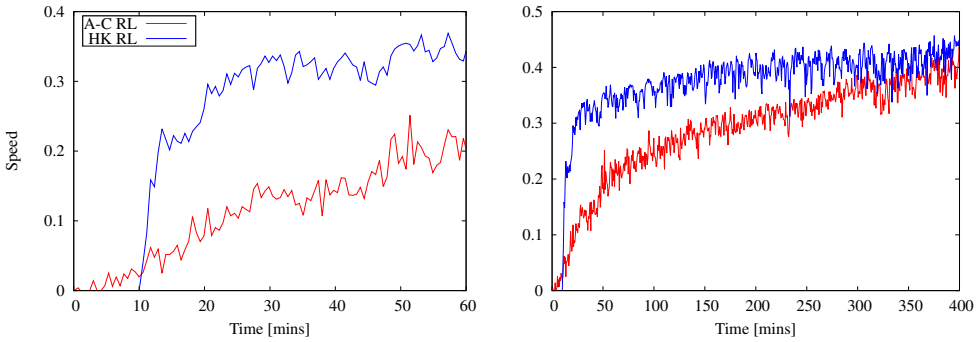


(a) SO policy (b) SO Value func. (c) Initial RL policy

Figure 5.1: Propagation of the exploratory policy through the value function to RL policy. The self-organised policy is propagated from (a) to (c) directly for positive value function (b) values (green). For a negative reward, pink colour in (b), the negative of the SO policy is propagated. The immediate reward is used to build the value function but not the policy. Note that the smoothness of the policy remains in the initial RL policy.

To test the approach, we set-up a walking task in the wentelteefje robot. The reward will be directly proportional to the absolute value of the velocity of the centre of mass

of the robot. In this robot, the legs rotate in a full circular motion. The motor command will control the direction and speed of rotation of the leg. A virtual leg is trained and will form a CPG whose motor signal is transmitted to the rest of the limbs either as an in-phase or as and anti-phase signal. This design lowers the dimensionality of the problem, making it tractable for classical RL. While the random initialisation of the two degrees of freedom may lead to local minima or slow convergence, a smoother function is brought about due to the training with the homeokinetic controller.



(a) (b)

Figure 5.2: Results of reinforcement learning with random initialisation (red line) of the parameters and with parameters shaped by homeokinetic controller (blue line). The six-legged robot receives reward based on horizontal speed. For the homeokinetic approach (HK RL), 10 minutes of exploration were used to pre-train the robot. During this time no rewards is used for control. After exploration, the actor-critic takes control. The same configuration and learning rates were used in both approaches when actor-critic RL took control. With HK RL, maximum reward is obtained significantly faster than classic actor-critic (A-C RL), as shown in (a). When the two systems continue learning, both strategies arrive at a comparable reward level around minute 300 (b). Exploration was taken only for 10 minutes, thus less than 5% of the total learning time was invested in exploration. This strategy allowed reaching the maximum reward in a third (around minute 100) of classical RL development.

Figure 5.2 shows the results for two average runs. For the unmodified actor-critic RL approach, both approximators are initialised randomly. In this case, RL takes control of the robot from the beginning. In our strategy, the robot is controlled by homeokinesis for the first 10 minutes. During this phase, the robot learns the exploratory policy and calculates the value function. After the probing time, classic actor-critic takes control. Figure 5.2a shows the first hour of the trial. In the exploratory time, there is no use

of the reward signal for control. When RL takes control with the learned policy, there is a considerable increase in the reward signal. Figure 5.2b shows a longer time frame window. The classical RL only achieves the same level of reward as the homeokinetic strategy only after 300 minutes.

Homeokinesis exploits embodiment. The produced behaviour is coherent with the robot configuration and its response from the environment. Learning from this type of exploration corresponds to unsupervised learning. A narrow state and action space is selected for exploration only based on input data. These subspaces are comprehensible for the robot and its interaction with the environment. When RL takes control, it is guided through the explored space that, in the best case, has a high reward. In this case, the supervised learning benefits from the exploratory experience. On the worst case, smooth actions are propagated with the only certainty that its inverse does not improve the performance. Moreover, the space reduction has been obtained inexpensively and based on embodied experience.

We should note that the effect of the discovered structure may not always be beneficial for the robot by itself, but the potential misguidance can be diminished by a manipulation of the value function. As the shape of the robot's body distinguishes one of the directions of movement, there is also a bias in the exploration towards the forward direction. If the goal change, e.g., finding a fixed point, then our algorithm would fail to provide a direct advantage. Nevertheless, the filtered propagation of the policy by the value function may still provide a better starting point than random initialisation by inducing smooth actions. Notwithstanding, our results confirm that even if the exploration does not directly bring about a coherent behaviour that will receive a high reward, it can still induce an acceleration of the learning of the task.

The proposed approach has been shown successful in a low dimensionality set-up. Nevertheless, function approximation does not easily generalise to high dimensions unless independence or hierarchical structures can be assumed. However, in robotic problems as well as in biological examples such assumptions are rarely justified. Often, the exploitable structure is not explicitly known. Thus, in the next section, we propose the extraction of reward-weighted behaviour in high-dimensional systems based on homeokinetic exploration.

5.4 Reward-Weighted Correlation

The exploitation of structures that are known in advance have been studied in (Martius and Herrmann, 2011). In the work, a track-like robot "armband" with different mechanical complexity was tested. The study shows that learning time can decrease even for an increase of the mechanical complexity of the robot if the complexity of the control problem was relatively low. The reason for this observation corresponds to the built-in interaction structure and that the robot was less likely to self-obstruct in the high-dimensional case. The speed-up saturated at a few tens of dimensions and the remaining learning time was low due to the homogeneity of the robot's configuration. Here, we study a more complex problem: a hexapod with twelve degrees of freedom that requires a measure of coordination for ambulation or navigation.

5.4.1 Rewarded Temporal Average

Our approach follows (Martius and Herrmann, 2012) by including the reward signal in the extraction of the interaction structure. We consider the correlation between sensory inputs and motor commands,

$$W_{ij}(t) = \frac{\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)(\mathbf{y}_j(t) - \langle \mathbf{y}_j \rangle) \rangle}{\sqrt{\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \rangle \langle (\mathbf{y}_j(t) - \langle \mathbf{y}_j \rangle)^2 \rangle}}, \tag{5.19}$$

where $\langle \cdot \rangle$ denotes a sliding temporal average with time constant τ_W. The vectors $\mathbf{x}$ and $\mathbf{y}$ represents the sensory input and the motor commands respectively with $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$. The matrix W correlates the inputs to the outputs. The relations between the different dimensions can be goal-oriented, thus creating a CPG for some specific target. Equation 5.19 can be transformed into a reward-related quantity by an appropriate weighting based on the rectified reward signal $r^{[+]}$.

$$W_{ij}^{r^{[+]}}(t) = \frac{\langle r^{[+]}(t)(\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)(\mathbf{y}_j(t) - \langle \mathbf{y}_j \rangle) \rangle}{\sqrt{\langle (r^{[+]}(t) - \langle r^{[+]} \rangle)^2 \rangle \langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \rangle \langle (\mathbf{y}_j(t) - \langle \mathbf{y}_j \rangle)^2 \rangle}}. \tag{5.20}$$

The reward signal is bounded to $r \in [-1:1]$, where $r^{[+]}$ correspondes to reward signals only when $r > 0$. In this way, only those sensorimotor couplings that directly contribute to the reward enter the average. The control weights are a smoothed version of the result of Equation 5.20.

$$\bar{W}_{ij}(t+1) = \varepsilon_W W_{ij}^{r^{[+]}}(t) + (1 - \varepsilon_W)\bar{W}_{ij}(t). \tag{5.21}$$

where $\varepsilon_W < 1$ is the adaptation rate.

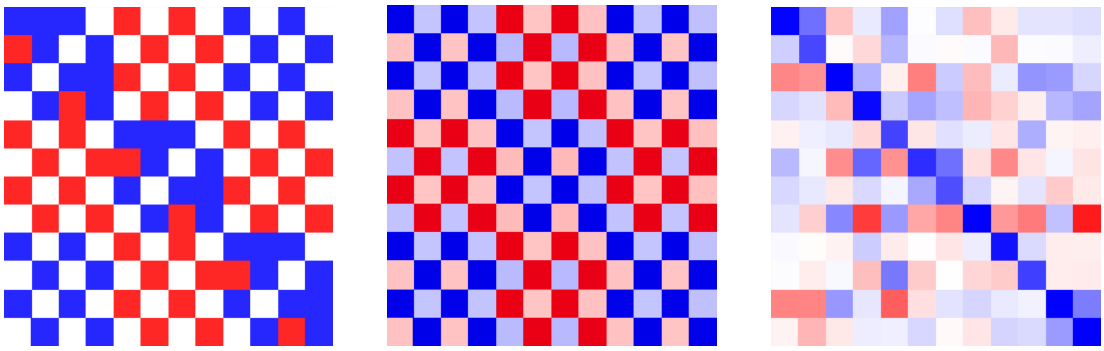
In this approach, we see reinforcement learning in its most basic idea: learning from experience (Sutton and Barto, 1998). The drive of the robot is towards the maximisation of the reward that is meassured after each action. This method is different from the usual temporal difference learning, but still remains as a valid reinforcement learning approach.

5.4.2 Learning Gait Patterns in a Hexapod

In the high-dimensional task, instead of learning with a classic reinforcement learning approach we try to discover the correlation between the different degrees of freedom based on the instantaneous reward. The exploration is produced by the homeokinetic controller, and the learning rule is based on Equation 5.21. The experiment runs on a 6-legged hexapod robot (Section 3.7) with $n = 12$ sensors and $m = 12$ motors. The reward signal is directly proportional to the horizontal speed with a positive value for movements in the direction from the body to the head. Negative values of reward are measured for backward walking.

Closed-loop feedback control is realised by a controller output related to the current input via $\mathbf{y}(t) = W\mathbf{x}(t)$. The CPG matrix can be used to perform an open-loop control of the robot. Also, by a minor phase shift it can control the robot in a closed-loop control. We use a baseline CPG matrix for comparison. The designed CPG controls the robot in a smooth and highly rewarded fashion (Figure 5.3a). In the second case, the feedback matrix was learned from the correlations observed in the first case (Equation 5.19, Figure 5.3b)), i.e. directly from the design behaviour without considering the reward. In the third case, the matrix was learned by the robot while exploring based on a homeokinetic controller (Eq. 3.18, Fig. 5.3c), considering the reward (Equation 5.20). In the images, the sensor inputs are presented as columns and the motor commands as rows.

The designed matrix (Figure 5.3a) generates a tripod gait in the hexapod robot. Note that some relationship between the different degrees of freedom (DoF) have not been set (white squares). The matrix learned from the design one (Figure 5.3b) has acquired the tripod gate. However, the matrix is blurred due to hardware-induced deviation from the ideal interaction matrix and the finding of new correlations. In the same figure, a sub-matrix $Q_{ij} = W_{ij}$ with $i = 1,2$ and $j = 1,2$ represents a 2×2 rotation



(a) (b) (c)

Figure 5.3: The matrix (a) contains the coefficients similar to a coupled CPG, which is sufficient to perform a tripod gait. The matrix on (b) has been learned by the system actuating over a close-loop disregarding the reward. On (c) the re-estimated matrix that was obtained from Equation 5.20 while the robot was exploring with the self-organised probing signal. In Figure 5.4, the reward obtained by the left and middle matrices can be seen. A value W_{ij} represents the relation of the sensor j to the motor command i. For the hexapod robot, every degree of freedom has a positional sensor and one actuator.

matrix:

$$Q = \begin{bmatrix} a & -b \\ b & a \end{bmatrix},$$

with $a^2 + b^2 \approx 1$. The leg produces a circular movement that is congruent with the designed original matrix, also inducing positive reward.

For the third matrix, the most noticeable relation appearing in the matrix is for the W_{ii} weights. This value, close to 1, represents the relationship with a motor command and its sensor position. Some rotational parameters appear at the second ($i = 3,4; j = 3,4$), forth ($i = 7,8; j = 7,8$) and fifth leg ($i = 9,10; j = 9,10$). Still, the appearance of coordinated gaits is not evident. Other collaborations between DoF in different legs are also difficult to identify. In order to get a better understanding of the behaviour of the robot, we will study some gaits generated by the close-loop control. However, first, we present the overall performance of the two learned behaviours.

The Figure 5.4 shows the performance for the two learned control matrices and the pure homeokinetic exploration. As expected, the reward of the designed matrix is consistent and positive throughout the experiment. The pure homeokinetic reward is small and also consistent in time. The SO controller is not promoted to follow any specific ac-

tion other than to explore coherently. That is why the overall reward oscillates around 0. The learned-from-reward exploration controller shows variable performance. There is a bigger amplitude of the reward with this approach. The robot behaves with a broad variety of actions that tends to maximise the reward but still not completely removing all the actions that leads to negative reward. Still, the robot is able to outperform the designed control at some time steps. Furthermore, when the performance is compared to the pure exploration mode, there is a significant improvement. Although the idea is not to compare the rewarded behaviour directly to the exploration performance, it is interesting to see that positive reward actions can be captured from an exploratory reward-less control. Due to the averaging nature of the approach, the matrix W had to be scaled so the robots could enter a closed-loop control cycle.

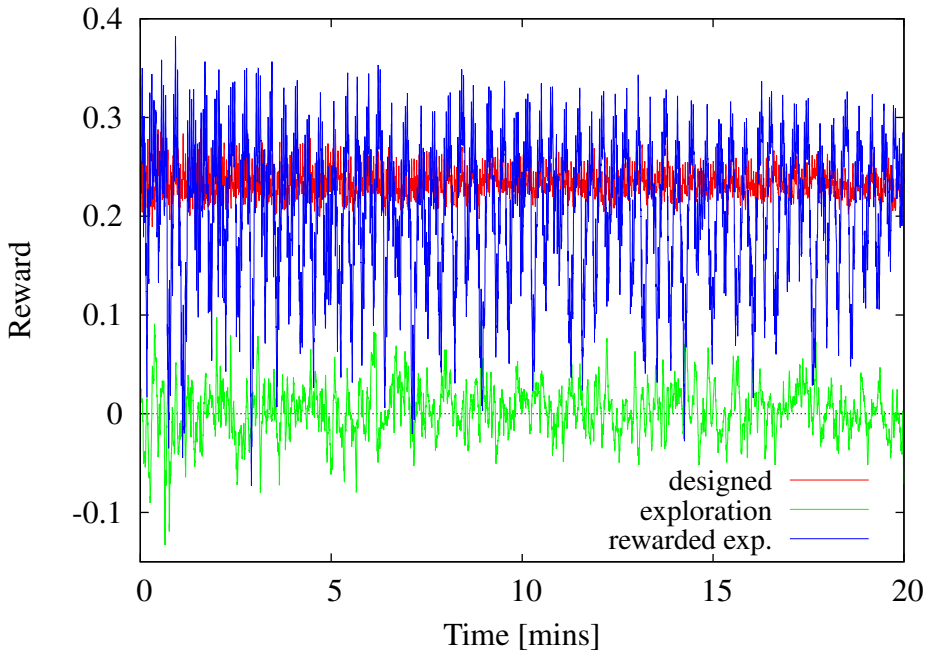


Figure 5.4: Results for the designed correlation, the homeokinetic exploration and the reward exploration. The reward is derived from the forward velocity of the hexapod. As expected, the designed tripod gait CPG (Figure 5.3b) produces a constant locomotion of the robot. Reward obtained with pure homeokinesis exploration is low, the robot is not considering reward under this mode. The correlation learned by rewarded exploration (Figure 5.3c) shows variable results, ranging from higher than designed to zero or even negative reward in some cases.

In order to further characterise the behaviour, we show in Figure 5.5 the positions of one leg of the robot on its x- and y-axis position. The top figure depicts a designed

rotation of one leg. When the y-axis position is minimum, the leg is in contact with the ground. During this period, the leg moves *backward* in the x-axis coordinate, thus impulsing the robot forward. A similar rotation can be appreciated in the middle figure. It represents the behaviour learned from the crafted one. However, the y-axis extension is not total, and when the leg is moving backward, it is not always in contact with the ground, thus losing overall impulse. The figure at the bottom, represent the behaviour learned by exploratory reward. It makes faster loops but the time touching the ground is minimal. As the results of the overall performance display, this behaviour produces a less stable reward maximisation. Furthermore, not every leg present a rotational behaviour. This learning strategy, as reinforcement learning, is affected by credit assignment. We expected that by a longer exploratory time the actions that do not cooperate with the overall reward were taken out of the average. On the contrary, our experiments show that the indirect nature of the reward with respect to a high-dimensional action space made the acquisition of the pure rewarding action cumbersome. As well, actions that entered the average can be counterfeit by the opposite action if the overall reward is positive. Nevertheless, a mostly positive reward behaviour has been learned within a small time of exploration and low computational cost. In Figure 5.6 the same leg trajectories are portrayed. The different use of the state space can be compared to the different controllers.

It is reasonable to think that the learning scheme based on Equation 5.20 will not be effective for all systems. More complex relations between reward and sensorimotor coupling than studied here are clearly possible. Nevertheless, it is not the goal to impose these relations precisely, but rather to introduce a bias into the self-organising system. Thus, any deviations between the true sensorimotor couplings and the relation that is implied by the guidance matrix $W^{r^{[+]}}$ are resolved by the exploratory behaviour of the self-organising controller. However, we should remark that a substantial deviation between guidance and realisable behavioural modes may compromise efficiency although the effectiveness of the control is not usually compromised.

5.5 Homeokinetic Reinforcement Learning

In order to find a control policy for an autonomous robot by reinforcement learning, the utility of some behaviour can be revealed locally through a modulation of the motor command by probing actions. For robots with many degrees of freedom, this type of exploration becomes inefficient. Hence, it is an interesting option to use an auxiliary

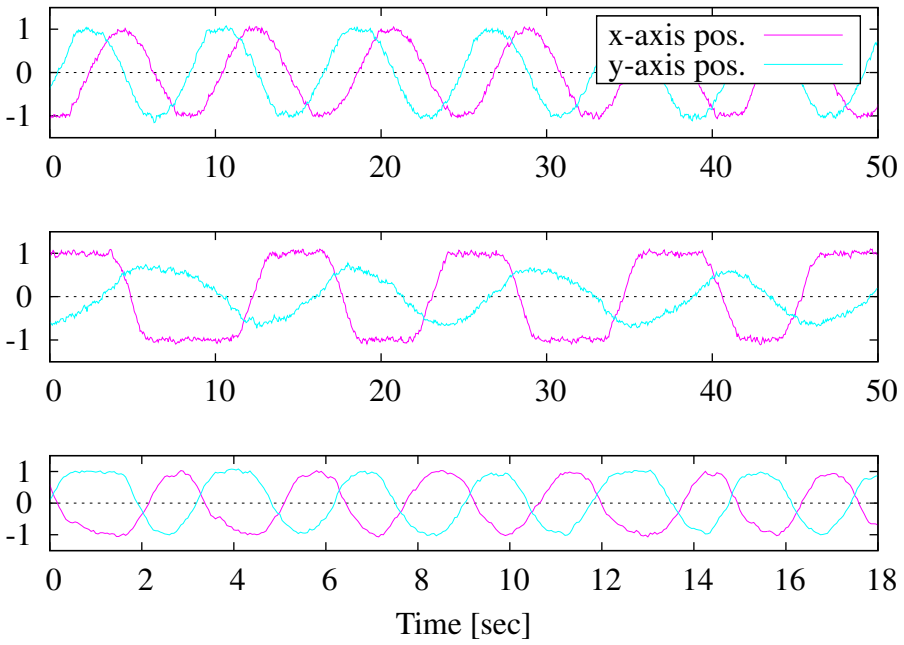


Figure 5.5: Considering a single leg controlled by a designed coupling matrix (top) we observe a phase shift between the horizontal and vertical actuation pattern. The movement of the learned from design leg (middle) does not follow the trajectory precisely, but keeps a similar phase shift. Using the present approach (bottom, Equation 5.21) the movement pattern becomes more smooth which may point to a reduced energy consumption, but the phase shift has increased, the speed of the robot (as implied by the guidance matrix $W^{r^{[+]}}$, Equation 5.20) being in the same range, see Figure 5.4.

controller for the selection of promising probing actions. We suggest optimising the exploratory modulation by a self-organising controller. The approach is illustrated by two control tasks, namely swing-up of a pendulum and walking in a simulated hexapod. The results imply that the homeokinetic approach is beneficial for highly complex problems.

Reinforcement Learning aims at solving dynamical optimisation problems. For this purpose, a utility function and/or a control policy is constructed (Equations 5.5 and 5.6). Optimal performance can be reached asymptotically under certain conditions. Optimal solutions relies on at least three assumptions that are hard to accomplish in practice: (1) we accurately know the dynamics of the environment; (2) we have enough computational resources to complete the computation of the solution; and (3) the Markov property (Sutton and Barto, 1998). Also, because the often slow decay of the learning rate in practical problems, only suboptimal solutions are found.

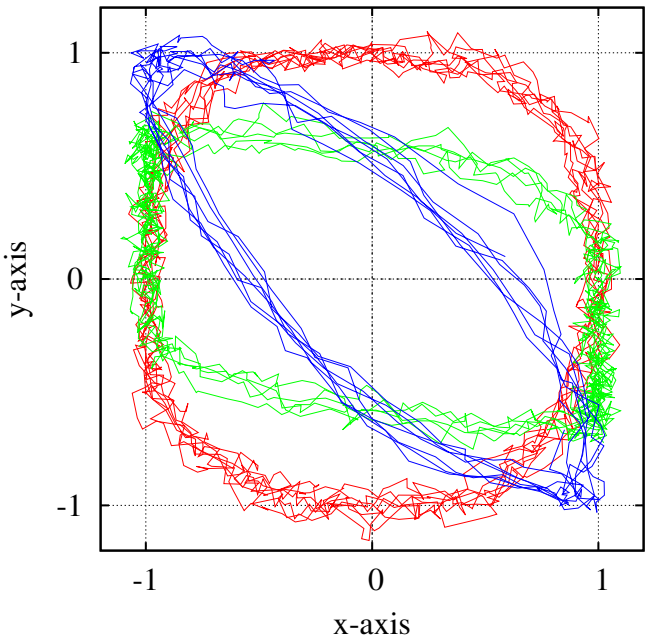


Figure 5.6: Configuration space representation of the trajectories from Figure 5.5. The red line represents the designed matrix, the green line is for the behaviour learned by following the first case without reward and the blue line represents the rotation learned by the system following Equation 5.21.

Exploration of the state space in high-dimensional system is an intractable problem due to the exponential growth of the space with respect to the dimensions. Gradient-based reinforcement learning can speed up the optimisation process, but is prone to local optima. Moreover, if the gradient is not known, then probing actions must be used in order to obtain gradient information. High-frequency probing (Wiener, 1948) tests two alternative actions virtually at the same time that seems appropriate for an autonomous agent which may not be able to apply different actions in the same state. In addition, the set-up of the probing actions requires some domain knowledge and becomes cumbersome in high dimensions. Furthermore, the priorities that should be used when sequentially probing the manifold of behaviours in robots with many degrees of freedom is unknown. A robot with an RL controller is biased to keep trying the path that is expected to give him the best reward in the future. Thus, seemingly non-rewarding nearby states are less likely to be explored.

We propose to use homeokinesis as an auxiliary algorithm that learns to probe the system. For this purpose, we will not follow the gradient of the utility function, but we will aim at maximising the learning success achieved by the probing actions.

This maxisimisation will help to obtain a more reliable representation of the utility function in shorter time while the reinforcement learning component will handle the actual increase of the expected reward.

As presented in Chapter 3, homeokinesis generates motors signals based on estimated next sensor values. The RL controller will use this signal as exploration mode and to update the parameters values in an actor-critic configuration. We replace the probing signal $\mathbf{n}$ in the actor's policy calculation (Equation 5.13) with the homeokinetic control command (Equation 3.1).

5.5.1 Self-Organised Reinforcement Learning Architecture

We present the dual architecture in Figure 5.7. The actor-critic RL, on the left side of the figure, produces the main component of the action. The actor A responds to the environment sensation $\mathbf{x}_t$ generating the policy action $\mathbf{u}_t$. The robot executes the action and receives new sensory information $\mathbf{x}_{t+1}$ from the environment. The reward $\mathbf{r}_{t+1}$ is calculated based on the actual state by R. Then, it is compared by the expected reinforcement V_{t+1} for the instantaneous state by the critic C. Following Equations 5.11 and 5.14, both the actor and the critic are updated. On the right side of the figure, a simplified view of the homeokinetic controller is presented (for the full scheme check Figure 3.1 in Section 3.2). The controller K inserts the homeokinetic signal as the actor's probing characteristic of the policy.

It is essential to the approach followed here that the robot monitors the full loop through the environment. This loop can be represented by a map of previous to new sensor values, but as well as a map from previous to new motor commands. The latter case is more convenient since often the dimensionality of the motor space is lower than that of the sensor space.

This control architecture corresponds to a closed-loop dual-control system (Feldbaum, 1961). Such systems have two mutually exclusive control signals. On one hand, the exploitation mechanism acts by following the actual knowledge of the system. On the other hand, exploration drives the system towards unknown states. Finding a balance between exploration and exploitation, in Bayesian terms, for optimal control is, in general, intractable (Dayan and Sejnowski, 1996). Some first proposed techniques include optimistic initial values (Moore and Atkeson, 1993), encouraging the system to take an action depending on how long has been since the agent tried that action in that state (Sutton, 1991). Other techniques propose the use of a world uncertainty statistical

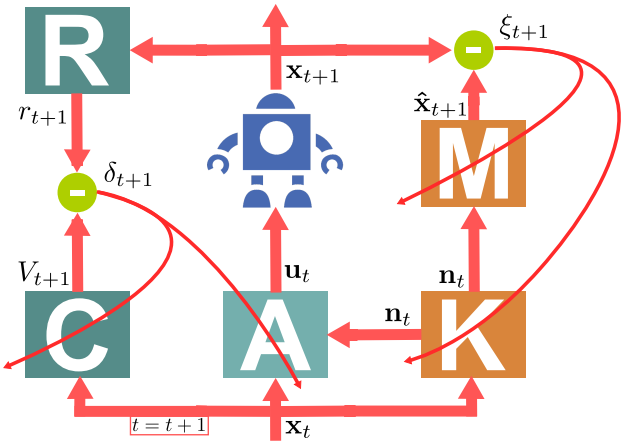


Figure 5.7: Architecture of the sensorimotor loop. The output $\mathbf{n}$ of homeokinetic controller K is used as the probing signal of the actor-critic RL. The actor A calculates the actual policy based on Equation 5.6. With the next time step state $\mathbf{x}_{t+1}$ the reward is calculated, and the critic C and the actor are updated following δ_{t+1}.

model as a way of exploration (Dayan and Sejnowski, 1996). Most of these models are not based on how the agent is uncertain about its world. Moreover, none of them bases the exploration on embodiment, as homeokinesis does. This last characteristic is what makes our approach suitable for complex interactions.

5.5.2 Experiments

We present a comparison of our approach with a standard version of the continuous time and space RL presented in Section 5.2. We assess the architecture with two tasks: first, swinging up a pendulum with limited torque (Doya, 1996), and second, a hexapod robot with twelve degrees of freedom. In the later case, the reward is the absolute value of the sagittal walking speed.

Various techniques have been proposed in order to approximate the relevant functions, e.g. kernel-based methods (Xu et al., 2007, 2002), normalised Gaussian networks (Sato and Ishii, 2000; Doya, 2000), Fourier basis function (Konidaris, 2008) and echo state networks (Jaeger, 2001; Szita et al., 2006). In our experiments, the RL actor and critic functions are implemented as a normalised Gaussian network (Section 4.2.2.3). The sigmoid function g for the RL actor is defined as,

$$g(x) = \frac{2}{\pi} \arctan\left(\frac{\pi}{2} x\right). \tag{5.22}$$

In the classic RL approach, the probing function is given by low-pass filtered noise:

$$\tau_n \dot{\mathbf{n}}(t) = -\mathbf{n}(t) + N(t), \tag{5.23}$$

where $N(t)$ is normal Gaussian noise.

The strength of the probing signal is weighted by σ. The idea is that while the reward becomes bigger, the probing input should become weaker (Gullapalli, 1990). The value is calculated by

$$\sigma = \sigma_0 \min\left\{1, \max\left\{0, \frac{V_1 - V(t)}{V_1 - V_0}\right\}\right\} \tag{5.24}$$

where V_0 and V_1 are the minimal and maximal levels of the reward. The homeokinetic controller follows the same configuration presented in Section 3.2.5. The learning rates for the self-organised controller are similar to those used in the previous experiments in Chapter 3. Both pendulum and robot are realised in the LpzRobots simulator (Martius, 2012).

5.5.2.1 Performance in a toy Example

The first task is the pendulum swing-up task. This experiment has low dimensionality, but the dynamics are complex enough to make it a viable test-bed. The number of sensors is $n = 2$, for position θ and angular velocity ω. There is one actuator ($m = 1$) that applies torque $T(t) = u(t)$. The task of the controller is to bring the pendulum to the upright position (see Figure 5.8 recalling it from Chapter 3).

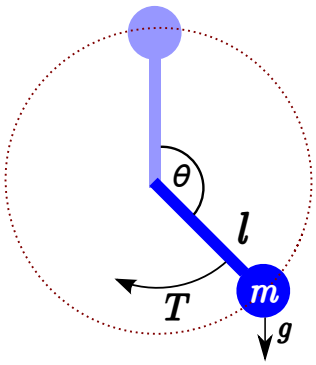


Figure 5.8: Swing-up pendulum with limited torque. The physical parameters are $m = 1$, $g = 9.8$, friction $\mu = 0.01$, $u^{\max} = 5.0$. The parameters of the RL controller are $\tau = 1.0$, $\sigma_0 = 0.5$, $V_0 = -1$, $V_1 = 1$, $\triangle t = 0.02$. The learning rate for the homeokinetic controller are $\varepsilon_A = 0.05$ and $\varepsilon_C = 0.01$.

For the pendulum swing-up task, we use the same configuration as in (Doya, 2000). The actor and the critic function are implemented by an RBFN with a 15×15 grid where the centres are evenly spaced between the dimensional ranges. The range of the angular position θ is $[-\pi, \pi]$ on the vertical line, and the angular velocity range ω is $[-2\pi, 2\pi]$ per unit time. The state-space corresponds to $\mathbf{x} = (\theta, \omega)$. The reward function is defined as,

$$r(\theta) = \cos(\theta), \tag{5.25}$$

it assumes the maximum value at the upright position and the minimum at the downward position of the pendulum.

Several trials are executed for each RL architecture. Each trial lasts for 20 seconds as long as the pendulum do not take more than two and half full circle turns in the same direction ($\mid \theta \mid < 5\pi$). If the accumulated angle surpasses this maximum value we assume that the trial has failed. At the end of a miscarried trail, a minimal reward is given for one second, and the trial is reinitialised in a random state. The performance of the trial is measured by the time when the pendulum is in the range $\mid \theta \mid < \pi/4$.

The policy, following Equation 5.13, is implemented as

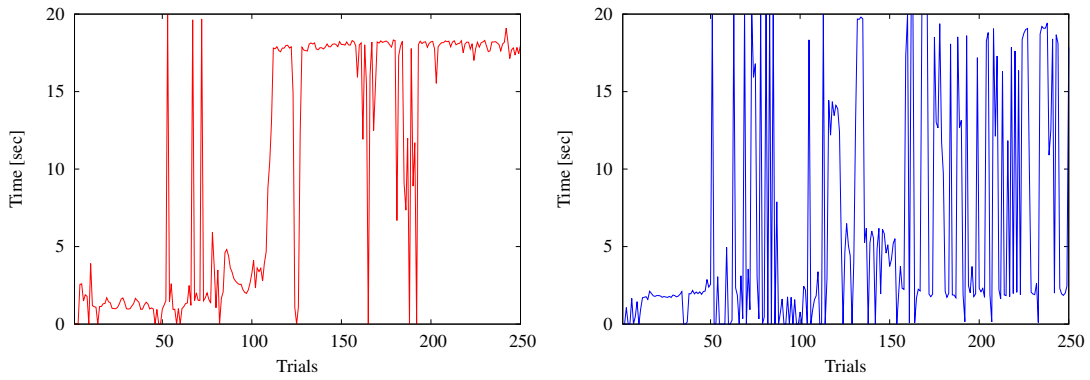
$$u(t) = u^{max} s\left(A(\mathbf{x}(t); \mathbf{w}^A) + \sigma n(t)\right), \tag{5.26}$$

with $u^{\max}$ the maximum applicable force. The swing-up pendulum problem is not trivial when $u^{\max} < mgl$ (See Figure 5.8). When the maximum force is smaller than the maximal load torque, the system cannot be driven arbitrarily from one position to another. Especially when starting from the resting position, $\mathbf{x} = (\theta = \pm\pi, \omega = 0)$, the pendulum has to build up momentum to be able to reach the highest position. Also, the system has to decelerate accordingly to not overshoot past the upright position.

The set-up of the homeokinetic controller is standard, with linear model and controller, and learning rates $\varepsilon_A = 0.05$ and $\varepsilon_C = 0.01$. With these learning rates, we promote a faster learning of the internal forward model compared to the dynamics of the controller. Note that no further tweak of these parameters is required as the homeokinetic controller update its internal values based on the response of the robot with the environment.

In Figure 5.9 we show two single set of trials. In Figure 5.9a, the classic actor-critic RL presents a better overall reward maximisation over all the trials. After finding a stable policy (around trial 120), the greedy actions are favoured. In the case of the homeokinetic RL (Figure 5.9b), the controller finds some beneficial solutions around trial 70. However, the controller keeps exploring systematically. The homeokinetic

controller, in general, avoids stagnation at fixed points as the maximum reward position for the pendulum. Even if the exploration is tampered by σ (Equation 5.24), the homeokinetic controller changes its internal parameters to increase sensitivity.



(a) Actor-Critic RL (b) Homeokinetic RL

Figure 5.9: Results of 250 trials of a single run of the pendulum swing-up task. The y-axis represents the time when the pendulum is in the range $\mid \theta \mid < \pi/4$. In (a) the actor-critic RL finds an almost optimal policy around trial 100. After that, RL applies a greedy action selection. In Figure (b), the homeokinetic controller rejects the optimal solution by augmenting the exploration signal. Thus, it continues exploring even after successful attempts around trial 70.

A comparison between the average results for the actor-critic RL controller and the homeokinetic controller are shown in Figure 5.10. The RL controller swiftly learns the swing-up task. The slope of the average reward is steepest; a stable performance is reached earlier and the total time spent in the upright position is longer. Interestingly, the SO controller never reaches a higher count of maximally rewarded states. Despite the low attained reward, there is a performance improvement. The improvement can be seen by the increase of the reward average further in time. Moreover, the detailed view from Figure 5.9b shows trial with maximum reward. These results evidence that the homeokinetic RL controller continues to explore new states even if the maximum of the reward function has been already discovered.

In Figure 5.11, we measure the performance of the homeokinetic RL architecture in dependency to the exploratory rate σ_0. With smaller values the importance of the probing signal in the actor's action decreases. On one hand, this restraint means that the whole controller has a smaller exploratory signal. Thus, the system tends to act in a more greedy way. On the other hand, the homeokinetic controller *feels* less the

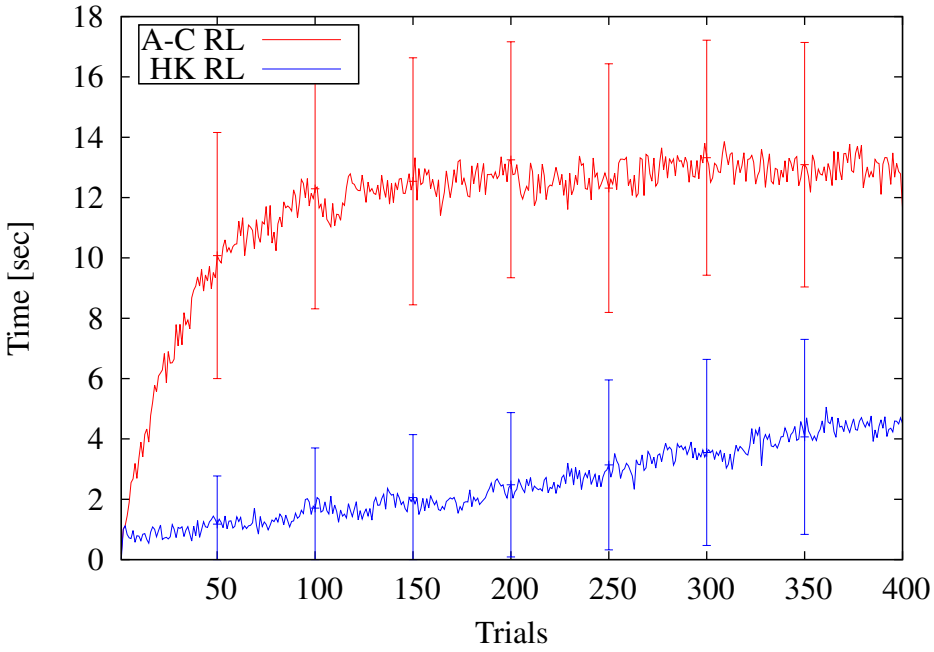


Figure 5.10: Average of 150 experiments, each with 400 trials of the swing-up task with actor-critic RL and homeokinetic RL (lower trace).

environment. Its actions are not significant on the policy. Thus, the next state depends in a lesser way on it. Since homeokinesis relies on feedback to regulate sensitivity and predictability, it fails to perform an embodied experience for the robot. Small σ_0 for random noise exploration also produces poor performance. This case is explained by the lack of a probing signal where the agent only explore by the non-optimal initialisation (that also affects homeokinetic exploration).

Because learning is driven merely by the correlation between exploratory action and utility function consistency, the results for this low-dimensional problem are little impressive. On the contrary, in high dimensional tasks, where exploration is a less trivial problem, the SO controller will allow the robot to keep exploring. Such that, local maxima of the expected reward or regions and directions with low gradients can be avoided easily.

5.5.2.2 Self-Organisation of Walking in a Hexapod

The hexapod has already been introduced in Section 3.2.5.2, Figure 3.7. It resembles an insect with three pairs of legs, two antennae and a thorax. A two-axis joint is placed where the legs meet the thorax allowing vertical and horizontal displacement, herein a servo motor actuate over each axis of the joint. In every axis, a sensor measures the

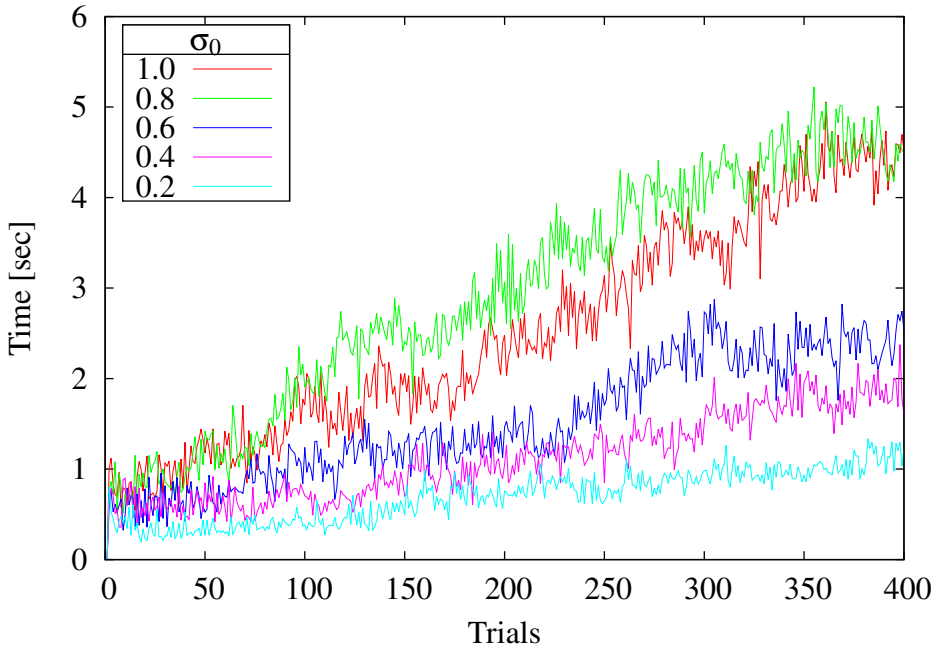
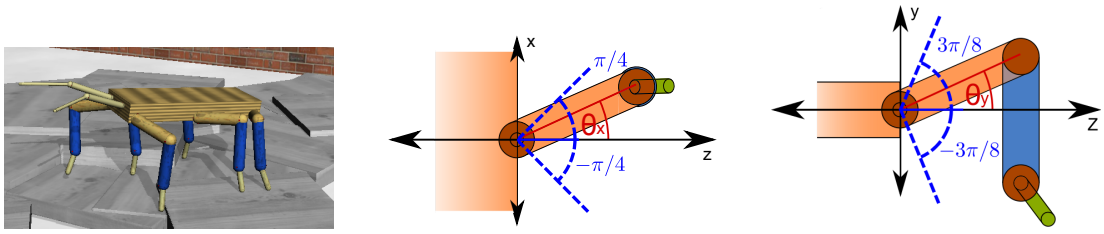


Figure 5.11: The effect of σ_0 in the homeokinetic RL architecture. At low values, the homeokinetic controller is not able to efficiently use the environment. The exploration renders useless for small probing rates.

angle θ with respect to the initial position and another sensor measures the angular velocity ω of the leg. The joint between the femur and the tibia passively actuate in one axis with a damping action as springs. The hexapod is a high-dimensional robot with 12 degrees of freedom ($m = 12$) and 12 sensors ($n = 12$) that defines a large state-space.



(a) LpzRobots simulator (b) Top view (c) Front view

Figure 5.12: Hexapod robot, 12×12 dimensions, the simulator version (a) adapted from (Martius, 2012). Figures (b) and (c) show the angles measured at each joint. A passive spring is attached to each tarsus.

The reward of the hexapod is the sagittal plane speed. Due to the symmetry of the robot no particular movement direction is implied, i.e. in some trials the robot

choose the direction of the antennae and in other the opposite direction. The set-up of the experiment is similar to the pendulum. A trial of 20 seconds is conducted by the controller, after that time the position and the velocities of each leg are set randomly. The performance of each trial is measured by the average speed of the trial. Again, normalised Gaussian networks are used as basis functions independently for each axis leg with 15×15 centres in the range $\theta_x \left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$ for horizontal displacement, and $\theta_y \in \left[-\frac{3}{8}\pi, \frac{3}{8}\pi\right]$ for vertical movements. The range of velocity for both axis is $\omega_x, \omega_y \in \left[-\frac{5}{4}\pi, \frac{5}{4}\pi\right]$. The maximum displacements of the legs are set, so they do not interfere with each other and to avoid over-extension, as can be seen in Figure 5.12. The maximum reachable speed of each degree of freedom was calculated based on the power of each motor joint.

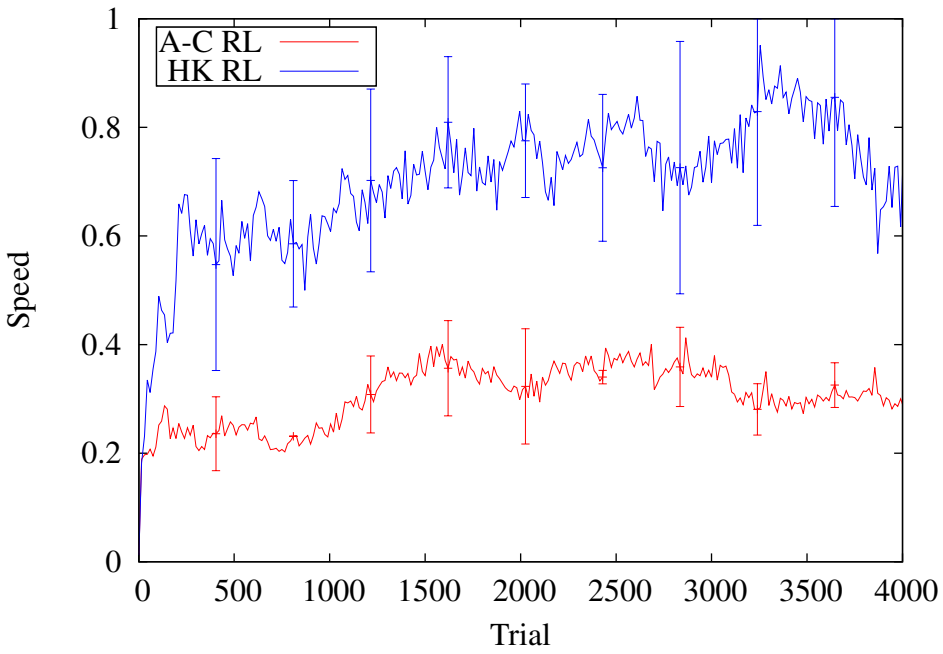


Figure 5.13: Average speed for the hexapod with actor-critic RL controller and with the homeokinetic (HK) controller during 4000 trials. Learning rate for homeokinesis are $\varepsilon_A = 0.05$ and $\varepsilon_C = 0.1$. The error bars indicate deviations over three runs for each control task. The prevalence of the HK method is seen as in this experiment the reward is in concordance with the exploratory mode of the controller. The AC method is not able to catch up with the improved reward of the HK implementation as the exploration is only random noise and does not include information from the interaction of the robot and the environment.

The results for the hexapod with actor-critic RL and the combined architecture

of homeokinesis and RL controller are shown in Figure 5.13. The self-organised RL controller shows better performance after a quick improvement in the first trials. For the proposed architecture, the average speed is higher than the maximal speed that was achieved by the actor-critic RL controlled robot. In this high-dimensional task, the role of exploration is critical for learning. The reduction of the state space due to coherent emergent behaviour has an overall benefit.

In this section, we have presented an integration of two approaches to the unsupervised generation of behaviour in robots. The interaction is based on an objective function that maximises the sensitivity of the learning systems with respect to mismatches in the utility function while simultaneously an RL component aims to maximise the future reward. We have tested our approach with two exemplary tasks of different complexity and have shown that:

- The exploration induced by the SO controller may counteract the reward maximisation in an optimally tuned low dimensional task, while,
- the SO controller seems to aid the learning process by guiding the exploration in a high dimensional task, and that,
- the variable coherence of the action modulation in the SO controller improves the capability of the algorithm to escape local minima and flat regions of the goal function.

For comparability of the two variants of learning, we ran the experiments with restarts after each trial. The state reposition is necessary for RL with random exploration, but it is not required in the self-organized variant. If a stable performance is reached at any local or global optimum the sensitivity of the SO controller increases until the state of the systems escapes from the stationary behaviour. Restarting may not be an option for an autonomous robot such that an SO controller may be required here also for practical reasons.

5.6 Discussion

In this chapter, we have shown three different ways of using a self-organised behaviour control for autonomous robots in combination with reinforcement learning. The main goal of these approaches is to exploit self-organised behaviour as exploratory signal. We compared our approaches against random noise and designed solutions. In the

first approach (Section 5.3), the control duality is solved by a temporal separation of exploration and exploitation. The system takes advantage of the homeokinetic exploration prior to the use of RL. The homeokinetic unsupervised learning policy can be assessed on-line by the value function. The on-policy learning of the value function allows discriminating potentially maximal-reward policy from those with low performance. After the exploration, the policy is used as the initial configuration of the RL actor. Our experiments show that this initialisation promotes a faster reward maximisation. This approach can be compared with other types of value initialisation. For example, an optimistic initialisation of the value function (Sutton and Barto, 1998). This method initialises the value function to higher values than the maximum expected reward for the optimal policy. After a visit to the state, the value function is updated. Thus, the policy will prefer unexplored states. This strategy promotes a more active exploration of the space compared to a random or minimum value initialisation. Still, no knowledge of the interaction between the agent and the environment is used. In large environments, this strategy is not suitable because of the infeasibility to visit every state. Also, a higher maximum reward is obtained at expenses of longer learning times. In the problems that we are interested, developmental robotics, homeokinesis provides a suitable way of exploring coherently without a prior knowledge of the system dynamics.

We have chosen to shape the RL actor and critic since they take control immediately of the RL agent. However, there is a direct implementation for this strategy to *model-based* RL (Sutton, 1990, 1991). In this approach, a forward model of the environment dynamics is acquired during RL development. The model is used to *plan* policies that will yield in higher future reward. Homeokinesis exploration can be used to train the RL forward model. Even more, the very same homeokinetic forward model can be used directly if an extended configuration is used.

Switching from exploratory mode to RL is an important aspect in this strategy. Exploration should end when the acquired information guarantees the discovery of the optimal policy by RL in minimal time. Of course, it is not possible to know a priori if these constraints will be met. Still, there are characteristics of the system that could be used as a signal to end the exploration. The amplitude or rate of change of the error E^A (Section 3.3) can give an indication that the actor has already learned the policy. A problem with said strategy is that a local error minimisation does not imply global learning. To ensure a non-local learning, we can use a temporal threshold of low error as benchmark for model learning. In practice, we used a fixed time of exploration

before switching to the RL mode. As a rule, at least enough time should be given for the models to learn. Short exploration time do not improve the learning times further since only a small portion of the search space has been visited. Longer than optimal exploration time also do not seem to improve further the learning times. In this case, the interesting states have been already been visited. Successful exploration time depends on the complexity of the system. For larger state and action space, there are other possible coherent behaviours to explore. The actor's policy is only one from all possible behaviours that homeokinetic produces.

The second strategy (Section 5.4), evaluates the correlations of sensory data and motor commands between DoF based on a high-level reward. The idea was to learn correlations instead of direct commands, thus allowing the inspection of systems with high-dimensionality. The obtained results show that starting from a pure homeokinetic behaviour and only extracting positive rewarded behaviour the system can achieve an overall improvement of performance. However, the results compared to a designed solution are not optimal. As well, more intricate relations between reward and sensorimotor coupling that the ones found by the learning strategy are possible. Nevertheless, the goal of the strategy is to introduce a bias on the exploratory behaviour implied by the guidance matrix $W^{r[+]}$. Still, some incipient gait-like behaviour is discovered, e.g., the rotation of a leg with two independent DoF. The discovery of such modes represents an effective information extraction. Similar to the proposed initialisation of the actor and critic, the system can use the correlations as a guidance of classic RL.

In the third strategy (Section 5.5), both exploration and exploitation occur at the same time. The robot takes greedy actions with respect to its actual knowledge but with a quota of exploration. Our experiments show that for high dimensionality, the system is favoured by the homeokinetic learning rule. One objection could be that the actions are only one part of the actual motor command realised by the robot. Nevertheless, the RL policy tends to homogenise with learning, i.e., for any position in the state space the actor provides a similar action with comparable response from the environment. Even more, only subtle change happens upon every visit. Hence, homeokinesis takes the RL action as part of the response of the environment and adjust its internal parameters to maximise sensitivity and predictability, maintaining the homeokinetic regime.

The main contribution of this chapter is to show that a coherent exploratory behaviour yields a better performance in goal-oriented tasks when compared to uninformed probing signals. Moreover, a self-organised controller can provide said behaviour inexpensively. The self-organising nature of the method allows its implemen-

tation in high-dimensional robots without explicit knowledge of its configuration and the environment. This characteristic makes it appropriate for developmental robots. Our strategies show an increment of the performance in the majority of the experiments that we carried on. Only, when the homeokinetic exploration was carried on altogether with RL and with a fixed point as a maximum reward, the results showed the worst performance. The kinetic nature of homeokinesis does not aid in stabilising around fixed points when the exploration and exploitation are combined in the same policy. Nonetheless, a disengagement of the homeokinetic exploration from the RL strategy can result in increased performance even for an incompatible high-level reward. In this case, our strategy serves as a feature extraction mechanism. Comprehensible behaviour and smooth transitions are basic building blocks for autonomous robots. Self-organised homeokinetic behaviour can obtain these structures, and then they can be exploited by goal-oriented learning strategies, as in this case reinforcement learning.

The SO exploration does not only provide a better probing signal in high-level behaviour, but also it provides smooth actions across the action space. These actions allow for a minimisation of the temporal credit assignment problem in RL. Neighbouring state space actions are closely related, thus, consecutive and continuous actions will usually also share related reward signals. Hence, reward obtained from temporally distant actions still relate to the actual actions. Another indirect advantage of the SO exploration is the dimensional reduction. Homeokinesis exploits the correlations between DoF, which later guides RL towards actions that have already proven to be compatible with the robot and the environment. Therefore, narrowing the exploration over an intractable environment.

One of the characteristic of homeokinesis is the fast adaptation to the environment as well as to changes in the configuration of the robot. If the robot were under sudden changes in its configuration, e.g., weight of some body part or the length of the legs, the controller will adapt in a bounded set of steps to the new configuration. The robot will maintain the exploratory characteristic. On the contrary, for the reinforcement learning part of the control, depending on the degree that the change affect the interaction of the robot and the environment, there will be a slower change on the policy. The policy does not implies generalisation. Thus, the actor and critic have to relearn for the new configuration.

We believe that this kind of strategies can lead to the advance of developmental robotics. The results are promising and the test in real robots is part of the next steps

in the line of research. The exploration of the environment in a free goal-less way based on embodiment, combined with high-level goal-oriented learning methods can be a key element in successful developmental autonomous robot.

Chapter 6

Conclusions

The main objective of this dissertation has been to study the self-organisation of internal models to improve development in autonomous robots. Our first step towards such goal was to assess different internal model implementations. We appraised models with different complexities, finding that the linear learning class provides optimal environment coverage for a homeokinetic robot. Later experiments showed that reducing the parameter space by embodied emergent behaviour aid learning in goal-oriented tasks. The internalisation of interaction dynamics by the internal models due to coherent behaviour prior to learning increases markedly the performance gain (Section 5.3.2). As well, dual-control with goal-oriented exploitation and self-organised bearing, benefit complex dynamical robots by guiding exploration in a comprehensible manner, e.g., pertinent to the robot configuration and the environment (Section 5.5.1). Also, we showed how high-level coordinated behaviour can be learned by extraction of rewarded sensorimotor coupling in intrinsically motivated behaviour (Section 5.4). We presented the novel use of artificial curiosity in the sensorimotor loop as a developmental driver in autonomous robots, showing its tendency to stagnating in homeostatic dynamics (Section 3.3.2).

In order to understand the paradigm of autonomous robot control, we studied the dynamics of the sensorimotor loop in Chapter 2. Following dynamical systems theory, we presented the fixed points, stability and bifurcation analysis both with and without bias parameter. The dynamics study was extended to homeokinesis in Chapter 3 as a manner to understand its implications in emergent behaviour. Also, we detailed the notions of self-organisation, guided SO and internal models in Chapter 2.

In Chapter 3, first, we studied self-organisation of behaviour based on the homeokinetic principle. This part of the chapter introduced the homeokinetic system dynamics

and the notion of how two opposite forces prompt an autonomous robot to maintain the dynamic range at the edge of chaos. Later in the chapter, we presented our approach to intrinsically motivated behaviour based on artificial curiosity. The idea was to analyse the suitability of artificial curiosity as a driver of behaviour in the sensorimotor loop. In the approaches (Schmidhuber, 1991b; Herrmann et al., 2000; Oudeyer et al., 2005), already model learning gain has been applied in robotics settings. Nevertheless, they include some external perturbation to incentive exploration, e.g., random state reposition. The main result in our approach was seemingly homeostatic dynamics, leading the robot towards stagnation. Even the introduction of an exploratory term did not solve stagnation and the robot was not able to maintain a dynamical working regime. A suitable probing mode remains as part of future work.

In Chapter 4, we evaluated how exploratory behaviour depends on different forward models configurations. In homeokinesis, this dependency is due to the controller adaptation based on the prediction error and the response of the model. The experiments included linear models with different learning rules and non-linear models in a homeokinetic robot. In this case, the most favourable behaviour requires flexibility to adapt to various situations and persistence to traverse most regions of the environment. We measure the performance as Pareto optimality for minimum prediction error and maximum environment coverage or traversed distance. Echo-State Network and a linear model with gradient decent learning rule showed the best performance. Both methods present fast adaptation to new situations and minimal prediction error when compared to the other methods. Further experiments showed that the performance of the ESN saturates with the complexity of the model when testing different sizes for the reservoir.

Three different approaches to guided self-organisation were presented in Chapter 5. First, we designed a novel dual-control architecture for exploration and exploitation. In this architecture, goal-oriented actions are promoted by reinforcement learning while embodied actions supplied by homeokinesis contribute towards exploration. In our experiments, results were dissimilar depending on the complexity of the system and the task. In a low-dimensional set-up with an unstable fixed point as task objective, a random probing signal delivered faster reward maximisation. On the contrary, on a high-dimensional system with mobility as the objective, our approach brought a swift and consistent increase of reward. This difference is due to two aspects. First, coordination of different degree of freedom (DoF) in a low-dimensional system can be found in linear time by exhaustive examination of the entire action space, or like in our

pendulum experiment, there is only one DoF. In a high-dimensional system, random exploration cannot handle the vast search space, but homeokinesis can bring about such coordination exploiting embodiment. Second, homeokinetic behaviour keeps the robot exploring the dynamical range by maximising the sensitivity, making it an antagonistic drive with respect to a fixed point goal.

In the second approach, we propose the self-organisation of parameters of the internal models prior to goal-oriented learning. We showed that the use of homeokinesis improved reward gain in robotics reinforcement learning when used for capturing relevant embodied behaviour. Homeokinesis provides an *inexpensive* policy, i.e., fast emergent behaviour without the need for expert knowledge, that narrows the action-state space. Also, if the reward signal is considered, the value function already captures the return of the homeokinetic policy. This reduction in the search space yields a faster reward maximisation and can be a key feature in complex developmental robotics.

The third approach presented in Chapter 5 promotes positive-rewarded coordination among the DoF for closed-loop control. This approach shows that effective coordination can be extracted from self-organised behaviour in high-dimensional systems by a temporal rewarded average of sensorimotor couplings. Extraction of useful coordinated behaviour is especially important in complex systems where the behavioural space is intractable.

The work presented in Section 5.5.1 has been published in (Smith and Herrmann, 2012). Also, the work presented in Section 5.3.2 and Section 5.4 has been published in (Smith and Herrmann, 2013).

In this dissertation, we have used self-organisation paradigm as a way to capture coherent behaviour and combine it with goal-oriented development in autonomous robots. We believe that self-organisation, and especially embodiment in robotics, can be an important strategy towards autonomous robots delivering intelligent exploratory modes and dimensional reduction.

A new and improved learning rule for self-organisation of the behaviour can be found in (Der and Martius, 2015). The differential extrinsic plasticity rule (DEP) produces seemingly purposeful and adaptive rhythmic behaviour. Such behaviour is the result of the underlying mechanism of spontaneous symmetry breaking. Compared to homeokinesis, the DEP resulting behaviour has a more rhythmic component. On a hexapod robot, different locomotion gaits emerge, e.g., synchronous wave and trot, and tripod gate. These behaviours are more closely related to optimal locomotion. Thus, the DEP learning rule is a strong candidate to test the methods presented in Chapter 5.

In the next section we present a more detailed plan on utilising DEP with reinforcement learning.

Future Work

The set of internal models implemented in Chapter 4 is not exhaustive. The implemented models were selected as a way to have a representative list of linear and non-linear algorithms with various architectures and learning rules. We are expanding the list concerning supplementary NNs, e.g., feedforward or recurrent, with different learning techniques, e.g., classic backpropagation or backpropagation through time. We are preparing a journal publication with the models already presented in this document plus additional ones.

Self-organisation of the parameters of the internal models was performed directly over the policy and value function of an actor-critic reinforcement learning set-up. The central idea was to use intrinsically motivated behaviour before the goal-oriented learning happens. This idea can be implemented into model-based reinforcement learning (Kuvayev and Sutton, 1996). In this type of learning, RL also builds a forward model for planning. In the general case, the model acquire the environment dynamics based on greedily selected actions. We propose the self-organisation of acquisition of the dynamics prior to RL. One of the points to study in this set-up is the model capacity. In this technique, the forward model acts as a non-local predictor. Thus, the model requires a larger memory capacity than the ones studied in this dissertation.

Deep learning is one of the topics with larger advances in machine learning in the last years. Classification problems and feature extraction have been highly successful (LeCun et al., 2010; Bengio et al., 2012). Self-organised behaviour can provide the large amount of data required for deep learning in an inexpensive way. Extracting basic features from coherent behaviour can reduce the search space in optimal goal-oriented learning significantly. As future work, we want to extract features from homeokinetic behaviour using deep learning techniques and then exploit this reduced search space with deep reinforcement learning (Mnih et al., 2015) in complex high-dimensional robots.

As mentioned in the previous section, DEP learning rule (Der and Martius, 2015), has further developed the methods for self-organisation of robotic behaviour. This new approach has the potential to improve the methods presented in Chapter 5. First, it is interesting to assess what kind of interaction between the DoF emerge in complex

robots, e.g., the wentelteefje presented in Section 3.2.5. Second, combine DEP with the methods in Chapter 5. Since DEP learning rule favours rhythmic behaviour, the optimal solutions in reinforcement learning could be found in less time. We believe that reward temporal average (Section 5.4) is the most suited method to exploit the emergent gaits when robots are controlled by DEP.

Bibliography

Ashby, W. R. (1960). *Design for a Brain*. Springer Science & Business Media.

Åström, K. J. and Wittenmark, B. (2013). *Adaptive control*. Courier Corporation.

Ay, N., Bernigau, H., Der, R., and Prokopenko, M. (2012a). Information-driven self-organization: the dynamical system approach to autonomous robot behavior. *Theory in Biosciences*, 131(3):161–179.

Ay, N., Der, R., and Prokopenko, M. (2012b). Guided self-organization: perception–action loops of embodied systems. *Theory in Biosciences*, 131(3):125–127.

Bagnell, J. A. and Schneider, J. G. (2001). Autonomous helicopter control using reinforcement learning policy search methods. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1615–1620. IEEE.

Baird III, L. C. (1993). Advantage updating. Technical report, DTIC Document.

Barto, A. G. and Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(1-2):41–77.

Barto, A. G., Singh, S., and Chentanez, N. (2004). Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of the 3rd International Conference on Development and Learning (ICDL 2004)*, pages 112–19.

Barto, A. G., Sutton, R. S., and Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-13(5):834–846.

Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.

Bengio, Y., Courville, A. C., and Vincent, P. (2012). Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR, abs/1206.5538*, 1.

Beni, G. (2005). From swarm intelligence to swarm robotics. In *Swarm robotics*, pages 1–9. Springer.

Bentivegna, D. C. and Atkeson, C. G. (2001). Learning from observation using primitives. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1988–1993. IEEE.

Berger, T. W., Chauvet, G., and Sclabassi, R. J. (1994). A biologically based model of functional properties of the hippocampus. *Neural Networks*, 7(6):1031–1064.

Berlyne, D. E. (1960). *Conflict, arousal, and curiosity.* McGraw-Hill Book Company.

Bernard, C. (1974). *Lectures on the phenomena of life common to animals and plants*, volume 2. Charles C Thomas Pub Ltd.

Blakemore, S., Wolpert, D., and Frith, C. (1998). Central cancellation of self-produced tickle sensation. *Nature Neuroscience*, 1(7):635–40.

Bogg, J. and Geyer, R. (2007). *Complexity, science and society*. Radcliffe Publishing.

Bongard, J., Zykov, V., and Lipson, H. (2006). Resilient machines through continuous self-modeling. *Science*, 314(5802):1118–1121.

Bradtke, S. J. (1993). Reinforcement learning applied to linear quadratic regulation. *Advances in Neural Information Processing Systems*, pages 295–295.

Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41.

Brooks, R. (1991). Intelligence without representation. *Artificial Intelligence*, 47:139–159.

Brooks, R. et al. (1986). A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23.

Broomhead, D. S. and Lowe, D. (1988). Radial basis functions, multi-variable functional interpolation and adaptive networks. Technical report, DTIC Document.

Buckner, R. L. (2010). The role of the hippocampus in prediction and imagination. *Annual Review of Psychology*, 61:27–48.

Butz, M. V., Herbort, O., and Hoffmann, J. (2007). Exploiting redundancy for flexible behavior: unsupervised learning in a modular sensorimotor control architecture. *Psychological Review*, 114(4):1015.

Camazine, S. (2003). *Self-organization in biological systems*. Princeton University Press.

Cameron, E. Z., Linklater, W. L., Stafford, K. J., and Minot, E. O. (2008). Maternal investment results in better foal condition through increased play behaviour in horses. *Animal Behaviour*, 76(5):1511–1518.

Cannon, W. B. (1932). The wisdom of the body.

Chang, F., Chang, L.-C., Huang, H.-L., et al. (2002). Real-time recurrent learning neural network for stream-flow forecasting. *Hydrological Processes*, 16(13):2577–2588.

Chentanez, N., Barto, A. G., and Singh, S. P. (2004). Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 1281–1288.

Chiel, H. J. and Beer, R. D. (1997). The brain has a body: adaptive behavior emerges from interactions of nervous system, body and environment. *Trends in Neurosciences*, 20(12):553–557.

Clark, A. (1997). *Being There: Putting Brain Body and World Together Again*. MIT Press, Cambridge, MA.

Clark, A. and Grush, R. (1999). Towards a Cognitive Robotics. *Adaptive Behavior*, 7(1):5–16.

Claus, C. and Boutilier, C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. In *AAAI/IAAI*, pages 746–752.

Cleveland, W. S. and Devlin, S. J. (1988). Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610.

Conant, R. C. and Ashby, W. R. (1970). Every good regulator of a system must be a model of that system. *International Journal of Systems Science*, 1(2):89–97.

Craig, J. J. (2005). *Introduction to robotics: mechanics and control*, volume 3. Pearson Prentice Hall Upper Saddle River.

Csikszentmihalyi, M. and Csikzentmihaly, M. (1991). *Flow: The psychology of optimal experience*, volume 41. HarperPerennial New York.

Davies, V. A. and Kemble, E. D. (1983). Social play behaviours and insect predation in northern grasshopper mice (Onychomys leucogaster). *Behavioural Processes*, 8(2):197–204.

Dayan, P. and Sejnowski, T. J. (1996). Exploration bonuses and dual control. *Machine Learning*, 25(1):5–22.

Deci, E. L. and Ryan, R. M. (1985). *Intrinsic motivation and self-determination in human behavior*. Springer Science & Business Media.

Der, R. (2001). Self-organized acquisition of situated behaviors. *Theory in Biosciences*, 120(3-4):179–187.

Der, R. (2003). Homeokinesis and the moderation of complexity in neural systems.

Der, R. and Liebscher, R. (2002). True autonomy from self-organized adaptivity. In *Proc. Workshop Biologically Inspired Robotics. The Legacy of Grey Walter*, pages 14–16.

Der, R. and Martius, G. (2012). *The playful machine*. Springer.

Der, R. and Martius, G. (2015). A novel plasticity rule can explain the development of sensorimotor intelligence. *CoRR*, abs/1505.00835.

Der, R., Steinmetz, U., Pasemann, F., et al. (1999). Homeokinesis - A new principle to back up evolution with learning. *Computational Intelligence for Modelling, Control, and Automation, Concurrent Systems Engineering Series*, 55:43–47.

Dickinson, M. H. (2015). Motor control: How dragonflies catch their prey. *Current Biology*, 25(6):R232–R234.

Dorigo, M., Trianni, V., Şahin, E., Groß, R., Labella, T. H., Baldassarre, G., Nolfi, S., Deneubourg, J.-L., Mondada, F., Floreano, D., et al. (2004). Evolving self-organizing behaviors for a swarm-bot. *Autonomous Robots*, 17(2-3):223–245.

Doya, K. (1992). Bifurcations in the learning of recurrent neural networks. *IEEE International Symposium on Circuits and Systems*, 6:2777–2780.

Doya, K. (1995). Supervised learning in recurrent networks. *Handbook of Brain Theory and Neural Networks. MIT Press, Cambridge*.

Doya, K. (1996). Temporal difference learning in continuous time and space. *Advances in Neural Information Processing Systems*, pages 1073–1079.

Doya, K. (1999). What are the computations of the cerebellum, the basal ganglia and the cerebral cortex? *Neural Networks*, 12(7):961–974.

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Computation*, 12:219–245.

Doya, K., Samejima, K., Katagiri, K.-i., and Kawato, M. (2002). Multiple model-based reinforcement learning. *Neural Computation*, 14(6):1347–1369.

Dréo, J. (2014). Ameisenalgorithmus — Wikipedia, die freie enzyklopdie. [Online; Stand 26. Januar 2015].

D'Souza, A., Vijayakumar, S., and Schaal, S. (2001). Learning inverse kinematics. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 1, pages 298–303. IEEE.

Duff, S. J. and Bradtke Michael, O. (1995). Reinforcement learning methods for continuous-time markov decision problems. *Advances in Neural Information Processing Systems 7*, 7:393.

Fagen, R. (1981). *Animal play behavior*. Oxford University Press.

Feldbaum, A. (1961). Dual control theory. *Automn. Remote Control*, 22:1–12.

Friston, K. (2010). The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138.

Frith, C. D., Wolpert, D. M., et al. (2000). Abnormalities in the awareness and control of action. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 355(1404):1771–1788.

Gershenson, C. (2007). *Design and control of self-organizing systems*. CopIt ArXives.

Gierer, A. and Meinhardt, H. (1972). A theory of biological pattern formation. *Kybernetik*, 12(1):30–39.

Gigliotta, O., Pezzulo, G., and Nolfi, S. (2010). Emergence of an internal model in evolving robots subjected to sensory deprivation. In *Proceedings of the 11th international conference on Simulation of adaptive behavior: from animals to animats*, SAB'10, pages 575–586, Berlin, Heidelberg. Springer-Verlag.

Goschin, S., Franti, E., Dascalu, M., and Osiceanu, S. (2007). Combine and compare evolutionary robotics and reinforcement learning as methods of designing autonomous robots. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 1511–1516. IEEE.

Goss, S., Aron, S., Deneubourg, J.-L., and Pasteels, J. M. (1989). Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, 76(12):579–581.

Gottlieb, G. (2007). Probabilistic epigenesis. *Developmental Science*, 10(1):1–11.

Grush, R. (2004). The emulation theory of representation: motor control, imagery, and perception. *Behavioral and Brain Sciences*, 27(03):377–396.

Gullapalli, V. (1990). A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural Networks*, 3:671–692.

Hager, W. W. (1989). Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239.

Haken, H. (1983). Synergetics. an introduction. nonequilibrium phase transitions and self-organisation in physics. *Chemistry, and Biology*, 3.

Haken, H. (2006). *Information and self-organization: A macroscopic approach to complex systems*. Springer Science & Business Media.

Haken, H. (2008). Self-organization. *Scholarpedia*, 3(8):1401. Revision 137295.

Hamed, N. (2007). *Self-Referential Dynamical Systems and Developmental Robotics*. PhD thesis, University of Leipzig.

Haruno, M., Wolpert, D., and Kawato, M. (2001). Mosaic model for sensorimotor learning and control. *Neural Computation*, 13(10):2201–2220.

Haykin, S. and Network, N. (2004). A comprehensive foundation. *Neural Networks*, 2(2004).

Herrmann, J. M., Pawelzik, K., and Geisel, T. (2000). Learning predictive representations. *Neurocomputing*, 32:785–791.

Heylighen, F. (2001). The science of self-organization and adaptivity. *The Encyclopedia of Life Support Systems*, 5(3):253–280.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* University of Michigan Press.

Huston, S. J. and Jayaraman, V. (2011). Studying sensorimotor integration in insects. *Current Opinion in Neurobiology*, 21(4):527–534.

Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2002). Movement imitation with nonlinear dynamical systems in humanoid robots. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1398–1403. IEEE.

Imamizu, H., Miyauchi, S., Tamada, T., Sasaki, Y., Takino, R., Pütz, B., Yoshioka, T., and Kawato, M. (2000). Human cerebellar activity reflecting an acquired internal model of a new tool. *Nature*, 403(6766):192–195.

Ito, M. (1970). Neurophysiological aspects of the cerebellar motor control system. *International Journal of Neurology*, 7(2):162–76.

Jaeger, H. (2001). The echo state approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148.

Jaeger, H. (2007). Echo state network. *Scholarpedia*, 2(9):2330. Revision 143667.

Jen, E. (2005). *Robust design: Repertoire of biological, ecological, and engineering case studies*. Oxford University Press New York.

Jong, N. K., Hester, T., and Stone, P. (2008). The utility of temporal abstraction in reinforcement learning. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 299–306. International Foundation for Autonomous Agents and Multiagent Systems.

Jordan, M. I. and Rumelhart, D. E. (1992). Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16(3):307–354.

Kawano, K., Takemura, A., Inoue, Y., Kitama, T., Kobayashi, Y., and Mustari, M. (1996). Visual inputs to cerebellar ventral paraflocculus during ocular following responses. *Progress in Brain Research*, 112:415–422.

Kawato, M. (1990). Feedback-error-learning neural network for supervised motor learning. *Advanced Neural Computers*, 6(3):365–372.

Kawato, M. (1999). Internal models for motor control and trajectory planning. *Current Opinion in Neurobiology*, 9(6):718 – 727.

Kawato, M., Furukawa, K., and Suzuki, R. (1987). A hierarchical neural-network model for control and learning of voluntary movement. *Biological Cybernetics*, 57(3):169–185.

Kemp, C. C., Edsinger, A., and Torres-Jara, E. (2007). Challenges for robot manipulation in human environments. *IEEE Robotics and Automation Magazine*, 14(1):20.

Key, S. V. V. and Baker, T. (1982). Trail pheromone-conditioned anemotaxis by the Argentine ant, Iridomyrmex Humilis. *Entomologia Experimentalis et Applicata*, 32(3):232–237.

Khansari-Zadeh, S. M. and Billard, A. (2012). A dynamical system approach to real-time obstacle avoidance. *Autonomous Robots*, 32(4):433–454.

Kitazawa, S., Kimura, T., and Yin, P.-B. (1998). Cerebellar complex spikes encode both destinations and errors in arm movements. *Nature*, 392(6675):494–497.

Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2005). All else being equal be empowered. In *Advances in Artificial Life*, pages 744–753. Springer.

Kobayashi, Y., Kawano, K., Takemura, A., Inoue, Y., Kitama, T., Gomi, H., and Kawato, M. (1998). Temporal firing patterns of Purkinje cells in the cerebellar ventral paraflocculus during ocular following responses in monkeys II. Complex spikes. *Journal of Neurophysiology*, 80(2):832–848.

Kohl, N. and Stone, P. (2004). Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 3, pages 2619–2624. IEEE.

Kohonen, T. (1998). The self-organizing map. *Neurocomputing*, 21(1):1–6.

Kondo, S. and Asai, R. (1995). A reaction-diffusion wave on the skin of the marine angelfish pomacanthus. *Nature*, 376(6543):765–768.

Konidaris, G. (2008). Value function approximation in reinforcement learning using the fourier basis. Technical report, University of Massachusetts Amhers.

Konidaris, G. and Barreto, A. S. (2009). Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems*, pages 1015–1023.

Kuvayev, L. and Sutton, R. S. (1996). Model-based reinforcement learning with an approximate, learned model. In *in Proceedings of the Ninth Yale Workshop on Adaptive and Learning Systems*.

Laurens, J., Meng, H., and Angelaki, D. E. (2013). Computation of linear acceleration through an internal model in the macaque cerebellum. *Nature Neuroscience*, 16(11):1701–1708.

Layne, J. R. and Passino, K. M. (1996). Fuzzy model reference learning control. *Journal of Intelligent and Fuzzy Systems*, 4(1):33–47.

LeCun, Y., Kavukcuoglu, K., and Farabet, C. (2010). Convolutional networks and applications in vision. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 253–256. IEEE.

Ljung, L. (1987). System identification: theory for the user. *PTR Prentice Hall Information and System Sciences Series*, 198.

Lopes, M. and Damas, B. (2007). A learning framework for generic sensory-motor maps. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1533–1538. IEEE.

Lungarella, M. (2007). Developmental robotics. *Scholarpedia*, 2(8):3104. Revision 91197.

Lungarella, M., Metta, G., Pfeifer, R., and Sandini, G. (2003). Developmental robotics: a survey. *Connection Science*, 15(4):151–190.

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560.

Mahadevan, S. and Connell, J. (1992). Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55(2):311–365.

Marshall, J., Blank, D., and Meeden, L. (2004). An emergent framework for self-motivation in developmental robotics. In *Proceedings of the 3rd international conference on development and learning (ICDL 2004), Salk Institute, San Diego*, volume 10.

Martius, G. (2010). *Goal-Oriented Control of Self-Organizing Behavior in Autonomous Robots*. PhD thesis, Göttingen University.

Martius, G. (2012). Lpzrobots simulator. http://robot.informatik.uni-leipzig.de/software.

Martius, G., Der, R., and Ay, N. (2013). Information driven self-organization of complex robotic behaviors. *PloS One*, 8(5):e63400.

Martius, G., Fiedler, K., and Herrmann, J. M. (2008). Structure from behavior in autonomous agents. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 858–862. IEEE.

Martius, G. and Herrmann, J. M. (2011). Tipping the scales: Guidance and intrinsically motivated behavior. In *Proceedings of European Conference on Artificial Life*, pages 766–775.

Martius, G. and Herrmann, J. M. (2012). Variants of guided self-organization for robot control. *Theory in Biosciences*, 131(3):129–137.

Martius, G., Herrmann, J. M., and Der, R. (2007). Guided self-organisation for autonomous robot development. In *Advances in Artificial Life 9th European Conference, ECAL 2007, Lisbon, Portugal*, volume 4648 of *Lecture Notes in Computer Science*, pages 766–775. Springer.

McCulloch, W. S. (1965). *Embodiments of mind.* The MIT Press.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133.

McFarland, D. and Bösser, T. (1993). *Intelligent behavior in animals and robots*. Mit Press.

Merfeld, D. M., Zupan, L., and Peterka, R. J. (1999). Humans use internal models to estimate gravity and linear acceleration. *Nature*, 398(6728):615–618.

Miller, W. T., Glanz, F. H., and Kraft, L. G. (1987). Application of a general learning algorithm to the control of robotic manipulators. *The International Journal of Robotics Research*, 6(2):84–98.

Mitchell, M., Hraber, P., and Crutchfield, J. P. (1993). Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems*, 7:89–130.

Miyamoto, H., Kawato, M., Setoyama, T., and Suzuki, R. (1988). Feedback-error-learning neural network for trajectory control of a robotic manipulator. *Neural Networks*, 1(3):251–265.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Moore, A. W. and Atkeson, C. G. (1993). Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13(1):103–130.

Nakanishi, J., Cory, R., Mistry, M., Peters, J., and Schaal, S. (2008). Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757.

Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E. (2006). Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pages 363–372. Springer.

Nguyen-Tuong, D. and Peters, J. (2010). Using model knowledge for learning inverse dynamics. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2677–2682. IEEE.

Nguyen-Tuong, D. and Peters, J. (2011). Model learning for robot control: a survey. *Cognitive Processing*, 12(4):319–340.

Nguyen-Tuong, D., Peters, J., Seeger, M., and Schölkopf, B. (2008). Learning inverse dynamics: a comparison. In *European Symposium on Artificial Neural Networks*, number EPFL-CONF-175477.

Nguyen-Tuong, D., Seeger, M., and Peters, J. (2009). Model learning with local gaussian process regression. *Advanced Robotics*, 23(15):2015–2034.

Nicolis, G. and Prigogine, I. (1977). *Self-organization in nonequilibrium systems*, volume 191977. Wiley, New York.

Nilsson, N. J. (2007). *The physical symbol system hypothesis: status and prospects*. Springer.

Nolfi, S. and Floreano, D. (2000). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. MIT press.

Norman, K. A. (2010). How hippocampus and cortex contribute to recognition memory: revisiting the complementary learning systems model. *Hippocampus*, 20(11):1217–1227.

Oudeyer, P.-Y. and Kaplan, F. (2007). What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics*, 1.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. (2007). Intrinsic motivation systems for autonomous mental development. *Evolutionary Computation, IEEE Transactions on*, 11(2):265–286.

Oudeyer, P.-Y., Kaplan, F., Hafner, V. V., and Whyte, A. (2005). The playground experiment: Task-independent development of a curious robot. In *Proceedings of the AAAI Spring Symposium on Developmental Robotics*, pages 42–47. Stanford, California.

Pavlov, I. P. (1927). *Conditional reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex*. Oxford University Press.

Peters, J. and Schaal, S. (2008). Learning to control in operational space. *The International Journal of Robotics Research*, 27(2):197–212.

Peters, J., Vijayakumar, S., and Schaal, S. (2003). Reinforcement learning for humanoid robotics. In *Proceedings of the Third IEEE-RAS International Conference on Humanoid Robots*, pages 1–20.

Pfeifer, R., Bongard, J., and Grand, S. (2007). *How the body shapes the way we think: a new view of intelligence*. MIT press.

Pfeifer, R., Scheier, C., and Illustrator-Follath, I. (2001). *Understanding intelligence*. MIT press.

Prokopenko, M. (2013). *Guided self-organization: Inception*, volume 9. Springer Science & Business Media.

Prokopenko, M. (2014). Grand challenges for computational intelligence. *Frontiers in Robotics and AI*, 1:2.

Rasmussen, C. E. (2006). *Gaussian processes for machine learning*. MIT Press.

Rauschecker, J. P. (2015). Auditory and visual cortex of primates: a comparison of two sensory systems. *European Journal of Neuroscience*, 41(5):579–585.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. In *ACM Siggraph Computer Graphics*, volume 21, pages 25–34. ACM.

Rivera, D., Morari, M., and Skogestad, S. (1986). Internal model control: PID controller design. *Industrial & engineering chemistry process design and development*, 25(1):252–265.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386.

Ryan, R. M. and Deci, E. L. (2000). Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary Educational Psychology*, 25(1):54–67.

Salge, C., Glackin, C., and Polani, D. (2013). Approximation of empowerment in the continuous domain. *Advances in Complex Systems*, 16(02n03).

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229.

Sato, M.-A. and Ishii, S. (2000). On-line em algorithm for the normalized gaussian network. *Neural Computation*, 12(2):407–432.

Schiller, U. D. and Steil, J. J. (2005). Analyzing the weight dynamics of recurrent learning algorithms. *Neurocomputing*, 63:5–23.

Schmidhuber, J. (1991a). Adaptive confidence and adaptive curiosity. In *Institut für Informatik, Technische Universität München, Arcisstr. 21, 800 München 2*. Citeseer.

Schmidhuber, J. (1991b). Curious model-building control systems. In *Neural Networks, 1991. 1991 IEEE International Joint Conference on*, pages 1458–1463. IEEE.

Schmidhuber, J. (2006). Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, 18(2):173–187.

Schmidhuber, J. (2010). Formal theory of creativity, fun, and instrinsic motivation. *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247.

Schölkopf, B. and Smola, A. J. (2002). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press.

Sharpe, L. L. (2005). Frequency of social play does not affect dispersal partnerships in wild meerkats. *Animal behaviour*, 70(3):559–569.

Shibata, T. and Schaal, S. (2001). Biomimetic gaze stabilization based on feedback-error-learning with nonparametric regression networks. *Neural Networks*, 14(2):201–216.

Sian, N., Sakaguchi, T., Yokoi, K., Kawai, Y., and Maruyama, K. (2006). Operating humanoid robots in human environments. In *Proceedings of the Robotics, Science & Systems Workshop on Manipulation for Human Environments, Philedelphia, Pennsylvania*.

Sirigu, A., Duhamel, J.-R., Cohen, L., Pillon, B., Dubois, B., and Agid, Y. (1996). The mental representation of hand movements after parietal cortex damage. *Science*, 273(5281):1564–1568.

Smith, A. W. and Zipser, D. (1989). Learning sequential structure with the real-time recurrent learning algorithm. *International Journal of Neural Systems*, 1(02):125–131.

Smith, O. J. (1958). *Feedback control systems*. McGraw-Hill.

Smith, S. C. and Herrmann, J. M. (2012). Homeokinetic reinforcement learning. In *Partially Supervised Learning*, pages 82–91. Springer.

Smith, S. C. and Herrmann, J. M. (2013). Self-organisation of generic policies in reinforcement learning. In *Advances in Artificial Life, ECAL*, volume 12, pages 641–648.

Snyder, L. (1999). This way up: illusions and internal models in the vestibular system. *Nature Neuroscience*, 2:396–398.

Sontag, E. D. (2003). Adaptation and regulation with signal detection implies internal model. *Systems & Control Letters*, 50(2):119–126.

Strogatz, S. H. (2001). *Nonlinear dynamics and chaos: with applications to physics, biology and chemistry*. Perseus Publishing.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44.

Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the seventh international conference on machine learning*, pages 216–224.

Sutton, R. S. (1991). Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163.

Sutton, R. S. and Barto, A. G. (1998). *Introduction to reinforcement learning*. MIT Press.

Szita, I., Gyenes, V., and Lőrincz, A. (2006). Reinforcement learning with echo state networks. In *Artificial Neural Networks–ICANN 2006*, pages 830–839. Springer.

Tani, J., Nishimoto, R., and Paine, R. W. (2008). Achieving "organic compositionality" through self-organization: Reviews on brain-inspired robotics experiments. *Neural Networks*, 21(4):584 – 603. Robotics and Neuroscience.

Tesauro, G. (1994). Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219.

Thach, W. (1998). A role for the cerebellum in learning movement coordination. *Neurobiology of Learning and Memory*, 70(1):177–188.

Thrun, S. B. (1992). Efficient exploration in reinforcement learning. Technical report, Pittsburgh, PA, USA.

Thrun, S. B. and Möller, K. (1992). Active exploration in dynamic environments. *Advances in Neural Inforamtion Preocessin Systems 4*.

Ting, J.-A., Kalakrishnan, M., Vijayakumar, S., and Schaal, S. (2009). Bayesian kernel shaping for learning control. In *Advances in neural information processing systems*, pages 1673–1680.

Tsui, K. M. and Yanco, H. A. (2006). Human-in-the-loop control of an assistive robot arm. In *Proc. RSS Workshop: Manipulation for Human Environments*. Citeseer.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind, 59*, pages 433–460.

Turing, A. M. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 237(641):37–72.

Varela, F. J., Thompson, E., and Rosch, E. (1992). *The embodied mind*. CogNet, MIT Press.

Vempaty, P. K., Cheok, K. C., and Loh, R. (2009). Model reference adaptive control for actuators of a biped robot locomotion. In *Proceedings of the world congress on engineering and computer science*, volume 2, pages 20–22.

Vijayakumar, S. and Schaal, S. (2000). Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space. In *International conference on machine learning, proceedings of the sixteenth conference*.

Volkinshtein, D. and Meir, R. (2009). Delayed feedback control requires an internal forward model. *Biological Cybernetics*, 105(1):41–53.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Westheimer, G. (2008). Was Helmholtz a Bayesian? a review. *Perception*, 37:642–650.

White, R. W. (1959). Motivation reconsidered: the concept of competence. *Psychological Review*, 66(5):297.

Whitehead, S. D. and Ballard, D. H. (1991). Learning to perceive and act by trial and error. *Machine Learning*, 7(1):45–83.

Wiener, N. (1948). *Cybernetics or Control and Communication in the Animal and the Machine*. Hermann Editions, Paris.

Williams, R. J. and Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

Wilson, R. A. and Foglia, L. (2011). Embodied cognition. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Fall 2011 edition.

Wise, R. A. (1980). The dopamine synapse and the notion of pleasure centers in the brain. *Trends in Neurosciences*, 3(4):91–95.

Wolpert, D. and Flanagan, J. (2001). Motor prediction. *Current Biology*, 11(18):R729–732.

Wolpert, D. M. and Ghahrami, Z. (2000). Computational principles of movement neuroscience. *Nature Neurosci*, 3.

Wörgötter, F. and Porr, B. (2008). Reinforcement learning. *Scholarpedia*, 3(3):1448. Revision 91704.

Xu, X., He, H.-g., and Hu, D. (2002). Efficient reinforcement learning using recursive least-squares methods. *Journal of Artificial Intelligence Research*, 16(1):259–292.

Xu, X., Hu, D., and Lu, X. (2007). Kernel-based least squares policy iteration for reinforcement learning. *Neural Networks, IEEE Transactions on*, 18(4):973–992.

Appendix A

Mathematical Derivations

A.1 Canonical Homeokinetic Learning Rule

To find the derivative of L^{-1} consider $LL^{-1} = I$, then

$$0 = \frac{\partial}{\partial p}\left(LL^{-1}\right) = \left(\frac{\partial L}{\partial p}\right)L^{-1} + L\frac{\partial}{\partial p}L^{-1}$$

thus

$$\frac{\partial}{\partial p}L^{-1} = -L^{-1}\left(\frac{\partial L}{\partial p}\right)L^{-1}. \tag{A.1}$$

Now consider two vectors a and b

$$a^\top\left(\frac{\partial}{\partial p}\left(LL^\top\right)^{-1}\right)b = 2a^\top L^{\top -1}\frac{\partial L^{-1}}{\partial p}b = -2a^\top\left(LL^\top\right)^{-1}\frac{\partial L}{\partial p}L^{-1}b$$

the basic relation $(AB)^{-1} = B^{-1}A^{-1}$, the definition in Eq. A.1 and $a^\top Q^\top b = b^\top Qa$ have been used in order to deal with $L^{\top -1}$. Following the gradient descent in Eq. 3.14 and absorbing the factor 2 into the learning rate yields the canonical learning rule (Eq. 3.17 in Section 3.2.2).

A.2 Response Model for RTRL

For a RTRL network, used as forward model for a homoekinetic controller, the response L has to be derived. The response is defined as:

$$L = \frac{\partial M\left(\mathbf{x}_t, K\left(x_t\right)\right)}{\partial x}. \tag{A.2}$$

For a RTRL network with activation function f and feedback weights W^{os}, the output is calculated as:

$$\hat{\mathbf{x}}_{t+1} = f\left(W^{yo}\mathbf{y}_t + W^{so}\mathbf{s}_t + W^{oo}\hat{\mathbf{x}}_t\right), \tag{A.3}$$

and the internal state update is:

$$\mathbf{s}_{t+1} = d\left(W^{ys}\mathbf{y}_{t+1} + W^{ss}\mathbf{s}_t + W^{os}\hat{\mathbf{x}}_t\right). \tag{A.4}$$

The derivation of L is as follows:

$$\frac{\partial M}{\partial \mathbf{x}} = F' \left(W^{yo} G' C + W^{so} \frac{\partial \mathbf{s}_t}{\partial \mathbf{x}} \right), \tag{A.5}$$

where G' as Eq. 3.21 and F' the diagonal matrix:

$$F'_{ij} = \delta_{ij} f' (q_i) = \delta_{ij} f'_i (\mathbf{q}), \tag{A.6}$$

$$\mathbf{q} = W^{yo} \mathbf{y}_t + W^{so} \mathbf{s}_t + W^{oo} \hat{\mathbf{x}}_t. \tag{A.7}$$

The derivative of the internal state $\mathbf{s}$ with respect to the input is:

$$\frac{\partial \mathbf{s}_t}{\partial \mathbf{x}} = D' \left(W^{ys} G' C \right), \tag{A.8}$$

where D', analogously to G' and F' is the diagonal matrix defined as:

$$D' = \delta_{ij} d' (k_i) = \delta_{ij} d' (\mathbf{k}), \tag{A.9}$$

$$\mathbf{k} = W^{ys} \mathbf{y}_{t+1} + W^{ss} \mathbf{s}_t + W^{os} \hat{\mathbf{x}}_t. \tag{A.10}$$

Assuming $\frac{\partial \hat{\mathbf{x}}_{t-1}}{\partial \mathbf{x}_t} = 0$ and $\frac{\partial \mathbf{s}_{t-1}}{\partial \mathbf{x}_t} = 0$, the model response finally is:

$$L = F' \left(W^{yo} G' C + W^{so} \left(D' \left(W^{ys} G' C \right) \right) \right). \tag{A.11}$$