Tom Ziemke
Christian Balkenius
John Hallam (Eds.)

LNAI 7426

From Animals to Animats 12

12th International Conference
on Simulation of Adaptive Behavior, SAB 2012
Odense, Denmark, August 2012, Proceedings

Springer

Lecture Notes in Artificial Intelligence 7426

Subseries of Lecture Notes in Computer Science

Tom Ziemke Christian Balkenius
John Hallam (Eds.)

From Animals to Animats 12

12th International Conference
on Simulation of Adaptive Behavior, SAB 2012
Odense, Denmark, August 27-30, 2012
Proceedings

Springer

Volume Editors

Tom Ziemke
University of Skövde
Informatics Research Centre
Kanikegränd 3A, 54134 Skövde, Sweden
E-mail: tom.ziemke@his.se

Christian Balkenius
Lund University
Cognitive Science
Lundagård, Kungshuset, 22222 Lund, Sweden
E-mail: christian.balkenius@lucs.lu.se

John Hallam
University of Southern Denmark
Mærsk McKinney Møller Institute
Campusvej 55, 5230 Odense, Denmark
E-mail: john@mmmi.sdu.dk

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-33092-6 e-ISBN 978-3-642-33093-3
DOI 10.1007/978-3-642-33093-3
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012945324

CR Subject Classification (1998): I.2, F.1, I.4, I.5, J.3, H.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This book contains the articles presented at the 12th International Conference on the Simulation of Adaptive Behavior (SAB 2012), held in Odense at the University of Southern Denmark in August 2012.

The objective of the biennial SAB conference is to bring together researchers in computer science, artificial intelligence, artificial life, complex systems, robotics, neurosciences, ethology, evolutionary biology, and related fields so as to further our understanding of the behaviors and underlying mechanisms that allow natural and artificial animals to adapt and survive in uncertain environments.

Adaptive behavior research is distinguished by its focus on the modeling and creation of complete animal-like systems, which – however simple at the moment – may be one of the best routes to understanding intelligence in natural and artificial systems. The conference is part of a long series that started with the first SAB conference held in Paris in September 1990, which was followed by conferences in Honolulu 1992, Brighton 1994, Cape Cod 1996, Zürich 1998, Paris 2000, Edinburgh 2002, Los Angeles 2004, Rome 2006, Osaka 2008, and Paris 2010, where the 20th anniversary of the conference was celebrated. In 1992, MIT Press introduced the quarterly journal Adaptive Behavior, now published by SAGE Publications. The establishment of the International Society for Adaptive Behavior (ISAB) in 1995 further underlined the emergence of adaptive behavior as a fully-fledged scientific discipline. The present proceedings provide a comprehensive and up-to-date resource for the future development of this exciting field.

The articles cover the main areas in animat research, including the animat approach and methodology, perception and motor control, evolution, learning and adaptation, and collective and social behavior. The authors focus on well-defined models, computer simulations or robotic models, that help to characterize and compare various organizational principles, architectures, and adaptation processes capable of inducing adaptive behavior in real animals or synthetic agents, the animats.

This conference and its proceedings would not exist without the substantial help of many people. We would like to thank the members of the Program Committee, who critically reviewed 66 submissions and provided detailed suggestions on how to improve the 44 articles accepted for inclusion in these proceedings and presentation at the conference (22 talks, 22 posters). We also thank, once again, Jean Solé for the artistic conception of the SAB 2012 poster and the proceedings cover.

August 2012

Tom Ziemke
Christian Balkenius
John Hallam

Organization

From Animals to Animats 12, the 12th International Conference on the Simulation of Adaptive Behavior (SAB 2012) was organized by the Mærsk Mc-Kinney Møller Institute of the University of Southern Denmark and the International Society for Adaptive Behavior (ISAB).

Conference Chairs

John Hallam	University of Southern Denmark, Odense, Denmark
Christian Balkenius	Lund University, Sweden
Tom Ziemke	University of Skövde, Sweden

Local Organization

John Hallam and Vibeke Nielsen, University of Southern Denmark

Program Committee

Frédéric Alexandre
Ronald Arkin
Angelo Arleo
Minoru Asada
Christian Balkenius
Ryad Benosman
Luc Berthouze
Josh Bongard
Nicolas Bredeche
Lola Canamero
Angelo Cangelosi
Ricardo Chavarriaga
Luis Correia
Nikolaus Correll
Kerstin Dautenhahn
Julien Diard
Stéphane Doncieux
Marco Dorigo
Boris Duran
Richard Duro
Michael Dyer
Luca Gambardella
Philippe Gaussier
Benoît Girard
John Hallam
Inman Harvey
Mark Humphries
Phil Husbands
Fumiya Iida
Hiroyuki Iizuka
Takashi Ikegami
Naoto Iwahashi
Frederic Kaplan
Toshiyuki Kondo
Robert Lowe
Davide Marocco
Jean-Baptiste Mouret
Ryohei Nakano
Stefano Nolfi
Pierre-Yves Oudeyer
Pietro Pantano
Rolf Pfeifer
Tony Prescott
Mikhail Prokopenko
Inaki Rano
Marc Schoenauer
Denis Sheynikhovich
Olivier Sigaud
Charles Taylor
Tim Taylor
Serge Thill
Peter Todd
Elio Tuci
Eiji Uchibe
Hiroaki Wagatsuma
Myra Wilson
Rachel Wood
Tom Ziemke

Sponsoring Institutions

Cyberbotics Ltd.
EUCog
ISAB
Lund University
University of Skövde
University of Southern Denmark

Table of Contents

Animat Approach and Methodology

Perception and Motor Control

Evolution

Learning and Adaptation

Collective and Social Behavior

Natural and Artificial Systems: Compare, Model or Engineer?

Kelly Vassie[1] and Giuseppe Morlino[2, 3]

[1] Unaffiliated, London, UK
kjvassie@ed-alumni.net
[2] Computer Science Department, Sapienza University, Rome, Italy
[3] Institute of Cognitive Sciences and Technologies, CNR, Rome, Italy
giuseppe.morlino@istc.cnr.it

Abstract. Some areas of biological research use artificial means to explore the natural world. But how the natural and artificial are related across wide-ranging research areas is not always clear. Relations differ further for bioengineering fields. We propose a taxonomy which would serve to elucidate distinct relations; there are three ways in which the natural is linked to the artificial, corresponding with distinct methods of investigation: i) a comparative approach (natural *vs* artificial) in which artificial systems are treated in the same way as natural systems, ii) a modeling approach (natural *via* artificial) in which we use artificial systems to learn about features of natural ones, and iii) an engineering approach (natural *pro* artificial) in which natural systems are used to draw inspiration for artefacts. Ambiguities about and between these approaches limit the development of fields and impact negatively on interdisciplinary communication.

Keywords: Artificial Life, Extended Mind, Thought Experiments, Modeling, Bioengineering.

1 Introduction

Distinction between two kinds of synthetic approaches to biology – i) comparative, such as ALife or Evolutionary Robotics and ii) the more widely known (and understood) modeling practice – are not entirely new. These approaches have previously been separated on the basis of: clarity or complexity [1]; methodology (Miller 1995 cited in [2], [3-5]; abstractness [6]; and as different levels of enquiry [7-8]. There are implicit arguments about the relationship between natural and artificial underlying each of these distinctions but these considerations are not seen as important. For example, in [5] Harvey et al. argue that Evolutionary Robotics (ER) 'is a new scientific tool', insofar as the methodological emphases (minimal cognition, existence proofs and reduction of bias) are very different from modeling. They claim that ER 'systems will not tell us how real cognitive systems work' whereas, for example, neuroscientific models might [5]. It is clear that the artificial system in modeling stands in for the natural system – because results about the model tell us

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 1–11, 2012.

how natural systems work. Yet it is not always clear how the ER system relates to a natural system. This is evidenced for example in the discussion of an ER simulation study into the origins of learning (Tuci et al. [9] cited in [5]). In this experiment some mechanisms for learning emerged, although no hypothesis could be made for these kinds of processes or the architecture that would support them – otherwise it could not be a study into the origins of learning itself. The learning mechanisms cannot be evidence for a natural system because the methodological processes have not specified a target system for this purpose, as modeling a learning mechanism would do [5]. What this ER experiment did show is that while an organism can evolve processes which enable it to learn, the actual mechanisms that emerged can only be used to help build concepts about learning. The relationship between natural and artificial in such work is not explicit; this has resulted in a negative view of simulation approaches. For example, Webb argues in [7] that because theoretical proofs eventually require comparison to the natural world they are basically a class of model, and if they don't represent anything "real" in the natural world they are (or should be) irrelevant to biological investigation. The issue for simulation work, if Webb's argument is accepted, is that it would be evaluated against the same requirements as modeling – justifying work on the basis of a concrete fit to empirical data [1], [7]. As we have just shown it is not empirical data that is generated but ideas about what mechanisms might be, and proof that learning can evolve from simple mechanical components. The outcome forced by Webb would not enable ALife researchers to develop scientific practices or revise relevant biological concepts (see [4] for an example), both of which are important for the advancement of this newer field. Furthermore, given the possibility that life *might* be artificially created, we would need a structure for the analysis of this artificial system because the artificial would have the same characteristic as a natural system, making it distinct from the representative characteristic of a model.

These two distinct relationships between the natural and artificial each give rise to their own epistemological concerns and considerations. One important concern is that the processes of simulation work in ways that go beyond (or abstract away from) our cognitive abilities. The argument that the non-anthropocentric process of simulation requires different epistemological considerations follows Humphreys (see [3]). Humphreys has different arguments from the one we present here, but we do have similar conclusions – that a "new-analysis" of the relation between artificial and natural is necessary, and that this includes making their epistemological concerns distinct. Our paper provides a structure for this analysis to take place.

As well as aiding the development of newer fields we see our work as providing an important framework for interdisciplinary communication. In light of the ever-greater specialisation within science – and, conversely, the rise of collaboration – our taxonomy offers a new tool for assisting professional dialogue and public science engagement. It is in this spirit that we include clarification of how the developing fields of bioengineering relate to the epistemological approaches of comparison and modeling. Finally, following Cordeschi [10] our intention here is not to carve unnecessary boundaries between approaches and opposing paradigms (i.e. we think that both comparative and modeling work is important, and within them, work from different kinds of theoretical positions).

2 Comparative, Modeling and Engineering Approaches

In the following sections we outline the three approaches – natural *vs* artificial, natural *via* artificial and natural *pro* artificial – discussing the role of artificial systems in each. Figure 1 provides a context for this taxonomy.

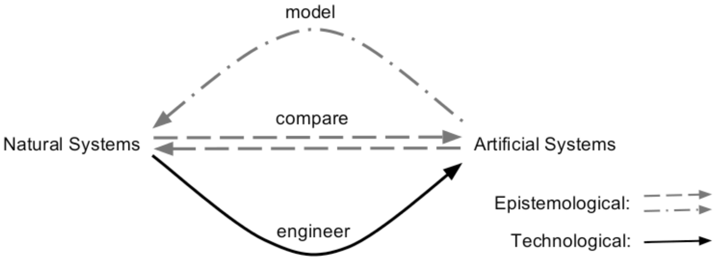


Fig. 1. Diagram showing the use of natural and artificial systems (arrows depict direction of system use)

We will show that the difference between comparison and modeling also concerns different levels of explanation and experimentation. We then argue that as the relation between the natural world and the artificial system is different, the epistemological issues must also be different. Further distinction will then be made between the epistemological (scientific) approaches of *vs* and *via*, and the engineering (technological) one of *pro*. We explain how the relationship between natural and artificial is reversed in these engineering approaches; even though models can be physically built as artefacts (albeit using engineering techniques), the aim of an actualised model is still the *explanation* of a natural phenomenon. However, before we outline these approaches we need to disambiguate the use of “simulation” to avoid confusion between simulations as used in the comparative approach and the simulation techniques used in modeling and engineering: in the comparative approach (*vs*), simulations are artificial systems used to build theories, question assumptions and explore mechanisms (as in ALife); in the modeling approach (*via*), simulation is used to animate the model. Once a model – representing a natural system – has been developed, the simulation – representing the operation of the model over time – can act as a descriptive or predictive tool. Finally in engineering processes (*pro*), simulations are rich in detail and used to trial technologies for economic reasons; they are a more efficient test than expensive hardware prototypes.

Comparing Systems: Natural *vs* Artificial. Comparative (animat) approaches[1] to science investigate the origins and mechanisms of phenomena. In treating artificial systems like natural ones we can deepen our understanding of nature, gain further insight, or develop a new means of problem-solving – for example Webb notes that

[1] We use the term “comparative approach” instead of the commonly used “animat approach” to show one alternative way to understand and explain the place of ALife in the biological sciences.

comparing a heart to a pump has an explanatory value distinct from modeling the process of blood circulation or creating an artificial heart [7]. The value lies in the pump being a kind of *source model* – usually seen as an artificial system that exists independently of the target system or hypothesis (i.e. the pump became a model for circulation after its creation) [7]. It is also common in the animat community to work with artificial and natural systems simultaneously (i.e. comparing the artificial and natural processes against each other, see [11] for an example). Thus comparison does not stop at treating artificial systems like natural ones; the term 'compare' implies that there is value in noting the similarities *and dissimilarities.* The *vs* approach *can* then be seen as a two-way explanatory relationship between the two kinds of system; which would have methodological similarities with ethological practices (e.g. [8]). Furthermore, this relation allows all possible similarities and dissimilarities to be addressed – i.e. counterfactual analysis. Under this approach we would identify at least *two* important kinds of experimentation. The first is *thought experiment* – 'devices of the imagination used to investigate the nature of things' [12]. Thought experiments have no accepted definition but are widely held to be a useful method for highlighting inconsistencies in theories and building intuitions [2]. A famous example of a thought experiment is Schrödinger's Cat. The experiment shows that a quantum system (a cat in a box with various items including a radioactive substance) can be in two states at once (a superposition), because until you open the box and observe whether the radioactive substance has decayed, there is no other option but to hold that cat is both alive and dead.[2] Secondly, we are introducing the notion of *extended experiment* to link the aims and methods of comparative approaches with those of the Extended Mind Thesis (EMT hereafter, [13]).[3] We see affinity between comparative approaches, which can promote understanding and allow new means of problem solving, and EMT, which holds that artefacts can be tools for thought – because they can function as, and therefore enhance, our cognitive processes. EMT holds that the physical mechanisms of our thinking extends beyond our biological boundaries when a two-way relationship between cognitive and external systems exists – for example using a smart phone as an external memory store, or a notepad to work out a sum [13]. Thus extended experiments can also be extended systems (as defined by EMT), because cognitive processing is enhanced and distributed across biological and technological boundaries: in devising new theories we're doing so with the additional processing power of technology, but it's a reciprocal relationship because our biological processes are necessary to make sense of a scientific experiment. An important outcome in viewing the process and evaluation of simulation work in this way is that "experiment as a tool to further understanding" underlines the distinct scientific character of simulation and the two-way relationship between natural and artificial. It has been argued that simulation experiments have no scientific value (reported by Di Paolo et al. in [2]): they are seen merely as computer programs in which symbols are re-arranged logically and as such cannot give rise to new knowledge. However, given that in our taxonomy extended experiments can be tied to

[2] In separating thought experiments from ALife simulations we are agreeing with Wheeler that 'ALife models are not thought experiments – philosophical or scientific' [14].

[3] See Humphreys [3] for an alternative (non-embodied) approach to defining hybrid investigations.

cognitive processes as tools for problem-solving, they can be more than "just a computer program" – they can be part of a process which facilitates conceptual development. Therefore, extended experiments do not *test* concepts; rather, they *come up with* concepts, aiding the (re-)formation of concept and theory. ALife holds a special role within this context because it i) serves as a bridge between disciplines [14], [16], ii) allows phenomena to be described in abstract terms [1] and, iii) helps us derive intuitions about life [2].

Evolutionary Robotics, like the origin of learning study mentioned above, is full of experiments that can be viewed as comparative (see also [5], [15] and [17]). Similarly, Ponticorvo and Miglino's work [4] compares a simulation with a variety of natural behaviours. Their research aims for *insight* into the many potential mechanisms that cause spatial behaviours, in order to create "theoretical proof" – backing up the intuition that mechanisms of orientation behaviours in natural systems do not require a modular neuro-cognitive system. Importantly, they conclude that modeling work would be needed to provide "actual proof" for spatial mechanisms [4]. Consequently the study can be seen as effective if it aids cognitive processes for a re-conceptualisation of spatial behaviours in natural systems, as opposed to providing actual evidence that architecture *is* non-modular. The epistemological concerns here are centred on how the artificial is compared to the natural at different stages of investigation, how results relate to theory or intuition about a natural phenomena, and how we learn from processes we cannot fully access.

Modeling Systems: Natural *via* Artificial. Scientific models can be conceptual, computational, or mathematical representations of nature. The empirical inquiry surrounding modeling centres on how closely a model "applies" to anything in the world so as to be a useful prediction or explanation [7]. Once the model is said sufficiently (or perhaps roughly) to predict or explain something, the hypotheses are said to be true [1]. The degree to which 'the hypothesis accounts for existing data and predicts new data from observations on the target phenomenon is taken to support its status as an explanation' [7]. Cordeschi explains further: 'The artefact therefore embodies the explanatory principles (the hypotheses) of the theory and is considered a test of the theory's plausibility' [10].[4] However, the hypothesis is only associated with a simplified or narrow element of the natural system, factoring out unrelated phenomena, to make analysis possible [10]. The process of modeling as a whole is the process of creating a representation of a target phenomenon, testing this artificial system, and evaluating the success against evidence about the target system: the artificial system models the natural – this is as depicted in Figure 1.

An inclusive account of what constitutes a model is required here so that more general or conceptual work is permitted. One such account is Barandiaran and collegues's [18-19]; they propose that there are four types of modeling distinguished by their degree of abstraction: *mechanistic*, characterised by an almost one-to-one correspondence between variables in the model and observables in the natural system; *functional*, which aims for behavioral or functional (rather than a variable-to-variable) correspondence between the model system and the target natural system; *generic*,

[4] To clarify, we're not arguing for a specific view on the relationship between model and theory we mean to show that the structure provided here can accommodate various positions.

covering a wide spectrum of phenomena in search for generic principles of complex systems; and *conceptual*, which does not target any particular natural system nor a collection of them – it is built from theories. As such conceptual models illustrate concepts by representing theoretical principles [18]. Despite the abstract nature of the latter two modeling types, the process of an artificial system standing in for the natural remains common, allowing an exploration of a biologically founded hypothesis, which is subsequently evaluated on the strength of its ability to predict or explain the target natural systems or natural phenomenon. Models can also be used as metaphor to support justification. There are thus different levels of explanation and investigation: the construction of the model, the analysis of the model, and the way that the model is used to aid explanation.

The epistemological concerns within the modeling approach – how we learn from models – have a complex history (see [10], [20-21]), especially in light of new debates on how modeling differs from ALife simulation [1], [14]. In modeling, the natural phenomenon is related to the stages of designing, building, manipulating and evaluating in different ways [20]. The scientific process and the epistemological processes are thus intricately linked – and there are two key arguments for this interconnection. According to Hughes [20] there are three stages involved in gaining knowledge from modeling. These are denoting (which links a theory or hypothesis of a natural system to the building of the model), demonstration (connecting the natural system to the representative model) and finally interpretation (linking the success of predictions to the explanation of the natural phenomenon). On Hughes's view there are three clearly defined relationships between the model and the target system – which should hold for any level of abstraction. Another influential, and similar, view comes from Morgan. Her argument says that we learn from models in two ways – from building them, and from manipulating them – but it is the "representational mechanisms" involved which underlie both [21]. From both Hughes and Morgan, the representative essence of modeling provides the foundation for epistemological gain, and subsequently for philosophical debate.

Engineering Systems: Natural *pro* Artificial. Engineers use natural systems to develop novel solutions to engineering problems and to construct technological artefacts; the artificial system in this approach is then the output of the process. Alongside the epistemological levels of *vs* and *via* there is a further "level" distinction – between epistemological approaches and technological approaches. A key characteristic of technological work is the separation of design and fabrication [22]. This *pro* approach includes, among others: *bioengineering* – taking what we know of natural systems and adapting it for the development of new engineering solutions. This kind of approach is widely used in synthetic biology; *biomimicry* – directly copying nature to create new technologies; and *natural computation*, incorporating the use of natural systems to develop alternative problem-solving techniques, the use of computers to synthesise natural phenomena, and the use of natural materials to compute [22]. Alongside electronic hardware, computation can be implemented in a range of media (e.g. silicone).

An example of bioengineering is Micro-Aerial Vehicles [24] – small, insect-like flying devices developed for robustness and efficiency. Applications include video surveillance, weather mapping and military surveys. An example of biomimicry is the creation of buildings that copy the structure of termite mounds for a more efficient, cheaper means of air circulation than air conditioning [25]. Meisel et al. in [22] classify the many types of naturally-inspired computation, including cellular automata, neural computation, evolutionary computation, swarm intelligence, artificial immune systems, membrane computing and amorphous computing.

3 Differentiating Epistemological Concerns

We have just outlined how comparison and modeling are distinct – based on how each approach uses artificial systems to learn about natural ones. In this section we develop this argument, showing that what follows from this are two distinct sets of epistemological concerns. We then differentiate engineering practice.

The separation of the way that simulation and modeling relate to natural systems is important for *four* key reasons. *One*, comparative experiments have been seen as unscientific because their relation to natural systems is not easily defined, as it is in the more established approach of modeling. In modeling the explanandum is clearly stated at every stage but in extended experiments it might not need to be. This would mean embracing the difference between a model and a source model (see section 2): the model relates to a pre-specified natural phenomenon at all stages of investigation, whereas simulation is allowed to be separate, or perhaps more "opaque" – as in the way Di Paolo et al. mean in [2]. With proper foundations for separation, ALife can grow as a field and define its own scientific methods and processes. New kinds of experimental work might be permissible under this distinct approach. *Two,* following these "unscientific" criticisms, the evaluation stage of simulations has been seen as unsuccessful when assessed against modeling criteria. So, contrary to Webb [7], the epistemological value of a study using artificial means cannot be measured solely on the basis of its direct impact on reality (or on data generation) – because this does not relate to experimentation within the comparative approach – which can allow counterfactual analysis and foundational explanations [26]. *Three,* this taxonomy facilitates interdisciplinary communication. In clearly defining the investigation within a specific approach other researchers can access the work of fields vastly different from their own. This might be useful, for example, given that thought experiments are in the same category as ALife simulation: some of the philosophical literature on thought experiments might help the development of Alife [1]. Better foundations for communication would also aid research across related fields: paired together, practitioners of ALife and economics, say, or neuroscience and ecology, might see structures and patterns in practice and methodology, hitherto invisible and mutually beneficial. *Four*, practice and terminology overlap in all the approaches, which might confuse elements of an investigation that should be separate, or that operate on a different level. An example relating to terminology would be "simulation" (see section 2). Webb's argument for conflating modeling and simulation [7] illustrates the problems that arise when different levels of practice are confused. She claims that because theoretical proofs require comparison to the natural world (in their evaluation) simulations should be seen as a class of model because they are *methodologically* similar. However, this argument conflates the levels (and processes) involved in relating natural to artificial. The evaluative processes involved in modeling and simulation may seem similar, but the scientific processes as a whole are different. We should not want to evaluate two different *processes* in the same way, even if they come up with the same answer, because we will have lost a distinction they started with: theory, empirical data and the natural world are different. For example, in modeling we are matching the *data* from the model with the known *data* about the natural system – this is because the natural system has been specifically and directly related to the model. In the evaluative stages of comparison, however, we are assessing and contrasting the artificial system itself with a natural system, or perhaps a concept. The relationship of each scientific process within each approach thus operates on distinct conceptual levels.

Following these level distinctions, the key epistemological difference lies in the way that comparative experiments can build or question theory, whereas models can represent or justify theory. Due to the character of their relationship with natural systems, comparative experiments and modeling have a number of distinct epistemological issues. We sum up the key philosophical questions in these approaches as:

- *Comparative* – how we can know something new from a process we cannot fully access; how knowledge is acquired when intuition is involved; how we simplify an experiment of a natural phenomenon which is a characteristic of many different kinds of species (e.g. learning); and how we might apply a re-conceptualisation of a theory into future scientific practice (see [2] for a discussion of many of these).
- *Modeling* – how models denote theory; whether models represent theories; how we build and simplify the natural phenomenon to create successful models (which as mentioned above is linked to producing evidence); and the processes of interpreting the models to form predictions [20].

The separation of epistemological concerns is important. If there are different routes to gaining knowledge there must also be different ways to question how we are gaining that knowledge.

Issues in the Confusion between the Approaches. The engineering approach is more obviously distinct than modeling and comparison. However, despite the clearer difference it could still be confused with elements of the other approaches. We would identify two potential mistakes: a) evaluating the validity or usefulness of an artefact on the basis of how much it is bio-inspired; or b) drawing inspiration from a natural system to construct an artefact as an explanation of a phenomenon. For example, in the case of the termite-inspired building, it *would* be a mistake to evaluate the usefulness of this building because it is a faithful replication of a termite mound. The extent to which it is bio-inspired bears no relationship to the success of the engineered artefact. Its success is in lowering building and operational costs and increasing efficient air circulation – removing the need for air conditioning. So, although the termite inspired building is a good example of a *pro* approach, it would not be hard to see how it might be misinterpreted as either a model of temperature control or as a building that didn't look like a termite mound. An example of how this error manifests in artefacts that are constructed by faithfully copying a natural system – because they consider naturally-inspired design an accurate explanation of a phenomenon – can be found in the accidents (and fatalities) that occurred during the first human attempts of flight. Machines were built to imitate the structure and shape of bird wings because these where seen as an explanation of flight dynamics [27]. The first significant successes in flight were achieved only when the principle of imitation began to be separated from the design process.

An example that perhaps embodies both the issues with levels of explanation, and the confusion between using natural systems and aiming to explain them, is the potential for misunderstanding early (Classic) Artificial Intelligence. In Classic AI the metaphor of the *mind as computer* operated at a different explanatory level to the actual engineering and logic-based foundations, which focused on designing more useful computer programs [28]. Drawing on Cordeschi's example (in [10]), it would be a mistake to use the metaphor as more than a kind of explanation – i.e. as a basis to engineer an artefact to explain a phenomenon. A specific example of the distinct levels is found in the "frame-problem". This was originally an engineering issue for logic-based systems [28]; Dennet [29] subsequently noticed the epistemological concerns. The engineering problem has now been solved but the epistemological one remains an open debate [28]. If we apply the epistemological issue to engineering an artefact we would be holding the idea that a classic model of computer vision in a human environment could "see". Clearly, classical approaches to computer vision are models that work in experimental environments and the Classic-based-logic is sound; the issue here lies in treating the system like a natural one – as both an epistemic and *technological* artefact. Treating the artificial system like a natural one we have shown as characteristic of the epistemological comparative approach. Computational theories of mind thus operate at an epistemological level not an engineering one. Your laptop does not exhibit human behaviour *even if* some of its processing is akin to the processes of your cognitive system. The *mind as computer* is not invalidated as metaphorical use of a model, but it is invalidated as an actual hypothesis of human intelligence as a whole – there are no existing robots that exhibit human intelligence based on this Classic logic [30].

4 Conclusions

Distinction between comparative, modeling and engineering approaches can serve science and technology, resulting in clearer objectives and more effective interdisciplinary communication. As well as supporting dialogues within the approaches, our taxonomy would also aid cross-approach communication. ALife seems especially well placed for this [14]. We would however, like to point out that some overlaps between approaches *may* be fruitful; one might even use artificial systems developed in a comparative framework as a basis for developing a model [4]. So, while we think that research must be classified clearly, we don't rule out the existence of additional approaches or advocate sharp distinctions that might hinder scientific or technological development. Application for future research includes the potential to develop new methodologies within the approaches. For example, the comparative approach could benefit from the practice of running experiments with natural and artificial systems in parallel, as outlined in [11]. Despite the greater issues faced in designing such an experiment (e.g. significant heterogeneity of natural and artificial systems) they might allow explanations of hypotheses without modeling. A further challenge would be explicating the relationship between artificial and natural in more complex cases, such as when a natural system is itself used as a model – sea slugs, cress and the common fruit fly are all now employed in science as (living) models of other systems. Finally, with the introduction of our *extended experiments* terminology under the comparative approach our aim is to cover a range of “actual” artificial systems acting as tools for conceptual thought: through this we hope to give “a new life to ALife”.

Acknowledgements. We are grateful to Roberto Cordeschi, Xabier Barandiaran, Adam Barnard, Rose Buckler, Sascha Griffiths and the committee reviewers for their comments and suggestions. The second author would also like to thank colleagues at ISTC-CNR for helpful discussions while this work was prepared.

References

1. Bedau, M.A.: Can unrealistic computer models illuminate theoretical biology? In: Wu, A.S. (ed.) Proceedings of the 1999 Genetic and Evolutionary Computation Conference Workshop Programme, Orlando, Florida, July 13, pp. 20–23 (1999)
2. Di Paolo, E.A., Noble, J., Bullock, S.: Simulation models as opaque thought experiments. In: Bedau, M.A., McCaskill, J.S., Packard, N., Rasmussen, S. (eds.) Seventh International Conference on Artificial Life, pp. 497–506. MIT Press, Cambridge (2000)
3. Humphreys, P.: The Philosophical Novelty of Computer Simulation Methods. Synthese 169(3), 615–626 (2009)
4. Ponticorvo, M., Miglino, O.: Encoding geometric and non-geometric information. A study with evolved agents. Animal Cognition 13(1), 157–174 (2010)
5. Harvey, I., Di Paolo, A.E., Wood, R., Quinn, M., Tuci, E.: Evolutionary robotics: a new scientific tool for studying cognition. Artificial Life 11(1-2), 79–98 (2005)
6. Beer, R.D.: Toward the evolution of dynamical neural networks for minimally cognitive behavior. In: Maes, P., Mataric, M., Meyer, J., Pollack, J., Wilson, S. (eds.) From Animals to Animats 4. Fourth International Conference on Simulation of Adaptive Behavior, pp. 421–429. MIT Press, Cambridge (1996)
7. Webb, B.: Animals Versus Animats: Or Why Not Model the Real Iguana? Adaptive Behaviour 17(4), 269–286 (2009)
8. Wood, R., Baxter, P., Belpaeme, T.: A review of long-term memory in natural and synthetic systems. Adaptive Behavior 20(2), 81–103 (2012)
9. Tuci, E., Quinn, M., Harvey, I.: An evolutionary ecological approach to the study of learning behaviour using a Robot-Based model. Adaptive Behavior 10(10), 201–221 (2003)
10. Cordeschi, R.: Steps Toward the Synthetic Method: Symbolic Information Processing and Self-Organizing Systems in Early Artificial Intelligence Modeling. In: Husbands, P., Holland, O., Wheeler, M. (eds.) The Mechanical Mind in History, pp. 219–258. MIT Press, Cambridge (2008)
11. Morlino, G., Giannelli, C., Borghi, A.M., Nolfi, S.: Developing the Ability to Manipulate Objects: A Comparative Study with Humans and Artificial Agents. In: Johansson, B., Şahin, E., Balkenius, C. (eds.): Tenth International Conference on Epigenetic Robotics (EpiRob10), pp. 169–170. Lund University Cognitive Studies (2010)

12. Brown, J.R., Fehige, Y.: Thought Experiments. In: Zalta, E.N. (ed.): The Stanford Encyclopedia of Philosophy, Fall 2011 edn. (2011)
13. Clark, A., Chalmers, D.: The Extended Mind. Analysis 58(1), 7–19 (1998)
14. Wheeler, M., Bullock, S., Di Paolo, E.A., Noble, J., Bedau, M., Husbands, P., Kirby, S., Seth, A.: The View From Elsewhere: Perspectives on ALife Modeling. Artificial Life 8(1), 87–100 (2002)
15. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press, Cambridge (2000)
16. Trianni, V., Tuci, E., Passino, K.M., Marshall, J.A.R.: Swarm cognition: an interdisciplinary approach to the study of self-organising biological collectives. Swarm Intelligence 5(1), 3–18 (2011)
17. Morlino, G., Trianni, V., Tuci, E.: Evolution of Collective Perception in a Group of Autonomous Robots. In: Madani, K., Dourado Correia, A., Rosa, A., Filipe, J. (eds.) Computational Intelligence. SCI, vol. 399, pp. 67–80. Springer, Heidelberg (2012)
18. Barandiaran, X., Moreno, A.: ALife Models as Epistemic Artefacts. In: Rocha, L.M., Yaeger, L.S., Bedau, M.A., Floreano, D., Goldstone, R.L., Vespignani, A. (eds.) Tenth International Conference on Artificial Life, pp. 513–519. MIT Press, Cambridge (2006)
19. Barandiaran, X., Chemero, A.: Animats in the Modelling Ecosystem. Adaptive Behavior 17(4), 287–292 (2009)
20. Hughes, R.I.G.: Models and Representation. Philosophy of Science 64(S), 325–336 (1997)
21. Morgan, M.S.: Learning from Models. In: Morgan, M.S., Morrison, M. (eds.) Models as Mediators. Perspectives on Natural and Social Science, pp. 347–388. Cambridge University Press, Cambridge (1999)
22. Meisel, M., Pappas, V., Zhang, L.: A taxonomy of biologically inspired research in computer networking. Computer Networks 54(6), 901–916 (2010)
23. Heinemann, M., Panke, S.: Synthetic biology—putting engineering into biology. Bioinformatics 22(22), 2790–2799 (2006)
24. Dantu, K., Kate, B., Waterman, J., Bailis, P., Welsh, M.: Programming micro-aerial vehicle swarms with karma. In: Liu, J., Levis, P., Römer, K. (eds.) 9th International Conference on Embedded, SenSys 2011, Seattle, WA, USA, November 1-4, pp. 121–134. ACM (2011)
25. Biomimicry Institute, `http://biomimicryinstitute.org/case-studies/case-studies/termite-inspired-air-conditioning.html` (last accessed March 2012)
26. Langton, C.G.: Artificial Life. In: Langton, C.G. (ed.) Proceedings of the Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems (ALIFE 1987), Los Alamos, NM, USA. Santa Fe Institute Studies in the Sciences of Complexity, vol. VI, pp. 1–48. Addison-Wesley, Redwood City (1989)
27. Lillienthal, O.: Birdflight as the Basis of Aviation. Translation from the second edition. Longmans, Green, and Col, London (1911)
28. Shanahan, M.: The Frame Problem. In: Zalta, E.N. (ed.) The Stanford Encyclopedia of Philosophy. Winter 2009 edn. (2009)
29. Dennett, D.C.: Cognitive Wheels: The Frame Problem in Artificial Intelligence. In: Boden, M. (ed.) The Philosophy of Artificial Intelligence. Oxford University Press, Oxford (1990)
30. Pfeifer, R., Bongard, J.C.: How the Body Shapes the Way We Think: A New View of Intelligence. MIT Press, Cambridge (2006)

LocoKit: A Robot Construction Kit for Studying and Developing Functional Morphologies

Jørgen Christian Larsen, David Brandt, and Kasper Stoy

University of Southern Denmark
{jcla,david.brandt,kaspers}@mmmi.sdu.dk
http://modular.tek.sdu.dk

Abstract. We describe a robot construction kit named LocoKit and a method for studying functional morphologies. LocoKit consists of simple mechanical parts that allow for construction of a wide range of morphologies and modular electronics for instrumentation and control. The method relies on LocoKit for constructing functional morphologies and an experimental setup borrowed from the study of functional morphology in animals. We demonstrate the use of LocoKit and the method in a case study on quadruped locomotion and conclude that the methodology represents a systematic and efficient approach to the study and development of functional robot morphologies.

Keywords: Robot construction kit, experimental methodology, functional morphology.

1 The Role of Functional Morphologies in Robotics

There is a potential to increase the adaptability, robustness and energy efficiency of robots, while decreasing their complexity and cost, by adapting the function of their morphology to support their desired behaviors[1]. Unfortunately, it is not clear how to perform this morphological adaptation systematically and efficiently. In an attempt to address this problem, we propose an iterative, bottom-up, and model-free methodology that relies on a robot construction kit named LocoKit, shown in Figure 6, and an experimental setup borrowed from the study of functional morphology in animals.

Our methodology is based on the assumption that a robot's behavior is a result of the interaction between its morphology, control, and environment[1]. It is also based on the assumption that, for robots with non-trivial morphologies, this interaction is so complex that modeling is intractable and thus our methodology is in line with one of the tenets of behavior-based robotics, namely, that *the world is its own best model* [2]. A consequence of this is that to we have to rely on a tedious trial-and-error process to develop a useful combination of morphology and control for a given environment. Our first proposal is to use a robot construction kit to make the development process more efficient. A robot construction kit allows us to rapidly explore a morphological space without the need at every iteration to make mechatronics from scratch, for instance, in a

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 12–22, 2012.

study related to the case study on locomotion we will present in Section 5, we were able to build and do experiments with 51 variations of the same morphology in one day. The second proposal is to use an experimental setup typically used in the study of functional morphology in animals. Basically, the experimental setup provides data that is much richer than what is typically seen in robotics. An experimental setup may for instance provide motor control outputs synchronized with high-speed video, three-dimensional motion capture of key points on the morphology (allowing for calculation of relative position, speed, and acceleration), and with output of transducers both internal and external registering mechanical variables (e.g. forces, velocity, acceleration, pressure, etc.). This rich data allows us to measure overall performance parameters like speed and energy consumption and, in addition, how specific elements of the morphology or controller contribute to the whole. This giving us valuable data to systematically understand and improve both morphology and control. It is our hope that the methodology described here with its focus on experimental methodology and the use of LocoKit may provide a way to increase our understanding of the role of functional morphologies. In addition to making it practical to take advantage of functional morphologies in robots.

In the following, we will briefly describe related work, the methodology, the implementation of the LocoKit robot construction kit, the experimental setup, and a case-study demonstration on robot locomotion demonstrating the methodology in practice.

2 Related Work

We will limit our review of related work to methodologies that consider the morphology of the robot important. An approach inspired by artificial life is to evolve morphology and behavior in simulation [3] and then transfer the result to the real world either through three-dimensional printing [4] or by using modular robots [5]. However, both methods are limited by the intrinsic limitations of how complex interactions can be simulated. Another similar approach is to use a mechanical system whose configuration can be changed and use this as a basis for evolution of morphology in the real world [6].

Our approach is also related to the bio-inspired approaches where animal morphology inspires robot morphology, e.g., [7]. In particular, the work that puts a strong emphasis on experimental analysis and validation [8,9]. However, in contrast to this work, we use the experimental data as a design tool and not only for analysis and validation.

LocoKit originated in the field of modular robots [10,11], but is more similar to construction toys such as Meccano and LEGO Mindstorms. In contrast to the toys, LocoKit is more geared towards building sensor-and-actuator-rich, dynamic robots with flexible morphologies. The contribution of this paper is an extended description of our methodology compared to previous work [12] and the introduction of LocoKit.

3 Methodology for Testing Morphological Hypotheses

The methodology we propose is outlined in Figure 1. The methodology is hypothesis-driven so the first task is to formulate a hypothesis regarding functional morphology. This hypothesis may have a biological origin, but this is not crucial. The second step is to build a robot using a construction kit to test the hypothesis. The construction of this first robot is guided by intuition and experience, however, as we will see errors or bad designs may be discovered and corrected in later iterations. Once the robot has been constructed, relevant morphological and motion data are measured. What to measure was outlined in the introduction and will be elaborated in the section on experimental setup later. This rich data provide the background for analyzing and explaining the function of the morphology. It is an advantage if it is possible to obtain measurements in a form that is comparable to that of other morphologies, biological or artificial, because that makes it possible to do a comparative analysis. This helps not only in analysing the data, but also in explaining the functionality of the morphology. Undesired functionality can often be tracked to mechanical or control problems in the robot design which then can be fixed in another iteration. Once the function of the morphology is acceptable from a technical point of view, the measured data can be used to support or reject the scientific hypothesis or, as is often the case, be used to refine the hypothesis.

The methodology is based on an experimental methodology that is common to all of science. The methodology is, however, optimized for the study of functional morphology. In particular, we suggest to use a robot construction kit and an experimental setup borrowed from biomechanics.

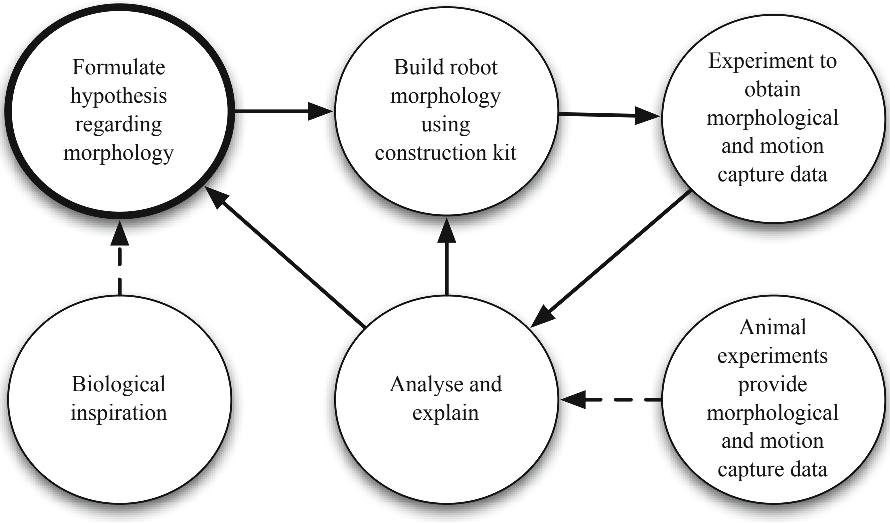

Formulate hypothesis regarding morphology
Build robot morphology using construction kit
Experiment to obtain morphological and motion capture data
Biological inspiration
Analyse and explain
Animal experiments provide morphological and motion capture data

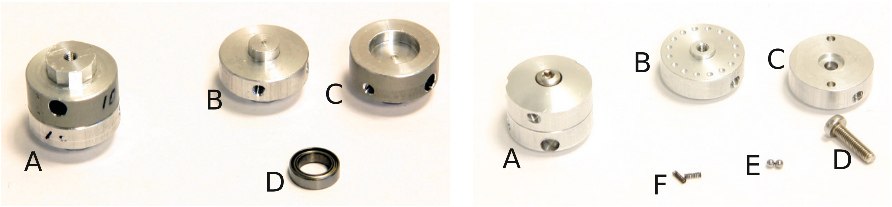
Fig. 1. Methodology for testing hypotheses regarding functional morphology (see main text for explanation)

4 LocoKit Implementation

4.1 Mechanical Components

The LocoKit elements connect with 4mm, off-the-shelf rods of varying lengths. We use two types of rods either carbon-fibre or fibre-glass enhanced plastic rods, but other materials are available as well. All of the construction elements of LocoKit can be connected to or mounted on these rods using one or more set screws. In the following four sections we will describe the four groups of mechanical components.

Structural Components. All morphological structures built from LocoKit are built from the rods described above and two mechanical connectors, a fixed joint and a rotary joint, both made in aluminium.



(a) Rotary joint (A). Left part of joint (B). Right part of joint (C). Bearing between right and left sides (D). The two sides are glued onto the bearing.

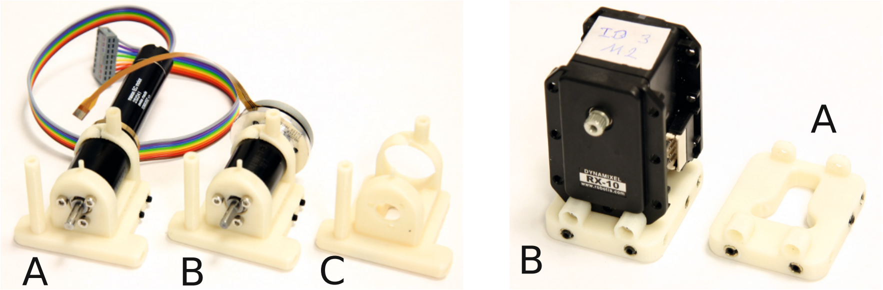
(b) Fixed joint (A). Left part of joint (B). Right part of joint (C). Mounting bolt (D). Balls (E). Springs pushing the spheres towards opposite side (F).

Fig. 2. LocoKit joints

The rotary joint, shown in Figure 2a, is used to connect two rods and allows the rods to rotate freely in parallel planes. On the sides of the rotary joint there are mounting points that allow for future, additional elements to be attached.

In Figure 2b the fixed joint is shown. This joint can, like the rotary joint, connect two rods, but in contrast to the rotary joint, it can fix the angle between them. The joint rotates freely initially, but provide tactile feedback at one of 12 evenly distributed angles due to the spring-load steel ball and the 12 matching holes on opposite sides of the joint. If the position is to stay fixed inside or outside the 12 preferred angles the outside screw can be fastened. This arrangement allows for rapid, precise construction of commonly used construction elements such as rectangles, triangles, etc., but leave the user free to use an arbitrary angle if needed.

Motor Mounts. In LocoKit we currently use three types of motors and two motor mounts since two of the motors use the same type of mount. Despite being different, the two mounts are designed to be interchangeable without changing the mechanical structure to which it is attached. We currently support Dynamixel Rx-10 motors, which together with its mount is shown in Figure 3b, and Maxon motors with a 22mm gearing, two of which are shown attached to their mount in Figure 3a.

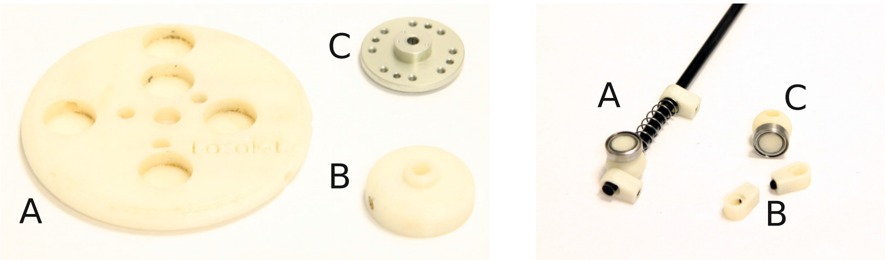


(a) Maxon motor mount (C), mounted with a 40W motor (A), and a 6W motor (B).

(b) Dynamixel motor mount (A) with motor mounted (B).

Fig. 3. LocoKit motor mounts

Transmission Components. In Figure 4a we see the components used to transfer the torque from a motor to the connection disk (A). The design includes two converter disks, (C) and (B), which allow us to transfer torque from both Maxon and Dynamixel motors. The connection disk has five holes at different distance to the center which provides the attached rod rotation with different offsets.



(a) (A) Connection disk. (C) and (B) Converter disks allowing us to use different motors on the same connection disk.

(b) (A) A rod compliantly attached to the connection disk. (B) Pieces that connect a rod to a connection disk.

Fig. 4. LocoKit transmission components

In Figure 4b is shown the components we use to attach a rod to a connection disk. The assortment of pieces allow us to mount a rod that rotates freely in a plane parallel to the connection disk and allow us to make connections that are compliant in the direction of the rod.

Miscellaneous Elements. In addition to the core elements presented so far, we also have a range of feet elements to be mounted at the end of a rod ranging from simple ball-feet to spring-load compliant feet. We also have elements for mounting batteries and power, control, and motor electronics boards, which we will describe below, to the frame of the robot.

4.2 Electronics

LocoKit electronics provide internal and external communication, processing, a sensor interface, low-level control of actuators, and a power supply for electronics as well as actuators. These functionalities have been distributed across three boards: power, processing and communication, and motor control. The processing and communication board also hosts the sensor interface. This design makes it easy to extend LocoKit with new boards, for instance for new types of actuators. It is also relatively easy to replace one of the existing boards with an improved version at a later stage.

For communication internally between the processor and the motor controller boards we use a full-duplex RS-485 bus operating at 1Mbps. In order to reduce the amount of wires the communication is wired through a connector which also distributes power.

Power Board. The power board provides a stable 24V voltage at a continuous current of up to 10A with an efficiency between 90-95% (a result based on experiments). The power board provides 240W, which is considered more than sufficient, while keeping weight to a minimum. We use lithium polymer batteries due to their high power to weight ratio. We use 6 cells in series giving a voltage of 18-25.2V. We have a battery pack that weighs 450g with a capacity of 2650mAh and another weighing 100g with a capacity of 600mAh for smaller robots. The power board also provides protection for the batteries.

Processor and Communication Board. The main purpose of the processor board is to provide a computational platform for control applications and data logging, a wireless interface to a PC and an interface to sensors. We use the commercially available Gumstix Overo Air board which includes: A 600MHz TI OMAP3 processor, 512MB of RAM, Wifi, Bluetooth and an microSD card reader. The use of Gumstix enable us to use Linux as the operating system for LocoKit, enabling us to use many standard software packages directly. Currently, we are using the Angstrom Linux distribution which is targeted at embedded systems. The Gumstix board is mounted on an interface board, which together make the processor and communication board. The interface board provides various functions and interfaces: low voltage regulation, three push buttons, 5 LEDs, 8 general purpose digital I/O, 4 analog inputs, an I^{2}C interface, an RS-485 interface, and finally a USB device port for debugging.

Motor Controller Boards. Each motor in the robot is controlled by a separate board, these boards implement the low-level motor control algorithms and the power electronics needed to operate the motors. The motor controller boards are implemented using a small 48MHz ARM7 processor which performs both RS-485 communication with the processing and communication board and low-level motor control. The motor is driven by a standard triple half-bridge. Besides the essential functionality a simple sensor interface is provided consisting of four pins which can be configured as digital inputs or outputs or as analog inputs. The physical connectors are compatible with most hall sensor based brushless DC motors from Maxon.

5 Case Study

In our case-study we will demonstrate the use of our methodology and LocoKit in practice. We will focus on the question of how the morphology of the robot influences locomotion. Specifically, how compliant feet influence the robots ability to walk.

5.1 Experimental Setup

The experimental setup is shown in Figure 5. The track consists of three force plates[1] organized in a row with two wooden tracks at both ends. Between two 60cm x 50cm force plates the smaller 30cm x 50cm force plate was installed. The force plates as well as the track modules were covered with carpet. For motion capturing 7 infrared high-speed cameras[2] were installed around the track operating at 250Hz. One camera was, in addition to automatic motion capturing, recording high-speed video of the robot.

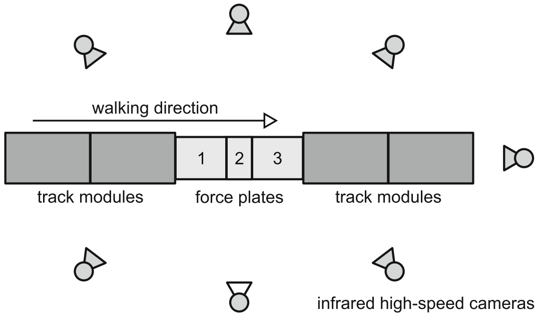


Fig. 5. The experimental setup consisting of three force plates and seven infrared cameras for motion capture. The camera with a white cap also recorded video.

[1] Two 60cm x 50cm, Type 9260AA6; one 30cm x 50cm, Type 9260AA3, Kistler, Winterthur, Switzerland.

[2] Oqus series, Qualisys, Gothenburg, Sweden.

5.2 SpringyBot Hypothesis

We use LocoKit to construct the robot, named SpringyBot, shown in Figure 6. The leg design has been inspired by concepts found in Scout [13] and iSprawl [14]. In Figure 6, a sketch shows how the legs of SpringyBot works. The continuous rotation of the connection disk at the top of the leg creates a foot trajectory as shown. The foot contains a spring which we hypothesize can keep the center-of-mass of the robot at a constant height as seen in a Spring Loaded Inverted Pendulum (SLIP) model [15].

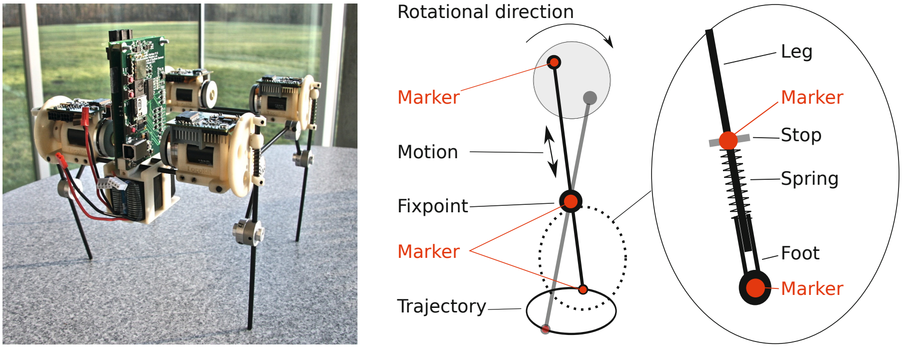
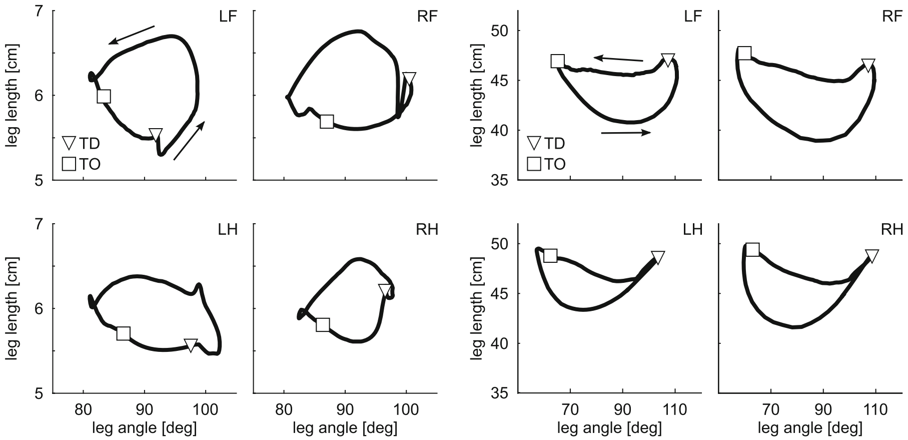


Fig. 6. The SpringyBot built from LocoKit and a schematic of its leg design. The leg is driven by the top disk and generates the trajectory shown. To be able to track key parts of the morphology infrared markers (red) have been placed at the positions shown.

5.3 Experiments

In our experiments we had the robot walking using a slow trot gait on the test track while being monitored by the cameras as shown in Figure 5. We divided the resulting data into gait cycles and extracted the total length of the leg, the leg's angle with respect to the ground, and the length of the compressible part of the leg (due to the spring). From this we also calculated the incompressible part of the leg for completeness.

Figure 7 shows phase plots of the total leg length as a function of leg angle for SpringyBot and for comparison a trotting dog. The discontinuity of the phase plots of SpringyBot is due to the spring since it is the only element of the leg morphology that can change rapidly. What can be seen from the phase plots, most clearly from the top-left one, is that just after contact with the ground and just before take-off the leg is shortened and lengthened, respectively. In the alternative visualization of the data in Figure 8, it becomes clear that it is the spring that on contact with the ground immediately compresses fully, and shortly before takeoff extends fully again and thus is compressed during most of the stance phase. In other words, the spring works like a shock absorber. This could be a desirable function of the morphology, but our goal was that the



(a) Phase plot for trotting SpringyBot

(b) Phase plot for trotting dog.

Fig. 7. Phase plot for each leg (L - left, R - right, F - front, H - hind, e.g. LF - left front leg) showing the length of the leg as a function of specific angle for SpringyBot and a trotting dog (▽: Touch down, □: Take off)

spring should shorten the leg during mid-stance to keep the center-of-mass at the same height similar to that of the SLIP model and the function of the dog leg. The implication for us then is that the spring is too short or perhaps too weak to support the weight of the robot and as a consequence we will in the next iteration try with first a longer and then a stronger spring. Another interesting research direction that this data indicates is to study the use of two springs, one for absorbing the initial impact and one for stabilizing the center-of-mass.

6 Discussion

The specific problem or solution is not important here. The point is that discovering the problem or even realizing it without the use of the rich experimental data would have been difficult because of the robot's dynamic behavior. A problem that will only be aggravated as we move towards faster locomotion. A second point is that the data is recorded in a way that makes it directly comparable to animal data which also can be an aid in debugging morphological problems. Finally, since the robot is built from LocoKit we can rebuild the robot quickly and perform yet another experiment and thus rapidly converge towards the desired functionality.

At a more general level, the case study demonstrates how we use our experimental methodology in practice in the context of understanding the role of morphology in locomotion. A crucial point about this case study and our research method in general is that experimental data is absolutely crucial. Experimental data represents the true function of the morphology taking all constraints of the

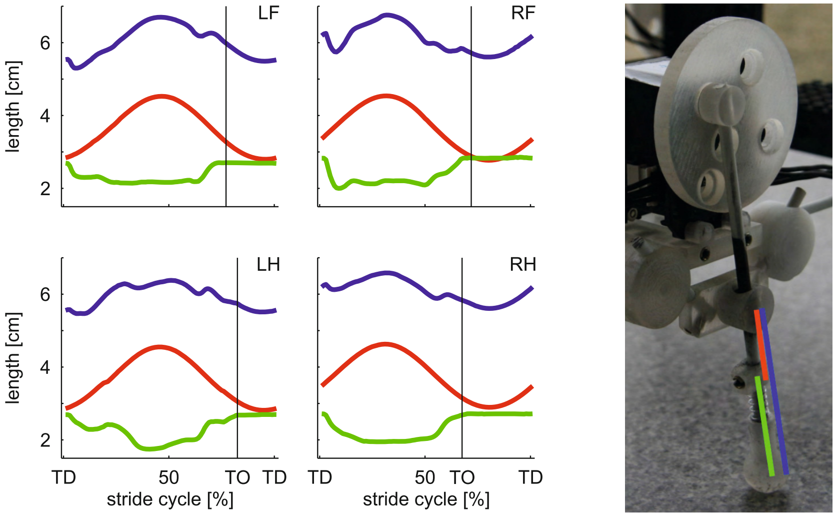


Fig. 8. Shows the length of different segments of the four legs during a stride cycle (colors refer to the photo on the right, and L - left, R - right, F - front, H - hind, e.g. LF - left front leg). The green curve represents the compressible part of the leg and shows how the spring is fully compressed during almost the entire stance phase.

environment into account and thus is a valuable design tool which in contrast to many other approaches may, if succesful, provide us with a functional robot at the end.

Modeling is not employed at this stage, but may be useful after the robot has been developed, for instance as a basis for simulation or comparison with similar robots. In other words, models are useful for analysis and comparison, but we think that experimental data is more valuable as a design tool for designing functional morphologies.

7 Summary

It is a significant challenge to understand and, for roboticists, take advantage of functional morphologies. In order to address this challenge, we introduced the LocoKit robot construction kit that allows for rapid exploration of a morphological space. LocoKit, together with an experimental setup borrowed from biology, also plays a crucial role in the proposed iterative, bottom-up, model-free method for the study of functional morphology in robots. We finally presented a case study showing how LocoKit and the methodology can be used in practice for the study of functional morphology in the context of locomotion. We conclude that LocoKit combined with our methodology allows for systematic and efficienct development and understanding of functional morphologies. As a whole, we hope that our contribution can aid in the understanding of and development of more morphologically advanced robots.

References

1. Pfeifer, R., Scheier, C.: Understanding Intelligence. The MIT Press (1999)
2. Brooks, R.: Elephants don't play chess. Robotics and Autonomous Systems 6, 3–15 (1990)
3. Sims, K.: Evolving 3D morphology and behavior by competition. In: Brooks, R., Maes, P. (eds.) Proc., Artificial Life IV, pp. 28–39. MIT Press (1994)
4. Lipson, H., Pollack, J.: Automatic design and manufacture of robotic lifeforms. Nature 406, 974–978 (2000)
5. Marbach, D., Ijspeert, A.: Co-evolution of configuration and control for homogenous modular robots. In: Proc., 8th Int. Conf. on Intelligent Autonomous Systems, Amsterdam, Holland, pp. 712–719 (2004)
6. Lichtensteiger, L.: Towards optimal sensor morphology for specific tasks: Evolution of an artificial compound eye for estimating time to contact. In: Proc., SPIE Sensor Fusion and Decentralized Control in Robotic Systems III, Boston, MA, USA, vol. 4196, pp. 138–146 (2000)
7. Spenko, M., Haynes, G., Saunders, J., Cutkosky, M., Rizzi, A., Full, R., Koditschek, D.: Biologically inspired climbing with a hexapedal robot. Journal of Field Robotics 25, 223–242 (2008)
8. Li, C., Hoover, A., Birkmeyer, P., Umbanhowar, P., Fearing, R., Goldman, D.: Systematic study of the performance of small robots on controlled laboratory substrates. In: Proceedings, Society of Photo-Optical Instrumentation Engineers Conference on Defense, Security, Sensing, Orlando, FL, USA, vol. 76790Z, pp. 1–13 (2010)
9. Nakatani, K., Sugimoto, Y., Osuka, K.: Demonstration and analysis of quadrupedal passive dynamic walking. Advanced Robotics 23, 483–501 (2009)
10. Yim, M., Shen, W.M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., Chirikjian, G.: Modular self-reconfigurable robot systems. IEEE Robotics & Automation Magazine, 43–52 (2007)
11. Stoy, K., Christensen, D.J., Brandt, D.: Self-Reconfigurable Robots: An Introduction. MIT Press (2010)
12. Larsen, J., Brandt, D., Stoy, K.: Systematic bottom-up robot design using a biomechanical experimental methodology. In: Proceedings, 15th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machine, Baltimore, MD, USA (submitted, 2012)
13. Buehler, M., Battaglia, R., Cocosco, A., Hawker, G., Sarkis, J., Yamazaki, K.: SCOUT: a simple quadruped that walks, climbs, and runs. In: Proceedings, IEEE International Conference on Robotics and Automation, Leuven, Belgium, pp. 1707–1712 (1998)
14. Kim, S.: iSprawl: Design and tuning for high-speed autonomous open-loop running. The International Journal of Robotics Research 25, 903–912 (2006)
15. Blickhan, R.: The spring-mass model for running and hopping. Journal of Biomechanics 22, 1217–1227 (1989)

An Introduction to the Analysis of Braitenberg Vehicles 2 and 3 Using Phase Plane Portrait

Iñaki Rañó

Institut für Neuroinformatik,
Ruhr-Universität-Bochum
inaki.rano@ini.rub.de

Abstract. Braitenberg vehicles are a widely used model of animal behaviour in robotics and Artificial Life. This paper presents the first comprehensive and formal analysis of the behaviour of Braitenberg vehicles 2 and 3. After a review of their intuitive behaviour we present their models as dynamical systems, that under circularly symmetric stimuli can be simplified and analysed using the phase plot technique. We prove that intuitive understanding is not enough to determine the potential behaviour of vehicles 2b and 3a and, under certain circumstances, they could behave as vehicles 2a and 3b respectively. This work paves the way to a proper understanding of Braitenberg vehicles, as it provides theoretical support for their many existing empirically grounded works.

1 Introduction

Braitenberg vehicles [1] have been used for decades on an empirical and intuitive basis on different areas like Artificial Life [2,3] or robotics. Each vehicle displays an expected behaviour according to the thought experiment presented in [1], where simple control mechanisms in the vehicle allow to implement adaptive behaviour. A key necessity to fully understand them from a theoretical viewpoint is the need to model the environment. The simplest vehicles actually model taxis behaviour [4], though the author provides metaphorical names for the resulting motion like "fear", "aggression" and "love". A key difference between the vehicles and existing models of animal steering [5] is that they inherently introduce the animal symmetry and restrictions to motion. Animals are very good at moving in the real world performing tasks like source seeking or trail following, very suitable tasks for some robotics applications. Therefore, these vehicles represent a good way to generate such behaviours.

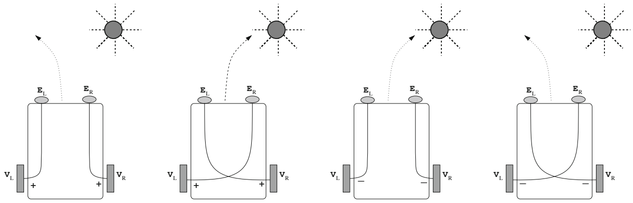
Even though Braitenberg vehicles prove empirically to be a useful mechanism to implement some behaviours, for instance in robotics, their applications rely only on empirical knowledge, which can be only achieved by trial and error. This paper presents the first qualitative review of Braitenberg vehicles 2 and 3, based on their model as non-linear dynamical systems. We prove that some trajectories the vehicles can show would actually not match the expected intuitive behaviour. The conditions under which the vehicles behave as expected are explicitly stated. Sticking to the spirit of the original idea of Braitenberg, the

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 23–32, 2012.

contents of the paper will be mostly theoretical with only qualitative analysis. The rest of the paper is organised as follows. The rest of this section reviews Braitenberg vehicle models 2 and 3 and presents some related works. Section 2 introduces their mathematical modelling, while the qualitative analysis of the trajectories is performed in section 3. Finally, section 4 draws some conclusions and presents future working lines.

1.1 Review of Vehicles 2 and 3

Braitenberg vehicles model apparently complex behaviour by including the effect of the environmental external stimuli. The vehicles model animal behaviour, abstracting and simplifying locomotion and perception, while capturing its main characteristics. Wheels abstract locomotion and focus on steering, so they can be used to model walking, swimming or crawling. This simplifies the control and analysis of motion, and is a good approximation because forward moving animals suffer from non-holonomic restrictions to motion like wheeled vehicles do [6]. On the perceptual side, animals are immersed in environments with many stimuli of different kinds, visual, acoustic or chemical among others. The vehicles can also perceive the stimuli at a point through their non-directional sensors. For vehicles 2 and 3 only a stimulus source is present. Usually a stimulus source produces a positive bounded stimulus that takes a maximum value where the source is, while the intensity decreases smoothly with the distance.



(a) Vehicle 2a (b) Vehicle 2b (c) Vehicle 3a (d) Vehicle 3b

Fig. 1. Braitenberg Vehicles 2 and 3

Fear and Aggression: Fear and aggression are two behaviours presented as vehicles 2a and 2b. According to this model, fear differs from aggression in the wiring between sensors and motors. We will speak of a-type vehicles for ipsilateral connection between sensors and motors, while contralateral connection will be noted b-type vehicles, as shown in figure 1. Vehicles 2, a- and b-type, have an increasing connection linking perception to action, represented by the + sign on figures 1(a) and 1(b). In both cases, the stimulus intensity is transmitted to the motor devices in such a way that a stronger stimulus produces a higher action response on the connected wheel. Therefore the velocity of the vehicle will be high when it is close to the source.

On the one hand, for ipsilateral (a-type) increasing connection, shown in Figure 1(a), the resulting behaviour is the so called fear. Since the wheel corresponding to the highest stimulated side will turn faster, the vehicle will move away from the source, or from high stimulus intensity areas. Moreover, the vehicle slows down far from the source, like if it was indeed frightened by the stimulus and safe far from it. On the other hand, the vehicle 2b, shown in Figure 1(b), models aggression. Since the connections are contralateral, the most stimulated side will have the smallest wheel velocity, and the vehicle will turn to face the stimulus speeding up for high values. This is why the behaviour is called aggression, the vehicle will reach a source with a maximal speed, like trying to hit it. Interestingly, it has been shown that, depending on the source and the connections, the aggression vehicle configuration produces trajectories different from the intuitive one [7,8].

Love or Taxis: Vehicles 3a and 3b display a behaviour named love, which matches animal tropotaxis [9], taxis with two symmetrically placed sensing organs. Like in the previous case, there are a- and b-type vehicles, with ipsilateral and contralateral connections between the sensors and the motors, but with a decreasing relation between the stimulus and the speed of the wheel (notice the minus sign in figures 1(c) and 1(d)). As pointed out in [10], the two vehicles produce different behaviours, named permanent love, the real taxis behaviour, and exploratory love in [10], some sort of negative taxis.

Figure 1(c) presents the a-type vehicle with a stimulus source close to the right sensor, such that the right wheel spins at a lower speed, and the vehicle turns towards the source with an overall decreasing velocity. If the stimulus source lays exactly in front of the vehicle, the trajectory will be a straight line with decreasing speed. Since the vehicle is assumed to be able to perceive the stimulus in any direction, if the source is located behind the vehicle, it will turn in the shorter angular direction to head the source while moving forward following a smooth trajectory. There is no certainty about the source being reached, since the turning speed could not be enough to head the proper direction and, therefore, the vehicle might miss it. The crossed connection presented in Figure 1(d) generates a different behaviour. Since the slow turning wheel is the one further from the stimulus, the vehicle will move away slowly when it is close to the source, but it will increase its speed as it moves further. The obtained behaviour corresponds to a negative taxis towards the stimulus as observed among some animals.

1.2 Related Works

Several kinds of Braitenberg vehicles have been used to provide robots with different abilities. Obstacle avoidance and target acquisition are implemented in [11], where Braitenberg vehicle 3a – to perform phototaxis – in a combination with a modified version of vehicle 2b – to avoid obstacles through infrared sensors – are used. In fact, the low level dynamical systems approach to behaviour generation [12] can be viewed as a Braitenberg vehicle with a stimulus build up from infrared sensor readings. Its motion is very smooth and natural, similar to the animal motion. Braitenberg vehicles are also used in [13] for local navigation, vehicle 2b is used for goal seeking, while vehicles 3b and 2a are used to avoid obstacles in the front and back of the robot respectively. The Braitenberg control system is a set of fuzzy rules that generates an offset velocity for each wheel instead of the velocity itself. Chemical source seeking is a very difficult problem implemented in robots using Braitenberg vehicles. An experimental analysis of vehicles 3a and 3b for odour source localisation is presented in [10], where the connection between sensors and motors is linear, though sensor readings are normalised and averaged. The sensor preprocessing is necessary due to the nature of the used stimulus.

Braitenberg vehicles are used as test-bed for evolutive behaviour generation sometimes implemented using neural networks. The work in [14] presents a four neuron model of cricket phonotaxis which can be comparable to the combination of vehicles 2a and 3b. Evolution of Braitenberg vehicle 2b using genetic algorithms is presented in [15], where the authors show that the evolved vehicles outperform free random evolution of agent morphology and control. In this work, fitness is measured in terms of a simulated reward, and the authors found that half of the individuals actually converged to vehicle 3a. Another example of evolutive strategies applied to Braitenberg vehicles can be found in [16]. In this case an initial random mixture of vehicles 3a and 3b is evolved on a Khepera robot to implement a wandering behaviour on a closed arena.

2 Mathematical Models of the Vehicles

Sticking to the original thought experiment of Braitenberg we will assume perfect stimulus sources and noiseless sensors. The first step to understand Braitenberg vehicles is to model the environment with all the existing stimuli. Since for vehicles 2 and 3 the environment consists on a single stimulus source which generates a time constant stimulus field, it can be modelled as a function of the position with a single maximum at the source. We will also assume an isotropic stimulus (no preferred emission direction) that decreases with the distance in the form of a two dimensional non-negative function $S(\mathbf{x})$ of the position $\mathbf{x} \in D \subset \Re^2$, with the source located at the origin of a Cartesian coordinate system. Therefore, the maximum of the stimulus is at the origin $S(\mathbf{0}) > S(\mathbf{x}) \forall \mathbf{x} \in D \subset \Re^2$. This means the gradient $\nabla S(\mathbf{0})$ becomes zero while the Hessian is a negative definite matrix $\mathbf{y}^t D^2 S(\mathbf{0}) \mathbf{y} < 0 \forall \mathbf{y} \in \Re^2$. The underlying assumption is that the function $S(\mathbf{x})$ is at least C^2.

For vehicles 2 and 3, there is a direct relation between the perceived stimulus and the velocity of the wheels that can be modelled as a function $F(s)$ taking non negative values, such that $v = F(s)$ where 's' is the stimulus on the sensor and 'v' is the speed of the wheel. The restriction $F(s) \geq 0$ forbids the vehicle to move backward, a biologically plausible assumption. We will restrict the analysis to smooth (C^2) $F(s)$ functions, which should be increasing for vehicles 2 and decreasing for vehicles 3, which means they have positive or negative derivatives respectively. The simplest example fulfilling this late constrain would be to select a positive or negative constant k and make $F(s) = k \cdot s$, however for $k < 0$ $F(s)$ would not be positive and we should rather choose $F(s) = g_0 + k \cdot s$ with $g_0 > 0$. Since $F(s)$ is increasing or decreasing for different vehicles, the compound function $F(S(\mathbf{x}))$ will have a maximum or minimum as it can preserve the maximum of $S(\mathbf{x})$ or turn it into a minimum.

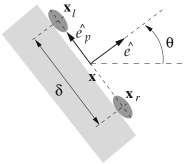


Fig. 2. Coordinate system at the front of the vehicle

From the velocities of each wheel, v_r and v_l, we can compute the overall motion of the vehicle as:

$$v = \frac{v_r + v_l}{2} \tag{1}$$

$$\omega = \frac{v_r - v_l}{d} \tag{2}$$

where d is the wheelbase of the vehicle.

2.1 Equations for a-Type Vehicles

Figure 2 shows the configuration of the sensors in front of the vehicle, where $\mathbf{x}$ represents the midpoint between the sensors separated by a distance δ. We can define the unit vector $\hat{e}^t = [\cos\theta, \sin\theta]$, pointing in the direction of the vehicle motion, and the vector $\hat{e}_p^t = [-\sin\theta, \cos\theta]$, orthogonal to $\hat{e}$, pointing to the left of the vehicle, forming a reference frame linked to the front of the vehicle.

For a-type vehicles the velocity of each wheel depends on the stimulus on the same side $v_r = F(S(\mathbf{x}_r))$ and $v_l = F(S(\mathbf{x}_l))$, where $\mathbf{x}_r$ and $\mathbf{x}_l$ are the positions of the right and left sensors. These velocities can be approximated by a Taylor series around the midpoint between the sensors as follows:

$$F(S(\mathbf{x}_l)) \approx F(S(\mathbf{x})) + \frac{\delta}{2}\nabla F(S(\mathbf{x})) \cdot \hat{e}_p \tag{3}$$

$$F(S(\mathbf{x}_r)) \approx F(S(\mathbf{x})) - \frac{\delta}{2}\nabla F(S(\mathbf{x})) \cdot \hat{e}_p \tag{4}$$

where $\nabla F(S(\mathbf{x}))$ is the gradient of the compound function evaluated at the midpoint between the sensors. Substituting equations (3) and (4) in (1) and (2), and using the unicycle motion model, the dynamical system describing the behaviour of a-type Braitenberg vehicles is:

$$\dot{x} = F(S(\mathbf{x}))\cos\theta \tag{5}$$
$$\dot{y} = F(S(\mathbf{x}))\sin\theta \tag{6}$$
$$\dot{\theta} = -\frac{\delta}{d}\nabla F(S(\mathbf{x}))\cdot\hat{e}_p \tag{7}$$

where $\mathbf{x} = [x, y]^t$. It is worth noting that, even though these equations model vehicles 2a and 3a, the actual shape of $F(S(\mathbf{x}))$ is very different, since for a stimulus source $F(S(\mathbf{x}))$ has a minimum at the origin for vehicle 3a, and a maximum for 2a. If the minimum of $F(s)$ for vehicle 3a takes the value zero the resulting dynamical system will have an equilibrium point at the origin, but it could be unstable. For vehicle 2a there is no stability point at the origin because it is a maximum of $F(s)$. Therefore, the behaviour of a-type vehicles is very different even they share the same dynamical model.

2.2 Equations for b-Type Vehicles

Vehicles of b-type have crossed connections between sensors and wheels, which means $v_r = F(S(\mathbf{x}_l))$ and $v_l = F(S(\mathbf{x}_r))$, but following the same procedure as in the previous section, the motion equations for this type of vehicles can be approximated as:

$$\dot{x} = F(S(\mathbf{x}))\cos\theta \tag{8}$$
$$\dot{y} = F(S(\mathbf{x}))\sin\theta \tag{9}$$
$$\dot{\theta} = \frac{\delta}{d}\nabla F(S(\mathbf{x}))\cdot\hat{e}_p \tag{10}$$

Even though the equations for vehicles 2b and 3b are the same, the properties of $F(s)$ are again different, as the dynamical system could have an equilibrium point for vehicle 3b but not for 2b. If the stimulus intensity reaching both sensors is the same, the vehicle will not turn since the turning rate is controlled by the directional derivative along the direction in which sensors lay $\hat{e}_p$. This is the result of sampling the stimulus at two different points. The forward velocity, on the other hand, is controlled by the intensity of the stimulus at the midpoint, as a first order approximation.

3 Analysis of the Resulting Behaviour

The above equations are valid for any kind of stimulus, but usually $S(\mathbf{x})$ is a function that only depends on the distance to the source, therefore, $F(S(\mathbf{x}))$ will also do. This means that the gradient of $F(S(\mathbf{x}))$ has only radial component. If the systems of differential equations (5), (6), (7) and (8), (9), (10) are converted to polar coordinates, the equations describing the motion of any Braitenberg vehicle can be stated as:

$$\dot{r} = F(S(r))\cos(\theta - \psi) \tag{11}$$

$$\dot{\psi} = \frac{F(S(r))}{r}\sin(\theta - \psi) \tag{12}$$

$$\dot{\theta} = \mp\frac{\delta}{d}\frac{\partial F(S(r))}{\partial r}\sin(\theta - \psi) \tag{13}$$

where (r, ψ) are the polar coordinates of the vehicle, and the sign on equation (13) depends on the type of vehicle, negative for a-type and positive for b-type.

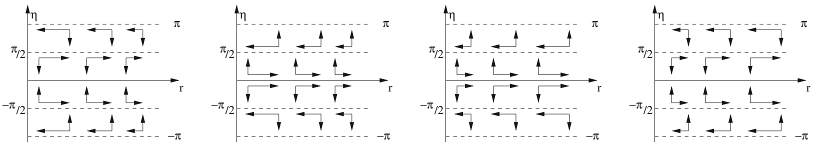
Since the angular variables always appear on the right hand side of the equations as the difference '$\psi - \theta$', a new variable $\eta = \psi - \theta$ can be defined to reduce the number of equations describing the evolution of the vehicle to:

$$\dot{r} = F(S(r))\cos\eta \tag{14}$$

$$\dot{\eta} = -\left[\frac{F(S(r))}{r} \pm \frac{\delta}{d}\frac{\partial F(S(r))}{\partial r}\right]\sin\eta \tag{15}$$

The new variable η represents the heading direction of the vehicle relative to the polar angular coordinate, such that when $\eta = 0$ the vehicle has the source on its back, if $\eta = \pm\pi/2$ the vehicle has the source to its left or right, and if $\eta = \pi$ the vehicle heads the source.

A dynamical system of two variables allows us to use the phase plane plot technique to analyse the behaviour of the vehicles. By drawing the qualitative vector fields generated by equations (14) and (15), we obtain all the trajectories at a glance, since they are the integral lines of the flow [17], i.e. trajectories come from following the directions of the arrows from a starting point on the plane. Given the conditions imposed to $F(S(\mathbf{x}))$ we can perform a qualitative analysis, which will depend on the sign of the discriminant function $G_{\pm}(r) = \frac{F(S(r))}{r} \pm \frac{\delta}{d}\frac{\partial F(S(r))}{\partial r}$. The phase plane portrait can be restricted to non negative radial values and to the angular range $\eta \in (-\pi, \pi]$, and it is presented for $G_{\pm}(r)$ functions that keep a constant sign in figure 3.



(a) Vehicle 2a. (b) Vehicle 2b. (c) Vehicle 3a. (d) Vehicle 3b.

Fig. 3. State space vector flows for circular symmetric conditions

3.1 Vehicle 2a

The reduced equations describing the behaviour of this vehicle under a circularly symmetric stimulus correspond to $G_{+}(r)$ in equations (14) and (15). By assumption $F(r) \geq 0 \forall r$, therefore the sign of the $\dot{r}$ component of the flow depends on the sign of $\cos\eta$, and it vanishes for $\eta = \pm\pi/2$, when the vehicle changes from approaching to moving away from the source or vice-versa. Specifically, since $\dot{r} > 0$ for $-\pi/2 < \eta < \pi/2$, the vehicle moves away from the origin when the source is on its back. Because $F(r)$ has its maximum at $r = 0$, the $\dot{r}$ component is larger close to the source. The angular component of the flow $\dot{\eta}$ vanishes, and changes its sign at $\eta = 0$ and $\eta = \pi$ since by assumption $F(s) > 0$ and $F'(s) > 0$, and, therefore, $G_+(r) > 0$. This means that the component of the flow $\dot{\eta}$ points in the opposite direction of $\sin\eta$, as plotted in Figure 3(a). There is no stability point unless $F(r) = 0$ for some r_0, and then there will be an attractor in the reduced state space at $(r_0, 0)$. In any case, there is an angular attractor at $\eta = 0$, which means the vehicle will always move away form the source.

3.2 Vehicle 2b

The equations for vehicle 2b correspond to $G_-(r)$, which means the radial component of the flow, shown in figure 3(b), is the same as for vehicle 2a. However, the sign of the discriminant function $G_-(r)$ can change for some r. If $G_-(r) > 0$, the flow is the same as for the vehicle 2a, and the resulting behaviour is fear instead of aggression. For $G_-(r) < 0$, the case presented in figure 3(b), the angular flow component points in the direction of $\sin\eta$ and the behaviour of the vehicle 2b is the one expected. If $G_-(r) < 0$, the angular variable has an attractor at $\eta = \pi$, the vehicle heading the source, but the dynamical system has no equilibrium point since $F(s) > 0$. It can be seen that there is a value r_0 such that the flow changes close to the origin, since $\lim\limits_{r\to 0^+} \frac{F(r)}{r} = \infty$ and $\lim\limits_{r\to 0^+} F'(r) = 0$. The consequence is that the vehicle moves away for small values of r and then it cycles around the source.

3.3 Vehicle 3a

The corresponding discriminant function for vehicle 3a is also $G_+(r)$, though in this case $F'(s) < 0$ and, therefore, the maximum of $S(r)$ is turned into a minimum. This corresponds to the vehicle slowing down close to the source and moving fast far from it, and, if the minimum value of $F(r)$ is zero, there is an equilibrium point at the origin. On the other hand, the sign of the radial component coincides with the previous sections and the flow, for the case $G_+(r) < 0$ is shown in figure 3(c), which corresponds to the taxis behaviour, i.e. the vehicle moving towards the source at $(0, \pi)$.

Since $F(s) > 0$ and $F'(s) < 0$ there could be a $r = r_0$ such that $G_+(r) = 0$ with a sign change in $G_+(r)$. However, contrary to the aggression case, if $\lim\limits_{r\to 0^+} F(r) = 0$ and since $\lim\limits_{r\to 0^+} F'(r) = 0$, the behaviour will not necessarily change close to the origin. If the source is initially on the back of the vehicle, $\eta \in (-\pi/2, \pi/2)$, it will move away and turn to head the stimulus, and it will reach the maximum distance from the source when $\eta = \pm\pi/2$. It is worth reminding, that the direction of the angular flow component can change making the vehicle move away from the source and challenging our intuitive understanding of vehicle 3a.

3.4 Vehicle 3b

The discriminant function for vehicle 3b is $G_{-}(r)$, which according to our assumptions cannot change its sign, since $F(s) > 0$ and $F'(s) < 0$. The corresponding flow on the phase plane is shown in figure 3(d), which has an angular attractor at $\eta = 0$, i.e. the vehicle with the source on its back, and can have an equilibrium point at $r = 0$ if $F(S(r))$ vanishes at the origin. However, analysing the phase plane, it is clear that the equilibrium is unstable, since any close trajectory will end up with the source behind the vehicle and escaping with increasing velocity, the above mentioned negative taxis. The behaviour of this vehicle is similar to the one of vehicle 2a but with increasing forward velocity as it moves far from the source.

4 Discussion and Further Work

Whilst previous works are mainly experimental and rely on empirical parameters adjustment, this paper presents the first formal analysis of Braitenberg vehicles 2 and 3, modelled as a dynamical system, and using the phase plane portrait. Even though their behaviour is easy to understand (reason why they are widely used for teaching) this paper shows that the real behaviour can be different from the intuitive one. Specifically, vehicles 2b and 3a can behave as 2a and 3b if there is a change on the corresponding discriminant function $G_{\pm}(r)$. This work supports and explains with a mathematical formulation existing empirical works of Braitenberg vehicles (and probably also explains unreported experimental results).

Relaxing the circularly symmetric assumption for the stimulus can generate more complex behaviour that needs to be analysed, while more specific conclusions for stimuli can be obtained if we assume $S(\mathbf{x})$ follows the inverse-square law or the inverse distance law. This paper deals only with the simplest theoretical configuration, a single stimulus source, but the environments can have different sources of several kinds. This will produce a richer behaviour and very complex equations that probably cannot be analysed analytically. However, numerical methods can be used for specific stimulus settings using the equations presented in this work.

References

1. Braitenberg, V.: Vehicles. Experiments in synthetic psycology. The MIT Press (1984)
2. Resnik, M.: LEGO, Logo and Life. In: Artificial Life. Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems, pp. 397–406 (1987)
3. Travers, M.: Animal construction kits. In: Artificial Life. Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems, pp. 421–442 (1987)

4. Fraenkel, G.S., Gunn, D.L.: The orientation of animals. Kineses, taxes and compass reactions. Dover Publications (1961)
5. Boeddeker, N., Egelhaaf, M.: Steering a virtual blowfly: simulation of visual pursuit. Proc. R. Soc. Lond. B 270, 1971–1978 (2003)
6. Arechavaleta, G., Laumond, J.P., Hicheur, H., Berthoz, A.: An optimality principle governing human walking. IEEE Transactions on Robotics 24(1), 5–14 (2008)
7. Rañó, I.: A bio-inspired controller to generate dense wandering workspace trajectories. In: Proceedings of the 7th Simposium on Intelligent Autonomous Vehicles (2010)
8. Rañó, I.: An Empirical Evidence of Braitenberg Vehicle 2*b* Behaving as a Billiard Ball. In: Doncieux, S., Girard, B., Guillot, A., Hallam, J., Meyer, J.-A., Mouret, J.-B. (eds.) SAB 2010. LNCS, vol. 6226, pp. 293–302. Springer, Heidelberg (2010)
9. Rañó, I.: A steering taxis model and the qualitative analysis of its trajectories. Adaptive Behavior 17(3), 197–211 (2009)
10. Lilienthal, A.J., Duckett, T.: Experimental analysis of gas-sensitive Braitenberg vehicles. Advanced Robotics 18(8), 817–834 (2004)
11. Bicho, E., Schöner, G.: The dynamic approach to autonomous robotics demonstrated on a low-level vehicle platform. Robotics and Autonomous Systems 21, 23–35 (1997)
12. Schöner, G., Dose, M., Engels, C.: Dynamics of behavior: theory and applications for autonomous robot architectures. Robotics and Autonomous Systems 16, 213–245 (1995)
13. Yang, X., Patel, R.V., Moallem, M.: A Fuzzy-Braitenberg Navigation Strategy for Differential Drive Mobile Robots. Journal of Intelligent Robotic Systems 47, 101–124 (2006)
14. Webb, B.: A Spiking Neuron Controller for Robot Phonotaxis, pp. 3–20. The MIT/AAAI Press (2001)
15. Mautner, C., Belew, R.K.: Evolving robot morphology and control. Artificial Life and Robotics 4, 130–136 (2000)
16. Salomon, R.: Evolving and optimizing Braitenberg vehicles by means of evolution strategies. Int. J. Smart Eng. Syst. Design 2, 69–77 (1999)
17. Kaplan, D., Glass, L.: Understanding Nonlinear Dynamics. Springer (1995)

Synthesis and Adaptation of Effective Motor Synergies for the Solution of Reaching Tasks

Cristiano Alessandro[1], Juan Pablo Carbajal[1], and Andrea d'Avella[2]

[1] Artificial Intelligence Laboratory, University of Zürich, Switzerland
{alessandro,carbajal}@ifi.uzh.ch

[2] Laboratory of Neuromotor Physiology, Fondazione Santa Lucia, Italy
a.davella@hsantalucia.it

Abstract. Taking inspiration from the hypothesis of muscle synergies, we propose a method to generate open loop controllers for an agent solving point-to-point reaching tasks. The controller output is defined as a linear combination of a small set of predefined actuations, termed synergies. The method can be interpreted from a developmental perspective, since it allows the agent to autonomously synthesize and adapt an effective set of synergies to new behavioral needs. This scheme greatly reduces the dimensionality of the control problem, while keeping a good performance level. The framework is evaluated in a planar kinematic chain, and the quality of the solutions is quantified in several scenarios.

Keywords: motor primitives, motor control, development.

1 Introduction

Humans are able to perform a wide variety of tasks with great flexibility; learning new motions is relatively easy, and adapting to new situations (e.g. change in the environment or body growth) is usually dealt with no particular effort. The strategies adopted by the central nervous system (CNS) to master the complexity of the musculoskeletal apparatus and provide such performance are still not clear. However, it has been speculated that an underlying modular organization of the CNS may simplify the control and provide the observed adaptability. There is evidence that the muscle activity necessary to perform various tasks (e.g. running, walking, keeping balance, reaching and other combined movements) may emerge from the combination of predefined muscle patterns, the so-called *muscle synergies* [1]. This organization seems to explain muscle activity across a wide range of combined movements [2–4].

The scheme of muscle synergies is inherently flexible and adaptable. Different actions are encoded by specific combinations of a small number of predefined synergies; this reduces the computational effort and the time required to learn new useful behaviors. The learning scheme can be regarded as developmental since information previously acquired (i.e. synergies) can be reused to generate new behaviors[5]. Finally, improved performance can be easily achieved by

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 33–43, 2012.

introducing additional synergies. Thus, the hypothetical scheme of muscle synergies would contribute to the autonomy and the flexibility observed in biological systems, and it could inspire new methods to endow artificial agents with such desirable features.

In this paper we propose a method to control a dynamical system (i.e. the agent) in point-to-point reaching tasks by linear combinations of a small set of predefined actuations (i.e. synergies). Our method initially solves the task in state variables by interpolation; then, it identifies the combination of synergies (i.e. actuation) that generate the closest kinematic trajectory to the computed interpolant. Additionally, we propose a strategy to synthesize a small set of synergies that is tailored to the task and the agent. The overall method can be interpreted in a developmental fashion; i.e. it allows the agent to autonomously synthesize and update its own synergies to increase the performance of new reaching tasks.

Other researchers in robotics and control engineering have recently proposed architectures inspired by the concept of muscle synergies. In [6] the authors derive an analytical form of a set of primitives that can drive a feedback linearized system (known analytically) to any point of its configuration space. In [7] the authors present a numerical method to identify synergies that optimally drive the system over a set of desired trajectories. This method does not require an analytical description of the system, and it has the advantage of assessing the quality of the synergies in task space. However, it is computationally expensive as it involves heavy optimizations. In [8] muscle synergies are identified by applying an unsupervised learning procedure to a collection of sensory-motor data obtained by actuating a robot with random signals. In [9] the architecture of the dynamic movement primitives (DMP) is proposed as a novel tool to formalize control policies in terms of predefined differential equations. Linear combinations of Gaussian functions are used as inputs to modify the attractor landscapes of these equations, and to obtain the desired control policy.

In contrast to these works, our method to synthesize synergies does not rely on feedback linearization, nor on repeated integrations of the dynamical system. The method is grounded on the input-output relation of the dynamical system (as in [8]), and it provides a computationally fast method to obtain the synergy combinators to solve a given task. Furthermore, our method is inherently adaptable as it allows the on-line modification of the set of synergies to accommodate to new reaching tasks.

2 Definitions and Methods

In this section we introduce the mathematical details of the method we propose. After some definitions, we present the core element of our method: a general procedure to compute actuations that solve point-to-point reaching tasks (see Sec. 2.1). Subsequently, in Section 2.2, we propose a framework for the synthesis and the development of a set of synergies.

Let us consider a differential equation modeling a physical system
$\mathcal{D}\left(\boldsymbol{q}(t)\right) = \boldsymbol{u}(t)$, where $\boldsymbol{q}(t)$ represents the time-evolution of its configuration

variables (their derivatives with respect to time are $\dot{\boldsymbol{q}}(t)$), and $\boldsymbol{u}(t)$ is the actuation applied. Inspired by the hypothesis of muscle synergies[3] [1], we formulate the actuation as a linear combination of predefined motor co-activation patterns:

$$\boldsymbol{u}(t) = \sum_{i=1}^{N_\phi} \boldsymbol{\phi}_i(t) b_i := \boldsymbol{\Phi}(t)\boldsymbol{b}, \tag{1}$$

where the functions $\boldsymbol{\phi}_i(t) \in \boldsymbol{\Phi}$ are called *motor synergies*. The notation $\boldsymbol{\Phi}(t)$ describes a formal matrix where each column is a different synergy. If we consider a time discretization, $\boldsymbol{\Phi}(t)$ becomes a $N\,\mathrm{dim}(\boldsymbol{q})$-by-$N_\phi$ matrix, where N is the number of time steps, $\mathrm{dim}(\boldsymbol{q})$ the dimension of the configuration space and N_ϕ the number of synergies.

We define *dynamic responses* (DR) of the set of synergies as the responses $\boldsymbol{\theta}_i(t) \in \boldsymbol{\Theta}$ of the system to each synergy (i.e. forward dynamics):

$$\mathcal{D}(\boldsymbol{\theta}_i(t)) = \boldsymbol{\phi}_i(t) \qquad i = 1 \ldots N_\phi. \tag{2}$$

with initial conditions chosen arbitrarily.

2.1 Solution to Point-to-Point Reaching Tasks

A general point-to-point reaching task consists in reaching a final state $(\boldsymbol{q}_T, \dot{\boldsymbol{q}}_T)$ from an initial state $(\boldsymbol{q}_0, \dot{\boldsymbol{q}}_0)$ in a given amount of time T:

$$\begin{aligned} \boldsymbol{q}(0) &= \boldsymbol{q}_0, \quad & \dot{\boldsymbol{q}}(0) &= \dot{\boldsymbol{q}}_0, \\ \boldsymbol{q}(T) &= \boldsymbol{q}_T, \quad & \dot{\boldsymbol{q}}(T) &= \dot{\boldsymbol{q}}_T. \end{aligned} \tag{3}$$

Controlling a system to perform such tasks amounts to finding the actuation $\boldsymbol{u}(t)$ that fulfills the point constraints[4] (3). Specifically, assuming that the synergies are known, the goal is to identify the appropriate synergy combinators $\boldsymbol{b}$. In this paper we consider only the subclass of reaching tasks that impose motionless initial and final postures, i.e. $\dot{\boldsymbol{q}}_T = \dot{\boldsymbol{q}}_0 = 0$.

The procedure consists of, first, solving the problem in kinematic space (i.e. finding the appropriate $\boldsymbol{q}(t)$), and then computing the corresponding actuations. From the kinematic point of view, the task can be seen as an interpolation problem; i.e. $\boldsymbol{q}(t)$ is a function that interpolates the data in (3). Therefore, a set of functions is used to build the interpolant trajectory that satisfy the constraints imposed by the task; these functions are herein the dynamic responses of the synergies:

$$\boldsymbol{q}(t) = \sum_{i=1}^{N_\theta} \boldsymbol{\theta}_i(t) a_i := \boldsymbol{\Theta}(t)\boldsymbol{a}, \tag{4}$$

[3] With respect to the model of time-varying synergies, in this paper we neglect the synergy onset times.

[4] In this paper we assume that the initial conditions of the systems are equal to $(\boldsymbol{q}_0, \dot{\boldsymbol{q}}_0)$.

where the vector of combinators $\boldsymbol{a}$ is chosen such that the task is solved. As mentioned earlier, if time is discretized, $\boldsymbol{\Theta}(t)$ becomes a $N\,\mathrm{dim}(\boldsymbol{q})$-by-$N_\theta$ matrix, where N_θ is the number of dynamic responses. The quality of the DR as interpolants is evaluated in Section 3.

Once a kinematic solution has been found (as linear combination of DRs), the corresponding actuation can be obtained by applying the differential operator; i.e. $\mathcal{D}\left(\boldsymbol{\Theta}(t)\boldsymbol{a}\right) = \tilde{\boldsymbol{u}}(t)$. Finally, the vector $\boldsymbol{b}$ can be computed by projecting $\tilde{\boldsymbol{u}}(t)$ onto the synergy set $\boldsymbol{\Phi}$. If $\tilde{\boldsymbol{u}}(t)$ does not belong to the linear span of $\boldsymbol{\Phi}$, the solution can only be approximated in terms of a defined norm (e.g. Euclidean):

$$\boldsymbol{b} = \arg\min_{\boldsymbol{b}} ||\tilde{\boldsymbol{u}}(t) - \boldsymbol{\Phi}(t)\boldsymbol{b}||. \tag{5}$$

When the time is discretized, all functions of time become vectors and this equation can be solved explicitly using the psuedoinverse of the matrix $\boldsymbol{\Phi}$,

$$\boldsymbol{\Phi}^+\tilde{\boldsymbol{u}} = \boldsymbol{\Phi}^+\mathcal{D}\left(\boldsymbol{\Theta}\boldsymbol{a}\right) = \boldsymbol{b}. \tag{6}$$

This equation highlights the operator $\boldsymbol{\Phi}^+ \circ \mathcal{D} \circ \boldsymbol{\Theta}$ ($\circ$ denotes operator composition) as the mapping between the kinematic combinators $\boldsymbol{a}$ (kinematic solution) and the synergy combinators $\boldsymbol{b}$ (dynamic solution). Generically, this operator represents a nonlinear mapping $\mathcal{M} : \mathbb{R}^{N_\theta} \to \mathbb{R}^{N_\phi}$, and it will be discussed in Section 4.

To assess the quality of the solution we define the following measures:

Interpolation error: Measures the quality of the interpolant $\boldsymbol{\Theta}(t)\boldsymbol{a}$ with respect to the task. Strictly speaking, only the case of negligible errors corresponds to interpolation. A non-zero error indicates that the trajectory $\boldsymbol{\Theta}(t)\boldsymbol{a}$ only approximates the task

$$\mathrm{err}_I = \sqrt{||\boldsymbol{q}_T - \boldsymbol{\Theta}(T)\boldsymbol{a}||^2 + ||\dot{\boldsymbol{\Theta}}(T)\boldsymbol{a}||^2}, \tag{7}$$

where $||\cdot||$ denotes the Euclidean norm, and the difference between angles are mapped to the interval $(-\pi, \pi]$.

Projection error: Measures the distance between the actuation that solves the task $\tilde{\boldsymbol{u}}(t)$, and the linear span of the synergy set $\boldsymbol{\Phi}$

$$\mathrm{err}_P = \sqrt{\int_0^T ||\tilde{\boldsymbol{u}}(t) - \boldsymbol{\Phi}(t)\boldsymbol{b}||^2 \mathrm{d}t}. \tag{8}$$

Forward dynamics error: Measures the error of a trajectory $\tilde{\boldsymbol{q}}(t, \boldsymbol{\lambda})$ generated by an actuation $\boldsymbol{\Phi}(t)\boldsymbol{\lambda}$, in relation to the task.

$$\mathrm{err}_F = \sqrt{||\tilde{\boldsymbol{q}}(T, \boldsymbol{\lambda}) - \boldsymbol{q}_T||^2 + ||\dot{\tilde{\boldsymbol{q}}}(T, \boldsymbol{\lambda}) - \dot{\boldsymbol{q}}_T||^2}. \tag{9}$$

Replacing $\tilde{\boldsymbol{q}}(t, \boldsymbol{\lambda})$, $\boldsymbol{q}_T$ and $\dot{\boldsymbol{q}}_T$ with their corresponding end-effector values provides the forward dynamics error of the end-effector.

2.2 Synthesis and Development of Synergies

The synthesis of synergies is carried on in two phases: exploration and reduction. The exploration phase consists in actuating the system with an extensive set of motor signals $\boldsymbol{\Phi}_0$ in order to obtain the corresponding DRs $\boldsymbol{\Theta}_0$. The reduction phase consists in solving a small number of point-to-point reaching tasks in kinematic space (that we call *proto-tasks*) by creating the interpolants using the elements of set $\boldsymbol{\Theta}_0$, as described in Eq. (4). These solutions are then taken as the elements of the reduced set $\boldsymbol{\Theta}$. Finally, the synergy set $\boldsymbol{\Phi}$ is computed using relation (2), i.e. inverse dynamics. As a result, there will be as many synergies as the number of the proto-tasks (i.e. $N_\phi = N_\theta$). The intuition behind this reduction is that the synergies that solve the proto-tasks may capture essential features both of the task and of the dynamics of the system. Despite the non-linearities of $\mathcal{D}$, linear combination of these synergies might be useful to solve point-to-point reaching tasks that are similar (in terms of Eq. (3)) to the proto-tasks (see Sec. 3).

The number of proto-tasks as well as their specific instances determine the quality of the synergy-based controller. To obtain good performance in a wide variety of point-to-point reaching tasks, the proto-tasks should cover relevant regions of the state space (see Sec. 3). Clearly, the higher the number of different proto-tasks, the more regions that can be reached with good performance. However, a large number of proto-tasks (and the corresponding synergies) increases the dimensionality of the controller. In order to tackle this trade-off, we propose a procedure that parsimoniously adds a new proto-task only when and where it is needed: if the performance in a new reaching task is not satisfactory, we add a new proto-task in one of the regions with highest projection error or we modify existing ones.

3 Results

We apply the methodology described in Section 2 to a simulated planar kinematic chain (see [10] for model details) modeling a human arm[11]. In the exploration phase, we employ an extensive set of motor signals $\boldsymbol{\Phi}_0$ to actuate the arm model and generate the corresponding dynamic responses $\boldsymbol{\Theta}_0$. The panels in the first row of Fig. 1 show the end-effector trajectories resulting from the exploration phase. We test two different classes of motor signals: actuations that generate minimum jerk end-effector trajectories (100 signals), and low-passed uniformly random signals (90 signals). In order to evaluate the validity of the general method described in Sec. 2.1, we use the sets $\boldsymbol{\Phi}_0$ and $\boldsymbol{\Theta}_0$ to solve 13 different reaching tasks without performing the reduction phase. The second row of Fig. 1 depicts the trajectories drawn by the end-effector when the computed mixture of synergies are applied as actuations (i.e. forward dynamics of the solution). It has to be noted how the nature of the solutions (as well as that of the responses), depends on the class of actuations used. The maximum errors are reported in Table 1. The results are highly satisfactory for both the classes of actuations, and show the validity of the method proposed. Since the reduction phase has not been performed, the dimension of the combinator vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ equals the number of actuations used in the exploration.

Table 1. Order of the maximum errors obtained by using $\boldsymbol{\Phi}_0$ and $\boldsymbol{\Theta}_0$ (no reduction phase)

	Min. Jerk	Random
err_I	10^{-15}	10^{-15}
err_P	10^{-5}	10^{-3}
err_F	10^{-4}	10^{-3}

The objective of the reduction phase is to generate a small set of synergies and DRs that can solve desired reaching tasks effectively. As described in Section 2.2, this is done by solving a handful of proto-tasks. The number (and the instances) of these proto-tasks determines the quality of the controller. Figure 2 shows the projection error as a function of the number of proto-tasks. The reduction is applied to the low-passed random signal set. Initially, two targets are chosen randomly (top left panel); subsequent targets are then added on the regions characterized by higher projection error. As it can be seen, the introduction of new proto-tasks leads to better performance on wider regions of the end-effector space, and eventually the whole space can be reached with reasonable errors. In fact, the figure shows that this procedure decreases the average projection error to 10^{-3} (comparable to the performance of the whole set $\boldsymbol{\Phi}_0$, see Tab. 1) and reduces the dimension of the combinator vector to 6, a fifteen-fold reduction. This result shows that a set of "good" synergies can drastically reduce the dimensionality of the controller, while maintaining similar performance. The bottom right panel of the figure shows the forward dynamics error of the end-effector obtained with the 6 proto-tasks. Comparing this panel with the bottom left one, it can be seen that the forward dynamics error of the end-effector reproduces the distribution of the projection error, rendering the latter a good estimate for task performance.

To further demonstrate that the reduction phase we propose is not trivial, we compare the errors resulting from the set of 6 synthesized synergies, with the errors corresponding to 100 random subsets of size 6 drawn from the set of low-passed random motor signals. Figure 3 shows this comparison. The task consists in reaching the 13 targets in Fig. 1. The boxplots correspond to the errors of the random subsets, and the filled circles to the errors of the synergies resulting from the reduction phase. Observe that, the order of the error of the reduced set is, in the worst case, equal to the error of the best random subset. However, the mean error of the reduced set is about 2 orders of magnitude lower. Therefore, the reduction by proto-tasks can produce a parsimonious set of synergies out of a extensive set of actuations. Evaluating the performance with different classes of proto-tasks (e.g. catching, hitting, via-points) is postponed to future works.

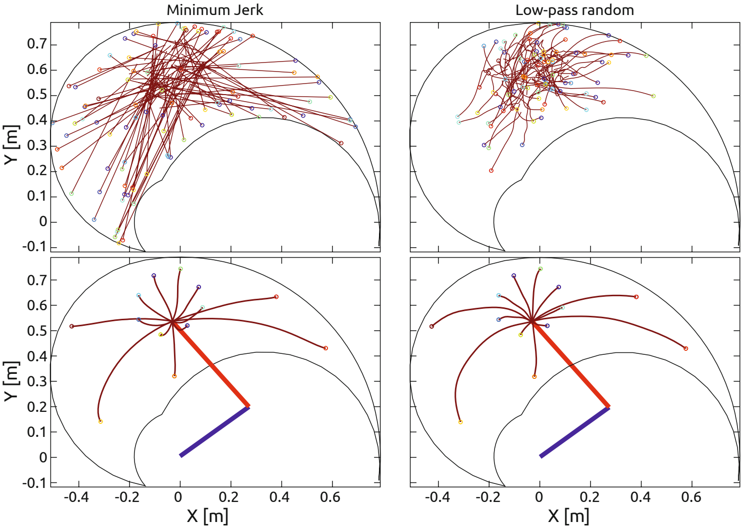


Fig. 1. Comparison of explorations with two different classes of actuation: minimum jerk and low-passed random signal. Each panel shows the kinematic chain in its initial posture (straight segments). The limits of the end-effector are shown as the boundary in solid line.

4 Discussion

The results shown in the previous section justify the interpretation of the methodology as a developmental framework. Initially, the agent explores its sensory-motor system employing a variety of actuations. Later, it attempts to solve the first reaching tasks (proto-tasks), perhaps obtaining weak performance as the exploration phase may not have produced enough responses yet (see the box-plots in Fig. 3). If the agent finds an acceptable solution to a proto-task, it is used to generate a new synergy (populating the set $\boldsymbol{\Phi}$), otherwise it continues with the exploration. The failure to solve tasks of importance for its survival, could motivate the agent to include additional proto-tasks; Figure 2 illustrates this mechanism. As it can be seen, the development of the synergy set incrementally improves the ability of the agent to perform point-to-point reaching. Alternatively, existing proto-tasks could be modified by means of a gradient descent or other learning algorithms. In a nutshell, the methodology we propose endows the agent with the ability to autonomously generate and update a set of synergies (and dynamic responses) that solve reaching tasks effectively.

Despite the difficulty of the mathematical problem (i.e nonlinear differential operator), our method seems to generate a small set of synergies that span the space of actuations required to solve reaching tasks. This is not a trivial result, since these synergies over-perform many other set of synergies randomly taken from the set $\boldsymbol{\Phi}_0$ (see Fig. 3). It appears as if the reduction phase builds features

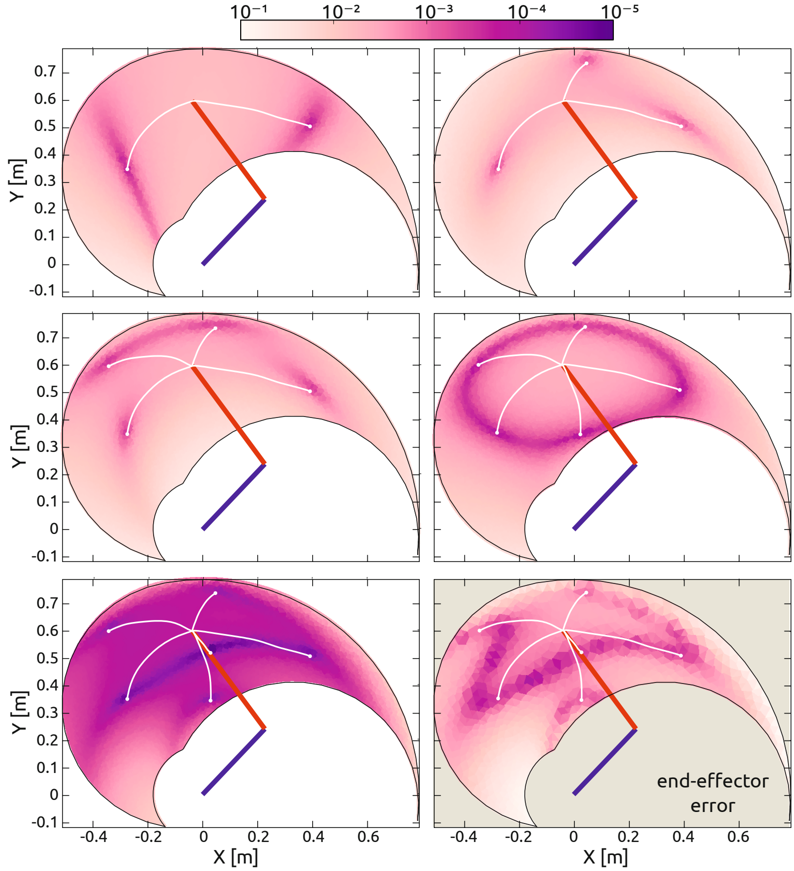


Fig. 2. Selection of targets based on projection error. Each panel shows the kinematic chain in its initial posture (straight segments). The limits of the end-effector are the boundary of the colored regions. The color of each point indicates the projection error produced to reach a target in that position. The bottom right diagram shows the forward dynamics error of the end-effector using 6 proto-tasks (6 synergies).

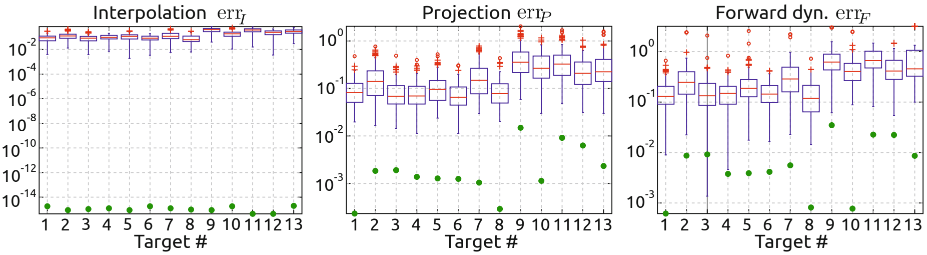


Fig. 3. Evaluation of the reduction phase. Errors produced by subsets randomly selected from the exploration-actuations (boxplots) are compared with the errors obtained after the reduction phase (filled circles).

upon the exploration phase, that are necessary to solve new reaching tasks. To verify whether solving proto-tasks plays a fundamental role, our synergies could be compared with the principal components extracted from the exploration set. This verification goes beyond the scope of this paper.

An important aspect of our method is the relation between $\boldsymbol{\Theta}$ and $\boldsymbol{\Phi}$ (see Eq. (2)). This mapping makes explicit use of the body parameters (embedded in the differential operator $\mathcal{D}$), hence the synergies obtained can always be realized as actuations. The same cannot be said, in general, for synergies identified from numerical analyses of biomechanical data. Though some studies have verified the feasibility of extracted synergies as actuations [12], biomechanical constraints are not explicitly included in the extraction algorithms. Additionally, Eq. (2) provides an automatic way to cope with smooth variations of the morphology of the agent. That is, both the synergies and their dynamic responses evolve together with the body. In line with [6, 7], these observations highlight the importance of the body in the hypothetical modularization of the CNS.

Once the task is solved in kinematic space, the corresponding actuation can be computed using the explicit inverse dynamical model of the system (i.e. the differential operator $\mathcal{D}$). It might appear that there is no particular advantage in projecting this solution onto the synergy set. However, the differential operator might be unknown. In this case, a synergy-based controller would allow to compute the appropriate actuation by evaluating the mapping $\mathcal{M}$ on the vector $\boldsymbol{a}$, hence obtaining the synergy combinators $\boldsymbol{b}$. Since $\mathcal{M}$ is a mapping between two finite low-dimensional vector spaces, estimating this map may turn to be easier than estimating the differential operator $\mathcal{D}$. Furthermore, we believe that the explicit use of $\mathcal{D}$ may harm the biological plausibility of our method. In order to estimate the map $\mathcal{M}$, the input-output data generated during the exploration phase (i.e. $\boldsymbol{\Phi}_0$ and $\boldsymbol{\Theta}_0$) could be used as learning data-set. Further work is required to test these ideas. Additionally, preliminary theoretical considerations (not reported here) indicate that the synthesis of synergies without the explicit knowledge of $\mathcal{D}$ is also feasible.

Finally, the current formulation of the method does not includes joint limits explicitly. The interpolated trajectories are valid, i.e. they do not go beyond the limits, due to the lack of intricacy of the boundaries. In higher dimensions, especially when configuration space and end-effector are not mapped one-to-one, this may not be the case anymore. Nevertheless, joint limits can be included by reformulating the interpolation as a constrained minimization problem. Another solution might be the creation of proto-tasks with a tree-topology, relating our method to tree based path planning algorithms[13].

5 Conclusion and Future Work

The current work introduces a simple framework for the generation of open loop controllers based on synergies. The framework is applied to a planar kinematic chain to solve point-to-point reaching tasks. Synergies synthesized during the reduction phase over-perform hundreds of arbitrary choices of basic controllers taken from the exploration motor signals. Furthermore, our results confirm that the introduction of new synergies increases the performance of reaching tasks. Overall, this shows that our method is able to generate effective synergies, greatly reducing the dimensionality of the problem, while keeping a good performance level. Additionally, the methodology offers a developmental interpretation of the emergence of task-related synergies that could be validated experimentally.

Due to the nonlinear nature of the operator $\mathcal{D}$, the theoretical grounding of the method poses a difficult challenge, and it is the focus of our current research. Another interesting line of investigation is the validation of our method against biological data, paving the way towards a predictive model for the hypothesis of muscle synergies. Similarly, the development of an automatic estimation process for the mapping $\mathcal{M}$ would further increase the biological plausibility of the model.

The inclusion of joint limits into the current formulation must be prioritized. Solving this problem will allow to test the method on higher dimensional redundant systems. Tree-based path planning algorithms may offer a computationally effective way to approach the issue.

Funds: The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013-Challenge 2 - Cognitive Systems, Interaction, Robotics- under grant agreement No 248311-AMARSi, and from the EU project RobotDoC under 235065 from the 7th Framework Programme (Marie Curie Action ITN).

Authors Contribution: **CA** and **JPC** worked on the implementation of the algorithm and the generation of the results reported here. The method was born during **JPC**'s visit to **AD**'s laboratory. **AD** provided material support for this development and uncountable conceptual inputs. All three authors have contributed to the creation of the manuscript. The authors list follows an alphabetical order.

References

[1] d'Avella, A., Saltiel, P., Bizzi, E.: Combinations of muscle synergies in the construction of a natural motor behavior. Nat. Neurosci. 6, 300–308 (2003)

[2] Ivanenko, Y.P., Cappellini, G., Dominici, N., Poppele, R.E., Lacquaniti, F.: Coordination of locomotion with voluntary movements in humans. J. Neurosci. 25(31), 7238–7253 (2005)

[3] Cappellini, G., Ivanenko, Y.P., Poppele, R.E., Lacquaniti, F.: Motor patterns in human walking and running. J. Neurophysiol. 95(6), 3426–3437 (2006)

[4] d'Avella, A., Fernandez, L., Portone, A., Lacquaniti, F.: Modulation of phasic and tonic muscle synergies with reaching direction and speed. J. Neurophysiol. 100(3), 1433–1454 (2008)

[5] Dominici, N., Ivanenko, Y.P., Cappellini, G., D'Avella, A., Mondì, V., Cicchese, M., Fabiano, A., Silei, T., Di Paolo, A., Giannini, C., Poppele, R.E., Lacquaniti, F.: Locomotor primitives in newborn babies and their development. Science 334(6058), 997–999 (2011)

[6] Nori, F.: Symbolic Control with Biologically Inspired Motion Primitives. PhD thesis, University of Genova (2005)

[7] Alessandro, C., Nori, F.: Identification of synergies by optimization of trajectory tracking tasks. In: The Fourth IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics, Roma, Italy, June 24-27, pp. 924–930 (2012)

[8] Todorov, E., Ghahramani, Z.: Unsupervised learning of sensory-motor primitives. In: Proc. 25th Int. Conf. IEEE Eng. Med. & Biol. Soc., pp. 1750–1753. IEEE (2003)

[9] Schaal, S., Peters, J., Nakanishi, J., Ijspeert, A.: Learning movement primitives. In: Dario, P., Chatila, R. (eds.) Robotics Research. Tracts in Adv. Rob., vol. 15, pp. 561–572. Springer (2005)

[10] Hollerbach, J.M., Flash, T.: Dynamic interactions between limb segments during planar arm movement. Biol. Cybern. 44(1), 67–77 (1982)

[11] Muceli, S., Boye, A.T., D'Avella, A., Farina, D.: Identifying representative synergy matrices for describing muscular activation patterns during multidirectional reaching in the horizontal plane. J. Neurophysiol. 103(3), 1532–1542 (2010)

[12] Neptune, R.R., Clark, D.J., Kautz, S.A.: Modular control of human walking: a simulation study. J. Biomech. 42(9), 1282–1287 (2009)

[13] Shkolnik, A., Tedrake, R.: Sample-based planning with volumes in configuration space. arXiv:1109.3145v1 (September 2011)

Gaze Allocation Analysis for a Visually Guided Manipulation Task

Jose Nunez-Varela[1,⋆], Balaraman Ravindran[2], and Jeremy L. Wyatt[1]

[1] School of Computer Science, University of Birmingham,
B15 2TT Birmingham, UK
{j.i.nunez,j.l.wyatt}@cs.bham.ac.uk
[2] Department of Computer Science and Engineering, IIT Madras,
600 036 Chennai, India
ravi@cse.iitm.ac.in

Abstract. Findings from eye movement research in humans have demonstrated that the task determines where to look. One hypothesis is that the purpose of looking is to reduce uncertainty about properties relevant to the task. Following this hypothesis, we define a model that poses the problem of where to look as one of maximising task performance by reducing task relevant uncertainty. We implement and test our model on a simulated humanoid robot which has to move objects from a table into containers. Our model outperforms and is more robust than two other baseline schemes in terms of task performance whilst varying three environmental conditions, reach/grasp sensitivity, observation noise and the camera's field of view.

Keywords: Gaze control, reinforcement learning, decision making.

1 Introduction

Acting under uncertain and incomplete information is required every time humans engage in some task. This uncertainty can be reduced by looking at relevant parts of the environment. The development of portable eye trackers has made it possible to study the role of human gaze during natural tasks, such as driving, tea and sandwich making, sports activities, etc. [10,12]. The key findings are that, where we look is determined by the task being performed, the purpose of looking is to reduce task relevant uncertainty, and that gaze patterns reflect extensive learning at several levels [12]. Furthermore, Johansson et al. [9] studied in detail the temporal and spatial relation between manipulation actions and gaze. Subjects had to grasp a bar and move it in a vertical plane to a target point, whilst avoiding an obstacle. The main conclusion was that gaze fixations specify landmark positions to which the manipulation actions are subsequently directed. In addition, empirical evidence from neurophysiological experiments has demonstrated that the human brain controls gaze via reward signals [8].

[⋆] We gratefully acknowledge the support given by CONACYT-Mexico (Reg.179604), UKIERI (SA06-0031), CogX (FP7-ICT-215181).

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 44–53, 2012.

In order to increase our understanding about how gaze can be coordinated, it is useful to create models that can be formally analysed in different conditions. Building on the work of Sprague et al. [11], we have defined a gaze allocation model based on a reinforcement learning (RL) framework [14] that allows for the control of the oculomotor system, multiple parallel motor systems, actions of variable duration, and that reasons about the expected actual reduction in uncertainty due to possible perceptual actions [4]. We implement and test our model using the iCub simulator[1] [2] (Fig. 1.A). Our manipulation task consists of picking up objects from the table top and then placing them inside one of two containers. In this work, the arms cannot interact with each other (e.g. to perform bi-manual grasps), so the task is divided into two sub-tasks, one for each arm. Thus, objects reachable by the right arm are placed inside the right container, with the same happening for the left side. A new object appears on the table every 60 seconds and also every time an object is put inside a container. We reward the robot for task performance. The key idea is to select the gaze at any moment that increases task rewards most by reducing task relevant uncertainty about the location of objects and containers. The only precisely known location to the robot is the centre of the table. In this paper we define our reward based model and characterise how task performance varies in terms of three environmental conditions, namely reach/grasp sensitivity in manipulation actions, the level of observation noise, and the size of the camera's field of view.

Besides Sprague et al. model, another reward based gaze control model was presented in [5] for a multitasking scenario. However, they do not consider multiple motor systems, only two fixation points are considered, and it is not a manipulation task. Other works have approached the problem of where to look based on models of saliency in order to detect regions of interest which may provide possible fixation points [16]. Nevertheless, these systems do not explicitly reason about how to reduce the uncertainty relevant to the task.

2 Coordinating Gaze and Actions

In this section we first describe how the task is modelled, then we explain the different components of our system and their interaction as Fig. 1.B illustrates. The robot initially learns the task and then performs the task by deciding what actions to perform and where to look.

2.1 Modelling the Task

As described in Section 1, the task is to pick up objects from the table top and then place them inside one of two containers. We consider two motor systems: the right and left arm/hand. We divide the task into two sub-tasks, one for each motor system. Furthermore, each sub-task is modelled with two levels. The

[1] The main advantage of using the iCub simulator is that it works with the same controllers used by the real robot. Thus, the transition between the simulation and the real robot is relatively straightforward.

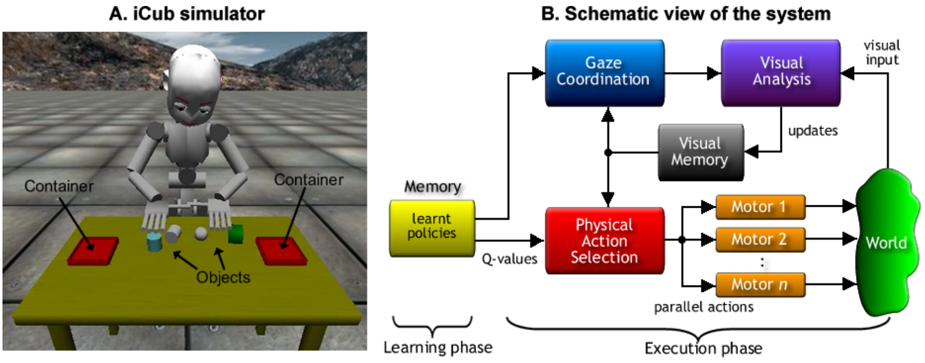


Fig. 1. A. Snapshot of the iCub simulator, where the task is to pick up and place objects from the table top to the containers. B. Schematic view of the system.

high-level models the sub-task qualitatively as a semi-Markov decision process (SMDP)[2] [13], with a discrete state representation. Whereas the low-level models each high-level action with a continuous state space representation.

In the high-level SMDP, each motor system $ms \in MS$ (where MS is the set of motor systems), is modelled as a tuple $\langle \mathcal{S}_{ms}, \mathcal{O}_{ms}, \mathcal{T}_{ms}, \mathcal{R} \rangle$, where $\mathcal{S}_{ms}$ is the set of discrete states, $\mathcal{O}_{ms}$ is the set of temporally extended high-level actions (called *options* as in [7]), $\mathcal{T}_{ms} : \mathcal{S}_{ms} \times \mathcal{O}_{ms} \times \mathcal{S}_{ms} \times \mathbb{N} \rightarrow [0, 1]$ is the transition probability distribution, where $\mathbb{N}$ is the set of natural numbers representing the execution time of each option, and $\mathcal{R} : \mathcal{S}_{ms} \times \mathcal{O}_{ms} \rightarrow \mathbb{R}$ is the reward function. For this domain, the reward function is identical for all motor systems.

In our case, an option $\mathcal{O}^j_{ms} \in \mathcal{O}_{ms}$ is modelled with continuous states $(\mathcal{S}^j)$ and actions $(\mathcal{A}^j)$, and with a non-stochastic transition function $(\mathcal{T}^j)$. Options are defined as commands provided by the motor controllers available to the iCub [3]. These controllers receive 3D positions in robot coordinates and they calculate and execute trajectories in the joint space of the robot.

The set of motor systems for the high-level SMDPs is defined as $MS = \{right\_arm, left\_arm\}$. A factorised discrete state space is used for both arms $(\mathcal{S}_{right\_arm}$ and $\mathcal{S}_{left\_arm})$, with the state variables: *armPosition* = {*onObject, onTable, onContainer, outsideTable*}, *handStatus* = {*grasping, empty*}, and *tableStatus* = {*objectsOnTable, empty*}. The set of options available for both arms $(\mathcal{O}_{right\_arm}$ and $\mathcal{O}_{left\_arm})$ are: *moveToObject*(2.65sec), *moveToTable*(2.95sec), *moveToContainer*(3.25sec), *graspObject*(2.87sec), and *releaseObject*(1.0sec). Next to each option is the average completion time in seconds, which was obtained by executing all options in sequence for 60 minutes. It is important to point out that for option *graspObject* we use a special command, defined in the iCub simulator, that makes the hand act like a magnet. Even though we simplify the problem of grasping, we can control its sensitivity by checking the offset between

[2] In contrast with the Markov decision processes (MDPs), SMDPs allow us to model actions with variable duration.

the centre of the hand and the centre of the object. Except for *releaseObject*, all options can fail if the offset between the centre of the hand and the desired final position is greater than some threshold (e.g. 1 cm). The iCub controllers have a limited accuracy and the minimum value of that threshold is 0.5 cm.

2.2 Visual Memory

The central component in the system is what we call the *visual memory* (Fig. 1.B), which captures the continuous state information about object pose needed for low-level control and supplies the discrete objects id's needed to create the high-level discrete state. We define the visual memory as the set of ordered pairs $\mathcal{VM} = \langle (e_1, bel(e_1)), (e_2, bel(e_2)), \ldots, (e_n, bel(e_n)) \rangle$, where e_i is the i^{th} entity of interest on the table, where an entity can be an object or a container. Every time a new entity is seen it is added to the visual memory. $bel(e_i)$ is a probability density (or belief state), associated with the location of the entity e_i. This location is used by the low-level options. Each object has a diameter of 4 cm, and its location refers to its centre projected in the X-Y plane in robot coordinates. Containers are 10x10x3 cm in width, length and height, and the location refers to its centre as well. The belief state for each entity e_i is approximated by a particle filter [15]. Each particle filter contains a set of particles $\mathcal{G}_i$, where each particle $g \in \mathcal{G}_i$ represents a possible location for entity e_i.

2.3 Learning Phase

As described above, each high-level SMDP models a given sub-task, and learning this sub-task is achieved via reinforcement learning (RL) using *SMDP Q-learning* [6]. Each motor system ms learns a policy $\pi_{ms} : \mathcal{S}_{ms} \to \mathcal{O}_{ms}$, that defines a mapping from states to options. A policy contains Q-values, i.e. the cumulative expected reward for each state-action pair. During learning we limit the number of objects appearing on the table to 10. We follow a minimal time to goal strategy, so for any option taken the robot receives -1 unit of reward. When the task is completed (i.e. no more objects appear on the table) it receives 0 units of reward. The robot initially learns how to perform the task under an assumption of complete observability, i.e. the visual memory contains the complete list of entities (e_i), and their corresponding belief ($bel(e_i)$) indicates the true location of that entity.

2.4 Execution Phase

During execution the robot is in charge of maintaining the visual memory, which does not contain any entity at the beginning. The robot has to look at entities in order to add its pair $(e_i, bel(e_i))$ into visual memory. Every time an entity e_i is observed, its belief $bel(e_i)$ is updated with new information.

Physical Action Selection: The robot selects options for each of its motor systems by following the *Q-MDP algorithm* [1] in terms of particle filters:

$$o_{ms} = \arg\max_{o \in \mathcal{O}_{ms}} \frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} weight(g) cost(g, o), \tag{1}$$

where $\mathcal{G}_i$ is the set of particles representing the belief state of entity e_i, and g refers to an individual particle. $weight(g)$ defines the weight given to the particle g. Each belief $bel(e_i)$ determines the likelihood of success or failure of options, which in turn determines changes in the discrete space described in Section 2.1. For instance, the discrete state variable *armPosition* takes the value *onObject* if and only if the offset between the centre of the hand and the centre of the object is less or equal than some threshold (e.g. 1 cm). The $cost(g, o)$ takes the value of $Q_{ms}(s, o)$ (i.e. the Q-value taken from the policy π_{ms}, where s is the current discrete state) if the offset between the centre of the hand and the particle g is less than or equal to some threshold, otherwise it takes the value of $\min_{o \in \mathcal{O}_{ms}} Q_{ms}(s, o)$, indicating that the option failed.

Gaze Coordination: The selection of perceptual actions works as follows:

1. Each entity e_i listed in visual memory represents a fixation point, i.e. a perceptual action $p \in \mathcal{P}$, where $\mathcal{P}$ is the set of perceptual actions at any given time. The number of perceptual actions varies depending on the number of entities in visual memory.
2. We define ms_f as the motor system that will benefit the most if it is given access to perception:

$$ms_f = \arg\max_{ms \in MS} \left[\max_{p \in \mathcal{P}} \{V^p_{ms}\} - \max_{o \in O_{ms}} \{V^o_{ms}\} \right], \tag{2}$$

where MS is the set of motor systems, and the expression inside the square brackets represents the expected gain that would result if gaze is allocated to motor system ms. V^p_{ms} is the expected value for motor system ms assuming perceptual action p is taken. V^o_{ms} is the expected value with the current uncertainty. In fact, $\max_{o \in O_{ms}} \{V^o_{ms}\}$ has been calculated during the option selection using (1), so we can cache this value. The difference between the maximum of these two values tells us how much we gain if gaze is allocated to this motor system. To calculate V^p_{ms} we follow:

$$V^p_{ms} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \max_{o \in O_{ms}} \left(\frac{1}{|\mathcal{G}_i|} \sum_{g \in \mathcal{G}_i} weight(g, \omega) cost(g, o) \right), \tag{3}$$

where Ω is the set of observations, ω is a particular observation, $\mathcal{G}_i$ is the set of particles, and g is a single particle. $weight(g, \omega)$ represents the weight of particle g after having observed ω, and $cost(g, o)$ takes the value of $Q_{ms}(s, o)$ if the offset between the centre of the hand and the particle g is less than or equal to some threshold, otherwise it takes the value of $\min_{o \in \mathcal{O}_{ms}} Q_{ms}(s, o)$.

This equation is trying to predict the value of moving the gaze to a specific fixation point by using “imaginary” observations ω to update the current belief, and then checking the effect this possible new belief would have in the selection of options.

3. After a motor system is selected in (2), we define p_f as the perceptual action that will be executed:

$$p_f = \arg\max_{p \in \mathcal{P}} \left\{ V^p_{ms_f} \right\}, \tag{4}$$

which is straightforward if we cache the results of $\max_{p \in \mathcal{P}} \{V^p_{ms}\}$ when calculating (2).

The observations ω are sampled assuming that the robot would fixate on the mean of the cloud of particles $(mean(\mathcal{G}_i))$. First, a number of particles g_j are uniformly selected. Then a bivariate Gaussian distribution is chosen for each g_j by indexing an observation model learnt off-line (described below), according to the location of particle g_j and the imaginary fixation point $(mean(\mathcal{G}_i))$ with respect to the also imaginary oculomotor position. From the chosen bivariate Gaussian distributions the observations Ω are sampled.

Visual Analysis: During a fixation, visual input is read from the right camera. The entities inside the camera's field of view (FoV) are detected[3], and their 3D locations (in robot coordinates) are calculated using the plane of the table to obtain an estimate of the depth. Since only one camera is used the estimated 3D locations are noisy[4]. In order to handle this noise during execution, an observation model was learnt off-line. This model is used for updating the particle filters and for sampling imaginary observations. The observation model was created by systematically moving the robot's gaze and an object in the robot space, and storing the coordinates depicted in Fig. 2.A. With these data points 405 bivariate Gaussian distributions were fitted, each representing a particular relationship between the position of the camera, the location of the fixation point and the object. Fig. 2 shows three of these distributions. Distributions B and D represent the noise when objects are found at the left and the right of the fixation point respectively, their location is less accurate. Distribution C represents the noise when objects are found in line with the fixation point, which gives a more accurate location. Finally, if an entity in visual memory is not seen for more than 1.5 seconds, Gaussian noise with zero mean and 1.0 cm of standard deviation is added to its current estimate.

[3] We have a noiseless object detection by using the simulator, this let us focus just on a single source of uncertainty, namely the estimated 3D locations.

[4] Using both cameras for triangulation results in a perfect estimate, as the cameras are perfectly aligned in the simulator.

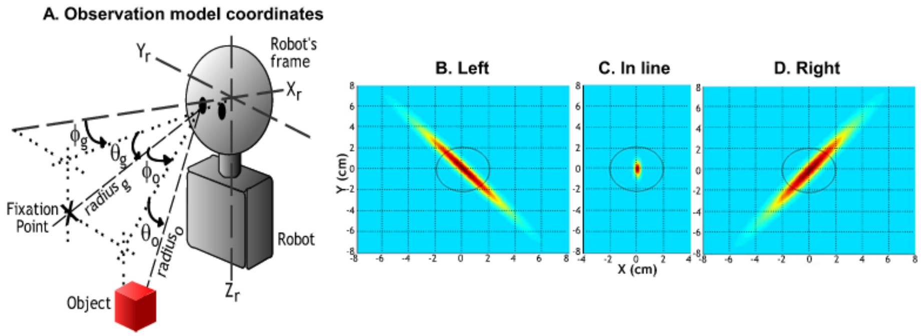


Fig. 2. Observation model. A. The coordinates specifying fixation points and object locations. B. Example of a distribution representing objects appearing at the left of the fixation point. C. Objects in line with the fixation point. D. Objects appearing at the right of the fixation point. The circle in the graph represents an object.

3 Experiments

We characterise our model by varying three environmental conditions: reach/grasp sensitivity in the manipulation actions, the level of observation noise and the camera's field of view. In order to compare our reward based model, we also implemented a *random*, and a *round robin* gaze allocation strategies.

Reach/Grasp Sensitivity Analysis: As described in Section 2.4, the success or failure of options is determined by a threshold. This threshold controls the sensitivity whilst reaching and grasping. We have defined six threshold values: 0.5, 1.0, 1.5, 2.0, 2.5 and 3.0 cm. Fig. 3.A shows the results of the average number of objects correctly placed in the containers for all gaze strategies. A total of 15 trials of 5 minutes each were performed for each strategy. The observation noise is 1.0, meaning that we use an unmodified version of the observation model, and the field of view (FoV) is 60°horizontally and 40°vertically (the default in the simulator), these angles refer to the complete FoV. The error bars represent the 95% confidence intervals. The results show how as the sensitivity increases (i.e. when it moves towards 0.5 cm), the performance of all three strategies decreases, since the task requires more accuracy. Notice that the ratio between the uninformed strategies (i.e. random and round robin), and the reward based strategy increases as the sensitivity increases. This means that if we require more accuracy, we also require an efficient way to allocate gaze. The main reason is that as the sensitivity increases the robot needs to fixate several times on an object before reaching and grasping can succeed, something that the reward based strategy is capable of doing but not the uninformed strategies.

Observation Noise Analysis: For these experiments we added noise to the observation model to test for robustness. As explained above, the observation model is a set of bivariate Gaussian distributions. Noise was added to the distributions by multiplying the two standard deviations that define each distribution by some factor (1.0, 1.5, 2.0 and 2.5, in this case). Fig. 3.B presents the results of the average number of objects correctly placed in the containers for all gaze strategies. A total of 15 trials of 5 minutes each were performed for each strategy. For this set of experiments the reach/grasp sensitivity is 1.0 cm and the FoV is 60°x40°. As the level of observation noise increases the performance of all strategies decreases. But notice how the ratio between the uninformed strategies and the reward based strategy grows as the observation noise increases, showing that the reward based strategy is much more robust to noise.

Field of View Analysis: For these experiments we vary the horizontal and vertical angles of the field of view with the values: (60°x40°), (50°x35°), (40°x30°), (35°x25°), (25°x20°). Fig. 3.C presents the results of the average number of objects correctly placed in the containers for all gaze strategies. A total of 10 trials of 5 minutes each were performed for each strategy. For this set of experiments the reach/grasp sensitivity is 1.0 cm and the observation noise is 1.0. As the field of view is reduced the performance of all strategies decreases, since less objects appear in the FoV. Although the reward based strategy outperforms the other two in all cases, it is interesting to notice that the performance of the uninformed strategies do not decrease so much. This is because as the FoV decreases *visual search* becomes a critical issue for the good performance of the task. The uninformed strategies are not too affected by this problem because they move the gaze more around the table, finding new objects indirectly. So, for instance, if the reward based strategy does not see objects on the left side of the table, the left arm remains idle. At the moment we have implemented a heuristic where every time the visual memory is empty the robot performs a systematic search for objects across the table. However, these results show that a better visual search strategy is required that can work together with the gaze allocation model in order to maintain a good performance.

There are two factors that should be taken into account for the analysis of all the previous results. First, the *peripheral information* has a major impact in the performance, especially when the FoV is large. We update all entities inside the FoV, this means that there might be objects that are never directly fixated but the robot still succeeds in grasping them, particularly if the sensitivity and/or observation noise decrease. This is why the performance of the uninformed strategies is so good in some cases. The effect of the peripheral information is more evident when the reach/grasp sensitivity threshold is between 2.0 to 3.0 cm, where round robin is even slightly better than our model. Second, the *decision time* is also an important factor. The reward based strategy takes in average 0.65 sec to decide where to look, compared to the almost immediate response of random and round robin. Unfortunately, it is not possible to directly compare our model to Sprague et al. model [11]. They model tasks with multiple concurrent goals within a single motor system, whereas we model tasks with multiple motor systems, with a single goal for each motor system.

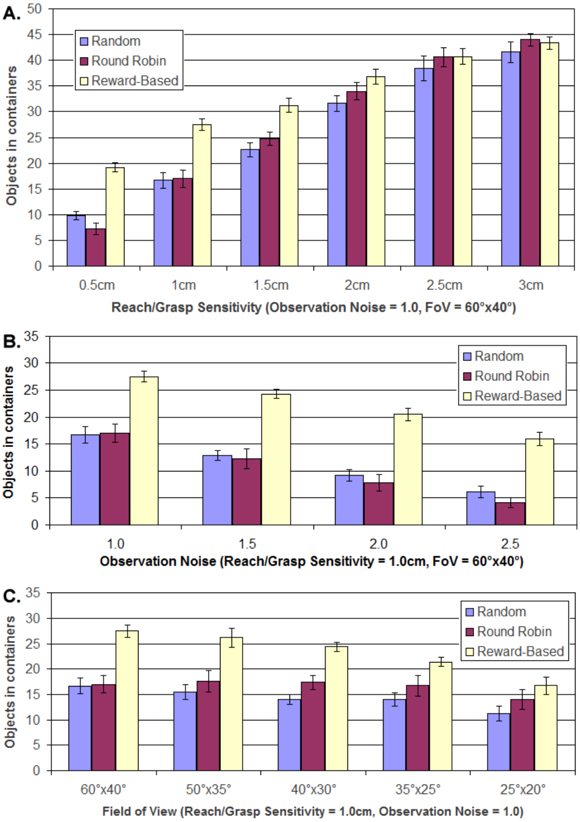


Fig. 3. A. Results for reach/grasp sensitivity analysis, with observation noise = 1.0 and field of view = 60°x40°. B. Results for observation noise analysis, with reach/grasp sensitivity = 1.0 cm and field of view = 60°x40°. C. Results for field of view analysis (horizontal x vertical angles), with reach/grasp sensitivity = 1.0 cm and observation noise = 1.0. The error bars represent the 95% confidence intervals.

4 Conclusions and Future Work

In this paper we have defined a gaze allocation model based on rewards by posing the problem of where to look as one of maximising task performance by reducing task relevant uncertainty. In particular, the robot selects the gaze at any moment that increases task rewards most by reducing task relevant uncertainty about object location. Our experiments show the behaviour of this reward based strategy compared with a random and round robin strategies, while varying three environmental conditions: reach/grasp sensitivity, observation noise, and the field of view. As the sensitivity and/or noise increases, the performance of all strategies decreases. If the FoV is reduced, the performance is expected to decrease, although we have concluded that a visual search mechanism is needed in order to maintain good performance. The results show that our reward based strategy outperforms the random and round robin strategies in all cases, except when reach/grasp sensitivity = 3.0 cm. This case helped us to identify two factors that should be taken into account in the analysis of the results, namely the peripheral information and the decision time. We are currently implementing a formal method for visual search instead of the current heuristic. Also, we are developing two more gaze strategies derived from our model, so that we can compare with other informed strategies, not just uninformed ones. We are also extending the system to take into account concurrent motor systems, so that the arms can interact with each other. Finally, another interesting extension would be to execute physical actions to help the perception process. For instance, if the object of interest is occluded by another object, it would be better to remove that object to get a better view.

References

1. Cassandra, A.R.: Exact and approximate algorithms for partially observable Markov decision processes. Ph.D. thesis, Brown University (1998)
2. Metta, G., et al.: The iCub humanoid robot: An open platform for research in embodied cognition. In: Proc. ACM Perf. Metrics for Int. Sys., pp. 50–56. ACM, New York (2008)
3. Pattacini, U., et al.: An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots. In: IEEE IROS, pp. 1668–1674. IEEE Press (2010)
4. Nunez-Varela, J., et al.: Where do I look now? Gaze allocation during visually guided manipulation. In: IEEE ICRA, pp. 4444–4449. IEEE Press (2012)
5. Karaoguz, C., et al.: Optimisation of gaze movements for multitasking using rewards. In: IEEE/RSJ IROS, pp. 1187–1193. IEEE Press (2011)
6. Bradtke, S., Duff, M.: Reinforcement learning methods for continuous-time Markov decision problems. Adv. in Neural Inf. Proc. Sys. 8, 393–400 (1995)
7. Sutton, R., et al.: Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. AI Journal 112(1), 181–211 (1999)
8. Schultz, W.: Multiple reward signals in the brain. Nat. Rev. Neurosci. 1(3), 199–207 (2000)
9. Johansson, R., et al.: Eye-hand coordination in object manipulation. Journal of Neuroscience 21(17), 6917–6932 (2001)
10. Land, M.: Eye movements and the control of actions in everyday life. Progress in Retinal and Eye Research 25(3), 296–324 (2006)
11. Sprague, N., et al.: Modeling embodied visual behaviors. ACM Trans. Appl. Percept. 4(2) (2007)
12. Hayhoe, M., Rothkopf, C.: Vision in the natural world. Wiley Inter. Reviews: Cognitive Science 2(2), 158–166 (2010)
13. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley-Interscience, New York (1994)
14. Sutton, R., Barto, A.: Introduction to Reinforcement Learning. MIT Press (1998)
15. Thrun, S., et al.: Probabilistic Robotics. MIT Press, Cambridge (2008)
16. Frintrop, S.: VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search. Springer, New York (2006)

Using Sensorimotor Contingencies for Terrain Discrimination and Adaptive Walking Behavior in the Quadruped Robot Puppy

Matej Hoffmann[1], Nico M. Schmidt[1], Rolf Pfeifer[1],
Andreas K. Engel[2], and Alexander Maye[2]

[1] Artificial Intelligence Laboratory, Department of Informatics, University of Zurich
Andreasstrasse 15, 8050 Zurich, Switzerland
{hoffmann,nschmidt,pfeifer}@ifi.uzh.ch
[2] University Medical Center Hamburg-Eppendorf
Dept. of Neurophysiology and Pathophysiology
Martinistr. 52, 20246 Hamburg, Germany
{a.maye,ak.engel}@uke.de

Abstract. In conventional "sense-think-act" control architectures, perception is reduced to a passive collection of sensory information, followed by a mapping onto a prestructured internal world model. For biological agents, Sensorimotor Contingency Theory (SMCT) posits that perception is not an isolated processing step, but is constituted by knowing and exercising the law-like relations between actions and resulting changes in sensory stimulation. We present a computational model of SMCT for controlling the behavior of a quadruped robot running on different terrains. Our experimental study demonstrates that: (i) Sensory-Motor Contingencies (SMC) provide better discrimination capabilities of environmental properties than conventional recognition from the sensory signals alone; (ii) discrimination is further improved by considering the action context on a longer time scale; (iii) the robot can utilize this knowledge to adapt its behavior for maximizing its stability.

Keywords: active perception, terrain recognition, object recognition, developmental robotics, adaptive behavior.

1 Introduction

In the majority of approaches to robot control the extraction and classification of features from the sensory input is a crucial processing step that has a critical effect on the behavioral performance of the artificial agent. Ever more complex methods are employed to detect type and position of objects, to recognize landmarks and obstacles, or to infer the spatial configuration of the surrounding area. In mobile robotics, for example, this problem is typically solved by employing several distal (non-contact) sensors: cameras, laser range finders, and possibly also radar. Terrain classification into traversable vs. non-traversable is done in a supervised manner through a set of labeled terrain examples [1]. This is used

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 54–64, 2012.

to update an internal representation of the world – a 2D occupancy grid that in turn is used for planning a collision-free path. Although recent studies suggest that the traditional "sense-think-act" approaches can also be extended to real-world environments, their task domain is still limited.

The inherent problem of these approaches, in our view, is that they treat perception as a separate, *passive* process that is detached from the agent's actions. A "sensory snapshot" of the environment is taken that is then mapped onto the states of an internal world model. However, we believe that perception in biological agents has a different character. First, it is active. This view can be traced back to the pragmatic philosopher John Dewey [3], and it was later picked up by research in active perception (see [4] for an overview). Second, perception occurs through the body. The information that reaches the brain is thus critically shaped by the active generation of sensory stimuli and by the agent's embodiment (this is quantified in [10], for instance). Sensorimotor Contingency Theory (SMCT)[15,14] as a representative of action-oriented approaches ascribes sensory awareness and perception to the exercise of knowledge about the lawful relations between actions and resulting changes in the sensory signals, called Sensory-Motor Contingencies (SMCs), instead of activating an internal representation of the perceived object.

We have recently developed a computational model for SMCs and demonstrated its application in an object-recognition task [11]. Here we apply the same model for controlling a robot with a completely different embodiment: a quadruped "dog" robot. We start by investigating how different gaits and terrains modulate the sensory information collected by the robot. Next we demonstrate that taking the action explicitly into account improves the terrain classification accuracy. Taking the context of longer sensorimotor sequences into account can further improve the classification performance. Finally, we show that the robot can successfully deploy its perception of the properties of different grounds to select gaits from a given repertoire to maximize its stability.

2 Related Work

The importance of sensorimotor information for object recognition in humans is evident from studies of neurological disorders [22], even though it is sometimes assigned only the role of a fall-back system [18]. In a scenario similar to ours, E.J. Gibson et al. [5] studied how infants perceive the traversability of the environment, implicitly taking into account their mode of locomotion – walking or crawling – and exploiting not only visual but also tactile information. In general, perceptual categorization in biological agents is a hard problem [7] resulting from a complex interplay of the brain, body and environment, and the individual effects are hard to separate. In this regard, robotics has provided efficient tools to test these effects independently.

First, Pfeifer and Scheier [16] have demonstrated how sensorimotor coordination can greatly simplify classification or categorization in a study where mobile robots distinguish between big and small cylinders by circling around them.

Whereas this would be very difficult from a static camera picture when the distance to the object is not known, different angular velocities resulting from circling around them render the problem much easier. Similar results emerged from studies in artificial evolution: the fittest agents were those engaging in sensory-motor coordinated behavior [2].

Second, perception can be facilitated by the morphology of the body and the sensory apparatus (see examples in [8]). In legged robots that engage in different terrains, proprioceptive sensors can be particularly useful. In a previous study in our platform, we have shown how information regarding the robot's position and orientation can be extracted [17]. A combination of proprioceptive sensors has been successfully employed in a terrain recognition task in a hexapod [6].

Third, the action that caused a sensory stimulation can be explicitly taken into account in a classification task. This has been done in [19], where sensory data resulting from different actions are clustered separately. In [20], traversability categories are predefined and the robot learns – for each action separately – a mapping from initial percepts to these categories.

Many more approaches employ some form of sensorimotor information, but to our knowledge the approach we will present here is one of the few in that actions play a constitutive role for the perception of the agent as proposed by SMCT. Our method allows for a context given by the sequence of previous actions, and it is inherently multimodal. In addition, we will test the hypothesis that longer sensorimotor sequences are needed for object categorization (i.e., the ground the robot is running on in our case). Furthermore, to demonstrate the behavioral relevance of the classification capabilities for the agent, we present a closed-loop system that employs the perception of the properties of different grounds to select gaits from a given repertoire to maximize stability.

3 Methods and Experiments

3.1 Robot and Experimental Setup

The Puppy robot (see Fig. 1 left) has four identical legs driven by position-controlled servomotors in the hips. It has passive compliant joints at the knees. We prepared five sets of position control commands for the servomotors, resulting in five distinct gaits (bound forwards, bound left/right, crawl, trot backwards), each of them with a periodic motor signal at 1 Hz. Four potentiometers measured the joint angles on the passive knee joints, and 4 pressure sensors recorded forces applied to the robot's feet. Linear accelerations (in X, Y, and Z direction) were measured by an onboard accelerometer. In total we used 11 sensory channels, jointly sampled at 50Hz.

To investigate the long-term properties of our approach, we additionally designed a model of Puppy in Webots [21], a physics-based simulator (see Fig. 1 right). For this model we used the same gait repertoire (2 gaits had to be adapted) plus 4 additional gaits (turn left/right, pace, walk), obtaining a repertoire of nine gaits. In both cases, gaits (actions) were exercised in 2-second-intervals during

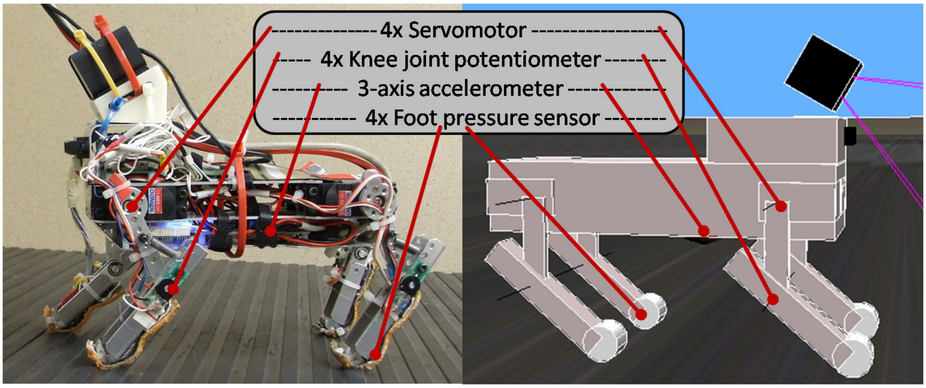


Fig. 1. Real and simulated robot and the sensor suite. The robot is 20 cm long. The camera and infrared sensors that are also mounted on the robot were not used in the experiments.

which the sensory data were collected, forming sensorimotor epochs of 2 seconds. At the end of each epoch the robot could change the gait.

For the real robot, we prepared a small wall-enclosed arena of 2x1 m. Four different ground substrates covered the ground: plastic foil, cardboard, Styrofoam and rubber. These materials differed in friction and also in structure (cardboard and rubber had ridges). In the simulator, the arena was much bigger in size (25x25 m), so encounters with the walls were much less frequent. The "foil", "cardboard", and "rubber" were flat but differed in Coulomb friction coefficients ($\mu = 2$, 11, and 20 respectively). To increase the differences between the substrates in the simulator, the "Styrofoam" ground ($\mu = 9$) was made uneven with randomly placed smooth bumps of up to 3 cm height.

3.2 Feature Computation

For effective processing of sensorimotor information we compressed the raw data by extracting some simple features. For the action space we chose a high abstraction level and used the gait as a single feature. In the sensory space, following a similar strategy as we used in [17], we took advantage of the periodic nature of the locomotion and created period-based features as follows: (1) sum of knee amplitudes of all legs in a period,[1] (2) sum of standard deviations of all knee joints, (3) sum of mean pressures in each foot, (4) sum of standard deviations of each foot pressure signal, (5-7) mean accelerations along X,Y, and Z-axis respectively, (8-10) standard deviations of the accelerometer signals. Since frequent gait transitions disrupt the locomotion and impact also the sensory values, only the last second (i.e. the second locomotion period) from each 2s epoch was used for the feature computation. Continuous feature values were used for classification (Section 4.1); for learning the sensorimotor contingencies and optimizing

[1] Note that the knees are passive compliant.

the behavior using a Markov model (Section 4.2), each feature was quantized to two levels only.

3.3 A Markov Model of SMCs

We employed the model that we presented in [11,12] with the necessary adaptations to the Puppy robot. The basic idea is to consider actions and resulting changes in sensory signals in an integrated manner, and to keep a record of sequences of actions and sensory observations. For each epoch, the action a (the gait in this case) and a vector of n sensory features observed during execution of a are concatenated to a single vector $ao(t) = [as_1 s_2 \dots s_n]$ that we call an action-observation pair. Based on the sequence of action-observation tuples that the robot experiences over time $c^h = [ao(t), ao(t-1), \dots ao(t-h)]$, the model samples the conditional probability distributions $P^h(ao(t+1)|c^h(t))$, i.e. the probability of experiencing a particular action-observation pair in the next time step given a finite history h of previous pairs. In this study we use $h = 0 \dots 4$. This probability distribution is what we call the extended Sensori-Motor Contingencies (eSMC) of an agent, and a particular combination of $ao(t+1)$ and $c^h(t)$ is a specific sample that in addition to its probability of occurrence can have other properties like a value.

3.4 Value System and Action Selection

We extended the basic idea of SMCT by a value system and an action selection algorithm. For each epoch t, we define the value[2] of the robot's state by a weighted sum of three components:

$$v(t) = -tumbled - 0.4regularity - 0.1speed$$

We used the signal of the accelerometer in Z direction to determine if the robot is upright ($tumbled = 0$) or has tipped over ($tumbled = 1$). The similarity of the sensory patterns at the knee joints between the first and second period during an epoch is reflected in the *regularity* value (1 for identical patterns during both periods), and the normalized velocity computed from the robot's global coordinates yields the *speed* value.

We have devised a stochastic action selection algorithm that attempts to optimize the temporal average of the internal value. It selects actions that have shown to activate eSMCs with high internal values, and explores the consequences of new actions when no or only bad prior experiences exist in a given situation. For each action-observation sequence $c^h(t)$ a record of actions executed next $a_{next}(c^h(t))$ and the average value $v(a_{next}(c^h(t)) = \sum_n v(t+1)/n$ is kept, where n is the number that action a_{next} was executed when context $c^h(t)$ was encountered, and $v(t+1)$ is the resulting value. Different history lengths h

[2] In reinforcement learning terms, this would be called reward - it is the immediate reward signal associated with each state.

may yield different value information. Since we consider longer matches between a particular action-observation sequence and the stored eSMCs as a more accurate estimation of the state, preference is given to the value information from longer matching histories. When the robot later experiences the same context again, it knows the average value of the actions it has tried before. Random values get assigned to the other actions. To avoid a predominantly random exploration in the initial learning phase when the robot has only little sensorimotor knowledge, the expected value for the most recently executed action is given by the internal value of the last epoch. This favors the continuation of successful actions, and switching to another action otherwise. The action with the highest expected value $\hat{a} = \arg\max_{a} v(a_{next}(c^h(t))$ is then executed with a probability $p(\hat{a}) = v(\hat{a}) + 1$.

4 Results

4.1 Perception and Discrimination of Different Grounds

In this section, we want to quantitatively assess the effect of considering actions and the resulting changes in sensory stimulation in an integrated manner. First, we compare the respective influence of the action (the gait the robot is running with) and the environment on the sensory data. Second, focusing on the ground discrimination, we demonstrate how explicitly incorporating the action that has induced a sensory stimulation improves the environment classification. Finally, we study the effect of longer sensorimotor sequences, testing our hypothesis that these are required for object categorization, whereby, from the robot's perspective, different grounds correspond to different objects in our scenario.

We have collected data from the real (4 x 20 minutes, i.e. 4 x 600 epochs) and simulated version of the robot (4 x 4 hours, i.e. 4 x 7200 epochs) running separately on the different substrates. After every epoch a new action was chosen at random. If the robot tumbled, it was manually (real robot) or automatically (simulator) returned to an upright position at the same location and two epochs following this event were discarded. A reflex for backing up from the walls was built in. Epoches when the robot was backing up (frequent in the real robot) were not discarded but entered the regular learning process. A naïve Bayes classifier (diagonal covariance matrix estimate, stratified 10-fold cross-validation) was trained to classify either the action or the ground substrate given the sensory observations and actions during the previous epochs.

Ground and Gait Discrimination from Sensory Data Only. To assess the dependencies of the sensory signals from the gait or ground, respectively, we collapsed the data across gaits (for assessing ground effects) or across grounds (for assessing gait effects). In the real Puppy, the classifier determined the correct gait from the sensory data in 72.4% of the cases, and in 81.6% in the simulation. In contrast, the ground recognition rates were lower, 67.2% for the real Puppy and 43.1% in the simulation (see also Fig. 2, top-most bars). This shows that gaits and grounds have a similarly strong effect on the sensory patterns in the real robot. In the simulation the different materials induce similar sensory patterns and hence, are difficult to distinguish. These figures serve as a baseline when we consider the classification of joint action and sensor information next.

Ground Discrimination Using Action Information. We separated the data into sets for each gait and classified the grounds on each set individually. Afterwards we averaged the ground recognition rate over all gaits. In comparison to the ground recognition using a single classifier, the action-dependent classification schema reaches an improved accuracy of 75.7% for the real robot. Considering only the gait yielding the best recognition rate, this value increases to 80.2%. In the simulation this increase is even more pronounced, from 43.1% to 62.9% and 78.3%, respectively (see Fig. 2, second bars from top). This indicates that taking the action that caused a sensory observation into account is more specific for the environmental condition than analyzing the sensory data alone.

Ground Discrimination Using Action Sequences. The sensorimotor patterns induced by a single action may often be similar even if the agent interacts with different objects. As suggested by SMCT, longer sequences of interaction with an object may be needed in order for the object to leave a unique "footprint". We confirmed this hypothesis by splitting the data further into sets for specific sequences of 2 or 3 consecutive actions, and averaging again over all sequences. The sensory feature vectors from consecutive epochs were concatenated. For a sequence of two gaits, the ground classification accuracy rises to 84.7% in the real robot, and to 70.6% in the simulation. Considering a sequence of 3 gaits further improves accuracy (see Fig. 2). Here, the gait sequence-specific classifier with the highest accuracy achieves a 100% recognition rate. This means that the sensorimotor patterns of this action sequence are apt for a reliable recognition of the different grounds.

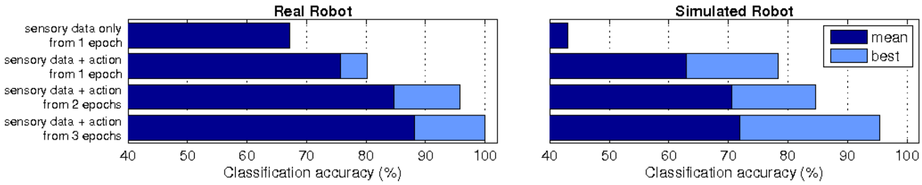


Fig. 2. Comparison of the ground classification accuracies when the action context is taken into account to different degrees. (left) Real robot. (right) Simulated robot.

4.2 Selecting Gaits to Optimize Behavior

Next we want to demonstrate how the better discrimination capabilities that result when a longer action context is considered can be used by the robot to improve its behavior. We let the simulated robot run on 4 different grounds,

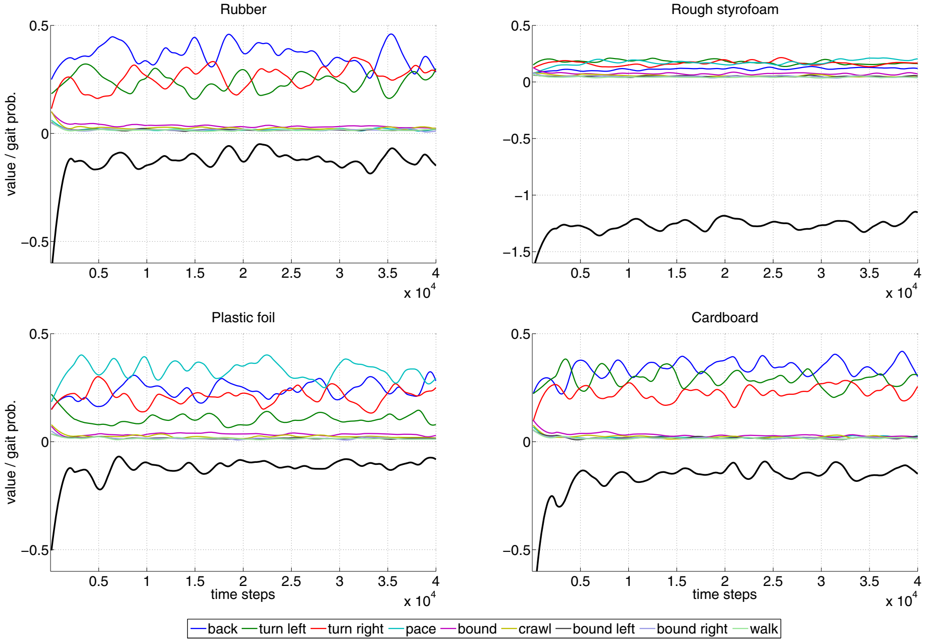


Fig. 3. Value (black curve under the abscissa) and gait selection frequencies (above) over time on 4 ground substrates (data from simulator). All curves have been smoothed with a weighted linear least squares and a 2nd degree polynomial model in a moving window of 5.000 samples. To appreciate the time course of the value function, the initially low values have been clipped. Note the different scale of the value function for rough styrofoam.

and used the Markov model (Sec. 3.3) to learn eSMCs for the 9 gaits from its repertoire. Each eSMC had an associated value given by the value function described in section 3.4.

With progressing sensorimotor knowledge, the robot preferred to choose gaits that improved its internal value, providing swift, smooth and stable locomotion. The plots of the value function in Fig. 3 show that a basic set of gaits that "feel good" to Puppy (i.e. maximize the value function) is found after only about 1.000 epochs (around 8 minutes). On cardboard it takes more than 2.000 epochs to arrive at a reasonable gait combination. Afterwards the robot tries to further improve its behavior by selecting from these comfortable gaits with different probabilities. As one would expect, the optimal gait sequence depends on the material properties of the grounds. Except for the plastic foil, Puppy prefers a mixture of walking back and turning left or right. It is most successful in epochs when it reduces the frequency of turns in favor of walking back. On plastic foil, the most successful gait is pacing, while turning left seems to be a less favorable gait. On cardboard, turning left is selected more frequently than turning right, though, while on rubber both turning actions are chosen with about the same frequency.

On the rough styrofoam, the value function is dominated by frequent tipping of the robot. Compared to the three flat grounds the value remains at a low level, and the separation into favorable and unpleasant gaits is less pronounced. The order of preference seems to be maintained, though.

The improvement of the internal value is not monotonic, but proceeds in a rather oscillatory manner. Intervals in which the robot had sufficient sensorimotor knowledge to optimize its behavior alternated with epochs in which it learned new eSMCs. With the sensorimotor knowledge growing, episodes with optimal behavior become more frequent and last longer. On cardboard, for example, behaviors that maximize the value function are found after about $2 \cdot 10^4$ epochs, and the increasing width of the peaks in the value function indicate that the robot spends more and more time in these optimal behaviors. A similar observation can be made on plastic foil. On rubber, the knowledge about favorable behavior around $2 \cdot 10^4$ seems to be lost afterwards, but it can be expected that the exploration process leads to a further improvement beyond the analyzed interval. Since the value function was designed to never reach zero, corresponding to a state of perfect harmony, the robot keeps on exploring the potential to further improve its fitness.

5 Conclusion and Future Work

In this study we have investigated sensorimotor classification of different substrates in a quadruped robot from the perspective of SMCT. First, we have demonstrated how sensory stimulation patterns critically depend on the actions the robot is exercising. If the robot wants to recognize the object or environment it is interacting with, like the terrain type in our case, the action (gait) that gives rise to the experienced sensory stimulation needs to be considered. In addition we have shown that deployment of longer action contexts further improves the discrimination capabilities. Our approach demonstrates that the robot successfully engages the acquired sensorimotor knowledge to optimize its behavior by selecting appropriate gaits on different ground substrates.

Apart from serving as a model of SMCT, our work has also substantial application potential. Autonomous, perception-based, off-road navigation is a hot research topic in mobile robotics (e.g., [9]). Unlike traditional approaches that rely on passive long-distance perception using high resolution sensors, we have hinted at the potential of a radically different approach: terrain perception through active generation of sensory stimulation in a multimodal collection of low-resolution sensors (for learning eSMCs, 1 bit per sensory channel was used). Taking action-observation sequences into account and exploiting the robot's rich body dynamics to simplify the structure of the sensory information, an advantageous transformation of the input space for classification can be achieved.

In the current study we have employed only proprioceptive and contact sensors. These have proven very effective in ground discrimination and, in conjunction with a simple one-step prediction of the best next action based on the current sensorimotor context, the robot could optimize its behavior. However, these sensors provide little information about the terrain beyond the robot's current location. Distal sensors (like infrared or vision), on the other hand, could provide information about future events that could likewise be exploited for perceptual categorization and further improvement of the behavior. A promising approach in this respect uses internal simulation in sensorimotor space to find action sequences that optimize the success of the agent with a longer temporal horizon [13,19]. Alternatively, reinforcement learning algorithms could be employed. Traversability in general may be a suitable touchstone to compare different approaches to use sensorimotor information for controlling robots. This will be the direction of our future work.

Acknowledgement. This work was supported by the Sino-German research training group CINACS, DFG GRK 1247, `www.cinacs.org`, and by the EU project 'Extending sensorimotor contingencies to cognition - eSMCs', FP7-ICT-270212, `esmcs.eu`.

References

1. Bagnell, J.A., Bradley, D., Silver, D., Sofman, B., Stentz, A.: Learning for autonomous navigation. IEEE Robotics & Automation Magazine 17(2), 74–84 (2010)
2. Beer, R.D.: The dynamics of active categorical perception in an evolved model agent. Adaptive Behavior 11, 209–243 (2003)
3. Dewey, J.: The reflex arc concept in psychology. Psychological Review 3, 357–370 (1896)
4. Engel, A.K.: Directive minds: how dynamics shapes cognition. In: Stewart, J., Gapenne, O., Di Paolo, E.A. (eds.) Enaction: Towards a New Paradigm for Cognitive Science, pp. 219–243. MIT Press, Cambridge (2011)
5. Gibson, E.J., Riccio, G., Schmuckler, M.A., Stoffregen, T.A., Rosenberg, D., Taromina, J.: Detection of traversability of surfaces by crawling and walking infants. Journal of Experimental Psychology 13(4), 533–544 (1987)
6. Giguere, P., Dudek, G.: Clustering sensor data for autonomous terrain identification using time-dependency. Autonomous Robots 26, 171–186 (2009)
7. Harnad, S.: Cognition is categorization. In: Handbook of Categorization in Cognitive Science. Elsevier (2005)
8. Hoffmann, M., Pfeifer, R.: The implications of embodiment for behavior and cognition: animal and robotic case studies. In: The Implications of Embodiment: Cognition and Communication, pp. 31–58. Imprint Academic (2011)
9. Jackel, L.D., Krotkov, E., Perschbacher, M., Pippine, J., Sullivan, C.: The DARPA LAGR program: goals, challenges, methodology, and phase I results. Journal of Field Robotics 23(11/12), 945–973 (2006)
10. Lungarella, M., Sporns, O.: Mapping information flow in sensorimotor networks. PLoS Computational Biology 2 e144(10), 1301–1312 (2006)
11. Maye, A., Engel, A.K.: A discrete computational model of sensorimotor contingencies for object perception and control of behavior. In: 2011 IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 3810–3815. IEEE (May 2011)
12. Maye, A., Engel, A.K.: Time Scales of Sensorimotor Contingencies. In: Zhang, H., Hussain, A., Liu, D., Wang, Z. (eds.) BICS 2012. LNCS (LNAI), vol. 7366, pp. 240–249. Springer, Heidelberg (2012)

13. Möller, R., Schenck, W.: Bootstrapping cognition from behavior – a computerized thought experiment. Cognitive Science 32(3), 504–542 (2008)
14. Noë, A.: Action in perception. MIT Press (2004)
15. O'Regan, J.K., Noë, A.: A sensorimotor account of vision and visual consciousness. Behavioral and Brain Sciences 24, 939–1031 (2001)
16. Pfeifer, R., Scheier, C.: Sensory-motor coordination: The metaphor and beyond. Robotics and Autonomous Systems 20, 157–178 (1997)
17. Reinstein, M., Hoffmann, M.: Dead reckoning in a dynamic quadruped robot: Inertial navigation system aided by a legged odometer. In: 2011 IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 617–624 (2011)
18. Sirigu, A., Duhamel, J.R., Poncet, M.: The role of sensorimotor experience in object recognition. A case of multimodal agnosia. Brain: A Journal of Neurology 114(pt. 6), 2555–2573 (1991)
19. Ugur, E., Oztop, E., Sahin, E.: Goal emulation and planning in perceptual space using learned affordances. Robotics and Autonomous Systems 59, 580–595 (2011)
20. Ugur, E., Sahin, E.: Traversability: a case study for learning and perceiving affordances in robots. Adaptive Behavior 18, 258–284 (2010)
21. Webots. Commercial Mobile Robot Simulation Software, http://www.cyberbotics.com
22. Wolk, D.A., Coslett, H.B., Glosser, G.: The role of sensory-motor information in object recognition: Evidence from category-specific visual agnosia. Brain and Language 94(2), 131–146 (2005)

How Walking Influences the Development of Absolute Distance Perception

Beata J. Grzyb[1], Angel P. del Pobil[1,2], and Linda B. Smith[3]

[1] Robotic Intelligence Lab., Jaume I University, 12071 Castellon, Spain
[2] Interaction Science Dept., Sungkyunkwan University, Seoul, Korea
[3] Cognitive Development Lab., Indiana University, Bloomington, IN, USA
{grzyb,pobil}@icc.uji.es, smith4@indiana.edu

Abstract. The emergence of upright locomotion in infants has been shown to influence and dramatically reorganize a myriad of behaviors. The change, however, does not always lead to improvements, but can also cause temporal instability and lost of integrity in many seemingly unrelated systems. Our empirical study showed that the onset of walking is also related to a disruption in infants' perceived reachability. This paper investigates the reorganization of the processes responsible for integration of different visual depth cues at the onset of walking and how such recalibration influences reaching behavior. A reward-mediated learning is employed to mimic the development of absolute distance perception in infants over a short developmental timescale.

Keywords: development of distance perception, locomotion and cognitive development, depth cue integration, infant reaching.

1 Introduction

When infants begin to self-locomote, they experience an extraordinary psychological reorganization. The onset of walking triggers staggering changes in perception, spatial cognition, and social and emotional development. It has been shown that independently walking infants have greater memory flexibility [6], a better understanding of object permanence [2], and enhanced emotional expressions [3]. Independent walkers spend significantly more time interacting with toys and with their caregivers, and also make more vocalizations and more directed gestures compared to non-walkers [7]. Although the onset of walking leads to rapid improvements in many aspects, it also contributes to momentary instability and lost of integrity in many seemingly unrelated systems. Learning to walk affects infants' reaching behavior, as some infants return to two-handed reaching behavior [8], and also affects infants' sitting posture by increasing the magnitude of distance-related sway properties [5].

Our empirical studies showed that 12-month-old infants reach more often than 9-month-old infants for distant – unreachable – objects [9], and that this transient disruption in perceived reachability is related to infants' walking ability [10]. We suggest that the processes responsible for integration of different

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 65–74, 2012.

visual depth cues reorganize themselves at the onset of walking so as to incorporate information from self-motion-based depth cues.

Reward-mediated learning has been shown to successfully mimic the development of near-optimal integration of visual and auditory cues in children [12,16]. Following this approach, we simulate the developmental process of distance perception for action in older infants. Our results show an increase in near-far space confusions after the recalibration of distance information in accordance with new motoric factors, providing further support for our hypothesis that recalibration of distance perception can cause distance errors seen in infants during the transition period to walking.

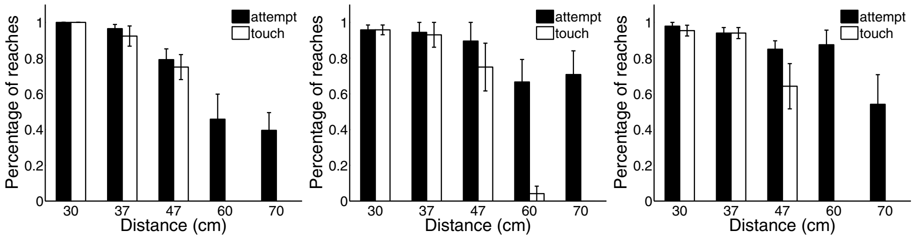
The paper is structured as follows. The next section provides a brief overview of our empirical studies with infants, which is followed by the discussion of the influence of the onset of locomotion on infants' distance perception. Section 4 provides the details of our model and results of computational simulation. We conclude the paper with the discussion of results and future work.

2 Observation Data

The main objective of our experiments was to see if, and if so, how infants' knowledge about their own body capabilities changes over a relatively short developmental timescale. Reaching provides a good measure of infants' body awareness, since to sucessfully reach for an object infants need to know not only the distance to the object, but also how far they can reach and lean forward without losing balance. In total, 8 9-month-old and 8 12-month-old infants participated in our study.

The procedure of the experiment was as follows. The infant was seated in a specially adapted car seat with the seatbelts fastened for security reasons. In order to keep her engaged and attentive during the entire experimental session, a colorful stimuli display was placed in front of her. The experimenter was sitting behind the stimuli display and presented a ball on a wooden dowel through the opening of the frame. The sequence of trials consisted of 9 distances (from 30 cm up to 70 cm) and begun and ended always with trials at close distances to keep the infant motivated. The order of distances, apart from the first two and the last two trials in the sequence was chosen pseudo-randomly. The sequence of distances was repeated up to three times. There was no explicit reward provided to the infant after the trial for any tested distance. This helped us to avoid situations where the infant could learn to make reaching movements just to communicate her interest in obtaining a reward. The entire experimental session was recorded with two cameras. These recordings were subsequentially viewed and infants' behavior scored.

The results of our first study revealed that 12-month-old infants, but not 9-month-old, continually reached for the unreachable objects. That was quite surprising as typically we would expect older infants to posses a better knowledge than younger ones. Since 12 months is the age around when the transition to walking occurs, we extended our experiment and recruited more infants with



(a) Non-walkers (b) Walkers with help (c) Independent walkers

Fig. 1. Mean percentage of reaches for 12-month-old infants

regard to their walking abilities. Our final sample was composed of 24 infants categorized into 3 equal-number groups, that is non-walkers, walkers with help, and independent walkers. Fig. 1 shows the mean percentage of reaches to objects placed at various distances. As is clearly seen, walkers (with and without help) reached more for distant distances than non-walkers. It is worth noting here that only one child was able to touch the ball at 60 cm distance, and none of these babies was able to make contact at 70 cm.

3 The Onset of Locomotion and Distance Perception

Infants are sensitive to a number of sources of depth information very early in life [13]. Most of the typical accounts, however, focus on information that specify only relative depth relations. In order to perceive distance veridically, however, infants need to perceive absolute distance information [4]. For example, infants need to know exactly how far to stretch an arm to reach for an object. Therefore, for absolute distance perception, various sources of relative depth information must be calibrated by one or another type of metric information. Motoric factors, and particularily locomotion experience, may be one type of such metric information.

The prelocomotor infants' visual system can detect the information for veridical distance perception, but only within a certain range [4]. At this stage of development, infants mainly use static depth cues to direct their gaze and reaching movements to nearby stimuli. Herein, as static depth cues we assume any depth cue that does not require any head or body motion. Although the effectiveness of these cues is limited to near space, such a solution seems to be optimal as infants' attention and exploration is also confined to the reachable space. Once infants are able to walk, other depth-specifying cues have to be used in order to correctly estimate distance to far objects. Locomotion, and especially walking, helps infants to calibrate distance information by drawing attention to previously unattended depth-specifying information. In other words, immobile infants see the world as "reachers', their attention and exploration is constrained to near space, and their actions are driven mainly by static depth cues. While being able to locomote they see the world as “walkers’, their attention and exploration is shifted from near to far space. The processes that integrate different depth cues begin to reorganize to incorporate also depth information from self-motion-based cues, such as motion parallax or motion perspective, and to scale the distance according to new motoric factors. We propose that such a change in distance perception provokes the momentary disruption in perceived reachability in infants.

Attention to environment was suggested to be one of the critical mechanisms by which developmental changes occur [4]. The ability to self-locomote lures attention to far space, especially to the location toward which the infant is moving. Such a reallocation of attention from near to far space modifies the use of various sources of depth information, which are related to accurate distance perception. It also leads to a change in the use of monocular static information specifying depth relations. Improved distance perception gradually leads to marked improvements in size and shape constancy at relatively large distances from the infant.

Similarly to attention, coupling between visual and vestibular information may also be considered an important parameter underlying the development of absolute distance perception. For successfully walking to a target, vestibular and peripheral optic flow information must be integrated with elaborate co-ordinated leg movements, together with mechanisms for analysing depth and distance in a central field, whereas for reaching and grasping the peripheral optic flow and vestibular information can be largely ignored while using depth and distance information in nearby space [1]. Vestibular information plays an important role in the utilization of motion parallax. As visual-vestibular coupling improves following the onset of locomotion, infants attend more to motion parallax. Thus, locomotor infants can become more aware than prelocomotors of the discrepancy between the depth relations specified by monocular static information and those specified by motion parallax. This explains a reduction in the tendency to reach for an apparently nearer of two objects on the basis of monocular static information in self-locomoting infants [4].

4 Methods

4.1 Task Setup

In our task each trial consists of the presentation of visual stimuli at various distances. Static depth estimation methods, i.e. stereopsis and familiar size, and self-motion-based depth cues, i.e. motion parallax, are used to calculate the distance. We assume that static depth estimation functions give accurate responses within reachable space, and their effectiveness worsen for distances outside of this range. Additionally, we assume that motion parallax always gives accurate distance estimation. Subsequently, a white noise with the standard error deviation $\sigma = 2$ and $\sigma = 3$ is added for the static depth cues function and for the self-motion-based cue function, respectively.

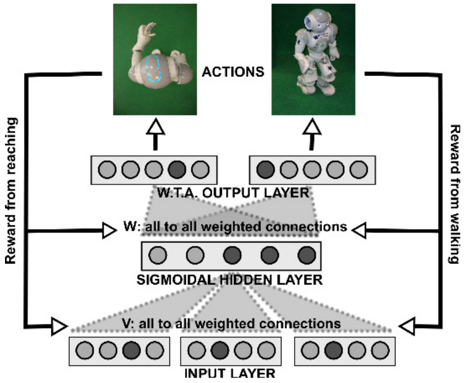


Fig. 2. General scheme of the reward-based learning model. Note that this paper discusses only the simulation of the model with no actual feedback from the environment. The model, however, is designed to be implemented on a NAO humanoid robot.

4.2 Reward-Mediated Learning

A three-layer neural network (see Fig. 2) is used to approximate the state-action mapping function. The input layer consists of $i = c * n$ neurons that encode the estimates of c different depth cues covering the n discretized distance units. Each input neuron has a Gaussian receptive field, centered on position $z_{c,n}$. The variance of these Gaussian receptive fields is in the order of the noise of the input stimuli. The input neurons x_i are all-to-all connected with weights $v_{i,j}$ to j neurons in the hidden layer.

A sigmoidal transfer function on the sum of the weighted inputs gives the outputs y_j of the hidden neurons:

$$y_j = \frac{1}{1 + e^{-\sum\limits_i v_{i,j} x_i}} \tag{1}$$

The hidden neurons are fully connected to output neurons k with weights $w_{j,k}$. All weights are drawn from uniform distributions, $v_{i,j}$ between -0.1 and 0.1, and $w_{j,k}$ between -1 and 1.

Each output unit represents an action. Two types of actions, that is reaching k_r and walking k_w, are possible. The action space is discretized with the *binning size* equal to 1 cm for reaching and 3 cm for walking action. The activation of the output neurons z_k is given by the weighted sum of the hidden layer activity, representing an approximation of the appropriate Q-value.

Based on the network's outputs, one action is chosen according to the *softmax* action selection rule [14]:

$$P_t(k) = \frac{e^{Q_t(k)/\tau}}{\sum_{b=1}^{n} e^{Q_t(b)/\tau}} \tag{2}$$

where $P_t(k)$ is the probability of selecting an action k, $Q_t(k)$ is a value function for an action k, and τ is a positive parameter called *temperature* that controls the stochasticity of a decision. A high value of τ allows for more explorative behavior, whereas a low value of τ favors more exploitative behavior.

After performing the selected action $\hat{k}$ the true reward $r(\hat{k})$ is provided. The reward is maximal when $\hat{k}$ equals the true object position k_t, decaying quadratically with increasing distance within a surrounding area with radius ρ (in our case $\rho = 4$) and zero otherwise.

$$r(\hat{k}|X) = max(0, (\rho - |\hat{k} - k_t|))^2 \tag{3}$$

To minimize the error between the actual and expected reward, we make use of the gradient descent method, which is widely used for function approximation, and is particularly well suited for reinforcement learning [14].

$$v_{i,j}(t+1) = v_{i,j}(t) - \epsilon(r_{\hat{k}} - z_{\hat{k}})(-w_{j,\hat{k}})y_j(1-y_j)x_i \tag{4}$$

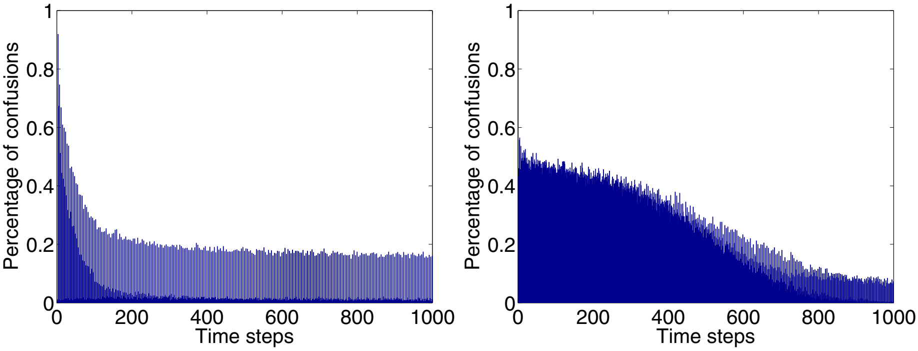
$$w_{j,\hat{k}}(t+1) = w_{j,\hat{k}}(t) - \epsilon(r_{\hat{k}} - z_{\hat{k}})(-y_j) \tag{5}$$

It is worth noting, that in the case of the update of weights $w_{j,k}$ only the output weights connected to the winning output unit $\hat{k}$ are updated. In our experiments, the learning rate ϵ, decreases exponentially, according to the formula $\epsilon(t) = \frac{\epsilon_0}{ceil(\frac{t}{v_\epsilon})}$, where $\epsilon_0 = 0.05$, and $v_\epsilon = 100000$.

4.3 Learning Sequence

As our goal is to mimic a developmental path for the integration of static and self-motion-based depth cues over a short developmental timescale, initially the neural network is trained only with static depth cues (i.e. stereopsis and familiar size). Additionally, the input space is constrained to the reachable distances, the infant's attention is fixed mainly on a near space. The action space is limited to reaching actions, that is only selection of reaching action k_r is possible. We start with a high temperature parameter $\tau = \tau_0$, so that the selection of action is only weakly influenced by the initial reward expectations. In this period, τ decreases exponentially with time $\tau(t) = \tau_0^{(\frac{v_\tau - t}{v_\tau})}$, where $\tau_0 = 10$ and $v_\tau = 50000$. The network is trained during 10000 time steps.

The onset of walking is simulated by providing the network with additional depth cues (i.e. motion parralax) and enabling the selection of walking action k_w. We test here two different scenarios. In the first one, we reset the temperature parameter τ, so that the selection of action is only weakly influenced by the initial reward expectations. In the other scenario, we left the parameter τ unchanged after learning the reaching space, so the selection of action was strongly influenced by the initial reward expectations. We compare the results obtained in these different scenarios in the next section.



(a) Low exploration, low τ

(b) High exploration, high τ

Fig. 3. Fraction of near-far confusions (mean over 10 time steps and 100 trained networks with different initializations)

5 Results

In order to compare the results of our simulation with the results obtained from our empirical study with infants, we calculated a number of near-far confusions during the learning process of far space. Herein, we defined the number of near-far confusions as a number of incorrect categorizations of far distance as near and vice versa. Fig. 3 presents mean values of near-far confusions over 100 trials with different network initializations (randomized weights) after the onset of walking for different testing scenarios (i.e. high and low level of exploration). As is noticeable, the number of near-far confusions at the begining is high in both cases. However, in the case of high value of temperature τ, the period for near-far confusions is prolonged. Although the network with lower exploration gives less near-far confusions, and learns far space representation much faster, its estimates are not as accurate as in the case of the network with higher exploration. The mean errors over 10 trials and over 100 trained networks are shown in Fig. 4. It is worth mentioning that the accuracy of the integration of various depth cues is affected by the binning size, which in our case was 1 cm for reaching and 3 cm for walking action.

Fig. 5 presents responses of neurons in the hidden layer of the network after learning of near and far space, respectively. Initially, the action space was limited and only half of the output neurons was trained. However, no explicit constraints were put on the hidden neurons. As it can be seen in Fig. 5a only approximatelly half of neurons in the hidden layer are used for encoding the input space. This behavior emerges during the training process. Comparing the activities of these neurons before and after learning of far space (the left side of the figures), it can be noticeable, that the activity does not change much during the learning process. Moreover, these neurons seem to be ordered in the order of increasing distance. That suggests that near-far confusions result from the reorganization

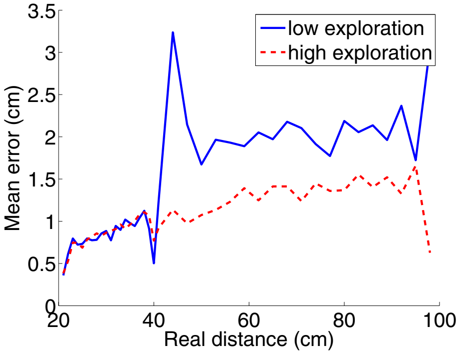
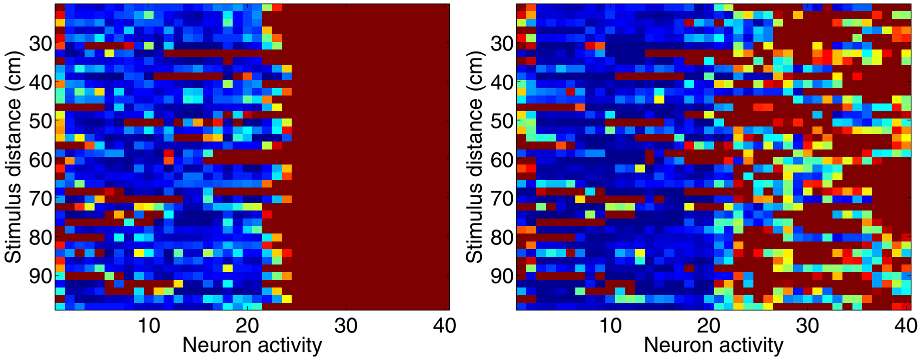


Fig. 4. Mean error of the networks in two different scenarios (mean over 10 trials and 100 trained networks with different initializations)



(a) After learning of near space (b) After learning of far space

Fig. 5. Responses of the neurons in the hidden layer of the network

of the network connections from the hidden to the output layer, which in turn may be thought of as the re-calibration of distance according to the motoric factors related to reaching and walking actions.

6 Discussion and Future Work

Our empirical findings that new walkers reach for unreachable objects provides further evidence that development is not necessarily continuous, and that once established skills are not invulnerable to changes. Our study shows that perception and cognition may not always guide action appropriately. The change in the perception of distance was suggested to contribute to the occurence of distance errors in infants. More specifically, on one hand a shift of attention from near to far space causes a reorganization of various visual depth-specifying cues, and on the other, walking promotes re-calibration of distance perception according to new motoric metrics. This hypothesis was tested with the reward-mediated learning model. The results of simulations showed an increase in the near-far confusions during the learning of far space representation, similarily as in the case of recent walkers. Thus, the perception of space, not only far space, but also near space, changes with the onset of locomotion. The newly developing representation of far space is being combined with the representation of near space to constitute a coherent space representation. This new space representation, however, arises out of high behavioral instability – a notion that is compatible with dynamical systems theory [15].

The results of simulations raise another important issue that is the role of commiting the errors during the process of learning. The network in the first scenario, where the level of exploration was kept low, exhibited a very short period of near-far confusions and was able to learn much faster the far space representation. The estimates of distance given by the network, however, were quite variable and less precise compared to the results given by the network in the second scenario, where the level of exploration was high at the begining of learning. Although such a network showed an extended period of near-far confusions, and learned much slower, it gave much better overall results. Similar relation between exploration and learning can be seen in children learning prospective control of walking. While walking on different grounds, children usually fall many times. Despite many failures, however, they intend to repeat the same action. Such repetitions may be helpful in learning of a causal relationship between falling and different predictive cues [11]. Similarly, in our case many different repetitions (and errors) are needed to efficiently integrate various visual depth-specifying cues and to calibrate space in reference to new motor metrics.

As in this model we used idealized input data, the most straightforward future extension of the proposed work is to implement the model on a real humanoid robot. Herein, our goal is twofold. The first goal is to verify our hypothesis with real visual data, both static- and self-motion-based depth cues. On the other hand, such a model for depth cue integration will provide a robot with a a robust and accurate distance perception for both reaching and walking actions. The final goal of our work is to provide the robot with a coherent near and far space representation which should be built in a dynamical way, through the active interaction with the environment in a similar way as infants do.

7 Conclusions

This paper discussed the phenomenon of distance errors seen in infants during the transition period to walking. We proposed that these errors result from the changes in infants' distance perception. More specifically, the processes responsible for integration of various visual depth cues reorganize so as to incorporate previously unattended depth-specifying cues and to re-calibrate distance with the reference to new motoric factors. The results of our simulations showed an increase in near-far confusions providing further support for our hypothesis.

Acknowledgments. This research was partly supported by WCU (World Class University) program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (Grant No. R31-2008-000-10062-0), by Ministerio de Ciencia e Innovación (FPU grant AP2007-02565, DPI2011-27846), by Generalitat Valenciana (PROMETEO/2009/052) and by Fundació Caixa Castell-Bancaixa (P1-1B2011-54).

References

1. Atkinson, J.: The developing visual brain. Oxford University Press Inc., New York (2000)
2. Bertenthal, B.I., Campos, J.J.: A systems approach to the organizing effects of self-produced locomotion during infancy. In: Advances in Infancy Research, pp. 1–60. Ablex, Norwood (1990)
3. Biringen, Z., Emde, R.N., Campos, J.J., Appelbaum, M.: Development of autonomy: role of walking onset and its timing. Perceptual and Motor Skills 106, 395–414 (2008)
4. Campos, J.J., Anderson, D.I., Barbu-Roth, M.A., Hubbard, E.M., Hertenstein, M.J., Witherington, D.: Travel broadens the mind. Infancy 1, 149–219 (2000)
5. Chen, L., Metcalfe, J.S., Jeka, J.J., Clark, J.E.: Two steps forward and one back: Learning to walk affects infants' sitting posture. Infant Behavior and Development 30, 16–25 (2007)
6. Clearfield, M.W.: The role of crawling and walking experience in infant spatial memory. J. Experimental Child Psychology 89, 214–241 (2004)
7. Clearfield, M.W.: Learning to walk changes infants' social interactions. Infant Behavior and Development 34(1), 15–25 (2011)
8. Corbetta, D., Bojczyk, K.E.: Infants return to two-handed reaching when they are learning to walk. Journal of Motor Behavior 34(1), 83–95 (2002)
9. Grzyb, B.J., Smith, L.B., del Pobil, A.P.: Reaching for the unreachable: Developmental decline between 9 and 12 months in the link between reaching behavior and the reachability of the target (submitted)
10. Grzyb, B.J., Smith, L.B., del Pobil, A.P.: Reaching for the unreachable: reorganization of reaching with walking (submitted)
11. Joh, A.S., Adolph, K.E.: Learning from falling. Child Development 77(1), 89–102 (2006)
12. Karaoguz, C., Weisswange, T.H., Rodemann, T., Wrede, B., Rothkopf, C.A.: Reward-based learning of optimal cue integration in audio and visual depth estimation. In: The 15th International Conference on Advanced Robotics, Tallinn, Estonia (2011)
13. Kellman, P.J., Arterberry, M.E.: The cradle of knowledge: Development of perception in infancy. MIT Press, Cambridge (1998)
14. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
15. Thelen, E., Smith, L.B.: A Dynamic Systems Approach to the Development of Cognition and Action. MIT Press (1994)
16. Weisswange, T.H., Rothkopf, C.A., Rodemann, T., Triesch, J.: Bayesian cue integration as a developmental outcome of reward mediated learning. PLoS ONE 6(7), 1–11 (2011)

Modelling Interaction in Multi-modal Affordance Processing with Neural Dynamics

Boris Durán and Serge Thill

Interaction Lab., University of Skövde,
P.O. Box 408, SE-541 28, Skövde, Sweden
{boris.duran,serge.thill}@his.se
http://www.his.se/coin

Abstract. Behavioral studies on the activation of affordances by understanding observation and action sentences on graspable objects show a direct relationship between the canonical orientation of graspable objects, their dimension and the kind of grip required by those objects to be grasped. The present work introduces the concepts of Dynamic Field Theory for modeling the results observed in the behavioral studies previously mentioned. The model was not only able to replicate qualitatively similar results regarding reaction times, but also the identification of same versus different object and the distinction between observable versus action sentences. The model shows the potential of dynamic field theory for the design and implementation of brain inspired cognitive systems.

Keywords: Neural dynamics, Dynamic Field Theory, affordances, modelling.

1 Introduction

1.1 Facilitation, Interference and Compatibility Effects

One of the tenets of embodied cognitive science [1] is that sensorimotor aspects of the brain are not merely purveyor or receiver of data but are actively involved in cognition. One of the psychological observations that are thought to be a result of such a direct involvement are for instance facilitation or interference effects in language [2]. For example in a study [3] participants were given a sentence describing the execution of an action and had to decide whether the verb was abstract or concrete, responding with either the hand or the foot if a presented verb was concrete (and refraining from responding otherwise). It was found that response times were slower if participants had to respond with the same effector necessary for executing the action described by the sentence compared to responding with the other effector. Other studies found facilitatory effects, e.g. [4,5]. A more thorough discussion, including a model unifying the apparent conflicting observations, can be found in [2].

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 75–84, 2012.

Compatibility effects are a related phenomenon in which response times in (for instance) classification tasks are modulated by a variable that has no direct effect on the task itself. In one study [6], participants had to classify objects as upright or upside down by pressing left and right keys on a keyboard. Response times were found to be faster if the handle of the object was on the same side as the correct response key than in the incongruent case, suggesting that the handle activated parameters related to reaching towards it even though it was irrelevant to the task. The issue is discussed in more detail, along with reviews of other studies and models pertaining to affordance-related facilitation, interference or compatibility effects in [7].

1.2 Understanding Affordance Processing for Artificial Cognitive Systems

Effects such as the ones discussed in the previous section are valuable insights since they shed some light on the processing in human cognition. If one subscribes to the idea that "intelligent" artificial cognitive systems should mimic corresponding human capabilities, then it is equally important to understand the precise cognitive mechanics underlying these capabilities. Psychological effects such as the ones discussed above impose constraints onto the system since they illustrate that apparently separate systems nonetheless appear to be modulated by each other. It is therefore a worthwhile activity to attempt the creation of computational models aiming at identifying the mechanisms which could give rise to facilitation, interference or compatibility effects, [2].

In the present paper, we address a multi-modal effect in affordance processing [8]. Participants were shown a sentence containing either an action (*e.g.* grasp) or observation (*e.g.* look at) verb followed by an object. They then had to decide whether or not a subsequent picture showed the object from the sentence. Results showed that response times were affected by verb type, object orientation in the picture and grip type required to grasp the object. Later work (Marino et al, in preparation) further shows that response times are affected by whether or not the size of the displayed object matches its usual size. It therefore appears that language comprehension forms a "motor prototype" of an object based on its affordances which is then used in the image identification task.

Affordance processing is, in general terms, highly relevant for artificial cognitive systems which need to interact robustly and autonomously in the real world (for instance robots, [9,10]). The experiment by [8] is of particular interest since it combines linguistic comprehension and visual processing, both required features for future robots. The work presented here is therefore a first step into the direction of general multimodal affordance processing models that address effects such as those discussed previously. Such models are, so far, rather scarce with most efforts focussed instead on affordance extraction (*e.g.* [11], see [12,7] for recent reviews). An exception is the model TRoPICALS [12], which specifically addresses compatibility effects in affordance processing. The present model is, in comparison, a more focussed model addressing more specific issue but that can, due to the modelling technique used, be extended in the future to integrate more effects.

1.3 Dynamic Field Theory

Dynamic field theory (DFT) [13] is a mathematical framework based on the concepts of dynamical systems and inspired by neurophysiology [14]. In a nutshell, fields represent populations of neurons and their activations follow continuous responses to external stimuli.

Fields are used to represent perceptual features, motion or cognitive decisions, e.g. position, orientation, color, speed. The dynamics of these fields allow the creation of peaks which are the units of representation in DFT [13]. Different configurations of one or more fields are possible and the designer is responsible for creating a proper connectivity and tuning of parameters. The result of activating this type of network is a continuously adaptive system that responds dynamically to any change coming from external stimuli. Dynamic fields have been previously used in affordance processing models: [15] presents an affordance competition hypothesis based on dynamic fields while the TRoPICALS model [12] uses them in some of its subsystems.

2 The Model

In the experiments studied in this work ([8], Marino et al. in preparation), stimuli are presented in two stages. First, subjects read a sentence on screen which contains a verb and the name of an object onto which the action represented by the verb will be performed. Second, the screen shows an image of an object in one of four combinations of size and orientation:

- RC: Real size and canonical orientation.
- RN: Real size and non-canonical orientation.
- SC: Scaled size and canonical orientation.
- SN: Scaled size and non-canonical orientation.

The real and scaled versions of an object represent a normal/average size of an object and its over-/under-sized version of it on screen respectively. The canonical orientation is that at which the presented object is found in normal situations, whereas the non-canonical orientation shows the object in an “upside-down” configuration.

The model assumes that a learning process on different objects has been performed throughout a life-time of experiences. Those previous experiences have shaped a long-term memory space composed of features such as size, orientation, color, weight, odor, etc. For the present study, the display gives us information about size, orientation and color which are the features used in this model.

These features are represented as follows: the size of an object in the screen is measured as the number of pixels it occupies; the orientation of an object is measured as the angle of its main axis with respect to the horizontal; finally, color is used as the third visual dimension in order to disambiguate between objects of similar size and orientation, e.g. an apple and an orange.

Groups of three two-dimensional fields are necessary for modelling the results of these experiments: size-orientation, size-color, and color-orientation. Besides the three two-dimensional fields containing the previous experiences with specific objects, long-term memory (LTM), it is necessary to work with three two-dimensional fields that will integrate the activity coming from LTM and the external stimuli (verbal and visual), see Fig. 1. In contrast to LTM, this new group of dynamic fields, which from now on will be called working memory (WM), contains interactions within each field and receives inputs from LTM and external stimuli.

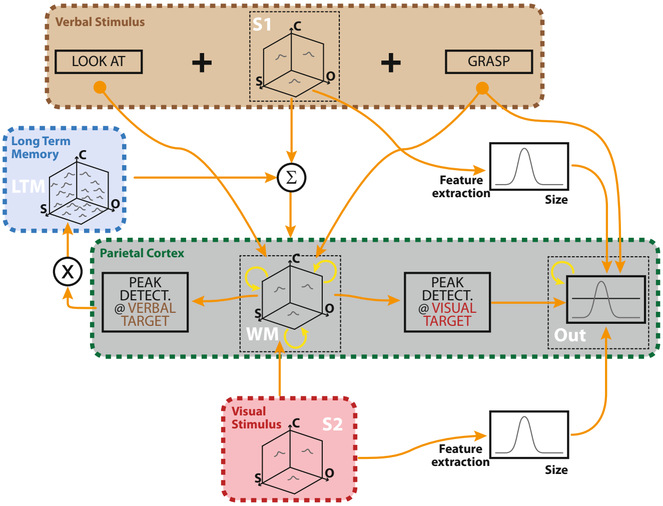


Fig. 1. Dynamic field model of affordances activated by verbal and visual stimuli

The dynamics of WM are tuned to work around a self-sustained attraction point. This means that once the verbal command has dissapeared, a short term memory persists, holding a peak above threshold level. Once a peak at the field site of the verbal stimulus is detected in WM, a constant feedback is given to LTM at the site of the current object. This follows the assumption that our minds give an extra "focus" to those objects that have called our attention, momentarily inhibiting other memories.

A visual stimulus adds new activity to the dynamics of a stable peak in WM. Depending on the position of this new building peak, the visual stimulus will either compete for the creation of a new peak or help mantain the previous peak. Detecting the position of this new peak in WM with respect to the position of the visual stimulus gives enough information to decide weather or not an object belongs to one of the four possibilities studied here, i.e. RC, RN, SC, SN.

Finally, the *size* dimension is used for measuring both the time needed to take a decision. A one-dimensional field is used to integrate both the verbal and visual stimuli and the activity from WM. This new field, labelled as "Out", contains also internal interactions and take as external inputs the extracted *size* dimensions of both the verbal and the visual stimuli. The position and amplitude of this new peak helps us decide weather the system goes for a precision or a power grip and also if the verbal object is the same as the visual object.

The mathematical formulation of the different blocks of the previous model is described at the in end of this document in Appendix A.

3 Results

Plots for two-dimensional fields versus their activations make use of 3 dimensions, Fig. 2. Groups of 3 countour plots arranged in a cartesian space will be used throughout this section to show the dynamics of WM.

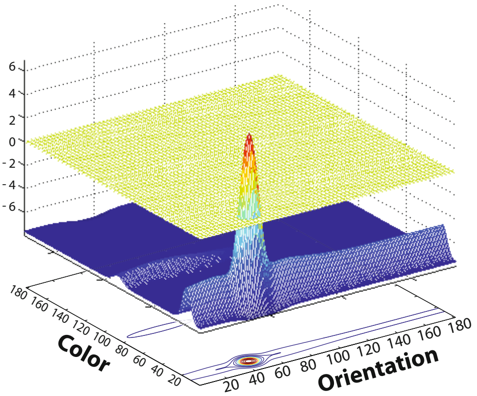


Fig. 2. Example of a countour plot (white plane) from the activation of a two-dimensional field. The zero threshold (green plane) is shown as a reference for the beginning of local excitation of peaks in the activity fields (blue plane).

As mentioned in the previous section, the model assumes that a long term memory module has collected the activity and associations of different features from different objects. In this example we have created a long term memory space containing two objects each one following a gaussian distribution in all dimensions, Fig. 3. The first object (Obj.A) was set to have an average size of 25 units, and the second (Obj.B) was set to 80 units; and a similar variance of 10 units. The hue component of their colors was used to set Obj.A at 15 and Obj.B at 85, and variance 10. Finally, it is assumed that the orientation of the objects also follows a Gaussian distribution, centred around the orientation where the object has been experienced most; for Obj.A it was set to 45 and for Obj.B at 30. The variance for this dimension was set to a large value (30 units) to simulate the multiple orientations that an object can be found in but considering a preferred orientation. All memories were normalized to a maximum amplitude of 1 unit.

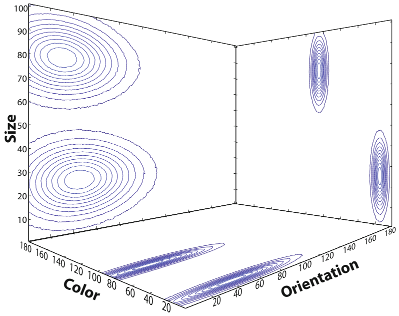


Fig. 3. Long term memory (LTM). Two different objects are defined within a 3D space at coordinates: Obj.A(S:25,O:45,C:15) and Obj.B(S:80,O:30,C:85).

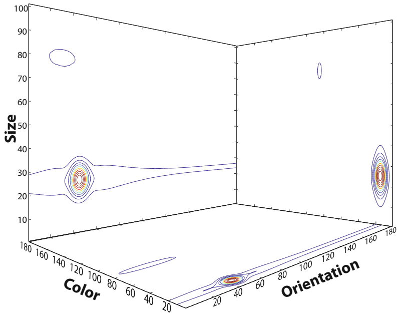


Fig. 4. Verbal stimulus. A peak is created at the location of Obj.A(S:25,O:45,C:15). The peak reaches a stable state over the zero threshold in all 2D fields.

The verbal stimulus was simulated as a gaussian peak with amplitude of 4 units, and variance of 5 units in all dimensions. The position of Obj.A, i.e. [45 25 15], in the three-dimensional space, i.e. [Size Orientation Color], was chosen to be the configuration for the verbal stimulus, Fig. 4. This stimulus was added for 100 timesteps to WM, enough time for the peak to reach a stable state.

Each of the cases studied in this experiment, i.e. RC, RN, SC, and SN, refers to a specific visual stimulus located in a different position of the three-dimensional space. Figure 5 collects the activation of all three fields after reaching a stable state, i.e. a peak has reached it's maximum. For this case, the peaks are located in the same or slightly close coordinates of the verbal stimulus since they are the most probable place where the long term memory has been shaped for Obj.A.

A real-noncanonical stimulus is simulated as a Gaussian input located at the same *size* and *color* coordinates but at a different orientation, Fig 6. The location of the orientation stimuli was set to 90 degrees more than the canonical orientation. In a real environment, having a 180 degrees difference between the canonical and non-canonical orientations would not make much difference in this model since both the 90 and the 180 distances are not within the variance width of the original memory.

The scaled-canonical and scaled-noncanonical stimuli follow the same configuration as the previous two cases but with a different position in the *size* dimension, Fig. 7, 8.

The model also considers the case in which the object presented in the visual stimulus is different from the one presented in the verbal stimulus, Fig 9. The peak created by the verbal stimulus is stable enough to keep its activity above the zero threshold when a new input, i.e. the visual stimulus, is now part of the dynamics.

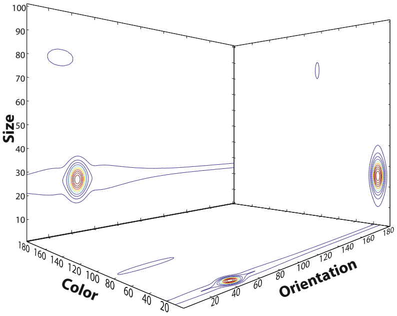


Fig. 5. A real-canonical (S:25,O:45, C:15) visual stimulus is presented, the previous peak created by the verbal stimulus remains active

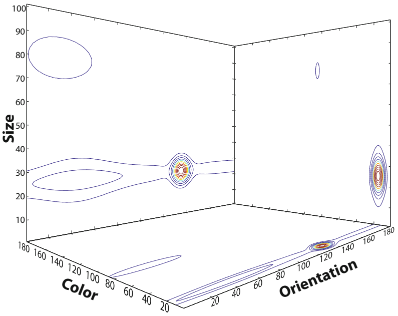


Fig. 6. A real-noncanonical (S:25, O:135,C:15) visual stimulus is presented, the previous peak (verbal stimulus) disappears and a new peak emerges at the location of the visual stimulus

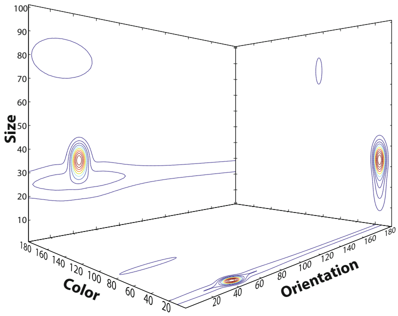


Fig. 7. A scaled-canonical (S:35,O:45, C:15) visual stimulus is presented, the previous peak (verbal stimulus) moves to the location of the visual stimulus

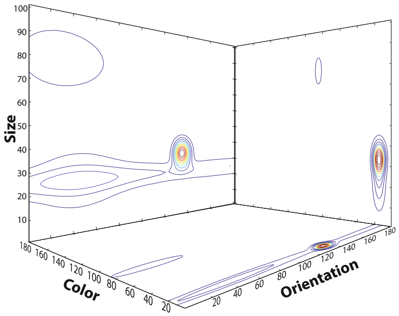


Fig. 8. A scaled-noncanonical (S:35, O:135,C:15) visual stimulus is presented, the previous peak (verbal stimulus) disappears and a new peak emerges at the location of the visual stimulus

Finally, Fig. 10 shows the reaction times of the proposed model for all cases and for two different verbs, 'look at" and 'grasp". As with the human trials, a set of 24 subjects for each verb, i.e. 'look at" and "grasp", was simulated by adding spatially correlated noise. Compared to the reaction times in human experiments the current model performs fairly well. The only case where a substantial difference can be seen is for the real-canonical case. The model shows a faster response to this kind of stimulus compared to human reaction times.

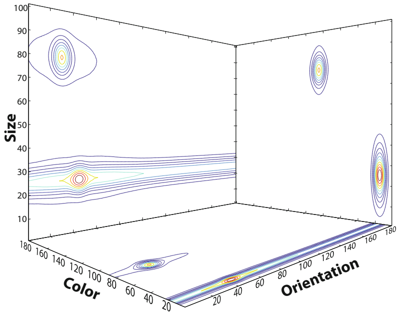


Fig. 9. Obj.B (S:80,O:30,C:85) presented as visual stimulus. The previous peak (verbal stimulus) remains, the new input is not strong enough to overcome the stabilized peak in working memory.

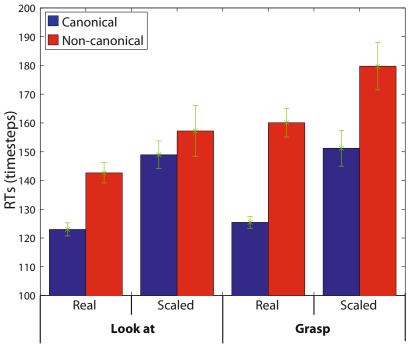


Fig. 10. Reaction times for size-orientation combinations of visual stimuli when using "look at" and "grasp" verbs

4 Discussion

The simulation trials presented here generated close results to human trials. The only case where it is possible to notice a significant difference is for the real-canonical case in both verbs. The model generates faster responses which, in other circumstances, could be considered as intuitive. Human trials show only a significant increase on reaction times when a "grasp" command is issued and a visual stimulus is presented with scaled-noncanonical object. This effect was successfully captured by the proposed model. Moreover, the model was able to detect the inconsistency of a different visual stimulus given a verbal command on a different object. A one-dimensional field was created as the final block that integrates the activity of the *size* dimension in all other fields. The advantage of using the *size* dimension is that this feature is directly proportional to the aperture of a gripper, thus making the whole sensori-motor loop a dynamic process. Last, but not least, the model was designed to autonomously decide weather or not to execute a motor action depending on the command that started the trial.

The integration of sensor modalities, and different features within each modality, in order to make sense of events around us seems to be a key point in the development of human cognition. Both short and long term memories play an important role in the emergence of affordances which, in the end could be seen as the desired outcome of this integration process. Therefore, finding a good model that offers scalability for the integration of different types of information is of utmost importance in the study of cognitive robotics.

The current work proposes a model of affordances activated by nouns of graspable objects. The model is based solely on the concepts of dynamic field theory. The use of dynamic fields allows us to integrate in a single framework interesting properties like the creation of short and long term memories, decision making processes, competition and collaboration dynamics, etc.

Dynamic field theory offers not only one-dimensional fields as units or modules of representation for physical and abstract dimensions. The possibility of having zero-dimensional fields, i.e. a single dynamic neuron, or n-dimensional fields each one representing low- or high-level features makes of this mathematical framework a powerful element to be considered in the creation of cognitive architectures.

5 Conclusions

A three-dimensional space was designed and used as a way of simulate affordances. Three different physical dimensions represent features of objects associated dynamically through two-dimensional fields. *Size* and *orientation* were the two main features used in the human trials; *color* was chosen as the third feature in order to disambiguate between objects. We argue that this 3D space is the simplest example of what could be seen as an hyper-space of associations between a large number of sensor modalities and physical properties.

The model presented here can be adapted to include other physical properties and simulate the responses from subjects of different ages. Future work includes the creation of long-term memories to replace the simulated objects used in this first design.

References

1. Chrisley, R., Ziemke, T.: Embodiment. In: Encyclopedia of Cognitive Science, pp. 1102–1108. Macmillan Publishers (2003)
2. Chersi, F., Thill, S., Ziemke, T., Borghi, A.M.: Sentence processing: linking language to motor chains. Frontiers in Neurorobotics 4(4) (2010)
3. Buccino, G., Riggio, L., Melli, G., Binkofski, F., Gallese, V., Rizzolatti, G.: Listening to action related sentences modulates the activity of the motor system: a combined tms and behavioral study. Cognitive Brain Research 24, 355–363 (2005)
4. Scorolli, C., Borghi, A.M.: Sentence comprehension and action: effector specific modulation of the motor system. Brain Research 1130, 119–124 (2007)
5. Borghi, A.M., Scorolli, C.: Language comprehension and hand motion simulation. Human Movement Science 28, 12–27 (2009)
6. Tucker, M., Ellis, R.: On the relations between seen objects and components of potential actions. Journal of Experimental Psychology: Human Perception and Performance 24(3), 830–846 (1998)
7. Thill, S., Caligiore, D., Borghi, A.M., Ziemke, T., Baldassare, G.: Integrating theories and computational models of affordance control and mirror neurons: A review and a model design (submitted)
8. Borghi, A.M., Riggio, L.: Sentence comprehension and simulation of object temporary, canonical and stable affordances. Brain Research 1253, 117–128 (2009)
9. Şahin, E., Çakmak, M., Doğar, M.R., Uğur, E., Üçoluk, G.: To afford or not to afford: A new formalization of affordances toward affordance-based robot control. Adaptive Behavior 15(4), 447–472 (2007)

10. Fitzpatrick, P., Metta, G.: Grounding vision through experimental manipulation. Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences 361(1811), 2165–2185 (2003)
11. Fagg, A.H., Arbib, M.A.: Modeling parietal-premotor interaction in primate control of grasping. Neural Networks 11, 1277–1303 (1998)
12. Caligiore, D., Borghi, A.M., Parisi, D., Baldassarre, G.: TRoPICALS: A computational embodied neuroscience model of compatibility effects. Psychological Review 117, 1188–1228 (2010)
13. Schöner, G.: Development as Change of System Dynamics: Stability, Instability, and Emergence. In: Toward a New Grand Theory of Development? Connectionism and Dynamic Systems Theory Re-Considered, pp. 25–49. Oxford University Press, New York (2009)
14. Amari, S.I.: Dynamics of pattern formation in lateral-inhibition type neural fields. Biological Cybernetics 27, 77–87 (1977)
15. Cisek, P.: Cortical mechanisms of action selection: the affordance competition hypothesis. Philosophical Transactions of The Royal Society B - Biological Sciences 362, 1585–1599 (2007)

Appendix

Verbal and visual stimuli follow a Gaussian distribution with gain $C_i = 4$ and standard deviation $\sigma_i = 5$, Eq. 1. Each memory included in LTM follows also a Gaussian distribution as in Eq. 1 but with $C_i = 1$ and $\sigma_O = 30, \sigma_S = 10, \sigma_C = 5$.

$$S_i(x, y) = C_i \; exp\left[\frac{-(x - x'_i, y - y'_i)^2}{2\sigma^2_{Si}}\right]; \; x, y \in \{Orientation, Size, Color\} \quad (1)$$

Field “Out” is integrated over a single dimension (Size), Eq. 2a. Verbal and visual stimuli add their *Size* component to this field as an external input (S) if and only if a peak is detected in all 3 fields of WM at the same time.

$$\tau \dot{u}(x, t) = -u(x, t) + h + S(x, t) + \int w(x - x') f[u(x', t)] dx' \quad (2a)$$

$$w(x - x') = C_{exc} \; exp\left[\frac{-(x - x')^2}{2\sigma^2_{exc}}\right] - C_{inh} \; exp\left[\frac{-(x - x')^2}{2\sigma^2_{inh}}\right] - g_{inh} \quad (2b)$$

Where $h = -3$ (*look-at*) and $h = -2$ (*grasp*), $f(u)$ represents a sigmoidal function with steepness $\beta = 4$, and $w(x)$ a mexican-hat kernel (Eq. 2b) with zero-mean, gains $C_{exc} = 12, C_{inh} = 7, \sigma_{exc} = 4, \sigma_{inh} = 10$ and $g_{inh} = 0.1$.

All three WM fields have similar dynamics as Eq. 2a but in these cases the integration is over two dimensions instead of one, Eq 3.

$$\begin{aligned} \tau \dot{u}(x, y, t) = & - u(x, y, t) + h + S(x, y, t) \\ & + \int\int w(x - x', y - y') f[u(x', y', t)] dx' dy' \end{aligned} \quad (3)$$

Where the resting level $h = -4$ and the kernel $w(x, y)$ used the same values as the 1D case with the exception of the global inhibition which was $g_{inh} = 0.025$ (*look-at*) and $g_{inh} = 0.04$ (*grasp*).

A Bio-inspired Model Reliably Predicts the Collision of Approaching Objects under Different Light Conditions

Ana Carolina Silva and Cristina Peixoto dos Santos

Industrial Electronic Department, University of Minho, Portugal

Abstract. In this paper, we present a model of the Lobula Giant Movement Detector, which is a part of a visual pathway responsible for triggering collision avoidance manouvres in the locust *Locusta Migratoria.* Also based on locust neural adaptation to transitions in light intensities, the model proposed here integrates a mechanism for light adaptation. The tests performed with the model demonstrate its ability to reproduce several characteristic properties of the LGMD response, including the firing rate profile for different visual stimuli. Additionally, results obtained for different light conditions show that the increase in the LGMD model efficiency is provided by the new mechanism of light adaptation. In here, the LGMD is presented as an ideal model to develop sensors for automatic collision detection.

Keywords: Bio-inspired model, Lobula Giant Movement Detector neuron, artificial neural networks, collision avoidance, spatiotemporal summation.

1 Introduction

Real-time collision detection in dynamic scenarios is a hard task if the algorithms used are based on conventional techniques of computer vision, since these are computationally complex and, consequently, time-consuming. On the other hand, bio-inspired visual sensors are suitable candidates for mobile robot navigation in unknown environments, due to their computational simplicity. In fact, some animals have been pressured by natural selection to obtain very complex behaviours, while maintaining a relatively simple nervous system. Insects are a perfect example. One advantage of studying the nervous system of insects is based on the fact that they have fewer neurons than other animals such as vertebrates. In addition, the properties of a single identified neuron can often yield general properties and mechanisms that are applicable to other systems [1].

One insect in particular, the locust, has evolved a dedicated and well-studied collision avoidance neural pathway that is responsible for generating collision avoidance behaviours to avoid predation and continual in-flight collisions with conspecifics [2]. The visual system of the locust is paramount to its survival, and acts as a great model system for study.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 85–95, 2012.

The *Locusta Migratoria* possess apposition compound eyes, highly tuned to activity during day light. However, it is known that locusts perform migration flights at low level intensities [3]. After a deep research it was found that, in environments with low level intensities, locusts improve the photon capture neurally by summing the outputs of neighbouring visual channels (spatial summation) and/or by increasing the length of time a sample of photons is counted by the compound eye (temporal summation). So, using spatiotemporal summation, locusts are able to extend their vision into dim light [4].

It is also known that locusts have a large neuron in their brain called the Lobula Giant Movement Detector (LGMD), that is tightly tuned to respond to objects approaching on a direct collision course [5,6,7]. On the other hand, non-colliding objects do not show the same increase in excitation and, by this reason, are unlikely to trigger avoidance reactions [2,8,9,10,11]. LGMD spikes are transferred to the Descending Contralateral Movement Detector (DCMD) which, subsequently, is connected to flight interneurons and motorneurons within the thoracic ganglia [2,10]. Therefore looming responses in this pathway may have consequences for collision avoidance behaviours.

Based on the relatively simple encoding strategy of the LGMD, we concluded that it is a good candidate as a bio-inspired model for collision detection. According to literature, the first physiological and anatomical bio-inspired model for the LGMD neuron was developed by Bramwell [12]. The model continued to evolve [13,14,15,16] and it was used in mobile robots and even in automobiles for collision detection. However, further work is needed to develop more robust models that can account for complex aspects of visual motion [10], as well as be able to work at different light intensities [4]. In this article, we are interested in integrating two recent LGMD models, [14] and [16], in order to take the advantage of noise immunity proposed in the first and direction sensitivity proposed in the second. Based on the principles of the locust visual system previously referred, we subsequently add a capability of light adaptation to the developed model.

In this article, the proposed model is verified when submitted to artificial visual stimuli of approaching and receding objects, including images with different light intensities and noise, and a real video captured by a camera. The results show that the system avoids approaching obstacles in an effective way despite these disturbances.

2 The Model

The biologically inspired neural network here proposed is based on previous models described on [13,14,15,16]. The modified neural network is shown on figure 1. In this model, we integrate an adaptative spatiotemporal summation system, which has the capability to adapt the neural network to different light intensity scenarios.

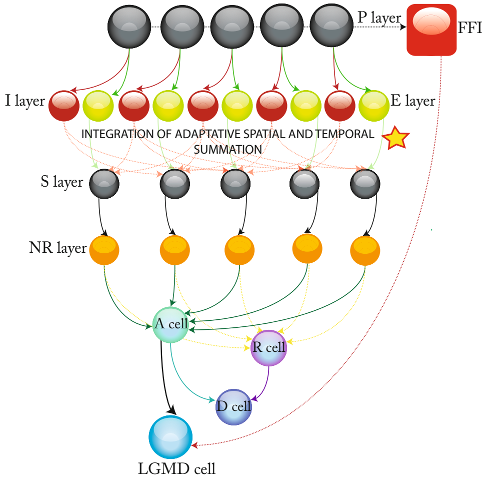


Fig. 1. Schematic illustration of the proposed LGMD model

A grayscale image of the camera current field of view, represented has a matrix of values (from 0 to 255), is the input to a matrix of photoreceptor units (P layer).

This layer calculates the absolute difference between the luminance of the current and of the previous input images, that is:

$$P_f(x, y) = |L_f(x, y) - L_{f-1}(x, y)|\,, \tag{1}$$

where $P_f(x, y)$ is the output relative to the cell in the (x, y) position at frame f, $L_f(x, y)$ and $L_{f-1}(x, y)$ are the captured luminance at position (x, y) for frames f and $f - 1$, respectively. This layer allows us to detect the object edges. The output of the P layer is the input of two different layers: the excitatory (E) and the inhibitory (I) layer. To the excitatory cells of the E layer, the excitation that comes from the P layer is passed directly to the retinotopic counterpart at the S layer. And the inhibition layer (or I layer) receives the output of the P layer and applies a blur effect on it, using:

$$I_f(x, y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} P_{f-1}(x + i, y + j) \cdot W_l(i, j),\ i, j \neq 0, \tag{2}$$

where $I_f(x, y)$ is the inhibition relative to the cell in the (x, y) position at frame f, $W_l(i, j)$, an empirically set kernel, represents the local inhibition weight. The inhibition from the (x, y) cell only spreads to the nearest neighbors and does not inhibits itself. This process is strongly supported by the biological nervous systems. In biology, an excited neuron does not inhibits itself, it inhibits the neighboring neurons, with a temporal delay associated to the inhibitory synapses (and this is the reason why we use the P cell excitement corresponding to the previous time-step, $f - 1$). Relative to the definition of the values holding by this kernel, the inhibition value of a particular cell is given by the distance at which a neighboring cell is located. The use of such kernel is also based on biological systems, since distant neurons inhibit a particular neuron with less intensity than those that are closest to a neuron, due to the decrement of the neuronal signal with increasing distance. Finally, the excitatory flux from the E cells and the inhibition that comes from the I cells are summed by the S cells (summing cells), using the following equation:

$$S_f(x,y) = E_f(x,y) - w_i.I_f(x,y),\ E_f(x,y) = P_f(x,y), \tag{3}$$

where w_i (a scalar) represents the inhibition strength. Based on [14], a new mechanism for the LGMD neural network was added to filter background noise. This mechanism, implemented in the *NR* layer, takes clusters of excitation in the S units to calculate the input to the LGMD membrane potential. These clusters provide higher individual inputs then the ones of isolated S units. The excitation that comes from the S layer is then multiplied by a passing coefficient $C\mathrm{e}_f$, whose value depends on the surrounding neighbours of each pixel, calculated as follows:

$$C\mathrm{e}_f(x,y) = \frac{1}{9}\sum_{i=-1}^{1}\sum_{j=-1}^{1} S_f(x+i,y+j) \tag{4}$$

The final excitation level of each cell in the *NR (*Noise-Reduction*)* layer, at frame f (NR_f), is given by:

$$\mathrm{NR}_f(x,y) = |S_f(x,y).C\mathrm{e}_f(x,y).w^{-1}| \tag{5}$$

$$w = \max(|C\mathrm{e}_f|)C_w^{-1} + \triangle c \tag{6}$$

C_w is set to 4, $\mathit{\Delta} c$ is a small number (0.01) to prevent w from being zero, and $\max(|C\mathrm{e}_f|)$ is the largest element in matrix $|C\mathrm{e}_f|$. Within the *NR* layer, a threshold filters the decayed excitations (isolated excitations), as:

$$\tilde{\mathrm{N}}\mathrm{R}_f(x,y) = \begin{cases} \mathrm{NR}_f(x,y),\ \text{if}\ \mathrm{NR}_f(x,y).C_{\mathrm{de}} \geq T_{\mathrm{de}} \\ 0,\ if\ \mathrm{if}_f(x,y).C_{\mathrm{de}} < T_{\mathrm{de}} \end{cases}, \tag{7}$$

where $C_{\mathrm{de}} \in [0,1]$ is the decay coefficient and T_{de} is the decay threshold (set to 20). The decay threshold here used was experimentally determined. The *NR* layer is able to filter out the background detail that may cause excitation. Hence, only the main object in the captured scene will cause excitation. The *LGMD* potential membrane K_f, at frame f, is summed after the *NR* layer,

$$\mathrm{LGMD}_f = K_f = \sum_{x=1}^{n}\sum_{y=1}^{m}(\tilde{\mathrm{N}}\mathrm{R}_f(x,y)), \tag{8}$$

where n is the number of rows and m is the number of columns of the matrix representing the captured image. The A (Approaching) and R (Receding) cells (adapted from [16]) are two grouping cells for depth movement direction recognition. The A cell holds the mean of three samples of the *LGMD* cell.

The R cell shares the same structure as the A cell but with a temporal difference, having one frame delay from A. According to the theory described above, it can be concluded that if the object is approaching $A_f > R_f$ and if the object is receding, $R_f > A_f$. The D cell or Direction cell ($\in \{-1, 0, 1\}$ in case of receding, no movement and approaching object, respectively) is used to calculate the direction of movement. This cell exploits the movement direction in depth. It is based on the fact that a looming object (approaching) gets larger whereas a receding object gets smaller. In a way to distinguish the movement direction detected by the D cell, a threshold mechanism was added, $T_D(0.05 \times n \times m)$, which was experimentally determined.

$$D_f = \begin{cases} 1, \text{ if } |A_f| - |R_f| \geq T_D \\ 0, \text{ if } T_D < |A_f| - |R_f| < T_D \\ -1, \text{ if } |A_f| - |R_f| \leq T_D \end{cases} \tag{9}$$

The LGMD membrane potential K_f is then transformed to a spiking output $k_f \in [0.5, 1]$ using a sigmoid transformation,

$$k_f = (1 + e^{-K_f \cdot ncell^{-1}})^{-1}, \tag{10}$$

where ncell is the total number of cells in the *NR* layer. The collision alarm is decided by the spiking of the *LGMD* cell.

A spiking mechanism was implemented using an adaptable threshold. This threshold starts with a value experimentally determined, T_s (0.88) and it is updated at each frame, through the following process,

$$T_s = \begin{cases} T_s + \triangle t, \text{ if } s_{av} > \mathit{\Pi} \text{ and } (T_s + \triangle t) \in [T_l, T_u] \\ T_s - \triangle t, \text{ if } s_{av} < \mathit{\Pi} \text{ and } (T_s - \triangle t) \in [T_l, T_u] \\ T_s, \quad \text{others} \end{cases}, \tag{11}$$

where $[T_l, T_u]$ defines the lower and upper limits for adaptation (T_l is 0.80 and T_u is 0.90) , $\mathit{\Delta} t = 0.01$ is the increasing step, $\mathit{\Pi} = 0.72$ is a threshold that limits the averaged spiking output s_{av},between frame $f - 5$ to frame $f - 2$,

$$s_{\text{av}} = 0.25 \sum_{i=2}^{5} s_{f-i}. \tag{12}$$

If the sigmoid membrane potential k_f exceeds the threshold T_s a spike is produced, as follows:

$$s_f = \begin{cases} 1, \text{ if } k_f \geq T_s \\ 0, \text{ others} \end{cases} \tag{13}$$

Finally, a collision is detected when there are n_{sp} spikes in n_{ts} time steps ($n_{\text{sp}} \leq n_{\text{ts}}$), where n_{sp} is 4 and n_{ts} is 5 (values experimentally determined).

$$C_f = \begin{cases} 1, \text{ if } \sum_{f-n_{\text{ts}}}^{f} s_f \geq n_{\text{sp}} \\ 0, \text{ others} \end{cases} \tag{14}$$

The escape behavior is initialized when a collision is detected. Besides that, the spikes can be suppressed by the *FFI* cell when whole field movement occurs. If it is not suppressed during the tuning of the robot, for example, the network may produce spikes and even false collision alerts due to sudden changes in the visual scenario.

The *FFI* cell is a cell which is very similar to the *LGMD* cell but receives the output from the P layer (and not from the S layer), as follows:

$$\mathrm{FFI}_f = \frac{\sum_{x=1}^{m}\sum_{y=1}^{n}|P_{f-1}(x,y)|}{\mathrm{ncell}}, \tag{15}$$

where P_{f-1} is the output of the P layer at frame $f-1$. If FFI_f exceeds a threshold T_{FFI}(experimentally set to 25), the spikes produced by the *LGMD* cell are automatically inhibited.

We have introduced a light adaptative system within the S layer, that comprises three different states depending on the mean gray scale value of the captured image: STATE 1 is activated in the presence of a very low light intensity scenario (mean gray scale value lower than 50); STATE 2 is activated in the presence of a medium light intensity scenario (mean gray scale value lower than 150); and STATE 3 is activated in the presence of a high light intensity scenario (mean gray scale value greater than 150).

In STATE 1 and STATE 2, a Gaussian spatial kernel (K_k) is chosen, and the following processment is done within the S layer:

$$S_f\,\mathrm{spatial}(x,y) = \sum_{i=a}^{b}\sum_{j=a}^{b} S_f(x+i, j+i).K_k, \tag{16}$$

where K_k ($k = \mathrm{State1}, \mathrm{State2}$) is a kernel that allows the spatial summation between neighboring pixels within the S layer; and a (b) is -2 (2) or -1 (1) depending whether on State 1 or 2, respectively. The K_{State1}has a gaussian distribution, with a standard deviation of 1, and a strenght of 20 units. The K_{State2}has a gaussian distribution, with a standard deviation of 0.5, and a strenght of 4 units. This kernel allows the spatial summation between neighboring pixels within the S layer.

Besides this spatial summation, when any of these states is activated, the model will activate a temporal summation mechanism, that sets the final value for each pixel of the S layer, as follows:

$$S_f(x,y) = \sum_{f-c}^{f} S_f\,\mathrm{spatial}(x,y), \tag{17}$$

where $c = -3$ or -1 whether on State 1 or 2, respectively.

In case STATE 3, due to the high intensity level, the spatial and temporal summation are not activated.

3 Experiments and Results

3.1 Implementation and Analysis of the Proposed LGMD Model

In order to assess the effectiveness of the proposed model, we develop a simulation environment in MATLAB, using a a Laptop (Toshiba Portegé R830-10R) with 4 GHz CPU and Windows 7 operating system. To test the efficiency of the LGMD model, different data sets were used. The first experiment was made on a simulated data set, showing a square approaching at different velocities, with a high noise level (500 pixels of random noise, corresponding to 5 percent of the image pixels) and with different light conditions. The second experiment was made with a receding stimuli, showing the same characteristics as previously described for the approaching stimuli. As previously mentioned, the input of the LGMD model are images, which represent a 2 dimensional information of light intensity. Lower intensity is represented by a lower gray scale value within the image. Artificial visual stimuli, with different mean of background intensity: A)10; B)30; C)50; D)70; E)90; F)120; G)200; H)255, were used to stimulate the LGMD model.

In our first experiment, we evaluate the LGMD model, by using a looming stimulus consisting of a solid square, for two different relations between the object half size (represented by l) and object velocity (represented by v). This ratio determines the time course of the angular size of the looming stimulus.We selected these two relations between l and $/v/$ due to their importance on the stimulation of the locust visual system [17][5].

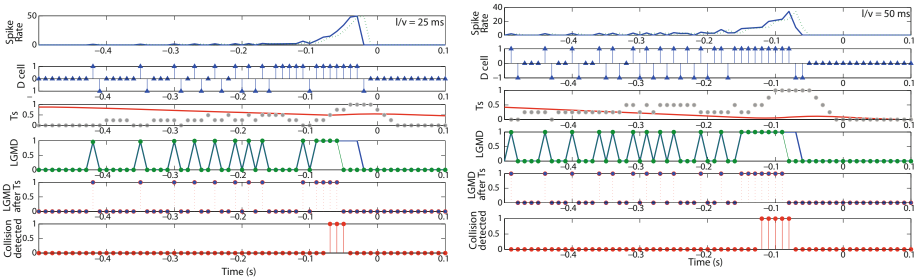


Fig. 2. LGMD model response to an approaching object with $l/|v|$ set at 25 milliseconds (left panel) and $l/|v|$ set at 50 milliseconds (right panel). In all these graphs, the zero value corresponds to the time of collision.

Figure 2 shows the output from the LGMD model, when using background H. In this figure, at each time step we can observe the result of different mathematical processing, corresponding to the layers of the proposed model, executed sequentially, necessary to detect an imminent collision. Observing the top panels of figure 2, in which it is represented the *Spike Rate* of the LGMD model (obtained by dividing the A_f value for the number of pixels in the captured image) we observe that the waveform obtained is consistent with the biological data reported to the same visual stimuli [10]. The analysis of these results showed that the LGMD neural network detected a collision at time -0.07 seconds (for $l/|v|$=25ms) and -0.12 seconds ($l/|v|$=50ms), being the simulated square located at 14 cm and 12 cm, respectively.

After this test, we decrease the background light intensity, following the order from G to A. For these tests, we obtained the results resumed in figure 3.

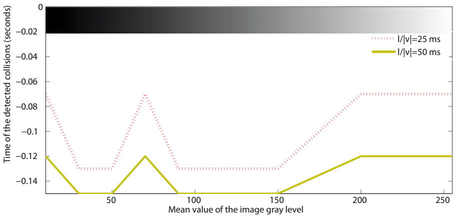


Fig. 3. LGMD model final output to $l/|v|=$ 25 (dots) and 50 milliseconds (line), when subjected to visual stimulus with different light intensities

At different mean values of the image gray level and for each $l/|v|$ relation, the output of the LGMD model relative to the detected collision (figure 3), did not always happen at the same instant (ranging from -0.07 seconds (or 14cm of distance) to -0.12 seconds (or 26 cm) for $l/|v|$=25ms and ranging from -0.12 (or 12 cm) to -0.15 seconds (or 15 cm) for $l/|v|$=50ms). Through these results we can verify that the distance at which the LGMD model detected a collision almost doubled in some situations. This happened due to the fact that we only have 3 different states of light adaptation. So, for the highest mean values of the image gray level within a state, the intensity of the image will be already too high for the size of the kernel used in the spatial summation processing, as well as for the temporal integration used. Consequently, the excitation level of the LGMD cell will strongly increase, leading to the generation of premature "collision detected" spikes. However, for $l/|v|$=50 ms, the collisions were detected when the object was located at 12 or 15 cm, depending on the mean value of the image gray level. Based on these results we can conclude that for high $l/|v|$ values, spatial and temporal summation has a lower effect on the distance at which the collisions were detected. After this first set of tests, we repeated the same stimuli but using a receding trajectory. In all the situations tested no collisions were detected, as expected.

In addition to these simulated visual stimuli, and in order to test the LGMD model in a real environment, we recorded a real video sequence, using a Sony Cyber shot digital camera 7.2 megapixels to obtain the video clip (figure 4, left panel).

The resolution of the video images is 640 by 480 pixels, with 30 frames per second of acquisition frequency . After the video recording and using a movie editor, we created two new video recordings, one showing an environment with high light intensity and other with very low intensity (see figure 4). By using these three video sequences, we were able to test three different stages of the

LGMD model light adaptation. Observing the left panel in figure 4, we observe that, for video sequences with low and medium intensity level, the collisions were detected at -0.08 seconds, ie, when the ball was at 26 cm relatively to the camera. However, for high intensity level, the collision was detected later (-0.04 seconds), ie, when the ball was located at 16 cm to the camera. Despite this difference, consequence of the distinct processing made in each state, the obtained results were very satisfatory, since the LGMD model here proposed is able to: remove the background noise, detect the direction of stimuli movement as well as auto-adapt to different light intensity scenarios, in a high effective way.

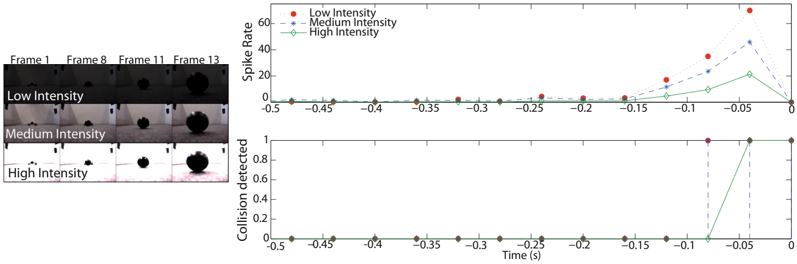


Fig. 4. LGMD model response to a real video sequence under different light conditions

Based on our results, we conclude that the LGMD model proposed here reliably detects collisions of objects of different sizes, different trajectories and under different light intensities. Our bioinspired model reproduces the response of the locust LGMD neuron and is able to account for spatiotemporal summation adaptation in different light conditions.

3.2 Comparative Analysis of Different LGMD Models Responses

As a second goal of the work here described, we evaluate the effectiveness of the proposed model in relation to others LGMD models proposed in literature (which for a better understanding we decided to call LGMD model 1 to the model proposed by [14] and LGMD model 2 to the one proposed by [16]). In order to accomplish this objective, after the computational implementation and stimulation of the LGMD model 1 and 2 with the artificial visual stimuli previously described, with $l/|v|$=50 ms and using different means of background intensities, we verify that the collisions were detected, by each LGMD models, at different time instants and, consequently, at different distances of the object relatively to the camera. The results obtained can be seen on figure 5. Relatively to the LGMD model 1, we can conclude that this model was only able to detect collisions for mean value of the image gray level superior than 120. Consequently, the LGMD model 1 will not be able to work at low light conditions. Since the collisions detected by the LGMD model 1, either with and without noise, are completely overlapping, we could also conclude that this model has high immunity to noise presence in the captured images.

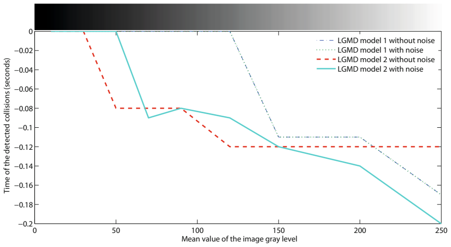


Fig. 5. LGMD model 1[14]and LGMD model 2[16] final output to $l/|v|$=50 miliseconds, when subjected to visual stimulus with different light intensities and with/without noise

Taking into account the results obtained with the LGMD model 2 when not subjected to noise, we can verify that this model is not able to work in very low ligh conditions (below 30) since, for these light intensity levels, the model was not able to detect collisions. For medium values, between 50 and 90, the model was able to detect collisions when the object was located at 8 cm relatively to the camera. And for higher values of gray level, this model was very stable and detected all the collisions at a distance of 12 cm. However, the response of the LGMD model 2 when stimulated with images with 500 pixels of noise, is very unstable concerning the distance at which it detects collisions. This happens due to the inability of this model to eliminate noise. Besides that, in this situation, the model can not detect collision for gray levels below 70.

4 Conclusions

In this paper, we have presented a neural model for obstacle detection and avoidance based on the percentual strategies used by *Locusta Migratoria*. The model reproduces the response of the LGMD neuron and is able to account for spatiotemporal adaptation in different light conditions. It was also shown that, unlike the previous LGMD models described in literature, the neural network here proposed can describe the response of the locust neuron in a wide range of stimulus conditions. For applications as collision detectors for robots, the model proposed is able to remove the noise captured by the camera, as well as enhance its ability to recognize the direction of the object movement and, by this way, remove the false collision alarms produced by the previous models when a nearby object is moving away. Furthermore, the model is able to autonomously adjust to different light conditions.

Acknowledgements. Ana Silva is supported by PhD Grant SFRH/BD/70396/2010. This work is funded by FEDER Funds through the Operational Programme Competitiveness Factors - COMPETE and National Funds through FCT - Foundation for Science and Technology under the Project: FCOMP-01-FEDER-0124-022674.

References

1. Zupanc, G.K.H.: Behavioral Neurobiology: An Integrative Approach. Oxford University Press (2010)
2. O'Shea, M., Williams, J.L.D.: The anatomy and output connection of a locust visual interneurone; the lobular giant movement detector (LGMD) neuron. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology, 257–266 (1974)
3. Chapman, R.F.: The insects: Structure and Function. Hodder and Stoughton, London (1980)
4. Warrant, E.J.: Seeing better at night: lift style, eye design and the optimum strategy of spatial and temporal summation. Vision Res. 39, 1611–1630 (1999)
5. Gabbiani, F., Krapp, H., Laurent, G.: Computation of object approach by a wide-field motion-sensitive neuron. J. Neurosci. 19, 1122–1141 (1999)
6. Gabbiani, F., Mo, C., Laurent, G.: Invariance of Angular Threshold Computation in a Wide-Field Looming-Sensitive Neuron. The Journal of Neuroscience 21(1), 314–329 (2001)
7. Gabbiani, F., Krapp, H.G., Koch, C., Laurent, G.: Multiplicative computation in a visual neuron sensitive to looming. Nature 420, 320–324 (2002)
8. Gray, J.R., Lee, J.K., Robertson, R.M.: Activity of descending contralateral movement detector neurons and collision avoidance behaviour in response to head-on visual stimuli in locusts. Journal of Comparative Physiology A, 115–129 (2001)
9. Rind, F.C.: Non-directional, movement sensitive neurones of the locust optic lobe. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology, 477–494 (1987)
10. Guest, B.B., Gray, J.R.: Respones of a looming-sensitive neuron to compound and paired object approaches. Journal of Neurophysiology 95(3), 1428–1441 (2006)
11. Gray, J.R., Blincow, E., Robertson, R.: A pair motion-sensitive neurons in the locust encode approaches of a looming object. Journal of Comparative Physiology A 196(12), 927–938 (2010)
12. Rind, F.C., Bramwell, D.I.: Neural Network Based on the Input Organization of an Identified Neuron Signaling Impeding Collision. Journal of Neurophysiology 75(3), 967–985 (1996)
13. Blanchard, M., Rind, F.C., Verschure, P.F.M.J.: Collision avoidance using a model of the locust LGMD neuron. Robotics and Autonomous Systems 30(1), 17–37 (2000)
14. Yue, S., Rind, F.C.: Collision detection in complex dynamic scenes using an LGMD-based visual neural network with feature enhancement. IEEE Transactions on Neural Networks 17(3), 705–716 (2006)
15. Stafford, R., Santer, R.D., Rind, F.C.: A bio-inspired visual collision detection mechanism for cars: combining insect inspired neurons to create a robust system. BioSystems 87, 164–171 (2007)
16. Meng, H., Yue, S., Hunter, A., Appiah, K., Hobden, M., Priestley, N., Hobden, P., Pettit, C.: A modified neural network model for the Lobula Giant Movement Detector with additional depth movement feature. In: Proceedings of International Joint Conference on Neural Networks, Atlanta, Georgia, pp. 14–19 (2009)
17. Badia, S.B.I., Bernardet, U., Verschure, P.F.M.J.: Non-Linear Neuronal Responses as an Emergent Property of Afferent Networks: A Case Study of the Locust Lobula Giant Movement Detector. PLoS Comput. Biol. 6(3), e1000701 (2010)

Synthesising a Motor-Primitive Inspired Control Architecture for Redundant Compliant Robots*

Naveen Kuppuswamy[1], Hugo Gravato Marques[1,2], and Helmut Hauser[1]

[1] University of Zürich,
Institute for Informatics, A.I Lab.,
Zürich 8050, Switzerland
naveenoid@ifi.uzh.ch
[2] ETH Zürich,
Dep. of Mechanical and Process Engineering, BIRL,
Zürich 8092, Switzerland

Abstract. This paper presents a control architecture for redundant and compliant robots inspired by the theory of biological motor primitives which are theorised to be the mechanism employed by the central nervous system in tackling the problem of redundancy in motor control. In our framework, inspired by self-organisational principles, the simulated robot is first perturbed by a form of spontaneous motor activity and the resulting state trajectory is utilised to reduce the control dimensionality using proper orthogonal decomposition. Motor primitives are then computed using a method based on singular value decomposition. Controllers for generating reduced dimensional commands to reach desired equilibrium positions in Cartesian space are then presented. The proposed architecture is successfully tested on a simulation of a compliant redundant robotic pendulum platform that uses antagonistically arranged series-elastic actuation.

1 Introduction

It has been argued that natural systems, in order to cope with uncertain, unstructured and dynamically changing environments evolve morphologies and material properties that are physically compliant (adaptable to external influences) and redundant (versatile in face of constraints), among other features [10]. The flip side of this argument is that the Central Nervous System (CNS) needs to cope with the large dimensionality thus induced. Even for simple end-point movements, a large number of muscles are recruited and, thus, have to be supplied with requisite input commands. Since the number of muscles is much higher than the number of variables in which the goal is defined, a single movement can be obtained by many different patterns of muscle activations; this is often referred to as Bernstein's *degrees of freedom problem* [3].

* This research has been supported by the European Community under the 7^{th} Framework Programme by the projects RobotDoc (Grant Agreement No. 235065), a Marie Curie Action ITN, and eSMCs (Grant Agreement No. 270212).

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 96–105, 2012.

This problem is also important from the point of view of designing robotic systems for the real world. The problem of controlling dynamically complex systems is typically approached by explicitly using the (*inverse*) kinematic or dynamical models of the plant. However, the computational complexity drastically increases with the number of degrees of freedom [6]. Learning and optimisation theory offer an alternative to solving the redundancy problem. However, optimization algorithms suffer from an exponential increment of the computational complexity as a function of the dimensionality of the search space, the so called "curse of dimensionality" [12], rendering them intractable. Unsupervised learning methods [13] have also been proposed in this context, however their scalability to complex redundant and compliant robotic systems is unknown.

An alternative paradigm would be to look for a reduced dimensional specification of the system behaviour; a problem that has been studied in the domain of Model Order Reduction (MOR). MOR techniques aim at reducing the order of a dynamical system while preserving the input-output relationship to the extent possible [1]. The robot control applications of MOR techniques is largely underexplored, and we could take inspiration from nature in deriving techniques.

In this context, there is significant biological evidence suggesting that a process of dimensionality reduction may be occurring in neural control mechanisms [7]. The discovery of spinal Convergent Force Fields (CFFs) and their linear combinations in frogs [8] provided neurological justification for the presence of *motor primitives*, which have been described as fundamental units of the motor control system, suitable combinations of which enable complex movements to be carried out. Until now, most approaches have aimed to identify and model primitives from observations of natural movements.

In this paper, we propose a framework for synthesising a motor-primitive inspired control architecture for redundant and compliant robots. The architecture is inspired by recent work in biology [2], which proposed a novel model for the synthesis of motor primitives of a frog's leg using MOR and optimisation of a cost. The work we present adapts and expands their technique to artificial systems, and as a preliminary result we focus on linear dynamical systems. The results are demonstrated in a simulated tendon driven robotic pendulum which uses antagonistically arranged series-elastic actuation.

This paper is organised as follows. The considerations underlying the proposed architecture are presented in Section 2. In Section 3, the algorithm for extracting motor primitives is described. The experiments and simulated results are presented in Section 4, followed by the conclusions in Section 5.

2 Proposed Control Architecture - Considerations

Motor primitives have been characterised [2] as spinally stored constraints on the motor input commands of the form,

$$u = U^*C, \tag{1}$$

where, $U^* = u^*_{1\cdots k}$ is a set denoted as the motor primitives, comprising of k primitives, and C is the vector of reduced dimensional control inputs, $C =$

$[C_1, \ldots, C_k]^T$. Each column of U^* can be thought of as a set of spinally stored muscle activations, similar to the activations produced by microstimulation of the frog's spine. In this formulation, each primitive u_i^*, is in the dimension of total number of muscles present [2], while only k primitives are needed and used to specify their motion. In this form, the primitives represent the basis vectors of the desired space of motor commands.

The approach of Berniker [2] to motor primitive synthesis used Balance Truncation [1], a control theoretic model reduction approach, to reduce the dimensionality of the mechanical system; the method relies on knowing the mechanical plant model. Furthermore, the approach assumes that motor primitives must be non-negative (real muscles cannot be negatively activated), orthogonal (act independently) and useful for generating commands (formalised based on mathematical properties of the equivalent reduced dimensional system); the primitive computation is based on optimising a corresponding cost function.

Ideally, in autonomous robots the architecture is synthesised in a self-organised manner, without knowledge of the full dimensional model in advance, which requires apriori system identification to be carried out. Also, some of the assumptions underlying the primitive synthesis approach [2] have to be adapted to comply with artificial actuation mechanisms. The technique proposed in this paper makes the following assumptions:

1. **Spontaneous Motor Activity Is Used to Collect a Dataset:** In order to self-organise a reduced dimensional model of the mechanical plant, spontaneous motor activity will be employed to perturb the system and collect a dataset characterising the behaviour, as described in Section 3.1.
2. **Statistical and Data Driven Methods Are Used to Reduce the Dimensionality:** The Oja rule [9] demonstrated the ability of unsupervised learning in a network of neurons, to perform Principal Component Analysis (PCA). Hence for dimensionality reduction, a PCA based method, Proper Orthogonal Decomposition (POD), will be used as described in Section 3.2.
3. **Primitives Can Also Be Negative as Motor Commands Can Be Negative for Artificial Systems:** For robotic systems, inputs may be negative as well, since typically most actuation mechanisms such as DC motors tend to exhibit bi-directionality at the output and bipolarity at the input. We address this consideration by proposing a technique for primitive synthesis using Singular Value Decomposition (SVD) described in Section 3.4.
4. **The Reduced Dimensional Model Is Utilised to Generate Control Inputs to Reach Equilibrium Positions:** Motivated by the equilibrium point hypothesis [7] we propose a reduced dimensional controller that generates required motor commands to reach Cartesian space equilibrium positions as described in Section 3.5.

These new assumptions will be utilised in the synthesis of the control architecture as described subsequently. As a preliminary exploration we shall constraint our proposal to linear dynamical systems, although it can potentially be adapted to nonlinear systems as well.

3 Synthesis Methodology

The proposed reduced dimensional architecture is synthesised using the methodology presented in Fig. 1. The various constituent processes are described in this section.

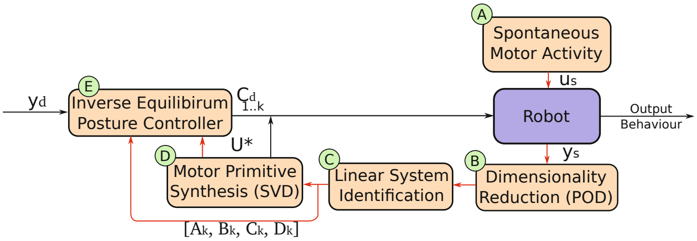

A
Spontaneous Motor Activity
u_s
Robot
Output Behaviour
y_d
E
Inverse Equilibirum Posture Controller
$C^d_{1..k}$
U*
D
Motor Primitive Synthesis (SVD)
C
Linear System Identification
B
Dimensionality Reduction (POD)
y_s
$[A_k, B_k, C_k, D_k]$

Fig. 1. Motor primitive-inspired Control Architecture - Synthesis Methodology. First the Robot is perturbed by Spontaneous Motor Activity $u(t)$ *(A)* to generate a dataset y_s and subsequently MOR (POD) *(B)* and Linear System Identification *(C)* are applied to yield a reduced order model. Primitives are then synthesised using SVD *(D)* and are combined with the reduced model in the equilibrium posture controller *(E)* to generate motor commands corresponding to desired behaviour goals in end-effector space.

3.1 Dataset Generation through Spontaneous Motor Activity

In mammals, the process of spontaneous motor activity (SMA) carries out muscle contractions in the absence of sensory stimulation. This type of motor activity has been observed during sleep throughout all developmental stages (including the foetal stage) [5]. One particular type of SMA observed is the Myoclonic twitch which spontaneously triggers independent contractions of individual muscles. Inspired by this process, we utilise independent and individual pulse inputs $u_s(t)$ (square signals of amplitude 0.01 and duration $2.8s$) to perturb the mechanical system, resulting in a motion output that can be recorded in the form of a dataset (see block *(A)* in Fig.1)of *snapshots* of the dynamical system as $\chi = [x(t_0), \ldots, x(t_i)]$, where $\chi \in \mathbb{R}^{N \times n_t}$ and $x(t_i)$ is the ${n_t}^{th}$ snapshot of the system, where n_t is the total number of snapshots in the dataset (or datapoints) and N is the state dimensionality.

3.2 Reduction Using Proper Orthogonal Decomposition (POD)

The next step is to reduce the dimensionality of the dataset using POD[1] as depicted in block *(B)* in Fig.1. Consider a linear dynamical system of the form below,

$$\dot{x} = Ax + Bu, \quad y = Cx, \tag{2}$$

[1] Also called PCA, Karhunen-Loeve decomposition or factor analysis.

where, $u \in \mathbb{R}^I$ is the input, $x \in \mathbb{R}^N$ is the state, $y \in \mathbb{R}^O$ is the output. The matrices A, B, and C are commonly called as the state, input, and output matrices, respectively. In this case, the reduction aims to find a lower dimensional representation z such that,

$$\dot{z} = A_k z + B_k u, \quad y = C_k z + D_k u, \tag{3}$$

where, $z \in \mathbb{R}^k$ is the state, and D is a feedthrough matrix compensating for steady state differences. We thus look to replace the N dimensional system by a nearly-equivalent (similar in behaviour) k dimensional system, where $k \ll N$. From the dataset χ collected in the previous stage, the Singular Value Decomposition (SVD) then can be used to obtain the *best* [11] reduced dimensional approximation $\hat{\chi}_k$ which minimises the norm $\| \chi - \hat{\chi}_k \|_2$. The SVD renders $\chi = U \varSigma V^T$, where $UU^T = I, VV^T = I$, and the singular values are ordered as $\sigma_1 \geq \dots \sigma_k \geq \dots \sigma_n$. We can then truncate χ to the first k singular values, by using the corresponding first U_k singular vectors as a basis of the k-dimensional subspace we are projecting the dataset to, as $z(t) = U_k x(t)$. The next step is to obtain the model parameters.

3.3 Identification on the Reduced Dimensional Dataset

To identify the reduced dimensional model, we employ system identification on the dataset $z(t)$ as depicted in block *(C)* in Fig.1 . Due to the assumption of linear dynamics, the dataset $z(t)$ obtained from POD will also be guaranteed to be linear [1] and linear least squares identification can be employed as,

$$[\dot{z}(t)] = [A_k, B_k][z^T(t), u^T(t)]^T, \quad [y(t)] = [C_k, D_k][x^T(t), u^T(t)]^T, \tag{4}$$

where, z is the new state variable of the dynamical system, A_k, B_k, C_k, and D_k are the reduced dimensional state, input, output and feedthrough matrices respectively. Note that $u(t)$ and $y(t)$ have not changed from the original system in Eq. 2.

3.4 Primitive Synthesis Using SVD

Once the reduced order model is obtained, primitives in the form of Eq. 1 are computed. An important criterion for the primitives is that ideally the commands generated in the reduced dimensional space are “useful” in the sense of their effect on the state [2]. This is ensured by allowing the primitives U^* to be orthogonal to the nullspace of the reduced dimensional input matrix B_k. Moreover, the primitives are orthogonal to each other (to allow spanning the control input space). Both these goals are accomplished by finding the singular vectors of B_k and choosing the last k of these vectors.

Consider the singular value decomposition of a matrix B_k, $B_k = U \varSigma V^*$. The null space of the input matrix B_k has as its basis, the last $n - k$ columns of the right singular vectors V^* of the decomposition [11]. Since the columns of the V^*

matrix are orthogonal to each other, the first k columns thus can be chosen as primitives, since they are both useful and orthogonal.

$$u^* = \mathcal{N}(B_k)^{\perp}, \quad u^* = V^*_{1\dots k}, \tag{5}$$

where the operator $\mathcal{N}()^{\perp}$ computes the nullspace complement of a matrix. Due to the availability of multiple high speed numerical SVD computing algorithms, primitives computation is faster than methods based numerical optimisation [2].

3.5 Feedforward Equilibrium Posture Controller Design

Once the primitives are computed, a controller can be designed as required. In [4] and [8], it is suggested that the controller is feedforward in structure and it generates the necessary commands to affect the equilibrium posture of the limb. For the obtained system in Eq. 3, the equilibrium state for a given input corresponds to $\dot{z} = 0$, which is therefore,

$$z = -A_k^{-1} B_k u, \quad y = \left[-A_k^{-1} B_k + D_k\right] u. \tag{6}$$

Since the input u is constrained according the motor primitive as in Eq.1, it is sufficient to compute the required reduced dimensional control inputs C_d for a desired output y_d where $C_i \in \mathbb{R}^k$. For this, the pseudo-inverse or the Moore-Penrose inverse (†) can be used to obtain,

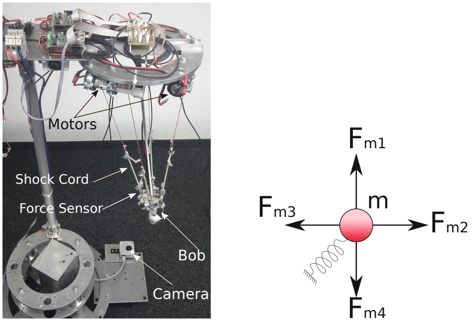
$$C_d = \left[\left(-C_k A_k^{-1} B_k + D_k\right) u^*\right]^{\dagger} y_d. \tag{7}$$

Note that if k is chosen to be of same dimensionality of the output y, Eq.7 is computed using a regular inverse instead of the pseudo-inverse and thus the redundancy problem is directly resolved.

4 Experiments and Results

4.1 Methods: Pendulum Robot and Simulation

The pendulum robot platform is a test setup built to investigate methods and techniques for developmental robotics. Loosely inspired by the human shoulder system, it consists of two mechanically independent pendula, each driven by 4 series elastic actuators coupled in an agonist-antagonist configuration, as shown in Fig.2a. Each muscle system can be actuated independently and includes force and elongation sensors. A camera is mounted on the base of the pendulum looking upwards to extract end-point position as a 2D position measured in the camera frame of reference. The dynamics of this robot is be assumed to be linear under the conditions of bounded amplitude motion due to the relatively long length of the muscles. Since the platform is driven by 4 motors, the input dimensionality is 4. Thus in order to be interesting, we must synthesise a controller with k primitives where $1 < k \leq 4$ to perform meaningful tasks in the 2D task space.



(a) (b)

Fig. 2. a) Pendulum robot platform and b) Linear System simulation of the Pendulum robot platform (the mass is that of the end-point bob)

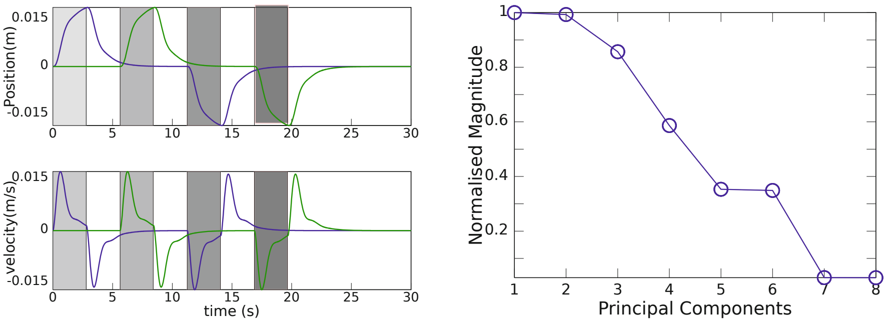
The simulator depicted in Fig.2b uses a linear approximation of the plant. The nonlinearities due to angles of force application are neglected as a simplification. The simulation implements the following model,

$$\begin{aligned} \ddot{x}_c &= -k_x x_c - b_x \dot{x}_c + \sum_{i=1}^{4} F_{m_i} cos(\theta_i), \\ \ddot{y}_c &= -k_y y_c - b_y \dot{y}_c + \sum_{i=1}^{4} F_{m_i} sin(\theta_i), \\ \dot{\alpha}_i &= \tau(u_i g - \alpha_i), \quad F_{m_i} = k_m \alpha_i, \end{aligned} \tag{8}$$

where, $k_{x,y}$ is the stiffness of restoring force to mean position, $b_{x,y}$, is the damping of the pendulum bob, $i \in [1,4]$ $\tau_{1 \cdots 4}$ are the time constants of the muscle (critically damped), $g_{1 \cdots 4}$ are the gains on the input signal, k_m is the muscle stiffness proportionality to its activation, and θ_i is the mounting angle of each of the spindle motors. Note that, x_c and y_c in this model are both state variables and should not be confused with the state x and output y of Eq.2. For this paper, the simulation constants were fixed as $k_{x,y} = 10$, $b_{x,y} = 5$, $\tau = 1$, $g = 5$, and $k_m = 5$ (currently being validated on the real robot platform). The model has dimensions 8 on state and 4 on input, corresponding to desired angular positions on DC motors with controllers of time constants τ and dc gain $g_{1 \cdots 4}$. The model was implemented in GNU Octave and integrated using the *ODE45* routine.

4.2 Results - Spontaneous Motor Activity and Dimensionality Reduction

A unit pulse input applied to each muscle sequentially to replicate the spontaneous motor activity in the form of single muscle twitches as shown in Fig.3a. The various state and output trajectories $y_s(t)$ and $x_s(t)$ respectively, were recorded and stored as a dataset and POD was used to reduce the dimensionality.



(a) (b)

Fig. 3. a) Single Muscle Twitching and output of the simulated robot and b) Principal components of the dataset $(y_s(t))$ -Scree Plot

The principal components are depicted in the scree plot in Fig.3b. The first k largest components were chosen to compute controllers with k motor primitives, where $1 < k \leq 4$, thus giving 3 types of controllers. Linear dynamical models of dimension k were then obtained by fitting in each case.

4.3 Results - Synthesised Motor Primitives and Task Performance

From the identified reduced dimensional linear state space, the primitives were synthesised using SVD for the cases of $k = 2, 3$, and 4 primitives. The computed primitives in each case are depicted in Table 1. The synthesised primitives are visualised by locating the resulting equilibrium points (at $\dot{x} = 0$) for a unit inputs applied to each of the reduced dimensional input C individually. Since the source system is linear, the equilibrium points obtained are unique and independent of the initial conditions as depicted in Fig. 4. Since any position in the Cartesian space can be obtained by using the right input C, the knowledge of these equilibrium points can be used to generalise to new points in the task space by using linear combinations similar to the biological case [8].

A reaching task was then computed using the controller form of Eq. 7 for a set of 3 controllers ($k = 2, 3, 4$), as shown in Fig. 5. In each case small offset errors result in steady state due to the quality of the obtained reduced dimensional model. This offset error could potentially be minimised if the model can be improved through subsequent stages of learning and adaptation. More complex

Table 1. Computed U^* for the cases of 2, 3, and 4 primitives

$k = 2$		$k = 3$			$k = 4$			
-0.53210	-0.49997	-0.56734	-0.41926	0.49501	-0.55065	-0.44407	-0.50324	0.49632
0.46796	-0.49313	0.56418	-0.56742	-0.34017	0.57413	-0.53848	0.34869	0.50874
0.54176	0.41663	0.43817	0.57504	0.48379	0.40476	0.59823	-0.47541	0.50227
-0.45208	0.57730	-0.40967	0.41423	-0.63655	-0.45091	0.39365	0.63178	0.49252

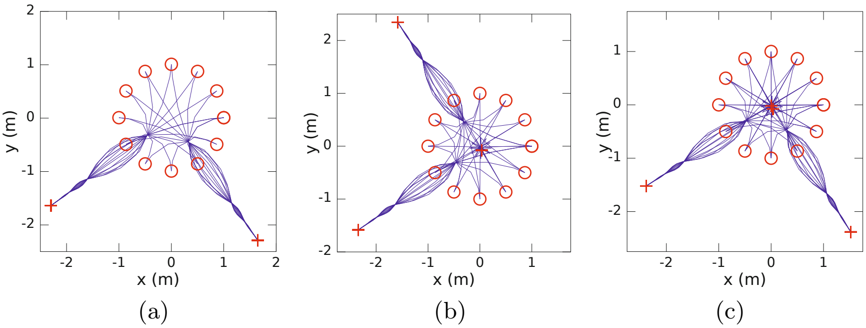


Fig. 4. Equilibrium positions of endpoint (*red* stars) while using (a) $k = 2$, and individual unit inputs in C i.e. $C_1 = 1, C_2 = 0$, and vice versa (b) $k = 3$, and individual unit inputs in C as before, and (c) $k = 4$, and individual unit inputs in C as before, the initial conditions (*red* circles) are chosen to lie in a circle about the center. The trajectories of the endpoints are in *blue* lines. In cases (b) and (c) the latter equilibrium points are are found to lie nearly at the origin of the workspace.

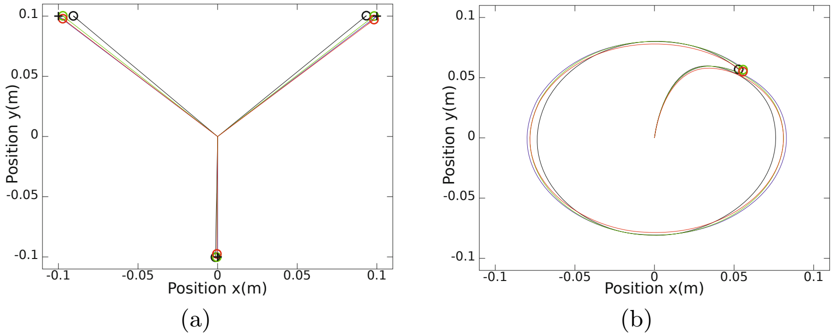


Fig. 5. Performance of the 3 controllers, $k = 2$ (*red*), $k = 3$ (*green*) and, $k = 4$(*black*) relative to an ideal controller (*blue*), in performing (a) reaching task in Cartesian Space to the positions $(0.1, 0.1)$, $(0, -0.1)$, and $(-0.1, -0.1)$, (b) continuous tracking task in Cartesian Space to a circle centered at $(0, 0)$ and diameter 0.15m. The trajectories obtained in each case are nearly identical.

desired trajectories using multiple waypoints can also be obtained using the controller as shown in Fig. 5.

5 Conclusions

This paper presented a motor primitive inspired architecture for reduced dimensional control of redundant compliant robots. Based on a biological model of motor primitives using model order reduction, considerations relevant to artificial system control were presented. A technique for self-organising a controller was presented, inspired by the concept of spontaneous motor activity. A reduced dimensional representation of the ensuing dataset was then used to synthesise motor primitives using SVD. The computed primitives were then utilised to compute the necessary control, across all of the inputs, for reaching fixed points in space. The proposed framework was tested on simulated version of a compliant redundant tendon driven robot platform. The preliminary simulation based results are promising and demonstrate the utility of the proposed technique for application to artificial systems. From an engineering viewpoint an extension of the work to the nonlinear systems such as kinematic chains is currently being carried out. An important consequence for biological systems arising as an extension of this work is an investigation on the relationship between dimensionality reduction and mechanical properties of biological systems.

References

1. Antoulas, A., Sorensen, D., Gugercin, S.: A survey of model reduction methods for large-scale systems. Contemporary Mathematics 280, 193–219 (2001)
2. Berniker, M., Jarc, A., Bizzi, E., Tresch, M.C.: Simplified and effective motor control based on muscle synergies to exploit musculoskeletal dynamics 106, 7601–7606 (2009)
3. Bernstein, N.: The Co-ordination and Regulation of Movements. Pergamo, Oxford (1967)
4. Bizzi, E., Mussa-Ivaldi, F.A., Giszter, S.: Computations underlying the execution of movement: a biological perspective. Science 253(5017), 287–291 (1991)
5. Blumberg, M.S., Lucas, D.E.: Dual mechanisms of twitching during sleep in neonatal rats. Behav. Neurosci. 108, 1196–1202 (1994)
6. Featherstone, R.: Rigid Body Dynamics Algorithms. Springer-Verlag New York, Inc., Secaucus (2007)
7. Latash, M.L.: Evolution of motor control: From reflexes and motor programs to the equilibrium-point hypothesis. Journal of Human Kinetics 19(19), 3–24 (2008)
8. Mussa-Ivaldi, F., Giszter, A., Bizzi, E.: Linear combination of primitives in vertebrate motor control. PNAS USA 91, 7534–7538 (1994)
9. Oja, E.: Simplified neuron model as a principal component analyzer. Journal of Mathematical Biology 15, 267–273 (1982)
10. Pfeifer, R., Lungarella, M., Iida, F.: Self-organization, embodiment, and biologically inspired robotics. Science 318(5853), 1088–1093 (2007)
11. Strang, G.: Linear Algebra and Its Applications. Brooks Cole (February 1988)
12. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
13. Todorov, E., Li, W., Pan, X.: From task parameters to motor synergies: A hierarchical framework for approximately-optimal control of redundant manipulators. Journal of Robotic Systems 22(11), 691–710 (2005)

Using Sensorimotor Contingencies for Prediction and Action Planning

Alexander Maye and Andreas K. Engel

University Medical Center Hamburg-Eppendorf
Dept. of Neurophysiology and Pathophysiology
Martinistr. 52, D-20246 Hamburg, Germany
{a.maye,ak.engel}@uke.de

Abstract. Sensorimotor contingency theory holds that the law-like relations between actions and contingent changes in the sensory signals constitute the basis for sensory experience and awareness in humans. These Sensory-Motor Contingencies (SMCs) are not only passively observed and recorded by the agent, but are actively exercised and used to control behavior. We have previously introduced a computational model of SMCs for robot control that employs a set of Markov models for the conditional probabilities of making sensory observations given an action. In this article we extend this model by showing how prediction and evaluation of future sensorimotor events can be achieved. We investigate this prediction and planning method in a scenario where the robot's actions do not take immediately effect, so that it has to plan ahead. Exploiting an action selection method that takes into account previous experiences, the robot learns to move in an energy-efficient, naturalistic manner and to avoid known obstacles. We also make a first step towards analyzing the robot's behavior in a dynamically changing environment.

1 Introduction

Sensorimotor Contingency Theory (SMCT, [8,6]) is an attractive alternative to conventional robot control architectures that rely on internal representations of the environment. It eliminates two of the main problems in current robotics, namely generating and maintaining internal models of the world. Instead it acknowledges the world as its own external representation, that can be probed and structured by actions. Sensory experience and perception are constituted by exercising Sensory-Motor Contingencies (SMCs), comprising previously learned knowledge of the structure of changes in sensory signals depending on the executed actions. Here the term exercising means that the agent does not only observe structures of sensorimotor coupling, but that its behavior is governed by this knowledge.

Recently we have introduced a computational model of SMCs [3]. It considers SMCs as probability distributions over pairs of actions and associated changes in sensory signals depending on a history of previous pairs of actions and observations. This approach can be formalized as a set of Markov models that take

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 106–116, 2012.

different history lengths of previous action-observation pairs into account. For controlling behavior the system keeps a record of the success of different actions in a given context, and when a similar context is encountered again, it knows the most appropriate next action or tries a new one.

In this article we would like to go a step beyond the control of the immediate sensorimotor interaction and introduce a simple method to generate predictions about sensorimotor events from known SMCs. This is an important function when the agent should have some form of sensory awareness or perception:

> For a creature (or a machine for that matter) to possess visual awareness, what is required is that, in addition to exercising the mastery of the relevant sensorimotor contingencies, it must make use of this exercise for the purposes of thought and planning. ([8], p. 944)

The main idea of our prediction method is to record information about the temporal order of the activation of SMCs, and to maintain it in a network of links between SMCs. The result is in principle a two-layer structure with sequences of action-observation pairs constituting SMCs, and sequences of SMCs forming a network that can be used for predictions.

In [4] we presented an application of our SMC model in a robot that by exploration learned the size of its confinement without using any distance sensors. In that study locomotion was controlled by setting directly the speed of the motors. This lead to abrupt, “robot-like” movements. Since the motor commands took immediate effect, a simple prediction about the best next action was sufficient. Here we use a motor controller that slowly accelerates the motors to the desired speed. On one hand this results in smooth and gentle movements, that have close resemblance with those of animals, and that significantly reduce wear and slip of the robot’s drive. On the other hand this requires the robot to have the ability to make predictions about the consequences on a longer term when selecting an action. To avoid a collision, for example, it would be too late to switch the motor command one time step before bumping into the object, since deceleration and acceleration in the opposite direction extend over several time steps. Therefore the robot needs the capability to plan ahead.

Two aspects of the proposed prediction method will be of interest. First, in addition to using previously observed SMC sequences, which corresponds to a mode of “remembering”, it is possible to arrange SMCs in different sequences that have not been encountered before, but still are compatible with the general context. This allows the system to exhibit a certain degree of “imagination”. And second, the accuracy and level of detail do not necessarily decrease with prediction depth, given that the environment is dynamically stable. At first glance this may seem to contradict our experience with the weather forecast, for example. But introspectively we know how it feels driving a Porsche [7] no matter whether we imagine driving it tomorrow or next week, and there is no loss of accuracy when we imagine the sequence of all things we will do tomorrow compared to just imagining a single activity.

2 Related Work

Compared to the number of robot studies on prediction and action planning that employ internal representations of the environment like, for example, in Simultaneous Localization and Mapping (SLAM), only few approaches exist that consider this problem from a sensorimotor perspective. An interesting model for learning delayed rewards using the environment as memory has been presented in [1]. The system described in [5] actively uses SMCs for planning trajectories. The approach employs artificial neural networks for learning forward and backward models of changes in the sensory signals depending on the robot's actions. A model for goal-oriented action planning based on the function of different brain areas is introduced in [2]. It recombines SMCs to generate action sequences in a similar manner like we will describe below.

For an autonomously acting robot it is not sufficient to only have the capability to entertain predictions about potential action sequences. Alternative courses of actions have to be evaluated, and an optimal behavior must be selected. In the field of reinforcement learning a large number of methods for evaluating action sequences and dealing with the credit assignment problem have been developed. Probably the most prominent algorithm is Q-learning [10] and its derivates. Apart from a value for each sensorimotor state, our method makes information about the observed frequency of this state, the likelihood of its realization, as well as the reliability of the prediction available. We therefore suggest an alternative method that takes this information into account.

3 Methods

3.1 A Markov Model of eSMCs

A detailed introduction to the basic idea of our model of SMCs has been given in [3]. Here we extend this model to facilitates prediction of sensorimotor events and action planning. While the agent executes an action a, sensory observations $o = [s_1 s_2 \dots s_S]$ from the S sensory channels are recorded. After each time step a new action-observation pair $ao(t)$ is available. These action-observation couplets are linked in a tree structure that reflects a finite history of experienced sensorimotor events (see Fig. 1). When a new action-observation pair is available, the respective node is activated. This means that the information stored in this node, basically a counter n and a value v, is updated, or that a new node is created. The key of each node is composed by concatenating the vectors encoding the action and the sensory features. At every time there is one active node at each level, and all active nodes share the same key given by the most recent action-observation pair. The active nodes are the roots for updating or creating the node on the next level upon arrival of a new action-observation pair. This way the path to each active node reflects histories of previous actions and sensory observations, with the length of this history corresponding to the level in the tree.

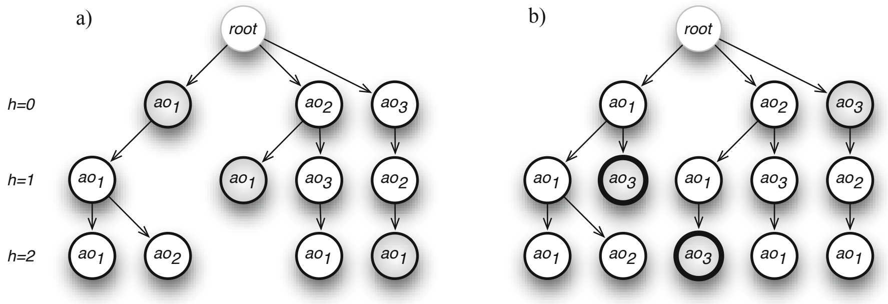


Fig. 1. Schema of the extended computational model of eSMCs. **a)**Active nodes (gray) when the action-observation pair ao_1 is encountered after ao_2 and ao_3. **b)** When ao_3 is encountered next, the corresponding node is activated at level $h = 0$ because this event has been seen before. Two new nodes are added (thick circles) representing event sequences ao_1ao_3 and $ao_2ao_1ao_3$. Each node carries information about the respective sequence, e.g. number of occurrence n and value v (not displayed).

It is easy to see that the probability of making a particular action-observation combination in the next time step $ao(t+1)$ conditional on a history of previous actions and observations can be computed at each node. Denoting the sequence of previous sensorimotor events by context $c = [ao(t)ao(t-1)\dots ao(t-h)]$, this conditional probability is given by $p(ao(t+1)|c) = n(aoc)/\sum_{i\in R} n(ic)$, where i runs over all keys of the successors R of the node addressed by path c at level h in the tree, and $n(aoc)$ is the number that the sequence $aoc = [ao(t+1)ao(t)ao(t-1)\dots ao(t-h)]$ of sensorimotor events has been observed. Therefore, each node represents the conditional probability distribution $P(ao(t+1)|ao(t), ao(t-1)\dots ao(t-h))$. We consider this probability distribution as an extended Sensory-Motor Contingency (eSMC) of the agent.

3.2 Generating Sensorimotor Predictions

When a match for the sequence of actions and observations that the agent has just experienced is found in the tree, the child nodes of the activated node can be used as a prediction of forthcoming sensorimotor events. Since the agent has experienced these longer sequences before, we consider this mode as one of remembering (Fig. 2a).

When no match is found for a given eSMC sequence, no matter if experienced or predicted, the oldest action-observation pairs are successively dropped until a match can be established. Since thereby eSMCs can get activated in a sequence that has never been experienced by the agent, we consider this mode as one of imagining. Longer predictions are generated by forward chaining, interleaving the two prediction modes as required.

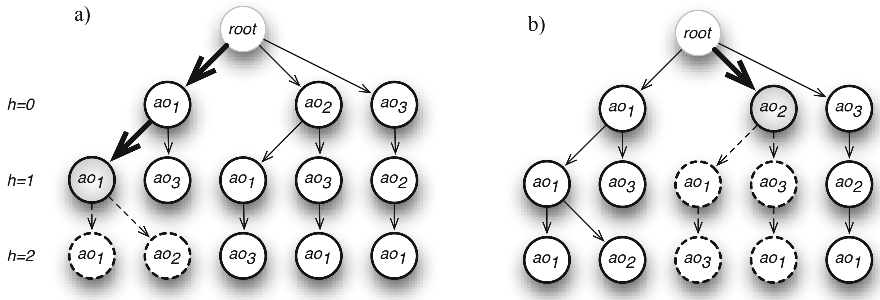


Fig. 2. a) Predicting future sensorimotor events by remembering previous sequences Supposed the agent has experienced the sequence of events ao_1ao_1. This sequence has a match at level $h = 1$. ao_1 and ao_2 are potential successors. **b)** Prediction by rearranging eSMCs in new combinations. For continuing the prediction of the branch $ao_1ao_1ao_2$, no match is found in the tree. The oldest events are successively discarded until a match is found again (ao_2). The resulting sequences are $ao_1ao_1ao_2ao_1ao_3$ and $ao_1ao_1ao_2ao_3ao_1$.

When matching a particular action-observation sequence in the tree of stored eSMCs, we suggest to search for the longest match, i.e. the one with the longest history length h. The longer this match is, the more information is used in identifying the current action context, and the more precise the predictions consequently will be.

3.3 Action Selection

The general idea for finding an optimal behavior is to combine the values associated with each action in a predicted sequence, and to select the sequence with the best value. Various methods for combining the values of future rewards for selecting optimal actions are used, but we would like to propose an approach that takes the specific information resulting from the prediction process into account.

The first information that we consider useful for evaluating action sequences is the average context size that was taken into account for generating each prediction. Basically this is the sum of the history lengths $h(t)$ of the eSMC used for generating the prediction for time step $t+1$. In terms of the eSMC tree this is the average path length used when iterating predictions. With H as the prediction depth or planning horizon this value is computed as $\bar{h} = \sum_{t=1}^{H} h(t)/H$. A large $\bar{h}$ indicates that large context sizes have been considered for the respective prediction, and it is regarded as more reliable, therefore. For each action sequence we keep only those predictions with a maximum $\bar{h}$.

In the next step the values associated with each predicted action-observation pair are combined into a single value for the respective sequence i. This is done by averaging over the values in each prediction step t, $\bar{v}_i = \sum_{t=1}^{H} v(t)/H$.

Now we have a single average value for each predicted sequence that in principle could be used to select the optimal behavior. However, since each action can result in different sensory outcomes, typically several versions for the same sequence of actions are generated. The last step is, therefore, to combine the predictions for one action sequence that differ in there course of sensory signals into an average expected value for this sequence. This is done by looking at the frequencies of encountering each action-observation pair in the predicted sequence, n. First we determine the minimum frequency n_{min} for each version i of a predicted sequence S of actions, $n_{min}(i) = \min_t n(t)$, and use this for computing a weighted average of individual values $\bar{v}_i$ for this sequence:

$$\bar{v} = \sum_{i \in S} \frac{n_{min}(i)}{\sum_{j \in S} n_{min}(j)} \bar{v}_i,$$

where i runs over all versions S of the same action sequence. This has the effect that versions of an action sequence with more frequently observed concomitant sensory features are weighted more strongly.

The action sequence with the best average value $\bar{v}$ is selected as candidate for execution. Finally we use the properly normalized value as a probability for executing the candidate action or an alternative. The lower the value of the candidate action, the more likely the alternative action is executed. This is a simple method to deal with the exploration-exploitation problem. The alternative action is the one with the least reliable prediction, i.e. the lowest $\bar{h}$.

4 Experimental Setting

4.1 Hardware

We implemented the model of eSMCs for controlling the Robotino® robot (Festo Didactic, Esslingen, Germany). It is equipped with an omnidirectional (Swedish) wheel drive, a webcam, 9 distance sensors, and a collision detector. For this study locomotion was restricted to forward and backward movements. Collisions are detected by a compressed air tube that is attached around the circular periphery, and that registers pressure changes. It does not yield directional information about the side of the collision. Readings of the instantaneous current consumption of the motors were used to detect whether the robot pushes against an obstacle that had not triggered the bumper. A custom-made accelerometer was attached to the chassis of the robot giving three-dimensional acceleration information. Camera and distance sensors were not used.

A major modification compared to the otherwise similar setup described in [4] is the motor control. Instead of setting the motors to constant speed v or $-v$, the motors are updated by $v(t+1) = v(t) + \max(v - v(t), \pm \Delta v_{max})$ for $v \lesseqgtr 0$. Here $\pm v$ is the forward/backward command output by the model, $v(t)$ the motor speed in the current time step, and Δv_{max} is a fixed step size for increasing or decreasing speed settings. We set $\Delta v_{max} = v$, so it takes two time steps to reverse the movement direction.

4.2 Robot Environment

The robot moves on a flat ground back and forth between two parallel walls. While the wall in the back of the robot triggered the bumper on collisions, the wall in the front had a protrusion that prevented further forward movement, but did not trigger the bumper[1] (Fig. 3). Consequently the robot could not rely on the bumper as a single sensory modality that flags collisions, but it had to learn the interaction between actions, and bumper and current signals to detect collisions instead. In addition it should learn to respond appropriately to collisions, and to move in a smooth and energy-efficient way.

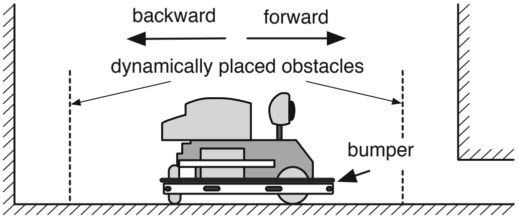


Fig. 3. Profile view of the spatial setup (adapted from [4])

To study the robot's adaptive behavior when the environment changes dynamically, we extended the setup by two obstacles at positions where the robot could not expect them, approximately 1/4 in front of each wall. When the robot reversed its movement direction at either wall, the obstacle in front of the opposite wall was pushed in the trajectory of the robot. Since the robot expected the wall to be further away, it reliably bumped into the obstacle, and the resulting behavior was analyzed.

4.3 Value System

Defining a value system is a method "to make an agent do something in the first place" [9]. Together with the eSMCs, their associated values are learned and used later to select actions that result in beneficial behavior. For each time step the value of the robot's current state was computed by a weighted average of signals from its sensors:

$$v = -bumper - \sum_{motors} 0.2 motor_{avg} - \sum_{motors} 0.2 motor_{inc} - 0.2 \max_{x,y,z}(|accel|)$$

[1] Both types of collisions had the side-effect of bringing the robot back to a perpendicular orientation between the two walls, that would otherwise be lost after moving several times back and forth due to the imprecision of the motors and the slip with the ground.

The bumper signal is 1 when a collision is detected and 0 otherwise. The motor current readings are averaged over each time step for each of the three motors ($motor_{avg}$). The difference of the average motor current in the last third and the first third of each time step ($motor_{inc}$) yields a signal for changing motor load. Finally, $accel = [-2 \ldots 2]$ indicates acceleration peaks, caused by a collision for example.

5 Results

In the first experiment the robot was free to explore eSMCs, and we were interested in the developing behavior. Executing the actions from the action selection schema (see Fig. 4), the robot moved between the endpoints of its confinement. Actions together with the resulting sensory signals built the eSMC tree structure, and information about the frequency of activation and the value for the agent were associated with each eSMC.

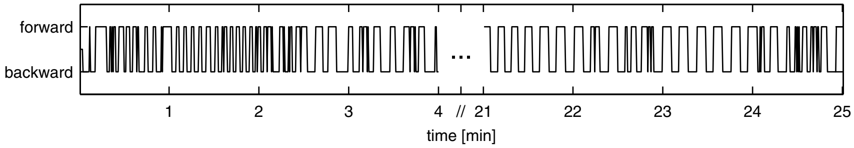


Fig. 4. Action sequences when the robot started learning eSMCs (1-4 minutes run time) and when behavior was guided by knowledge of eSMCs (after 21-25 minutes, same run)

With progressing knowledge about the sensorimotor laws the robot moved with straight transitions between the walls, and reversals of movement directions well before imminent collisions. This behavior minimized motor current consumption, acceleration peaks, and in particular collisions that triggered the bumper (see Fig. 5). Note that the robot uses the full extent of the confinement and not only the central part, since only in this way it can minimize motor currents and acceleration peaks.

The second experiment aimed at the question how the robot behaves with respect to unexpected changes in its environment. In continuation of the first experiment we put a new obstacle in the trajectory of the robot. Interestingly the behavior for the two collision types was markedly different (see Fig.6). While on rear collisions the robot simply switched to the opposite movement direction and silently moved away (left column), it apparently did not know how to handle front collisions. It kept bumping into the obstacle for some time, occasionally going back, but then pushing again in the obstacle. The obvious explanation for this behavioral difference is that during the initial learning period the robot has made experience with backward movements that triggered the bumper, but never with forward movements that did the same, since the protrusion in the

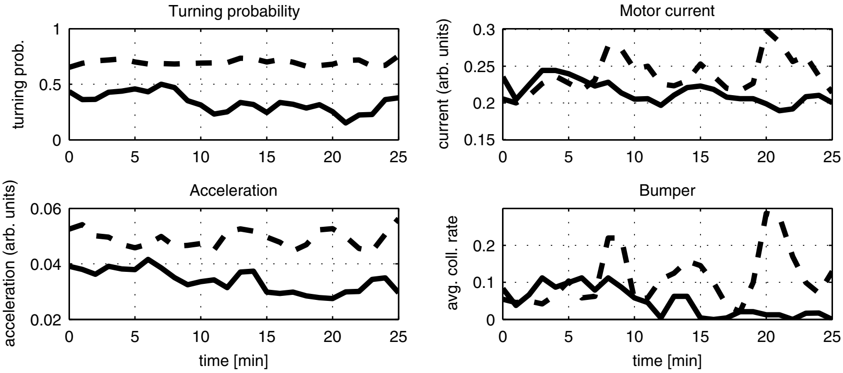


Fig. 5. Time course of the sensory signals in the three modalities that affect the internal value associated with each eSMC: acceleration, current consumption, and bumper. The solid line is from a run with the described action selection schema, the dashed line resulted from randomly switching between forward and backward actions. Turning probability is not considered for the internal value, but gives an idea how straight the robot moved (low values). All curves are smoothed with a moving average window of 4 minutes.

front wall spared the bumper. The experience of a bumper signal while moving forward is a new eSMC, therefore, and the robot starts to explore behavioral alternatives to deal with this situation.

The curves in Fig. 6 show that in both cases the collision came unexpectedly, because the history length of previous action-observation pairs that he robot was able to match with this collision dropped always to zero. For rear collisions, the eSMCs at the first level in the tree had information about the optimal action sequence to follow, and the context, or “awareness” of the situation quickly built up again. In contrast, there were no eSMCs for bumper triggering front collisions in the tree, and this situation called for exploration.

6 Discussion

The experimental results are interesting in multiple respects. First, the robot is able to learn how to avoid collisions with static obstacles without using any distal sensors like distance sensors or the camera. It “knows” that it can safely move in one direction for a certain number of time steps, but then better turns in the other direction. This can be interpreted as an understanding of the spatial range across that the robot acts. We would like to emphasize that this understanding does not build on an internal representation of this space, but exclusively on the learned eSMCs. Second, one should remember that an action resulting from the action selection schema can have very different effects on the sensory signals and the overt behavior of the robot, depending on the context of the previous actions. Current consumption and accelerations for the same action differ, for

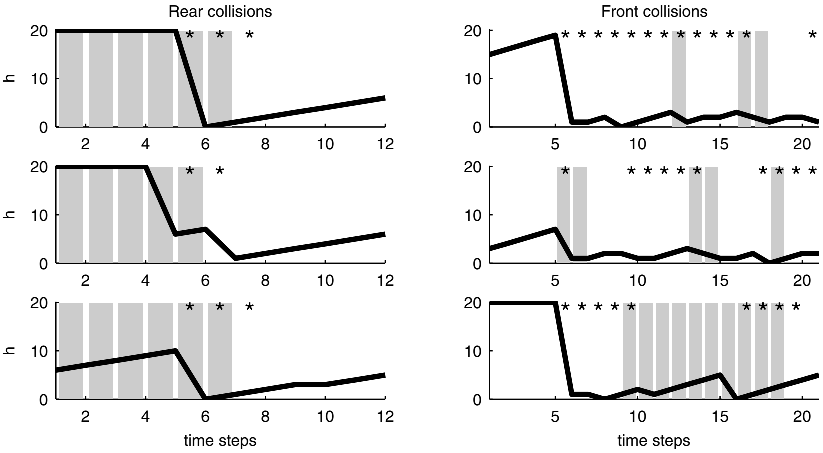


Fig. 6. Three examples for the behavior on rear (left column) and front collisions (right column). The background shade shows for each time step the selected action (backward-grey, forward-white). Asterisks mark time steps when the bumper was triggered. The curves display the maximum history length that was used for planning. Note that for reversing movement direction it takes two time steps for the motor controller until the robot actually moves into the new direction.

example, when the robot is accelerating, decelerating, or in uniform motion. This shows that sensorimotor knowledge has to account for the action context, and the proposed tree structure for eSMCs is a model that reflects this fact.

The second experiment shows that the robot is able to exploit its sensorimotor knowledge when the environment is changing. It also demonstrates that it automatically switches into an exploration mode when it encounters new eSMCs. This is in accordance with our view that there is no strict distinction between exploration and exploitation phases, as is sometimes found in robotic control architectures. In our rather cognitive approach the agent is exercising its sensorimotor knowledge and acquiring new eSMCs throughout life time.

An interesting question is how the approach presented in this study works when the agent has a larger action repertoire and lives in more complex environments. Then the time to sample the higher-dimensional space of actions and observations may become very long, and hence the agent may spend excessive time on exploration before showing reasonable behavior. Clearly a complete exploration of action-observation space in such scenarios is impossible, but also not necessary in our view. A small set of sensorimotor knowledge that locally optimizes the behavior of the robot may be sufficient to deploy the robot, and the proposed action selection schema will extend this knowledge towards the global optimum by exploring alternative action courses with a probability that is inversely proportional to the success of known actions. A second argument against excessive exploration phases is our observation that the ratio between eSMCs that an agent actually experiences throughout lifetime and all possible eSMCs is typically very small: In [3] the agent experienced only 0.05% of all possible eSMCs with history length 2, and in this study only about 0.02%. At the next history level (length 3) with $27 \cdot 10^6$ potential eSMCs, only 293 ($\approx 1 \cdot 10^{-7}$%) have been encountered in this study. This shows that the learning process is not dominated by the sheer size of the state space, but rather by the actual situatedness of the agent. This is why we anticipate similar results in more complex settings, and corresponding experiments are underway.

Technically optimal implementations may be required when extending our approach to a larger action repertoire of the robot or a more complex environment, though. For example, value propagation and graph searching techniques can be used to reduce the time complexity when searching for promising action sequences. Extending this conceptual study to real-world scenarios will be the focus of our future work.

Acknowledgment. This work was supported by the Sino-German research training group CINACS, DFG GRK 1247, www.cinacs.org, and by the EU project 'Extending sensorimotor contingencies to cognition - eSMCs', IST-270212, esmcs.eu.

References

1. Bovet, S., Pfeifer, R.: Emergence of delayed reward learning from sensorimotor coordination. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2272–2277 (August 2005)
2. Duff, A., Sanchez-Fibla, M., Verschure, P.F.M.J.: A biologically based model for the integration of sensory-motor contingencies in rules and plans: A prefrontal cortex based extension of the distributed adaptive control architecture. Brain Research Bulletin 85(5), 289–304 (2011)
3. Maye, A., Engel, A.K.: A discrete computational model of sensorimotor contingencies for object perception and control of behavior. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 3810–3815. IEEE (May 2011)
4. Maye, A., Engel, A.K.: Time Scales of Sensorimotor Contingencies. In: Zhang, H., Hussain, A., Liu, D., Wang, Z. (eds.) BICS 2012. LNCS (LNAI), vol. 7366, pp. 240–249. Springer, Heidelberg (2012)
5. Möller, R., Schenck, W.: Bootstrapping cognition from behavior – a computerized thought experiment. Cognitive Science 32(3), 504–542 (2008)
6. Noë, A.: Action in perception. MIT Press (2004)
7. O'Regan, J.K., Noë, A.: What it is like to see: A sensorimotor theory of perceptual experience. Synthese 129(1), 79–103 (2001); ArticleType: research-article/Issue Title: Perception, Action and Consciousness/Full publication date: October 2001/Copyright © 2001 Springer
8. O'Regan, J.K., Noë, A.: A sensorimotor account of vision and visual consciousness. Behavioral and Brain Sciences 24, 939–1031 (2001)
9. Pfeifer, R., Scheier, C.: Understanding Intelligence. MIT Press (September 2001)
10. Watkins, C.J.C.H., Dayan, P.: Technical note on Q-learning. Machine Learning 8, 279–292 (1992)

A Bio-inspired Control System for a Wearable Human-Machine Interface

Michele Folgheraiter[1], Mathias Jordan[1], Jan Albiez[1], and Frank Kirchner[1,2]

[1] German Research Center for Artificial Intelligence (DFKI),
Robotics Innovation Center Robert-Hooke-Str. 5, D-28359 Bremen, Germany
{michele.folgheraiter,mathias.jordan,jan.albiez,frank.kirchner}@dfki.de
[2] University of Bremen, Robotics Lab. Robert-Hooke-Str. 5,
D-28359 Bremen, Germany

Abstract. A control system is presented that integrates bio-inspired with classical control techniques to govern the forearm joint of a wearable haptic interface. The neural circuit is based on the architecture of the human segmental reflexes, and the neurons are represented by the combination of a first order linear differential equation and a sigmoid or a piecewise-linear activation function. Due to a long term adaptive mechanism that considers the state of the interface and the interaction force with the user, the stiffness of the joint is regulated according to the particular motion and task at hand. Experimental results showed that the proposed control architecture is able to improve the interface performances in terms of responsiveness, as well as to implement a safety behavior that intervenes in case of harmful external forces.

Keywords: Bio-inspired Control System, Human-Machine Interaction, Wearable Exoskeletons, Stiffness Adaptation.

1 Introduction

Human-machine interfaces are becoming more and more important in our society. They range from simple graphical menus which mainly involve the vision system, to more complex ones that consider also haptic feedbacks or even biosignals (e.g. EEG, EMG etc.) in order to control the behavior of the machine. When the artifact to interact with is represented by a robotics system, special attention should be paid in order to guarantee the comfort, and even more importantly, the safety of the human. Although most of the robotic systems do not require a strong interaction with the operator, which is mainly involved in supervising the machine, there are different type of robotic devices that are intended to heavily interact with humans. Among these artificial exoskeletons can be used both to support/enhance the body during daily activities, or to operate remote robotics systems.

The control strategies of most of the state-of-the-art haptic interfaces are based on approaches that were mainly developed for traditional robotic systems. Widely used are e.g. the admittance and impedance control. In an admittance

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 117–126, 2012.

controller the interaction force with the user is measured and indirectly regulated via a position control loop. In an impedance controller, on the contrary, the interaction force is directly regulated according to a reference calculated via a measured position and a defined world model.

Although such control techniques were successfully applied to artificial exoskeletons [9], their capability to adapt to the user's behavior is rather limited. Furthermore, due to their artificial nature, the integration of biosignals results sometimes unintuitive.

In the field of biorobotic a big effort was already dedicated to develop neurocontrollers based on models [1,3] of the human nervous system. In [2] and [5], e.g., physiologically analogous artificial neural networks were applied to control the motion and the stiffness of anthropomorphic manipulation systems. However, to the best of the authors knowledge, up to now only little effort was invested to transfer such control techniques also to wearable exoskeletons. It is in the authors point of view that a better integration between the interface and the human can be achieved: Enhancing the information exchange between the two systems. Reproducing in the machine the same behaviors and control strategies that are found in the biological nervous system.

The first problem is not directly tackled in this work. However, it is worth saying that the coupling between the human and the interface can be in general achieved in two different ways: via direct connections with the nervous system using surgery implanted electrodes , or with external connections furnished with superficial electrodes (e.g. EEG, EMG) like in [7] or sensors (e.g. force transducers, temperature sensors etc.). This work proposes a control approach that integrates classical and bio-inspired techniques to govern the forearm joint of a wearable arm exoskeleton. In particular an artificial neural circuit, that partially resembles the structure of the myotatic and inverse-myotatic segmental reflexes, is used to modulate the joint stiffness, implement a safety mechanism that limits the maximum interaction force, and modulate the actuator torque in order to fastness the joint responses. The paper is structured as follows: Sect. 2 introduces the fundamentals of the human segmental reflexes, Sect. 3 describes the architecture of the control system, Sect. 4 presents the experimental results, and finally Sect. 4 draws the conclusions.

2 The Human Segmental Reflexes

Reflexes are involuntary control mechanisms intended to regulate the behavior of the human muscles and articulations [8]. Although they are able to operate autonomously, their activity can be modulated by neurons located in the motor and in the somatosensory cortex. Segmental reflex circuits have efferent connections that propagate signals toward the motoneurons, and afferent connections (e.g *Ia* and *Ib* afferent) that receive the information from muscles and pain receptors.

There are mainly three kinds of segmental reflexes: the myotatic-reflex (or stretch reflex), the inverse-myotatic reflex (or Golgi tendon organ reflex) and the flexion-withdrawal reflex (or nociceptive reflex). In this work only the first two reflexes are considered and therefore briefly introduced in the following.

The Myotatic Reflex coordinates agonist and antagonist muscles in order to compensate for noise forces that alter the position of the joint. In the neural circuit (Fig. 1 a) *Ia* afferent fibers (from primary spindles) form excitatory synapses with the *α-motoneurons* that innervate the synergic muscle. The *Ia* afferents innervate also inhibitory interneurons whose axons project to *γ-motoneurons* that control the antagonist muscle. When the muscle is stretched *Ia* afferents are activated and this increases the firing rate of the corresponding *α-motoneuron* that in turn produces inhibition of motoneurons supplying antagonist muscle. The myotatic-reflex has an important role in regulating the normal muscle tone. Gamma motoneurons control intrafusal muscle fibers; contraction of these increases the sensitivity of the stretch receptors. This turns into a stronger response of the *Ia* afferent fibers to a muscle stretch. Gamma activity can also be controlled by descending pathways originating from superior neural centers.

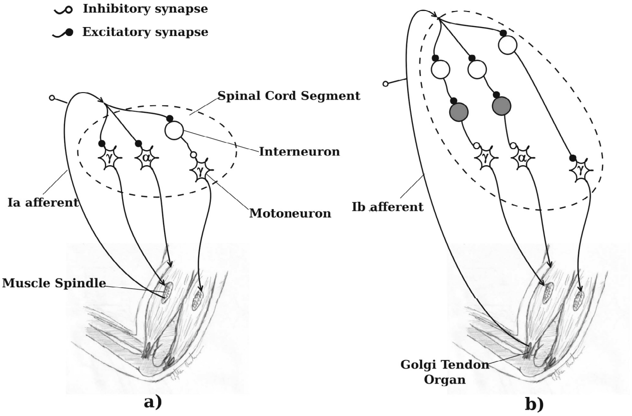


Fig. 1. The natural circuit: a) Myotatic Reflex Circuit; b) Inverse Myotatic Reflex Circuit

The Inverse Myotatic Reflex has the main function to avoid damages to muscles and tendons when they are overloaded by unexpected forces. When the muscle is rapidly stretched, the tension on the Golgi tendon organ increases, and as well as the *Ib* afferent fiber activity. *Ib* afferent fibers, on their turn, innervate interneurons that inhibit the *α-motoneuron* and the *γ-motoneuron* of the synergic muscles (Fig. 1 b) and excite the *γ-motoneuron* that controls the antagonist muscles. This coordinated control action decreases rapidly the stiffness of the joint and lowers, in this way, also the load on tendons and articulations.

3 The Bio-inspired Control Scheme

The system to be controlled is represented by a wearable exoskeleton intended to operate remote robotic systems. The interface (see Picture 2) consists of 9DOFs, six of which are hydraulically actuated and two are only measured in position. Thanks to the presence of three contact points with the limb, namely shoulder, upper arm, and forearm, the interface is capable of displaying complex force patterns to the user, and thus enhancing the interaction with the remote environment(more details in [4]) .

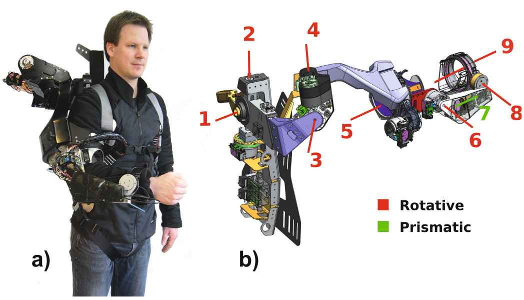


Fig. 2. a) The VI-Bot exoskeleton developed at the Robotics Innovation Center, DFKI Germany; b) CAD representation of the interface with indicated the position and the kind of the available degrees of freedom

The present paper particularly deals with the control system that governs the elbow and forearm joints of the exoskeleton (respectively joints 6 and 7 in Fig. 2b). The proposed control structure is based on a combination of bio-inspired and classical control techniques. In comparison with [9] and [7] where only classical force control schemes are used to regulate the exoskeleton's joints, our control architecture, by integrating artificial neural circuits, has the potentiality to better harmonize with the human nervous system enhancing in this way the performances and the safety features of the interface. The actuation mechanism is composed of a rotational DOF, in proximity of the user's elbow, and a prismatic one located along the forearm (3). This specific configuration is intended to compensate the misalignments between the exoskeleton joints and the user's articulations and therefore to avoid undesired interaction forces with the user limb. For simplicity, the parallel kinematic structure here is depicted as a triangle where the vertexes are represented by passive rotational joints. Specifically, vertexes 1 and 2 identify the hydraulic actuator, and vertexes 2 and 3 constitute the extremities of the pneumatic spring (pneumatic actuator). The contraction of the pneumatic spring has the effect to shorten one edge of the triangle and therefore to decrease the distance between the exoskeleton elbow joint and the wrist connection. The hydraulic joint is controlled in torque via a classical PID controller in anti-windup configuration ("Low level Torque Control" Fig. 3). The torque reference is settled by a force control module that implements different functions, like gravity compensation and force feedback generation. Furthermore its value can be modulated by biosignals, e.g. EEG and EMG can be used to predict/detect the arm motion and therefore to prepare/enhance the exoskeleton movement. The neural circuit is composed by six neurons and is responsible for actively controlling the stiffness of the pneumatic spring when interacting with the user. If for some reason, the forearm performs an unexpected movement, a safety mechanism is activated to avoid dangerous forces applied to the user's arm. The inputs of the neural circuit are the signals coming from the sensory system of the exoskeleton, while the outputs the signals that effect the behavior of the two actuators. Each neuron in the circuit is represented by a first order differential equation (Eq. 1) which is based on Grossberg model (a detailed description of this model together with the conditions for the system stability can be found in [6]).

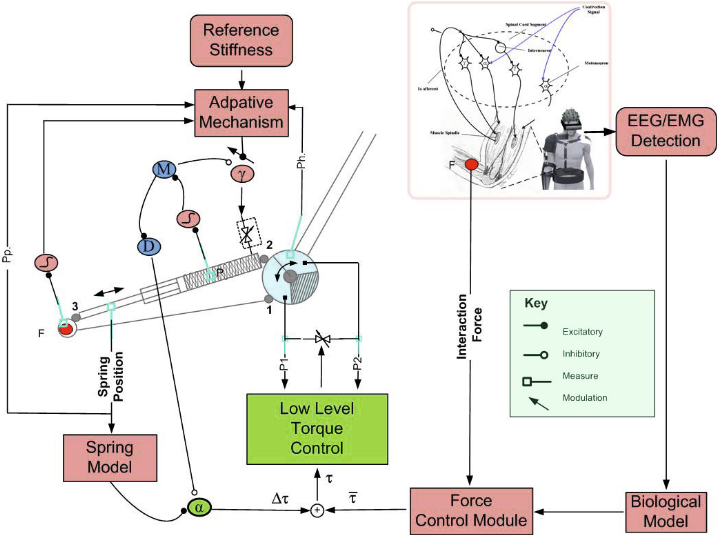


Fig. 3. The implemented control strategy for the parallel forearm structure. Each neuron has excitatory and/or inhibitory inputs represented in Eq. 1 by the variables I_{exc_k} and I_{inh_k} respectively.

$$T \cdot \dot{S}_j = (A - S_j) \cdot \sum_{k=1}^{n} W_{k+} \cdot I_{\text{exc}_k} - (B + S_j) \cdot \sum_{k=1}^{m} W_{k-} \cdot I_{\text{inh}_k} - S_j \cdot Fc \quad (1)$$

In Eq. 1 S_j represents the potential of the neuron, T the time constant that defines the dynamics of the neuron, $W_{k+}, W_{k-} \in [-1, 1]$ the k^{th} synapse that modulates the corresponding input, Fc is a forgetting constant, A and B are respectively the maximum and the minimum values for the potential, and I_{exc_k} and I_{inh_k} are the k^{th} excitatory and inhibitory inputs for the neuron, respectively.

Two kinds of activation functions are used according to the function of the neuron: a piecewise-linear function or a sigmoid function Eq. 2.

$$\mathrm{th}(x) = \begin{cases} x & \text{if } -1 \le x \le 1 \\ -1 & \text{if } x < -1 \\ 1 & \text{if } x > 1 \end{cases} \quad , \ \mathrm{sig}(x) = \frac{1}{1+\exp(a-bx)} \tag{2}$$

The behavior of each neuron's membrane is shaped by adjusting the time constant T and the forgetting constant Fc. In particular the smaller is T the higher is the frequency response of the neuron and vice versa. The parameters of the sigmoid activation function a and b were calculated in order to shape the function in a way to increase or reduce the distance between the saturation zone and the region where the input is highly attenuated.

In the bottom part of the artificial neural circuit (Fig. 3) the *α-motoneuron* is receiving as excitatory input the force signal produced by the pneumatic spring, which is calculated by the static model of the device (more details about the model in [4]) and the measured contraction. The neuron also receives a secondary inhibitory input signal that is originating from a *threshold-neuron*. This artificial cell has the function to monitor the pressure P inside the pneumatic spring and to increase rapidly its potential when this pressure overcomes a defined threshold. In a normal working condition the *α-motoneuron* is effecting the reference torque calculated by the "Force Control Module" (which e.g. compensates for gravity, calculates the proper force feedback, etc.), and its activity is directly related with the level of compression of the pneumatic spring. This has the effect to enhance the elbow movement according to the will of the user and therefore to mimic the behavior of the human muscle-spindle in regulating the contraction of the muscle according to an external load. In the upper part of the circuit the *γ-motoneuron* activity regulates the stiffness of the pneumatic spring. It receives an excitatory input from the "Stiffness Controller" and an inhibitory input that is originating from the *threshold-neuron*. When the force acting on the forearm is overcoming a defined limit, the safety mechanism comes into play. The high activity of the *threshold-neuron* has the effect to decrease the potential of both *gamma* and *alpha* motoneurons and therefore to reduce the pressure in the pneumatic spring and to inhibit the effect of torque modulation respectively. The *memory-neuron* (in Fig. 3 identified by the letter M), that is located between the *threshold-neuron* and the *γ-motoneuron*, has the function to prolong the interval this inhibition persists. Furthermore, to avoid a rapid rise of the pressure when the external force is removed from the pneumatic spring, a *damper-neuron* with long latency time (in Fig. 3 identified by the letter D) is located between the *memory-neuron* and the *α-motoneuron*.

3.1 Stiffness Adaptation Mechanism

The stiffness of the human muscular system is mainly regulated by the central nervous system. According to the task at hand and the past experiences in performing it, by learning, the brain is able to increase the movements' smoothness and at the same time to decrease the muscle stiffness in order to improve the energy efficiency [8]. In the proposed control scheme it is assumed that a middle level control strategy is defined to implement this regulation. However, an additional adaptation mechanism was also implemented in the low level neural circuit. More specifically the reference stiffness signal is modulated by a synapse that changes in the range [0 1] (for the pneumatic spring 0 corresponds to zero stiffness and 1 to the full stiffness set-point furnished by the middle level controller) according to Eq. 3,

$$T_a \cdot \dot{w}_s = \eta \cdot (P_H \cdot P_P) \cdot (1 - w_s) - F_c \cdot w_s \tag{3}$$

where P_H and P_P are the position of the hydraulic actuator and the pneumatic spring respectively, both normalized in the range [0 1] where zero means no contraction for the pneumatic spring and complete extension for the hydraulic actuator (elbow complete extended), F_c is a forgetting constant, and η is a learning constant that modulates the adaptation mechanism, whose value depends on interaction force F between the exoskeleton and the forearm according to Eq. 4.

$$\eta = \begin{cases} 0 & \text{if } F \leq F_{Th} \\ \overline{\eta} & \text{if } F > F_{Th} \end{cases} \tag{4}$$

Due to the proposed mechanism the synapse efficiency increases whenever a rapid movement occurs that makes the interaction force to overcome the threshold F_{Th} and at the same time a correlation exists between the movement of the pneumatic spring and the hydraulic actuator. This, when the reference stiffness is not zero, has the effect to increase both the pressure in the pneumatic-spring, in order to make the joint more stiff, and the excitation of the *α-motoneuron* due to the same amount of spring compression. In other words, this gives the joint a lower stiffness, lower precision, and lower reactivity during slow movements, and a high stiffness, precision and reactivity in the opposite case.

3.2 The Artificial Control Circuit and Analogies with Is Natural Counterpart

In the natural alpha-gamma loop (Fig. 1) the gamma motoneuron sets an equilibrium position for the muscle-spindle. Due to the fact that the external fibers of the muscle-spindle are connected with the muscle's fibers, a change in the load will effect both the muscle and muscle-spindle lengths. This in turns will alter the activity of the *α-motoneuron* that will change the contraction of the muscle in order to restore the original equilibrium position.

In our architecture the function of the spindle is replaced by a position sensor that measures the length of the pneumatic spring and effects the force in the hydraulic actuator; therefore we have a serial configuration instead of a parallel one. The sensor that measures the operational pressure of the pneumatic spring can be related instead to the Golgi tendon organ. This receptor is located in the terminal part of the muscle, and detects the force the muscle applies via the tendon to the bone.

Furthermore, the artificial synapse that brings the reference stiffness to the *γ-motoneuron* is adaptive. This somehow mimics the behavior that in the human reflex contracts the muscle spindle to increase or decrease its sensitivity when discrepancies exist between the intended and actual muscle length.

4 Experimental Results

The control architecture was first implemented and tested in Matlab-Simulink environment, afterwards compiled using the *RapidSTM*32 tool-chain, and finally flashed on a dedicated electronic board equipped with an ARM Cortex STM32f103VE micro-controller. The integration of the differential equations representing each neuron was carried out in a discrete way with a time step of $10ms$ which is a good trade-off between precision and computational workload.

Table 1 reports the chosen parameters for the neurons present in the circuit. Note that the *alpha* and *gamma* motoneurons are characterized by smaller time constants T allowing these cells to follow the inputs with a faster dynamics than the other neurons. The *M-neuron*, in comparisons with the other neurons, has one-order smaller value for the forgetting constant that lets the artificial cell to keep its potential for a longer time and therefore "to remember" its input values.

Table 1. The principal parameters of the neurons present in the circuit. The neurons that miss the constants a and b have a piecewise-linear activation function instead of sigmoidal one. W_{i+} and W_{i-} stand for excitatory and inhibitory synapses respectively.

Neuron	T	Fc	a	b	W_{1+}	W_{2-}
threshold-neuron	0.1	0.1	546	678	0.6	-
α-motoneuron	0.01	0.1	-	-	1	1
γ-motoneuron	0.01	0.1	-	-	0.6	1
M-neuron	0.1	0.01	341	678	1	-
D-neuron	0.1	0.1	-	-	0.9	-

The behavior of the control system was tested directly on the elbow and forearm joints of the exoskeleton. In order to prove the efficacy of the safety mechanism an external force was suddenly applied to the pneumatic spring and the output of the *α-motoneuron* monitored. Such case represents an exceptional operative condition for the haptic interface where the compression is generated by the misalignment between the user's articulation and the exoskeleton joints. All the sensory signals were normalized in the range [0 1], this to be compatible with the neurons' inputs. Figure 4a (first graph) reports the change of the pressure (9^{th} second) in the pneumatic spring due to the force applied.

How it is possible to notice, when the pressure overcomes a defined threshold, the *threshold-neuron* increases rapidly its activity and as a consequence also the *memory-neuron* and the *damper-neuron* potentials increase. Due to the fact that *threshold-neuron* output inhibits the *γ-motoneuron*, the pressure in the pneumatic spring decreases along with the interaction force between the exoskeleton

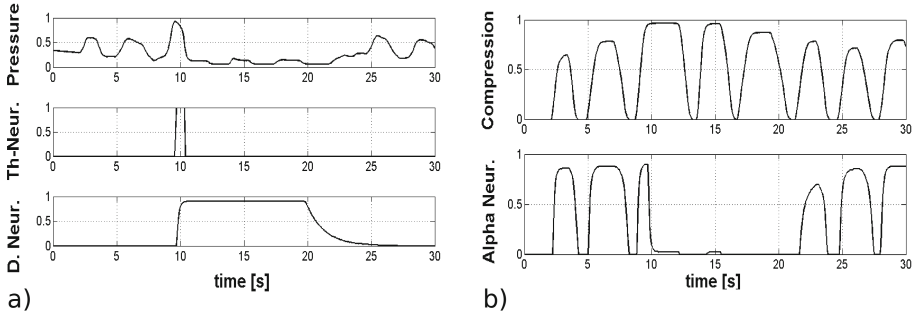


Fig. 4. a) How the pressure increase activates the safety mechanism; b) How the inhibition mechanism modulates the effect of the *α-motoneuron* on the joint torque

and the user limb. The same inhibition signal is reaching also the *α-motoneuron* thanks to the *damper-neuron* (Fig. 4b). This is needed to decrease the influence that the motoneuron has on the reference torque of the hydraulic actuator and therefore to avoid that an unpredictable movement of the pneumatic spring generates big forces on the elbow joint of the human limb.

The plots in Fig. 5a show how, in a normal working condition, the *α-motoneuron* activity has the effect to increase the reactivity of the joint. How it is possible to notice, when the *α-motoneuron* is not inhibited the velocity in average is increased of 30%.

The synapse modulation, that allows the regulation of the joint's stiffness, is presented in Fig. 5b. The parameters chosen for Eq. 3 were $T_a = 0.1$, $\overline{\eta} = 0.1$, $F_c = 0.005$, and $F_{Th} = 0.2$. The forgetting constant was chosen twenty times smaller than the learning constant to stabilize the behavior and to limit the energy consumption needed to change the stiffness.

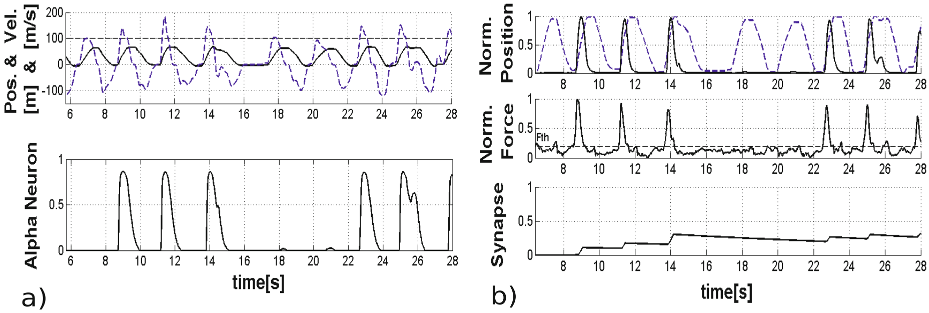


Fig. 5. a) Increasing of the joint's velocity (first plot, dashed line) due to the influence of the *α-motoneuron* (second plot); b) Synapse adaptation (third plot) due to the correlation between the position of the pneumatic spring (first plot dashed line) and the hydraulic actuator (first plot continuous line)

5 Conclusions

In this paper an hybrid control system that combines bio-inspired and classical control techniques is presented. In particular a neural circuit based on the architecture of the human segmental reflexed is integrated to influence directly the control action of a torque-regulated hydraulic joint. This special architecture allows to integrate more easily biosignals, and aims at improving the integration between the machine and the human. The neural controller, thanks to the presence of specific threshold-neurons, is able to detect when the interaction force between the user and the interfaces is overcoming dangerous values, and by an inhibition mechanism to lover the pressure inside the pneumatic spring. The displacement of this actuation element, in normal working condition, has also the effect to speed up the joint motion, improving in this way the user experience in interacting with the exoskeleton. Finally an adaptation mechanism was introduce to modulate the joint stiffness and to adjust the behavior of the interface according to the specific motion and usage.

Acknowledgments. This work was supported by the German Bundesministerium für Bildung und Forschung (BMBF, grant FKZ 01IW10001).

References

1. Bullock, D., Grossberg, S.: VITE and FLETE: neural modules for trajectory formation and postural control. In: Hersberger, W. (ed.) Volitional Control, pp. 253–297. Elsevier Science Publishers (1989)
2. Chou, C.P., Hannaford, B.: Study of human forearm posture maintenance with a physiologically based robotic arm and spinal level neural controller. Biological Cybernetics 76(4), 285–298 (1997)
3. Cisek, P., Grossberg, S., Bullock, D.: A cortico-spinal model of reaching and proprioception under multiple task constraints. Journal of Cognitive Neuroscience 10(4), 425–444 (1998)
4. Folgheraiter, M., de Gea, J., Bongardt, B., Albiez, J., Kirchner, F.: Bio-inspired control of an arm exoskeleton joint with active-compliant actuation system. Applied Bionics and Biomechanics 6(2), 193–204 (2009)
5. Folgheraiter, M., Gini, G.: Human-like reflex control for an artificial hand. BioSystem Journal 76(1-3), 65–74 (2004)
6. Grossberg, S.: Nonlinear neural networks: Principles, mechanisms, and architectures. Neural Networks 1(1), 17–61 (1988)
7. Kawamoto, H., Lee, S., Kanbe, S., Sankai, Y.: Power assist method for hal-3 using emg-based feedback controller. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1648–1653 (October 2003)
8. Steward, O.: Functional Neuroscience. Springer (2000)
9. Veneman, J., Kruidhof, R., Hekman, E., Ekkelenkamp, R., Van Asseldonk, E., van der Kooij, H.: Design and evaluation of the lopes exoskeleton robot for interactive gait rehabilitation. IEEE Transactions on Neural Systems and Rehabilitation Engineering 15(3), 379–386 (2007)

A Finite Element Method of Electric Image in Weakly Electric Fish

Sejoon Ahn and DaeEun Kim

Biological Cybernetics Lab.
School of Electrical and Electronic Engineering
Yonsei University
Shinchon, Seoul, 120-749, South Korea
{ahn.sj,daeeun}@yonsei.ac.kr

Abstract. Weakly electric fish can generate an electric field with their electric organ (EO). Through 14,000 electroreceptors distributed on their skin, they can sense the electric field perturbation induced by a nearby prey or object. Many researchers have studied to reveal the mechanism of electrolocalization. Simulations are typically based on an analytical model which represents the EO as a set of charge, or a model based on finite-element method (FEM) in a 2-dimensional space. In this paper, we show a 3-dimensional FEM model to test the electric perturbation of various shapes of objects. Using this model, we show that a measure of caudal relative slope or tail-side half width at half maximum can estimate the lateral distance of a target object regardless of its size, shape and rostrocaudal position.

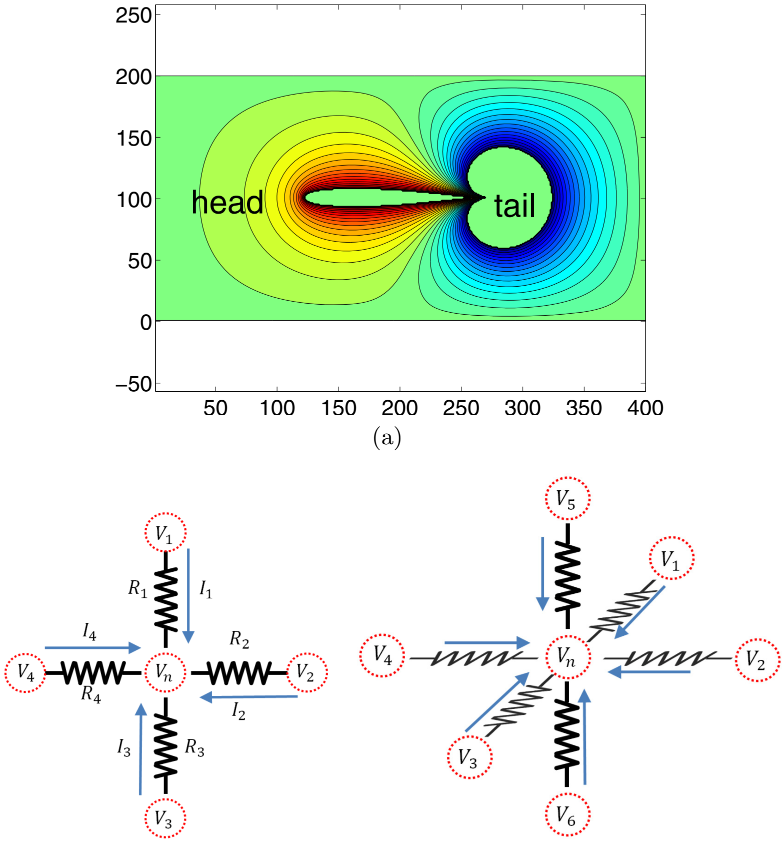
Keywords: electrolocalization, weakly electric fish, finite element method, relative slope, THWHM.

1 Introduction

Weakly electric fish are able to detect nearby objects in the dark where visual cues are absent. They have a special electric organ which can produce electric potential which are called eletric organ discharge (EOD). Weakly electric fish are able to measure the perturbation of an electric field caused by an object or a prey. In order to understand the mechanism of how the electric fish can sense a target object, an analysis of electric discharge and its effect with a target object has been done. von der Emde et al. [8] did experiments with a pulse type fish, *Gnathonemus petersii*, and has proposed a measure called 'slope to amplitude ratio' that an electric fish might use to measure the distance of a near by object [8].

Chen et al. [2] showed that the electric field of weakly electric fish could be modeled with distributed charges. Their model of electric field is close to real biological data. Electric field perturbation induced by a sphere object near the electric fish could also be calculated by an analytical equation derived by Rasnow [4]. The analytic equation can explain many ideas related to the electrolocation

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 127–135, 2012.



(a)

(b)

(c)

Fig. 1. FEM model (a) electric field generated by an electric fish; the contour lines represent the equipotential lines of the electric field generated by the electric fish (b) a typical FEM model designed in 2-dimension slave; there will be 4 resistors surrounding each element (c) the FEM model designed in 3-dimension space; this is our current model

process of weakly electric fish [5–7]. Through the theoretical model, it is reported that even tail-bending movements can be involved with electrolocation of a target object [5]. However, the analytic model of electric field has its limitation on modelling the field perturbation of multiple objects [7].

The analytical model [2, 4–7] can be applied to a single sphere object. It also assumes that the electric field passing through a target object is constant [4], and it has a simple model over the relative resistance on the internal body of a fish, the skin, and the water. The analytic model can only be applied to a sphere object, not cubes or other shapes of objects. As an alternative, a finite element method can be tested over various shapes of objects or multiple objects. A 2-dimensional FEM model has been used to find the electrosensing property of weakly electric fish [3], but it is not a realistic model for electric fish.

In this paper, we test a 3-dimensional FEM model of electrolocation procedure of weakly electric fish. Our model has an advantage over the previous analytical models; the electric field perturbation can be simulated even for an arbitrary shape of target objects. As an electrolocation measure, the relative slope or full-width at half-maximum (FWHM) can be used [1, 2, 6, 7]. We found that the caudal slope or tail-side half-width at half-maximum (THWHM) in an electric image provides a cue to estimate the lateral distance of a target object. In this paper, we will see how much those measures will be robust for sphere or cube objects at different distances with the 3-dimensional FEM model.

2 Method

2.1 3-Dimensional Model of Electric Fish

The electric organ can be modeled as a set of distributed poles. [2, 6, 7].

$$V(\overrightarrow{x}) = \sum_{i=1}^{m} \frac{q/m}{|\overrightarrow{x} - \overrightarrow{x}_p^i|} - \frac{q}{|\overrightarrow{x} - \overrightarrow{x}_n|} \tag{1}$$

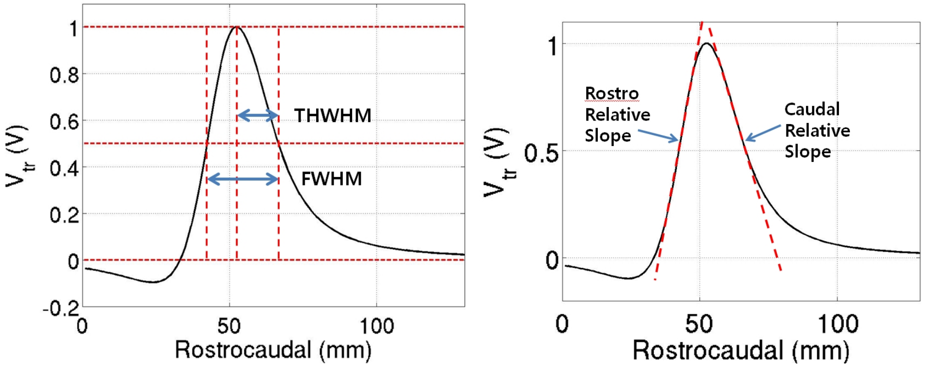
where $\overrightarrow{x}$ indicates an arbitrary point in space represented as a vector, q is the normalization constant of point charge (mV cm), and m is the number of poles (m=155). In our FEM model, a mesh of resistors are given as shown in Fig 1(b)-(c). The distributed poles (which represent the electric organ) are represented in the FEM model as a series of elements which have constant voltage values (voltage source). The size of the water tank used for simulation is 40cm×20cm×20cm, where each size of each gridpoint is 1mm×1mm×1mm (total 16,000,000 elements are available) .

Using the fact that the total amount of the current that comes in or out from one element is zero by Kirchhoff's law, a linear equation is given for each element. To measure the electric field perturbation caused by an object, the base potential on the skin is initially calculated without an object, which is only influenced by electric poles. Then the object perturbation voltage is compared with the base potential.

By using a set of equations assigned to each element in the FEM model, the electric field perturbation can be measured at the electroreceptors on the skin. The transdermal voltage difference forms an electric image.

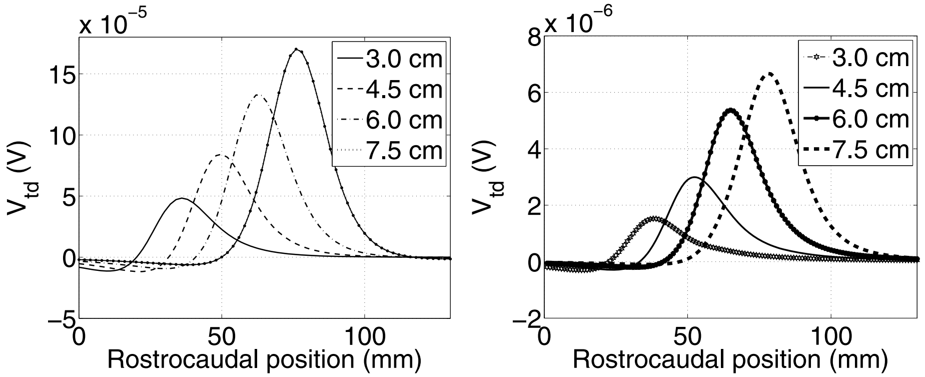
2.2 Relative Slope and THWHM

Electrolocalization procedure estimates the rostrocaudal distance and the lateral distance a target object near the electric fish. Many researchers have suggested electrolocation measures. von der Emde et al. [8] proposed a slope-to-amplitude ratio in an electric image, and Chen et al. tested a full-width at half-maximum for the lateral distance [2].



(a) (b)

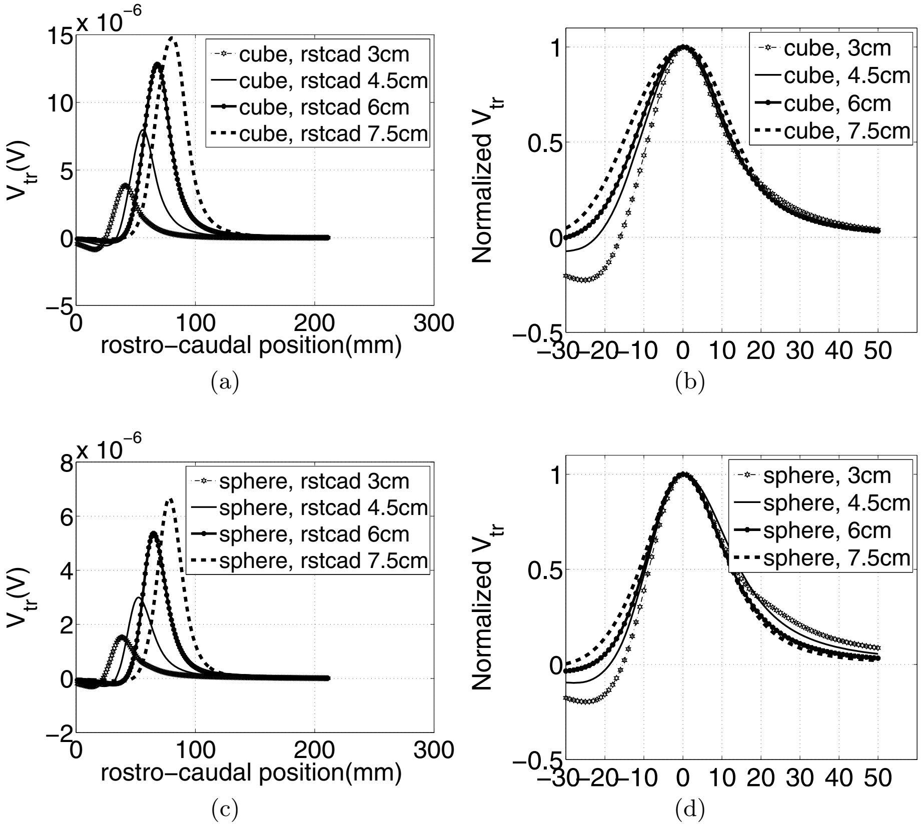
Fig. 2. Measure used in electric images (a) FWHM and THWHM (b) relative slope (rostal and caudal side)



(a) (b)

Fig. 3. Transdermal voltages; spherical object diameter: 20mm, lateral distance: 30mm, rostrocaudal distance: 30, 45, 60, 75mm (a) simulation with analytical model [7] (b) simulation with the 3-dimensional FEM model

The slope-to-amplitude ratio can be interpreted as the maximum slope of a normalized electric image which is also called relative slope. von der Emde et al. used the rostral side of the electric image for the relative slope (in other words, the image near the head). Sim and Kim (2012) showed that the caudal slope provides a cue to estimate the lateral distance, regardless of the rostrocaudal position of a target object [7]. In this paper, we will test a new style of measure THWHM, tail-side half width at half maximum (caudal slope) and the caudal relative slope in an electric image. THWHM checks the half-width of the caudal



(a)

(b)

(c)

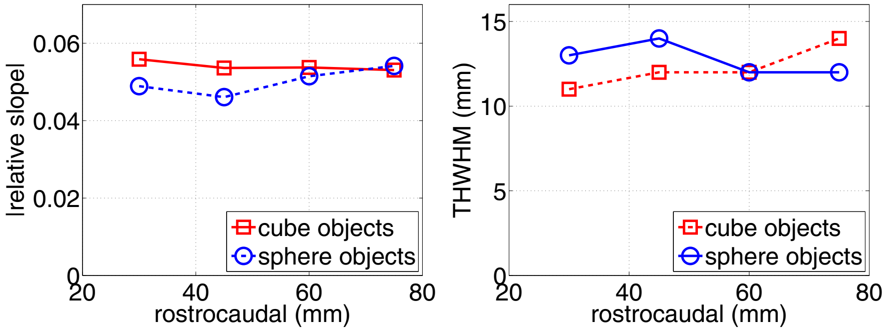
(d)

Fig. 4. Simulation results with cubes and spheres (a)/(c) is the electric image for the case of cube/sphere. (b)/(d) shows the normalized electric image where the rostro-caudal location, when peak value occurs, is set to zero. While the rostal side of the slopes vary quite severely, the caudal side shows a similar pattern. This means that the relative slope or THWHM on the caudal side may be used as a measure to estimate the lateral distance, while the relative slope on the rotral side will probably not.

side at half-maximum of an electric image, rather than the full width – see Fig. 2. The caudal slope measures the relative slope at caudal side of the electric image.

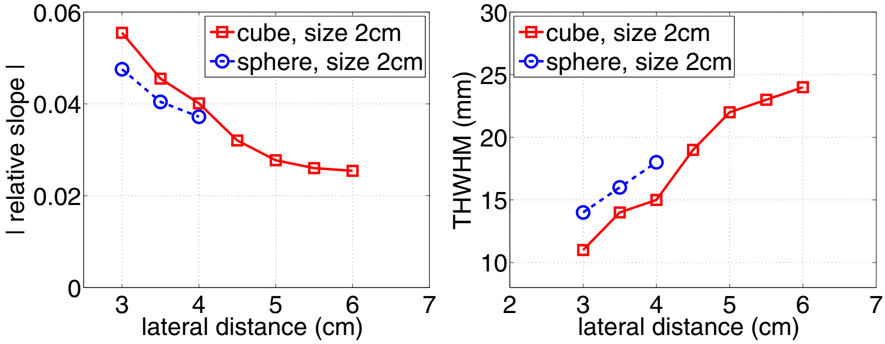
3 Experiments

We first tested two methods, analytical model [7] and the FEM model over a sphere object to validate the FEM model. As expected, the two approaches have the same style of electric images over varying positions of a target object as shown in Fig. 3.



(a) (b)

Fig. 5. Comparison cube and sphere with various rostrocaudal distance (a) relative slope for metal cubes and spheres; even as the rostrocaudal distance of the objects vary, the relative slopes have relatively small transitions (b) THWHM with the same image



(a) (b)

Fig. 6. A cube or sphere simulated under a fixed rostraocaudal distance; cube is 2cm in diameter with fixed rostrocaudal distance, 4.5cm. As the cube's (or sphere's) lateral distance varied from 3cm to 6cm (or 3cm to 4cm) with a 0.5cm interval (a) the relative slope comparison (b) the THWHM comparison.

For our simulation experiments, we placed metal cubes or spheres at a fixed lateral distance and various rostrocaudal distances, and the electric image is shown in Fig. 4. As the object gets closer to the tail, the peak value of electric images increase. This is because the intensity of the electric field is higher in the region near the tail.

Unlike the analytical model [2, 7], our FEM model can simulate the electric image for cube objects. The general trend of electric images for cubes and spheres are similar, but it seems that cube objects make the electric field distorted more severely. We normalized the electric images into the scale [0,1], and we observed much variation on the rostral side of the image. In contrast, the caudal side of the normalized electric image could be a good measure for lateral distance estimation.

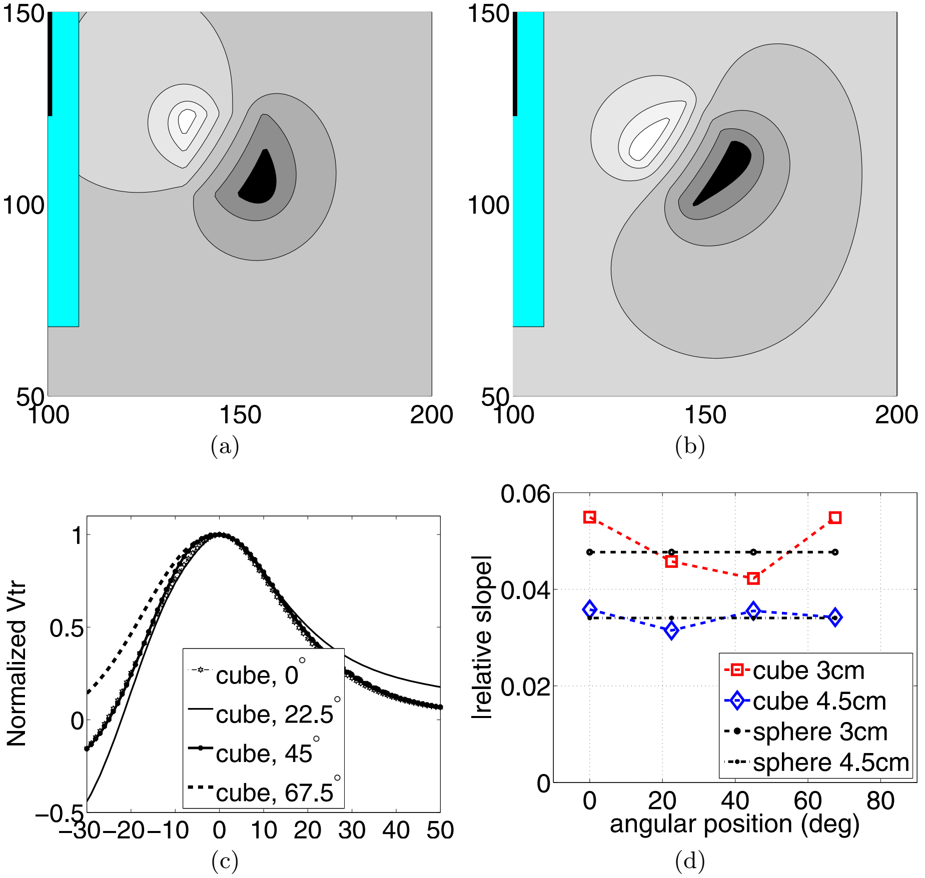
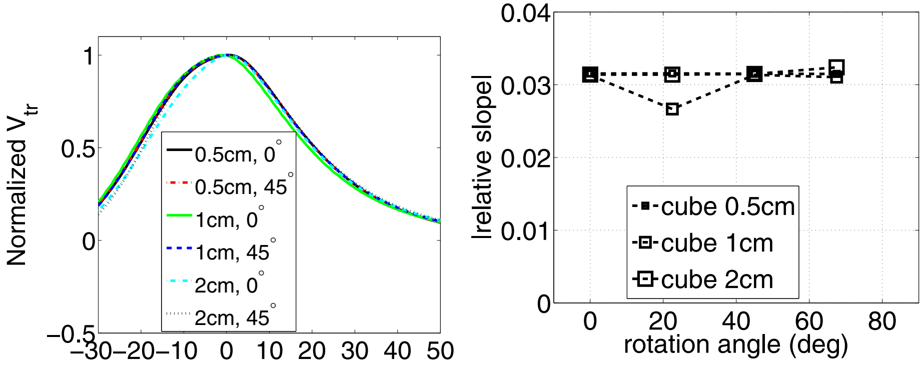


Fig. 7. Comparison of electric images with varying angles and varying distances of cubes. Spheres were also simulated at the same lateral distances for comparison (a,b) figures which represents the voltage perturbation due to a nearby cube. The bar on the left represents the electric fish head region. The cube is rotated with 0 and 45 degrees. (c) normalized electric images of a cube with varying rotation angles (d) the relative slope (on the right side of the image) for cubes varying lateral distance and rotation angle. For each lateral distance(3cm, 4.5cm), a sphere was placed for simulation.

In Fig. 5, the comparison of electric images for cubes and spheres is shown. Fig. 5(a) shows the absolute caudal slope and the slopes of cube objects is slightly larger than those of spheres. The relative slope has a little change for variation of rostrocaudal positions of a target object. Fig. 5(b) shows the results with the THWHM measure. A little fluctuation of the measure can be found with the measure.



(a) (b)

Fig. 8. Electric images and relative slope for cubes with varying sizes and varying angles (the lateral distance and rostrocaudal distance of the objects were fixed to 4.5cm, but the cube sizes and rotational angles varied) (a) normalized electric images (only with rotation angle 0 and 45 degrees and cube size 0.5cm, 1cm and 2cm; total 6 cases displayed) (b) relative slope for all cases (12 cases displayed)

As an object gets closer to the electric fish, the electric image has a sharper curve which means that the relative slope increases and the THWHM decreases. Interestingly cubes and spheres have different values at the electrolocation measures. The electric fish may perceive a sphere to be a little further away compared to the same sized cube, since the relative slope of a cube at a fixed lateral distance is larger than that of a sphere. This result is consistent with the biological experiment result by von der Emde et al.[8].

We check how the electric fish senses cubes at different angles. We used a cube whose side length is 2cm, located at the rostrocaudal position 4.5cm from the head, and at varying lateral distances, 3cm and 4.5cm . We tested the rotation angles, 0, 22.5, 45, 67.5 degrees. For comparison, a sphere for each lateral distance was also simulated. In Fig. 7(a)-(b), the voltage perturbation due to the cube object is shown (at a lateral distance of 4.5cm). The electric images depending on rotation angles of cubes are displayed in Fig. 7(c). The rostral side images have relatively more variation. The caudal side of the electric images are considered for the lateral distance estimation. Fig. 7(d) shows the results with the caudal relative slope. The relative slope of the spheres were given as a reference. From this result, the rotating angle of a cube influences the electric image pattern. However, the overall pattern clearly shows that the closer the object is, the larger the relative slope becomes, regardless of the shape or rotation angle. The fluctuation of relative slope is larger when the object is closer to the electric fish.

Fig. 8 shows another simulation results with varying sizes of cubes. The cube width is either 0.5cm, 1cm or 2cm. Fig. 8(a) is result of the normalized electric images, shows the relative slopes with varying sizes of cubes or rotating angles. The relative slope has only a small change for rotating angles and size variations of cubes at a far distance.

4 Conclusion

Our 3-dimensional FEM model shows similar results with the analytical model [7], and we can test various object shapes or multiple objects. When we investigated the electric images of cubes or spheres, we confirm that caudal relative slope can be a cue for the lateral distance of a target object, regardless of the rostrocaudal position, as Sim and Kim [7] suggested the measure for a sphere object. In our simulation experiments, the caudal relative slope or THWHM gives an approximate estimation of lateral distance even for a cube. The rotation angle of cubes might influence the electric field perturbation, but it seems that cubes at relatively large distances may have a minor effect on the perturbation.

Here, we have simulated the electric image for cubes and spheres at varying distances. To validate our FEM model, physical experiments could be done in the future. Our FEM model can consider the relative resistivity of fish's internal body, fish skin, and water, and so this model has great potential of application. In this paper, we tested a single object, but we can test multiple objects and their interference in the electric image for the future works. It might even be possible to accommodate skin capacitance or object capacitance into the model and measure phase shift of EOD waveform. We still need further work to improve our model. As this model could become more accurate, it might give a crucial key to understanding the mechanism of the electrolocalization. This model could also help one to develop an artificial sensory system, since it could predict the electric potential field in more realistic situations.

Acknowledgements. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No.2012-0001626).

References

1. Babineau, D., Longtin, A., Lewis, J.E.: Modeling the electric field of weakly electric fish. Journal of Experimental Biology 209(18), 3636 (2006)
2. Chen, L., House, J.L., Krahe, R., Nelson, M.E.: Modeling signal and background components of electrosensory scenes. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology 191(4), 331–345 (2005)
3. Fujita, K., Kashimori, Y.: Modeling the electric image produced by objects with complex impedance in weakly electric fish. Biological Cybernetics 103(2), 105–118 (2010)
4. Rasnow, B.: The effects of simple objects on the electric field of apteronotus. Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology 178(3), 397–411 (1996)
5. Sim, M., Kim, D.: Electrolocation based on tail bending movements in the weakly electric fish. Journal of Experimental Biology 214, 2443–2450 (2011)
6. Sim, M., Kim, D.: Electrolocation with an electric organ discharge waveform for biomimetic application. Adaptive Behavior 19(3), 172 (2011)
7. Sim, M., Kim, D.: Electrolocation of multiple objects based on temporal sweep motions. Adaptive Behavior 20(3), 146–158 (2012)
8. von der Emde, G., Schwarz, S., Gomez, L., Budelli, R., Grant, K.: Electric fish measure distance in the dark. Nature 395, 890–893 (1998)

Multimodal Integration of Visual Place Cells and Grid Cells for Navigation Tasks of a Real Robot

Adrien Jauffret, Nicolas Cuperlier, Philippe Gaussier, and Philippe Tarroux

ETIS laboratory, UMR 8051
F-95000 - Cergy Cedex, France
LIMSI laboratory, CNRS(UPR3251)
91403 Orsay Cedex, France

Abstract. In the present study, we propose a model of multimodal place cells merging visual and proprioceptive primitives. First we will briefly present our previous sensory-motor architecture, highlighting limitations of a visual-only based system. Then we will introduce a new model of proprioceptive localization, giving rise to the so-called grid cells, wich are congruent with neurobiological studies made on rodent.

Finally we will show how a simple conditionning rule between both modalities can outperform visual-only driven models by producing robust multimodal place cells. Experiments show that this model enhances robot localization and also allows to solve some benchmark problems for real life robotics applications.

Keywords: Grid cells, Bio-inspired robotics, Multimodal integration, Sensory-motor navigation & mapping, Neural networks.

1 Introduction

Ethological studies of animal navigation show that a wide variety of sensory modalities can be used by animals to navigate and self localize in an unknown and complex environment. Since the startling discovery by OKeefe and Dostrovsky [1] of the spatial correlates of neural activity in the hippocampal system (HS) of rodents, some work has been done to investigate the neural bases of animals spatial learning (see [2] for a short review). Originally found in HS, place cells are pyramidal neurons exhibiting high firing rates at a particular location in the environment (place field). Cells with similar properties but with larger place field have also been found in the Enthorinal Cortex (EC). As firing of place cells persists in the dark it has also been suggested that other senses (proprioception, touch, smell) might contribute as well [3–8].

Later in 2005, Hafting and Moser discorvered grid cells in the dorso-lateral band of the medial EC (dMEC) [9]. These cells present spatial firing fields forming regular triangle-pattern (grid) that tiled the environment. They could be the basis of a cognitive map of Euclidean space. Each grid is defined by 3 parameters: frequency (distance between two vertex), phase (spatial shift) and orientation. Grid cell activity does not require visual input, since it remains unchanged in

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 136–145, 2012.

absence of any visual cue (dark) even if the bumps of activity tend to spread due to accumulation of errors by the integration process [9].

Three main classes of models have been proposed: recurrent network models based on continuous attractor dynamics [10, 11], independent-neuron models based on oscillatory interference [12] and models using a residue number system [13].

Following our previous work [14] based on a residue number model, we first present a robotic implementation of this model able to exhibit grid cells like firing pattern. Results underline the key role played by visual inputs. We show that without visual recalibration, grid cells firing seems scrambled, according to biological results [9]. Next we present a model, based on a pavlovian conditioning rule, that merges signals coming from visual cells and grid cells into multimodal place cells. Experiments on a real robot show how grid cells information can be enough to self-localize in the dark on short distances. We also show how grid cells activity help to greatly reduce visual ambiguity, giving robust multimodal place fields. Finally, we will briefly discuss how this model behaves when the robot is kidnapped and shifted to another location.

2 Modeling Place Cells from Visual Information

In previous works, we developped a model of the hippocampus in order to obtain visual place cells (VPCs) [15] that allowed controlling mobile robots for visual navigation tasks [16, 17]. The embedded pan-tilt camera allows the capture of several images (actually 15) corresponding to a 360 degrees panorama. A gradient image convolved with a DoG (difference of gaussian) filter allows to highlight a set of salient points in the scene (curvature points at a low resolution). A log polar transform of a small circular image centered on each focus point (local view of 16*16 pixels) is computed in order to improve the pattern recognition against small rotations and scale variations.

Then, a neural network learns place cells that code information about a constellation of landmarks in the scene (5 landmarks per images) (figure 1). Activities of the different place cells depend on the recognition level of landmarks. Robustness comes from the large number of local views extracted (75 per panorama) and the only use of a competition between place cells (see [18] for detailed parameters). In our model, local views correspond to the ”what” information coded in the perirhinal cortex or in other areas of the ventral visual pathway of the rat temporal cortex [19]. The absolute position of these local views (the ”where” information) is provided by the parietal cortex through the parahippocampal region. The merging of ”what” and ”where” information may be performed in the superficial layer of the enthorinal cortex or in the postrhinal cortex [20, 21].

A neural network learns to associate a particular PCs with an action (a direction to follow in our case). This sensory-motor architecture is named Per-Ac [22] and allows the robot to learn simple but robust behaviors.

Even if our architecture has been succesfully tested in small sized environments (typically one room), a visual-only based mechanism shows limitations

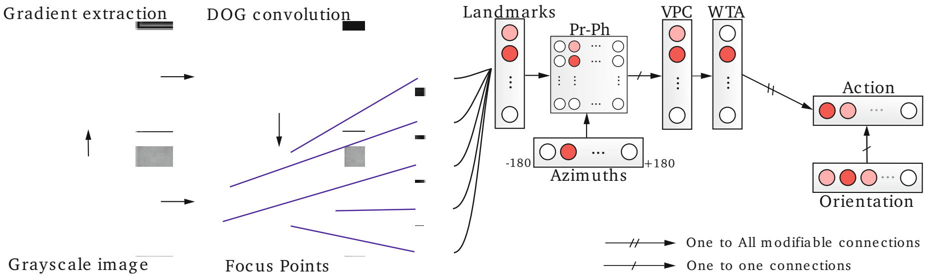


Fig. 1. Sensorimotor model relying on vision. The gradient image is convolved with a difference of gaussian filter. Local maxima of the resulting image correspond to points of interest on which the system focuses on to extract local views. A Place Cell (PC) learns to recognize a specific landmarks-azitmuths constellation. An action is associated with this PC. This association is learned by a least mean square algorithm (LMS), after what the system is able to move in the learned direction when the associated PC wins.

when trying to scale to larger and more complex ones (multi-room, outdoors). First the large number of PCs needed to cover this kind of environment introduces a computational problem that highly decreases the robustness of the localization. Then a lot of mistakes are due to the ambiguous nature of the visual modality. Indeed the activity of a PC can highly responds for different locations with identical visual panorama (corridor for instance). We propose to overcome these issues by adding a bio-inspired localization mechanism based on proprioception, following our hypothesis of the way grid cells work.

3 Modeling Grid Cells from Extra Hippocampal Path Integration

3.1 A Plausible Model of Grid Cells

Here we present a model for generating grid cells from path integration. Our simplest model of grid cells (GCs) is based on various modulo's operator applied on path integration [14] (figure 2). The path integrator is supposed to be stored outside the hippocampal system. The activity D_i of a neuron belonging the path integration field (associated with direction θ_i) is discretized over a new field of neurons $E^i_j = round(\frac{D_i.N_E}{D_{max}})$ where D_{max} is the maximum value of the distance that can be computed by the neural field, N_E is the number of neurons on each field used to discretized each analog activity on the path integration field. Then a modulo operator is used to compress the field E^i_j by projection.

$$M^n_k = \begin{cases} 1 & if\ k\ =\ ArgMax(E_j)\ mod\ \lambda^n \\ 0 & otherwise \end{cases} \quad (1)$$

with λ^n the value of the modulo used to build grid n.

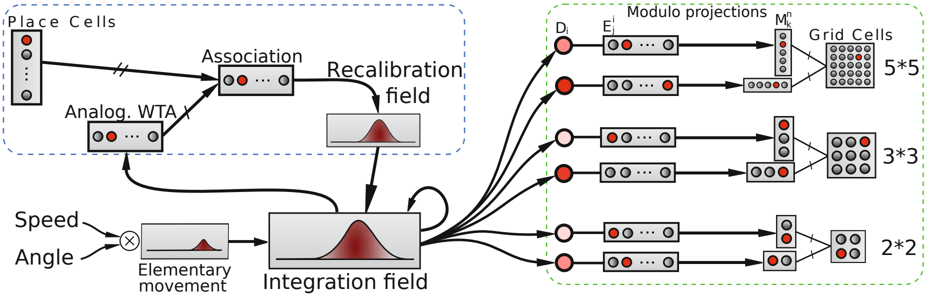


Fig. 2. Linear speed and absolute orientation are used to characterize movement unit and so generate global path integration on a neural field. A recalibration mechanism associate a VPC with the argmax of the neural field (distance and orientation are stored). It allows the system to limit cumulative error on this field when it later well recognizes the corresponding VPC. Path integration field is then used to build grid cell activity without any cartesian map. Activities of randomly chosen pairs of neurons D_i in that field are discretized on other fields E_i^j. Those fields are compressed by simple modulo projections M_k^n. The conjonction of 2 codes of 2 projections is sufficient to obtain grid cells.

A recalibration mechanism, using a Widrow and Hoff learning rule [24, 26], associate each new VPC with the current path integration field activity (one-shot learning), so that it can be recalibrated when that VPC is later well recognized, like [10, 27–30] (actually when the winning cell activity reach an absolute threshold of 0.9 and a difference relative to VPCs mean activity of 0.4).

3.2 Recreating Grid Cells Activity on 3 Experiments with a Real Robot

Typical experiments made on rodent consist in recording the activity of grid cells in dMEC while the rat (around 20cm large) freely moves in a circular enclosure (2m of diameter) during 30 minutes. Our experiments run in almost similar conditions since a real robot (around 40cm large) randomly moves in an hexagonal park (4m of diameter) during the same period of time. Position and simulated grid cells are simultaneously recorded. Ultrasound sensors are used by the robot to avoid obstacles and stay inside the hexagonal playground.

In a first experiment, no recalibration is allowed. We note a fast drift of grid cells activity induced by cumulative errors on path integration. All cells present a blurred activity without grid-like pattern because of a 30 minutes errors cumulation (figure 3).

In a second experiment, a VPC is learned at the center of the hexagon and visual calibration is allowed. We obtain coherent grid-like pattern but error remains too high to correctly visualize small grids. Error's amplitude is directly linked with the recalibration area (figure 3).

To avoid this problem, we made a third experiment with a touch-like reset cue at the corner of 2 walls. The origin of the path integration is set in this

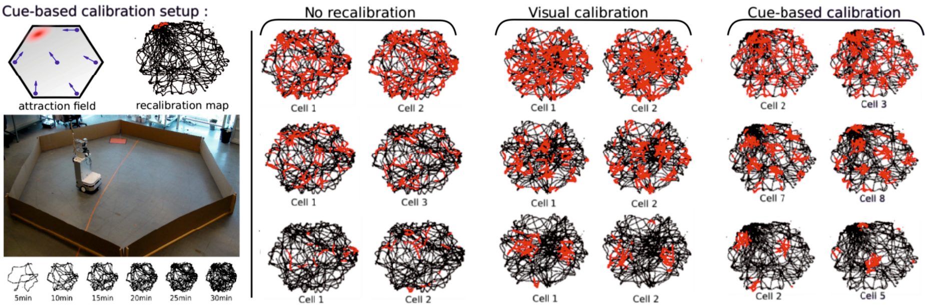


Fig. 3. Experiments made with a real robot (around 40cm large) randomly moving in an hexagonal enclosure (4m of diameter) during 30 minutes (trajectory in black, neural activity in red). **Left** : Experimental setup: for the cue-based calibration experiment, path integration is set in one corner of the park and an attraction field is learned around it allowing the robot to autonomously go recalibrate itself every minute. **Right** : Results of the third experiment show coherent regular hexagonal pattern of different phases and modulos. Those patterns are quite similar to thus obtained with rodents.

corner and a visual attraction field is learned around it by a few sensory-motor associations. It allows the robot to autonomously converges into the reset area by visual recognition (homing behavior). The reset cue is a red paper sticked on the floor and a color detector is used by the robot to perceive this goal. A simple counter triggers a periodic drive allowing an homing behavior every minute. The drive is reset each time the goal is reached, allowing the robot to switch back to a random exploration strategy. This solution avoids errors accumulation over more than one minute and so gives us the precision needed to display small grid. But it introduces a static error directly linked with the size of the reset field.

Results show well-defined pattern for the different modulo factors. Those grid activities are congruent with neurobiological records in rodents EC [9]. Grid shared the same spacing for the same modulo and the same orientation for the same discretization factor. Nevertheless, each cell produces a grid of different phase.

4 Building Robust Multimodal Place Cells from Visual and Grid Cells

4.1 A Pavlovian Model of Merging

In this section, we propose a simple merging mechanism which can take advantage of allothetic and idiothetic information. This mechanism pops up the synergy between both modalities.

There are many ways of merging different information sources, and it is known as a difficult problem, especially when the nature of the sources are highly different. Sensory modalities must be recoded into a common format before they can be combined. The task is described as the recoding problem by Pouget and Deneve [23].

Our model is based on the learning of associations between VPCs and the whole GCs pattern. The system associates a particular GCs state (conditional stimulus) with the current winning VPC (inconditional stimulus) by a classical conditionning rule [24, 26], as for the recalibration mechanism (one-shot learning), like [27]. We test it with a normalized least mean square algorithm (NLMS) [25] trying to predict the visual state from GCs activities. A simple weighted sum allows the merging of VPCs and odometric place cells from GCs.

The activity of a multimodal place cell MPC_k is given by a simple linear combination of VPC and $PredVPC$ activities:

$$MPC_k = \alpha.VPC_k + (1 - \alpha).PredVPC_k$$

with VPC_k the activity of the corresponding visual place cell, $PredVPC_k$ the activity of the corresponding place cell predicted by the grids (NLMS output), $\alpha \epsilon$ [0;1] a ponderation factor (0.5 in our case) and $k \epsilon$ [0;K] the indice of the cell.

The activity of a place cell predicted by grid cells is given by:

$$PredVPC_k = \sum_{k=0}^{K-1} w_k.G_k$$

where w_k is the weight of the synapse coming from the corresponding grid cell G_k.

4.2 Results Obtained during a Multi-room Indoors Experiment

In order to test the robustness and the generalization capabilities of our architecture, we made several indoor navigation experiments in a 25x15m environment (our laboratory). For analysis purpose, we supervized the robot learning to recognize 19 places (each 1.5m) on a multi-room trajectory. The trajectory starts in one room, pass through a corridor and ends in a second room (mostly similar to the first one). Next, the robot followed 5 different parallel trajectories, imposed by a remote control. Visual recalibration is allowed (calibration driven by well recognized VPCs only, no periodic homing behavior).

Those five trajectories permit to cover a large space near the learned path, in order to test generalization capability (figure 4).

The visual recognition system allows great generalization capability (large place field) but present small perception mistakes due to cue redundancy. On contrary, the proprioceptive recognition system presents well-defined place fields without any ambiguity, but is subject to the classical cumulative error of odometry. This induces a very precise discrimination for small scales but a shifted localization for larger one. Results show interesting emergent characteristics since the merging mechanism keep a correct localization even if errors simultaneously happen on the 2 modalities. Merging modalities hightlights contingencies and reduces non-contingent activities. Two low-level activities at the same time are more coherent than a singural high-leveled one.

Finally, MPCs are robust to ambiguity and keep large generalization properties. To show the deterministic nature of the results, we repeated the experiment a dozen of times in a changing environment (ambiant light and furnitures

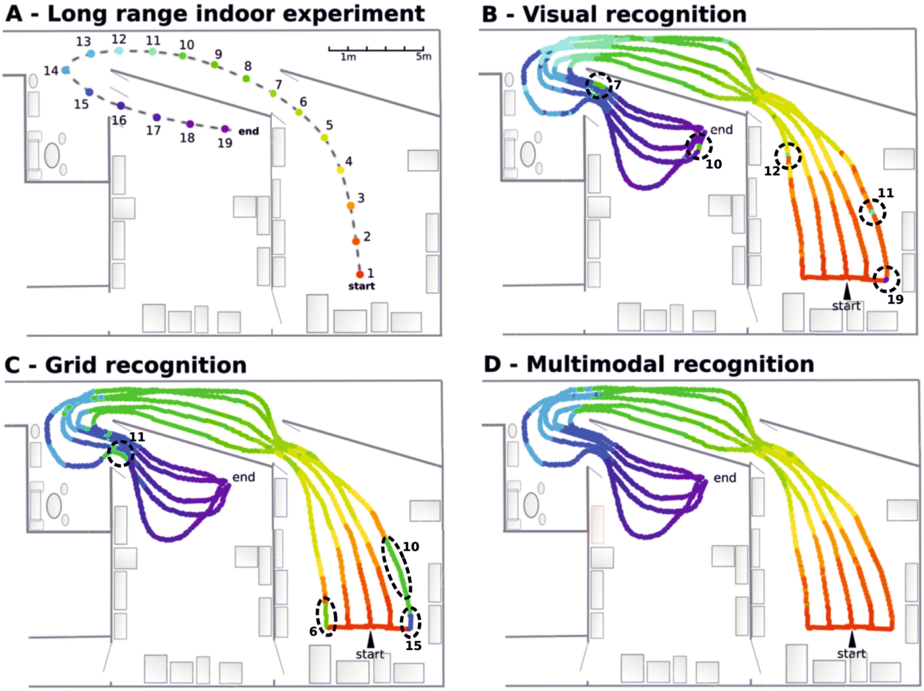


Fig. 4. Long range navigation in an indoor environment: **A** - Experimental setup: The robot learns 19 regularly-spaced places (each 1.5m), starting from a room, passing through a corridor and ending on a second room. **B** - Visual recognition obtained for 5 different trajectories. Each color is associated to one visual place cell. Results show great generalization capabilities but present ambiguities (dotted circles. Numbers correspond to perceived places.). **C** - Grid recognition for the same trajectories. Grid fields are smaller but without any ambiguities. A place is not recognized if the robot is too far away from the learned place. **D** - Multimodal recognition obtained by merging visual and grid place cells. The synergy of both modalities shows well defined areas even if the robot is far from the learned trajectory.

changing, persons moving). We also studied what happen when the robot is lifted, blindfolded, then transported to another place. If this place is already known and highly recognized, the robot recalibrates its path integration field to a previously learned value. We made several kidnapping events in order to test the robustness of this mechanism. Each times, the robot typically runs 5 meters before to recalibrate its odometry. Thanks to the merging mechanism, the perceived location never stays wrong for more than 2 meters. This recalibration mechanism allows the robot to always keep consistancy between vision and proprioception and never getting lost near learned locations.

4.3 Topology Matters : On the Need for Convolving Grid Cells

In our model, all grids present binary fields (activated or not) so that the pattern generated by the conjunction of 3 grids is a three-steps stair shaped. Moreover

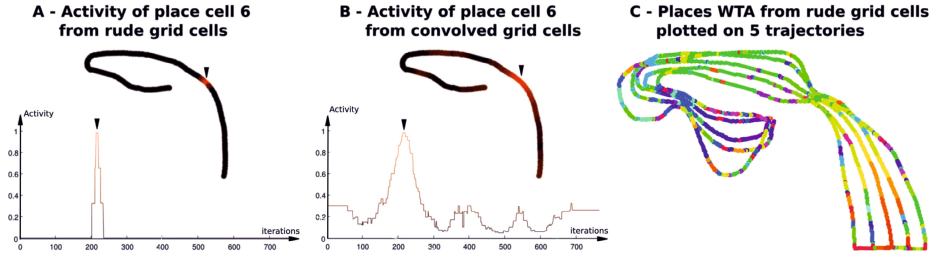


Fig. 5. Example of the diffusion mechanism used to generates topology. **A** - Activity of motor place cell 6 from binary grid cells, in space and time on a multi-room trajectory (before to be convolved). The place field is a thin three-steps stair shaped. **B** - Activity of motor place cell 6 from convolved grid cells. The convolution act as a diffusion mechanism spreading activity on neighboring grid cells. Motor place cells show gaussian shaped activity allowing generalization capabilities. **C** - Results obtained for winning motor place cells from binary grid cells (no diffusion) during the previous indoor experiment. It shows a lot of errors that can be easily removed by adding the diffusion mechanism.

the 3 differents modulo factors are relatively prime so that the activity of the conjunction pattern is often completely different of the previous one for only few movements. Narrow place cells are treated like there is no proximity distance between them. Each grid cell is considered as an orthogonal input so that grid networks doesn't benefit from any topology. That is the reason why we propose to convolve each grid network with a pyramidal-shaped mask, giving to the system more generalization capabilities. This technics allows to spread field activities over neighboring cells by using a natural torus topology [11], and so generates continuity. In return, it looses the ability to distinguish between 2 near places if the distance between them is smaller than the mask size (figure 5). We used this convolution method in the multi-room experiment presented in section 4.2. Without that convolution, the system can't work.

Such method confront us with its biological plausibility since experiments made on rodent only show binary activities. It questions about how is it possible for mammals to correctly navigate using grid cells binary fields. We argue that the topology needed can naturally emerge from the large number of grid cells in rodent's brain.

5 Discussion

In this incremental design approach, one objective of our robotic experiments is to show the limitations of models (proof by failure). Hence, highligting the need to take into account new cerebral structures or interactions between them allows us to propose a more coherent model for a better understanding of explored brain structures.

Our experiments emphasized some issues while trying to scale our architectures to larger environment: ambiguities coming from the visual modality have been identified as the major problem. Consequently, we extended our architecture by modelling grid cells networks and we presented a robotic experiment that can account for their firing properties. Then, we present a simple merging mechanism exploiting these grid cells to desambiguate visual perception and generate robust multimodal place cells. We show that it succesfully overcome the perception ambiguity problem and it stay robust even if the system is blindfolded or kidnapped, then lifted to another place. Moreover, results show an emergent characteristic hightlighting contingency and reducing singular activities since the merging mechanism keeps a coherent localization even if errors simultaneously happen on both modalities.

We also underline the need to spread grid cells binary activity to neighboring cells to create a topology inducing interesting generalization properties. It questions about the biological plausibility of such ad-hoc method. We claim that such topology can naturally emerge from the large number of grid cells in rodent's brain.

Our current work focuses on switching navigation strategies according to an emotional metacontroller based on bayesian inferences. In the same time, we are performing long range (several kilometers) outdoor navigation experiments based on these models.

Acknowledgment. We would like to thank AUTO-EVAL and the NEUROBOT French ANR project.

References

1. O'Keefe, J., Dostrovski, J.: The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. Brain Research 34, 171–175 (1971)
2. Cuperlier, N., Quoy, M., Gaussier, P.: Neurobiologically inspired mobile robot navigation and planning. Frontiers in NeuroRobotics 1(1) (2007)
3. Markus, E.J., Barnes, C.A., McNaughton, B.L., Gladden, V.L., Skaggs, W.E.: Spatial information content and reliability of hippocampal CA1 neurons: Effects of visual input (2004), doi:10.1002/hipo.450040404
4. Muller, R.U., Kubie, J.L., Ranck Jr., J.B.: Spatial firing patterns of hippocampal complex-spike cells in a fixed environment. The Journal of Neuroscience 7(7), 1935–1950 (1987)
5. Quirk, G.J., Muller, R.U., Kubie, J.L.: The firing of hippocampal place cells in the dark depends on the rat's recent experience. The Journal of Neuroscience 10(6), 2008–2017 (1990)
6. Schenk, F., Grobety, M.C., Lavenex, P., Lipp, H.-P.: Dissociation between Basic Components of Spatial Memory in Rats. Science Series D 82, pt. III, 277–300 (1995)
7. Schenk, F., Lavenex, P.: Olfactory traces and spatial learning in rats. Institute of Physiology, University of Lausanne (1998)
8. Wallace, D.G., Hines, D.J., Pellis, S.M., Whishaw, I.Q.: Vestibular Information Is Required for Dead Reckoning in the Rat. The Journal of Neuroscience 22(22), 10009–10017 (2002)

9. Hafting, T., Fyhn, M., Molden, S., Moser, M.B., Moser, E.I.: Microstructure of a spatial map in the entorhinal cortex. Nature 436(7052), 801–806 (2005)
10. Fuhs, M.C., Touretzky, D.S.: A Spin Glass Model of Path Integration in Rat Medial Entorhinal Cortex 26(16), 4266–4276 (2006)
11. McNaughton, B.L., Battaglia, F.P., Jensen, O., Moser, E.I., Moser, M.-B.: Path integration and the neural basis of the 'cognitive map'. Nature Reviews Neuroscience 7, 663–678 (2006)
12. Burgess, N., Barry, C., O'Keefe, J.: An oscillatory interference model of grid cell firing (2007), doi:10.1002/hipo.20327
13. Fiete, I.R., Burak, Y., Brookings, T.: What Grid Cells Convey about Rat Location. The Journal of Neuroscience 28(27), 6858–6871 (2008)
14. Gaussier, P., Banquet, J.P., Sargolini, F., Giovannengeli, C., Save, E., Poucet, B.: A model of grid cells involving extra hippocampal path integration, and the hippocampal loop. Journal of Integrative Neuroscience 6(3), 447–476 (2007)
15. O'Keefe, J., Nadel, N.: The hippocampus as a cognitive map. Clarendon Press, Oxford (1978)
16. Gaussier, P., Revel, A., Banquet, J.-P., Babeau, V.: From view cells and place cells to cognitive map learning: processing stages of the hippocampal system. Biological Cybernetics 86, 15–28 (2002)
17. Banquet, J.-P., Gaussier, P., Quoy, M., Revel, A., Burnod, Y.: A hierarchy of associations in hippocampo-cortical systems: Cognitive maps and navigation strategies. Neural Computation 17(6), 1339–1384 (2005)
18. Giovannangeli, C., Gaussier, P., Banquet, J.P.: Robustness of visual place cells in dynamic indoor and outdoor environment. International Journal of Advanced Robotic Systems 3(2), 115–124 (2006)
19. Kolb, B., Tees, R.: The Cerebral Cortex of the Rat. MIT Press (1990)
20. Suzuki, W.A., Miller, E.K., Desimone, R.: Object and place memory in the macaque entorhinal cortex. J. Neurophysiol. 78(2), 1062–1081 (1997)
21. Burwell, R.D., Hafeman, D.M.: Positional firing properties of postrhinal cortex neurons. Neuroscience 119(2), 577–588 (2003)
22. Gaussier, P., Zrehen, S.: Perac: A neural architecture to control artificial animals. Robotics and Autonomous Systems 16(2-4), 291–320 (1995); Moving the Frontiers between Robotics and Biology
23. Pouget, A., Deneve, S., Duhamel, J.-R.: A computational perspective on the neural basis of multisensory spatial representations. Nature Reviews Neuroscience 3, 741–747 (2002), doi:10.1038/nrn914
24. Widrow, B., Hoff Jr., M.E.: Adaptive Switching Circuits. IRE WESCON Convention Record 4, 96–104 (1960)
25. Haykin, S.: Adaptive Filter Theory. Prentice Hall (2002) ISBN 0-13-048434-2
26. Pavlov, I.P.: Conditioned reflexes. Routledge and Kegan Paul, London (1927)
27. Strosslin, T., Sheynikhovich, D., Chavarriaga, R., Gerstner, W.: Robust self-localisation and navigation based on hippocampal place cells. Neural Networks 18(9), 1125–1140 (2005)
28. Arleo, A., Smeraldi, F., Hug, S., Gerstner, W.: In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) Advances in Neural Information Processing Systems, 1(ii), pp. 89–95. Citeseer (2001)
29. Herrmann, J.M., Pawelzik, K., Geisel, T.: Self-localization of autonomous robots by hidden representations. Autonomous Robots 7(1), 31–40 (1999)
30. Mataric, M.J.: A Distributed Model for Mobile Robot Environment-Learning and Navigation, MIT EECS Master's Thesis (January 1990)

Towards a Neural Hierarchy of Time Scales for Motor Control

Tim Waegeman, Francis Wyffels, and Benjamin Schrauwen

Department of Electronics and Information Systems
Ghent University, Ghent Belgium
http://reslab.elis.ugent.be

Abstract. Animals show remarkable rich motion skills which are still far from realizable with robots. Inspired by the neural circuits which generate rhythmic motion patterns in the spinal cord of all vertebrates, one main research direction points towards the use of central pattern generators in robots. On of the key advantages of this, is that the dimensionality of the control problem is reduced. In this work we investigate this further by introducing a multi-timescale control hierarchy with at its core a hierarchy of recurrent neural networks. By means of some robot experiments, we demonstrate that this hierarchy can embed any rhythmic motor signal by imitation learning. Furthermore, the proposed hierarchy allows the tracking of several high level motion properties (e.g.: amplitude and offset), which are usually observed at a slower rate than the generated motion. Although these experiments are preliminary, the results are promising and have the potential to open the door for rich motor skills and advanced control.

Keywords: Locomotion Control Hierarchy, Adaptive control, Feedback control, Central Pattern Generator, Reservoir computing (RC).

1 Introduction

Animals show remarkable rich motion skills, they are able to run and walk over uneven and difficult terrain without the need to think about breathing, muscle control or low level sensory feedback processing. Instead, they think on a more high level such as which obstacles are approaching and how they can avoid them.

According to [1], many aspects of brain functions can be explained by a hierarchy of temporal scales at which representations of the environment evolve. The higher level encodes slower contextual changes in the environment or body while at the lower level faster variations due to sensory processing are encoded.

Other biological research suggests that specialized neural circuits, so called central pattern generators (CPGs), located in the spinal cord are responsible for generating rhythmic activations needed for body function including the contractions of a heart or lungs and control of muscles for walking [2]. Implying that some aspects of brain functions are offloaded to regions outside of the brain. A lot these findings are based on the study of the locomotion of a lamprey which is a primitive fish (in [3] an extensive review is presented). For instance, researchers

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 146–155, 2012.

discovered [4] after extracting and isolating the spinal cord from the body, that the spinal cord, when excited with electrical stimulations, will produce fictive locomotion. This indicates that sensory information is not needed to generate such rhythmic patterns. However, it plays a crucial role in shaping the generated pattern to keep the coordination between the CPGs and the body. In [5] they demonstrated that a mechanically driven treadmill can induce a normal looking walking gait in a deprecated cat.

These findings suggest that locomotion can be represented by a hierarchy of modules which interact with each other on different timescales which simplifies high level control and at the same time allows fast action against perturbations. For example, slight irregularities in the terrain are compensated very fast by the morphology of the legs without the need of the brain to intervene. Larger irregularities are handled by a slower reflex motion of which the runner becomes finally conscious at the highest but slowest contextual level.

Biological research is often used to improve the abilities of robots. For example, in [6] a salamander robot is controlled by a network of CPGs imitating the spinal cord of a salamander. In this work we investigate a multi-timescale control hierarchy applied on a robot leg. Each of the layers in the proposed hierarchy uses a random dynamical system of which only the readout layer is trained (eg. Reservoir Computing systems [7]). On the lowest level, the fastest timescale, the passive compliance of the leg interacts with the environment. The leg is driven by a pattern generator which generates learned rhythmic motor signals and gets feedback from the rotary encoders in the leg. On the highest level, we use a controller which reacts to slower contextual changes. This hierarchy separates the motor commands from the functional control which is done by the high level controller. Furthermore, by using a Reservoir Computing network on each layer, we open the door for potentially rich motor skills and advanced control which is topic for future investigation.

The remainder of this paper is structured in six Sections. In Section 2 we start by giving a rough overview of the proposed hierarchical control scheme. Next, in Section 3 we elaborate more deeply on the core technique used to build our hierarchy. After that, the two main building blocks of our control hierarchy are discussed in more detail. Afterwards, the hierarchy is validated by means of three preliminary experiments in Section 6. Finally, we end this paper by giving our conclusions.

2 Proposed Hierarchy

In all vertebrates, neural circuits located in the spinal cord can be found that are responsible for generating rhythmic activations used for locomotion. These neural circuits are called Central Pattern Generators (CPGs) and are currently modeled by roboticists to control robot locomotion. One of the key advantages that can be identified is that these CPGs typically have only a few control parameters and thus reduce the control problem [3].

In this work we propose the use of a hierarchical (artificial) neural system for adaptive locomotion control. This system, illustrated in Fig. 1(a), consists of

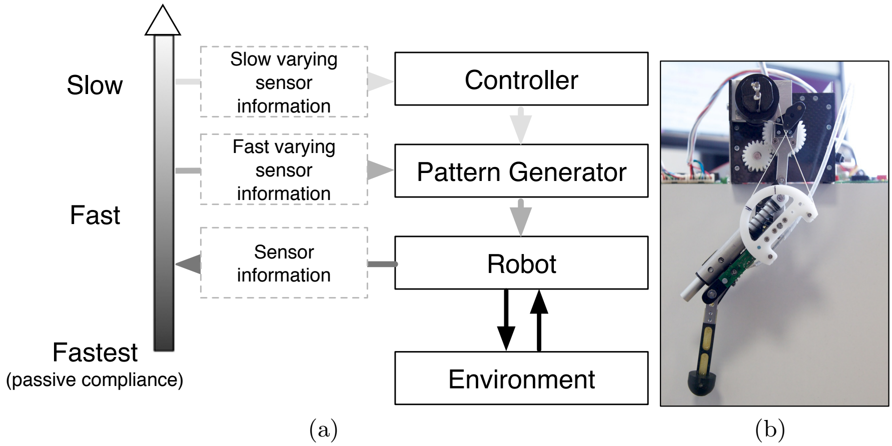


Fig. 1. (a) We present an overview of the control hierarchy. Our approach uses two building modules working on different time scales. The first module, the pattern generator, operates at a fast time scale and gets feedback from fast varying sensors. The parameters of the pattern generator are adapted by a controller, operating at a slower time scale, which gets feedback about slowly variating motion properties. The environment is included to illustrate that very fast perturbations caused by interacting with the environment are handled by the passive compliance of the leg. (b) Shows the actual leg of the Oncilla robot build in the AMARSi consortium on which the experiments were performed.

two building blocks: a pattern generator and a controller. On the lowest level, a pattern generator operates at a fast time scale and embeds a learned rhythmic pattern which is given to the motors of the robot. The rotary encoders of the motor system provide the pattern generator with direct feedback. Only environmental changes which are are unhandleable by the passive compliance of the leg, will be visible for this encoder. On the highest level, and thus slower time scale, the controller tunes the parameters of the pattern generator online in such a way that it keeps track of the slow varying parameters of the resulting motor. To achieve this, the sensor information presented to the controller is preprocessed by calculating for example amplitude and offset. As a result, the proposed hierarchy operates at multiple time scales which allows the use of a more advanced controller (which often acts slower) while the pattern generator can be kept relatively simple and can immediately act on for instance perturbations. In other words, the motor commands generated by the pattern generator are separated from the functional control which is done on a higher level.

3 Reservoir Computing

The core technique used for each of the building blocks in our hierarchical control approach is Reservoir Computing (RC). This is a collection of efficient training methods for random dynamical systems in which only the readout weights are trained [7]. In the past, RC has been independently introduced as Echo State Networks [8] and Liquid State Machines [9]. In previous work, RC has already proven its capabilities in a broad range of applications including robot localization [10], chaotic time series prediction [11] and speech recognition [12]. Additionally, researchers are making efforts to directly implement such systems on hardware [13].

The most commonly used flavor of RC is the Echo State Network approach which uses a random recurrent neural network of sigmoidal neurons. Training such a system starts by randomly creating the weight matrices $\mathbf{W}_{\text{res}}^{\text{res}}$, $\mathbf{W}_{\text{inp}}^{\text{res}}$, $\mathbf{W}_{\text{out}}^{\text{res}}$ and $\mathbf{W}_{\text{bias}}^{\text{res}}$ (these weights are usually drawn from a uniform or a normal distribution) which respectively determine reservoir-to-reservoir, input-to-reservoir, output feedback and bias weights. The reservoir weight matrix $\mathbf{W}_{\text{res}}^{\text{res}}$ is typically scaled such that the spectral radius, e.g. the largest eigenvalue, is smaller than 1. This guarantees that the entire system is operating at the edge of chaos, where its computational power is greatest [14]. Some learning paradigms, such as FORCE learning [15], require that the spectral radius is larger than 1 as long as additional inputs are able to restrain the system's dynamics.

After constructing the weight matrices the network can be simulated. Therefore, every time step, the neuron states are updated using the following equation:

$$\begin{aligned}\mathbf{x}[k+1] = (1-\lambda)\mathbf{x}[k] +& \\ \lambda \tanh\Big(\mathbf{W}_{\text{res}}^{\text{res}}\mathbf{x}[k] + \mathbf{W}_{\text{inp}}^{\text{res}}\mathbf{u}[k] + \mathbf{W}_{\text{out}}^{\text{res}}\mathbf{y}[k] + \mathbf{W}_{\text{bias}}^{\text{res}}\Big).& \qquad (1)\end{aligned}$$

The states $\mathbf{x}[k+1]$ at time step $k+1$ depend on the previous states $\mathbf{x}[k]$, input $\mathbf{u}[k]$, a bias and the (optional) output feedback of the system $\mathbf{y}[k]$. By changing the leak-rate λ, the system's dynamics can be tuned effectively [8].

For training the output weights $\mathbf{W}_{\text{res}}^{\text{out}}$ a learning algorithms is needed that rapidly reduces the difference between the actual and desired output, and keep it small while converging to a set of fixed output weights. The resulting weights maintain a small error without further modification [15]. Recursive Least Squares (RLS) is one of those learning rules that satisfy all conditions for FORCE learning. During training, the reservoir states are updated according to equation 1 while at every time step the readout weights $\mathbf{W}_{\text{res}}^{\text{out}}$ and the output $\mathbf{y}[k+1]$ are adjusted according following RLS equations:

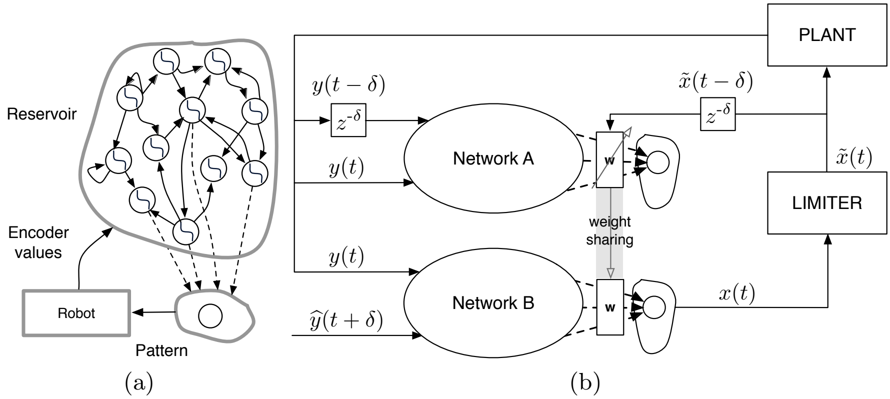
$$\mathbf{y}[k+1] = \mathbf{W}_{\text{res}}^{\text{out}}\mathbf{x}[k+1] \qquad (2)$$

$$\mathbf{e}[k+1] = \mathbf{y}[k+1] - \mathbf{y}_{\mathbf{desired}}[k+1] \qquad (3)$$

$$\mathbf{P} = \mathbf{P} - \frac{\mathbf{P}\mathbf{x}[k+1]\mathbf{x}^{\mathbf{T}}[k+1]\mathbf{P}}{1+\mathbf{x}^{\mathbf{T}}[k+1]\mathbf{P}\mathbf{x}[k+1]} \qquad (4)$$

$$\mathbf{W}_{\text{out}}^{\text{res}} = \mathbf{W}_{\text{out}}^{\text{res}} - \mathbf{e}[k+1]\mathbf{P}\mathbf{x}[k+1]. \qquad (5)$$

Here $\mathbf{e}[k+1]$ is the error at time step $k+1$ and $\mathbf{P}$ is an estimation of the inverse of the correlation matrix. After training, the weights $\mathbf{W}_{\text{out}}^{\text{res}}$ are kept fixed and the system can be used. However, when online adaptation is necessary, it is possible to train continuously.



(a) (b)

Fig. 2. The two main building blocks of our hierarchical approach. On the left (a) one can see a schematic overview of a Reservoir Computing based pattern generator. The rotary encoders in the robot leg are used for feedback. On the right (b) an overview of the controller we use to modulate the pattern generator (the plant) is shown.

4 Modulatable Pattern Generator (MPG)

As we discussed in Section 2 the robot is controlled directly by a pattern generator which is able to imitate demonstrated rhythmic motions. The pattern generator is implemented in a RC-network. In previous work we showed that by using additional inputs, the generated patterns can be modulated [16]. This work was extended in [17] by adding the capability to encode discrete and rhythmic motion patterns into a single recurrent neural network as respectively a limit cycle and a fixed point attractor. Typical parameters that are used for this are summarized in Table 1. More recently, in [18] a new method to modulate the shape of a learned rhythmical pattern was illustrated based on tuning the bias weights of the neurons instead of using additional inputs. In summary: the influence of adding a small bias to each neuron is determined after training. Therefore, each neuron is perturbed separately with a small constant bias. After perturbing each neuron, one can observe the influence of this on the properties of the output signal. For each property that one wants to track, a control vector can be composed which can be used to modulate the output signal. In this work we will use this modulation approach to change the properties of the generated signal. We only determine the control vector that influences the signal amplitude and offset. More complex property transformations will be shown in future work.

5 Controller

In [19] and [20] we introduced a novel feedback controller which learns to control a plant (dynamical system) by online learning an inverse plant model based on real-time controlled plant-input/output pairs. In parallel, this preliminary model is used to actually control the system, producing a new plant-input output pair

Table 1. Summary of all parameters in a pattern generating RC-network with their typical values

Parameter	Description	Value
N	number of neurons	100 to 2000
λ	leak-rate	0.01 to 1
ρ	spectral radius	0.99 to 1.5
β	bias weight variance	0 to 1
ω	output feedback scale	0 to 10
α	learning rate, only FORCE learning case, determines initialization of $\mathbf{P_{init}} = \frac{\mathbf{I}}{\alpha}$	0.1

which gradually improves the inverse model. At the core of this feedback controller we use a RC-network to accommodate the inverse model. However, as described in [20], any dynamical system with a high dimensional state representation can be used to accommodate such model as well. In this paper we apply the same feedback controller (shown in Fig. 2(b)) to modify the bias vector to MPG neurons, which are sensitive to the amplitude and offset of the generated motion. Although the MPG is fully responsible for the produced motion, the controller allows the MPG to track a desired amplitude and offset which are changing more slowly compared to the position of the motor.

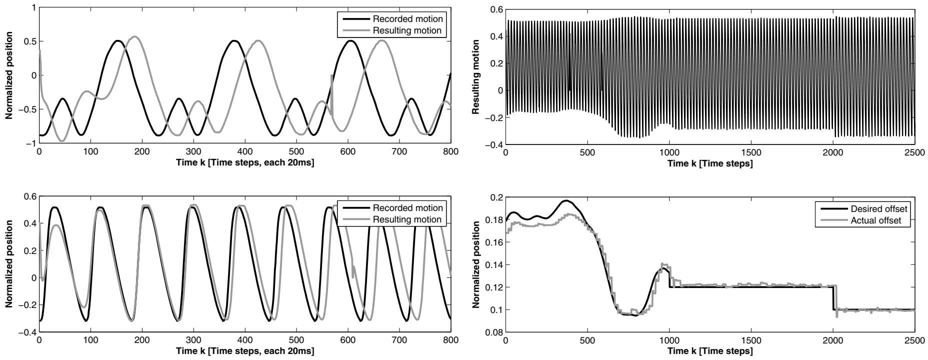
As shown in Fig. 2(b), the feedback controller uses two identical RNN of which only the inputs differ. The output weights of Network A are trained (applying RLS) by observing plant-input/output pairs $((y(t-\delta), y(t)), (x(t-\delta)))$. These output weights are used by Network B to produce a plant-input $x(t)$ given the actual and desired plant-output, $y(t)$ and $\hat{y}(t+\delta)$ respectively.

6 Experiments

In this section we apply the discussed control hierarchy on a prototype robot leg (of the Oncilla robot platform) which is developed in the AMARSi consortium and shown in Fig. 1(b). This robot leg is controlled by a motor control board which in turn is driven by a small computer. However, because of the computational limitations of this onboard computer and to ensure the desired communication timings, all calculations are offloaded to a much more suited computational unit. In this work we want to demonstrate the above described control hierarchy concept on a simple task. Although the control of multiple servos is possible, we will limit the amount of controlled servos to 1. The robot leg is controlled by a simple P-controller which converts the positions, generated by the MPG, to a torque signal. However, to allow for changes in the robot dynamics to be visible in its motion, the used P-parameter is smaller than optimal and the amount of torque is limited. In Table 2 we show the associated parameters concerning the used feedback controller and MPG setup. Additionally, the different timings are given at which each system is interacting with another system. As mentioned before, the used feedback controller is interacting at a much slower rate compared to the MPG's control rate.

Table 2. Summary of all our setup parameters used in the experiments

Parameter	Pattern generator	Controller
N	500	500
λ	0.14	1.
ρ	1.4	1.
α	0.1	0.01
time scale	20ms	100ms
input scaling	1.0	0.1
β	0.5	0.5

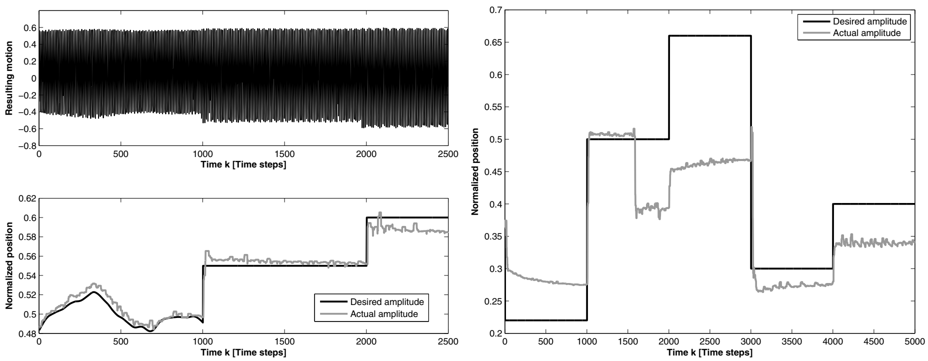


(a) (b)

Fig. 3. (a) Shows two different recorded motions together with the actual reproduction by the robot leg. (b) Depicts the actual motion during offset control (top) together with the desired and actual offset (bottom).

6.1 Learning by Imitation

By limiting the servo's torque the robot leg can be back-driven, allowing the demonstration of a given motion. In this work we impose a mixed sinusoidal motion which afterwards is used to train the MPG-network. When training is completed, the necessary gradient vectors to the MPG-neurons that affect the amplitude and offset, are computed. In Fig. 3(a) the actual trained leg motion is shown for two different imposed patterns which are shown as well. The first pattern is a mixed sine pattern while the second motion is a sinusoidal pattern with a slower and faster phase. The latter is similar as in a swing/stance phase gait. Both resulting motions show a phase shift caused by integrating the robot leg into the feedback loop. A change in the generated pattern has to propagate through the dynamics of the robot leg, before the correct leg angle is visible for the leg encoder. Because of the feedback loop, this variating delay eventually affects the generated pattern. This illustrates that a higher level control is necessary to modulate the pattern generator such that these dynamics are taken into account.



(a)

(b)

Fig. 4. (a) Depicts the actual motion during amplitude control (top) together with the desired and actual amplitude (bottom). (b) This plot illustrates how the proposed hierarchy reacts to changes in the dynamics of the robot leg during amplitude control.

6.2 Motion Modulation by Controlling the MPG

After learning the recorded motion, the feedback controller is applied to modulate the amplitude and offset of the motion, which are only observable on a slower time scale. As mentioned before, this can be achieved by controlling the bias of amplitude/offset responsive MPG-neurons. Fig. 3(b) shows the desired and actual offset which is controlled by the feedback controller. To control the offset, the highest level of our proposed hierarchy was interacting with the MPG every 100 ms (5 times slower than the interaction rate of the MPG). Fig. 4(a) demonstrates a similar experiment but for amplitude control. Additionally, the actual resulting positions are depicted at the bottom of both Fig. 3(b) and 4(a).

6.3 Adapting to Changes in Robot Dynamics

In the previous experiment we showed that the generated motion can be modulated. However, we want to investigate the capability of the proposed hierarchy to adapt to changes in the dynamics of the robot or in its environment. In our experimental setup we are limited in introducing changes to adding weight to the robot leg. We increase its mass by hanging an extra weight at the tip of the leg. As a result, the amplitude of the motion will be reduced and the offset will move closer to the lowest point of the leg. However, this switch in dynamics will cause the inverse model of the feedback controller to adapt to these changes as well. As a result, during amplitude control the amplitude will eventually converge again to its desired value. In Fig 4(b) after 1500 time steps a mass of 100 g is added to the leg. After adjusting its internal model, the controller start compensating for the extra weight at time step 3000 by controlling the bias of the MPG.

7 Conclusions

Roboticists are often inspired by biology to improve the abilities of robots. One of the main directions is the use of central pattern generators which reduce the dimensionality of the locomotion control problem. Inspired by this, we proposed the use of a multi-timescale hierarchical controller that uses random dynamical systems for each layer. On the lowest level, a pattern generator is able to embed any rhythmic signal. This pattern generator interacts directly with the motor of the leg on a fast timescale. On the highest level, and thus slowest time scale, a controller tunes the parameters of the pattern generator online such that it keeps track of the slow varying parameters of the resulting motion. To achieve this, the sensor information presented to the controller is preprocessed by calculating for example the amplitude and offset. Since the controller acts on a slower timescale, this controller can be very advanced and might consist of a very large random dynamical system. On the other hand, the pattern generator is fast enough to react immediately on small perturbations which can not be compensated by the morphology of the robot (passive compliance).

By means of three preliminary experiments on the AMARSi Oncilla leg, we validated the proposed control hierarchy. In a first experiment we showed that the hierarchy is able to capture a (by hand) shown rhythmic motion pattern which is embedded by the pattern generator. In the second experiment we illustrate that the higher level controller is able to track slow varying properties such as amplitude and offset, by only controlling the bias of the pattern generator. Finally, in the third experiment, we demonstrated that the control hierarchy is able to deal with new situations such as changes of the leg weight.

We are now planning to apply the proposed control hierarchy on the in the AMARSi consortium developed quadruped Oncilla robot. We will mainly try to tackle two problems: (1) controlling the stability of the robot by considering a measure of stability as the slow control property to track, and (2) we will embed multiple gaits for the Oncilla robot.

Acknowledgment. This work was partially funded by a Ph.D. grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) and the FP7 funded AMARSi EU project under grant agreement FP7-248311.

References

1. Kiebel, S., Daunizeau, J., Friston, K.: A hierarchy of time-scales and the brain. PLoS Computational Biology 4(11), e1000209 (2008)
2. Stein, P.: Neurons, networks, and motor behavior. The MIT Press (1999)
3. Ijspeert, A.: Central pattern generators for locomotion control in animals and robots: a review. Neural Networks 21(4), 642–653 (2008)
4. Cohen, A., Wallen, P.: The neuronal correlate of locomotion in fish. Experimental Brain Research 41(1), 11–18 (1980)

5. Rossignol, S.: Locomotion and its recovery after spinal injury. Current Opinion in Neurobiology 10(6), 708–716 (2000)
6. Ijspeert, A., Crespi, A., Ryczko, D., Cabelguen, J.: From swimming to walking with a salamander robot driven by a spinal cord model. Science 315(5817), 1416–1420 (2007)
7. Verstraeten, D., Schrauwen, B., D'Haene, M., Stroobandt, D.: An experimental unification of reservoir computing methods. Neural Networks 20, 391–403 (2007)
8. Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks. German National Research Center for Information Technology, Tech. Rep. GMD Report 148 (2001)
9. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Computation 14(11), 2531–2560 (2002)
10. Antonelo, E.A., Schrauwen, B., Stroobandt, D.: Event detection and localization for small mobile robots using reservoir computing. Neural Networks 21, 862–871 (2008)
11. Jaeger, H., Haas, H.: Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. Science 308, 78–80 (2004)
12. Schrauwen, B., D'Haene, M., Verstraeten, D., Van Campenhout, J.: Compact hardware liquid state machines on FPGA for real-time speech recognition. Neural Networks 21, 511–523 (2008)
13. Paquot, Y., Duport, F., Smerieri, A., Dambre, J., Schrauwen, B., Haelterman, M., Massar, S.: Optoelectronic reservoir computing. Scientific Reports 2, 1–6 (2012)
14. Legenstein, R.A., Maass, W.: Edge of chaos and prediction of computational performance for neural microcircuit models. Neural Networks, 323–333 (2007)
15. Sussillo, D., Abbott, L.: Generating coherent patterns of activity from chaotic neural networks. Neuron 63(4), 544–557 (2009)
16. Wyffels, F., Schrauwen, B.: Design of a central pattern generator using reservoir computing for learning human motion. In: Proceedings of the ECSIS Symposium on Advanced Technologies for Enhanced Quality of Life, pp. 118–122 (2009)
17. Waegeman, T., Wyffels, F., Schrauwen, B.: A discrete/rhythmic pattern generating RNN. In: Proceedings of the 20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 567–572. Ciaco - i6doc.com, Louvain-la-Neuve (2012)
18. Li, J., Jaeger, H.: Minimal energy control of an ESN pattern generator. Jacobs University, Tech. Rep. (2011)
19. Waegeman, T., Schrauwen, B.: Towards Learning Inverse Kinematics with a Neural Network Based Tracking Controller. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part III. LNCS, vol. 7064, pp. 441–448. Springer, Heidelberg (2011)
20. Waegeman, T., Wyffels, F., Schrauwen, B.: Feedback control by online learning an inverse model. IEEE Transactions on Neural Networks and Learning Systems (submitted)

Modelling Walking Behaviors Based on CPGs: A Simplified Bio-inspired Architecture

Cai Li, Robert Lowe, and Tom Ziemke

Interaction Lab., University of Skövde, Sweden
{gauss.lee,robert.lowe,tom.ziemke}@his.se

Abstract. In this article, we use a recurrent neural network including four-cell core architecture to model the walking gait and implement it with the simulated and physical NAO robot. Meanwhile, inspired by the biological CPG models, we propose a simplified CPG model which comprises motorneurons, interneurons, sensor neurons and the simplified spinal cord. Within this model, the CPGs do not directly output trajectories to the servo motors. Instead, they only work to maintain the phase relation among ipsilateral and contralateral limbs. The final output is dependent on the integration of CPG signals, outputs of interneurons, motor neurons and sensor neurons (sensory feedback).

Keywords: CPGs, the NAO robot, Interneuron, Motorneuron.

1 Introduction

Central pattern generators have been investigated in robotics applications for many years now. Roboticists have proposed many useful but different CPG models. Generally these CPG models mainly focus on the following three aspects: a. integration of Sensory feedback: the famous sensor-driven Runbot[2] is a very interesting example of this; b. topology of CPG network: this refers to the number and alignment of different CPG neurons. A good permutation of CPG neurons not only simplifies the control problems but also guarantees the stability of the whole network[11]. Using group theory is a very good example on this; c. the connections of CPG neural networks[15]: the models focusing on this often use exploration-exploitation algorithms(e.g.reinforcement learning[8], genetic algorithm[7]) to find out the proper weights within this network. In this approach, the sensory feedback and even the topology are both considered to be determined by the algorithm. However, this also involves too many dimensions so that the convergence of the algorithm becomes uncertain. In order to find a simplified architecture to contain all the three aspects, we propose a bio-inspired architecture integrating them. This architecture (Figure.1) is inspired by the findings of Amrollah[1] and the work of Nassour[9].

As a model inspired from the neurophysiological spinal structure of human beings[4], we simplify the whole structure into a hierarchical model with three main parts. The core CPG architecture previously used to model crawling and infant early walking behaviors[5,6] is located in the spinal cord, only maintaining

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 156–166, 2012.

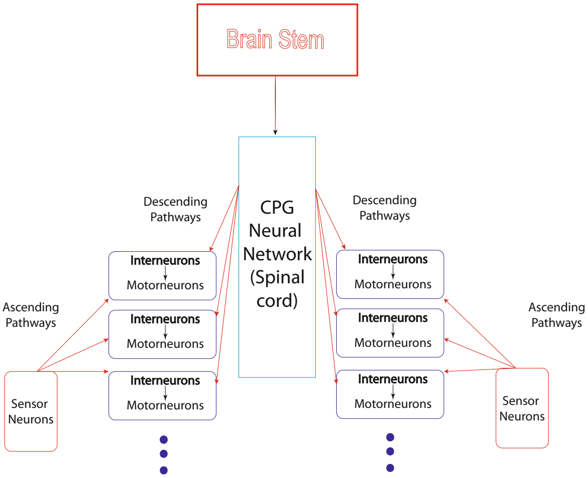


Fig. 1. The brainstem emits the control signal to adjust the oscillatory patterns of CPGs located in the spinal cord. The ascending and descending pathways are communication channels between environment and the robot. Each block including interneurons (IN) and motorneurons (MN) represents the control flow from IN to MN which integrates sensory feedback in order to reshape the CPG signals. So IN neurons are called pattern formation neurons as well. The whole architecture is inspired by Amrollah's CPG model[1] and human neurophysiology.

phase relations. The CPG signals go through the interneurons (IN) and motorneurons (MN) which integrate two kinds of sensory feedback: proprioceptive and exteroceptive feedback. Both of these two types of feedback are acquired via sensors of the robot through sensor neurons. Finally the output to each joint is entirely reshaped by the IN and MN. From the perspective of neurophysiology, this model includes two pathways: the ascending pathway and the descending pathway. Both of these two constitute the interaction between humans and environment. In the same way, the robot can interact with the sensed world via its sensors and form stable walking gaits.

In this article, a complete model of CPG is proposed and explained in the second part. The third part covers the functions of IN and MN and how they are simply modelled. The implementation on both simulated and physical robots is analyzed in the fourth part. Finally some conclusions and future work are proposed.

2 The CPG Neural Network

In this work, the use of CPG neural networks is to maintain phase relations among all the joints. In our model, a four-cell core architecture is used to produce a primitively stable anti-phase relation between contralateral limbs (both legs and arms). Since this four-cell architecture has been proved to be structurally stable[11], the phase relation between arms and legs do not change. On the other hand, even though the four-cell network is simple, it has been sucessfully used to implement crawling behaviors[6] and our work could be regarded as an extended or topologically "developed" model of the core architecture. Actually, the main 4-cell architecture is very general in controlling some other four-limb robots, like the salamander robot[3] and puppy robot[12].

In our model, we use another two pairs of CPGs to control the knee and ankle parts of the NAO robot. The schema of our extended full architecture is shown in Figure.2. Each CPG neuron is modelled by a Hopf oscillator (see equations(1)(2)(3)) with adjustable swing and stance phase.

$$\dot{x_i} = a(m - {x_i}^2 + {y_i}^2)y - w_i x_i \tag{1}$$

$$\dot{y_i} = a(m - {x_i}^2 + {y_i}^2)x - w_i y_i + \sum \alpha_{ij} y_j \tag{2}$$

$$w_i = 2 \cdot \pi(\frac{w_{swing}}{(1 + e^{-100y_i})} + \frac{w_{stance}}{(1 + e^{100y_i})}) \tag{3}$$

Here x and y are the two dimensions of Hopf oscillator.w is the total frequency of the oscillator which is controlled by w_{swing} and w_{stance}. w_{swing} and w_{stance} determine decrease and increase speed of the periodic signal. a is the convergence rate and m is the amplitude of this oscillator. The advantange of this oscillator is that the speed of upward and downward wings of the periodic signals can be changed according to different requirements of locomotions. α_{ij} is the connection weight from neuron i to j.

In the CPG network, the knee and ankle parts are synchronized with the main joints (hips and arms) of which the anti-phase movement has been developed during early infancy[14]. According to dynamic systems theory, how each joint moves in a walking behavior depends on the interaction with the physical body and the environment. Even though the early infant walking could be modelled simply based on the signals directly from CPG outputs[5], with the development of muscle and neural systems, infants start to change their walking patterns with the maturation of their sensorimotor systems. Therefore, after integration with sensory feedback, the CPG signals are futher reshaped to be adaptive to the posture change and environmental alteration.

3 Modelling of Interneurons and Motorneurons and Sensory Feedback Integration

Interneurons and motorneurons are very helpful on postprocessing CPG signals and preprocessing sensory feedback. Interneurons, in most vertebrates or invertebrates, are mainly in charge of reshaping CPG outputs with feedback. Meanwhile, connected to muscle fibers, motorneurons directly integrate outputs of all the interneurons (pattern reform neurons) with muscle extremities. Even in a very primitive mollusc called Clione lamicina with only two main CPGs working together with interneurons and motorneurons, a swimming behavior can be

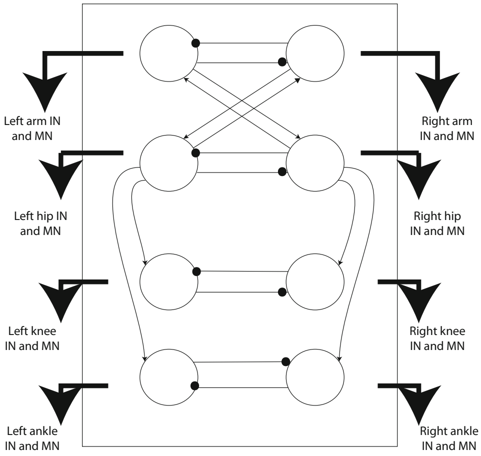


Fig. 2. The CPG neural network used to model the spinal cord part. Each neuron is an independent oscillator. The upper four neurons form the core four-cell architecture. Knee and ankle parts are synchronized by two independent pairs of anti-phase oscillators. The output of each CPG neuron is directly sent to interneuron level to integrate with sensory feedback. This schema here only depicts the sychronization of all the *pitch* joints. There are two kinds of connections in this recurrent neural network. One is the subtractive connection represented by lines with dot ends. The other is additional connection represented by arrow lines.

flexibly formed[10]. A generic bio-inspired complete CPG model has been identified by Rybak[13] and Amrollah[1](see Figure.3). Inspired from their work and investigations, we simplify the model and embed our CPG architecture in it.

In our work, the interneurons and motorneurons are all modelled by sigmoid functions of which the nonlinearity is extremely useful in reshaping CPG outputs. The two rythmic generators (Figure.3) are replaced by our two-layer RGs for each joint (See Figure.4). The sensor neurons are modelled by sigmoid functions which are used to normalize all the sensor values between 0 and 1. This idea is inspired by Geng's work on Runbot[2].

There are two kinds of sensor neurons in this model. One is the proprioceptive sensor neuron and the other is the exteroceptive sensor neuron. The ES (extensor sensor) and FS (flexor sensor) neuron (equations(4)(5)) are both the ones acquiring proprioceptive information from each joint. In our work, the inputs of these neurons are joint values obtained from joint sensors. In our implementation, we set the $\theta_{threshold}$ to 0 since the mutually opposite sigmoid functions limit the joint motion in a certain scope according to the requirement (See Figure.6). Actually, in robotic systems, each joint of humanoids has its own extension and flexion extremities in different behaviors in order to stablize the limit-cycle attractors of the whole body. When a joint is close to the extensor extremity, the flexor is autonomously activated to pull back the joint.

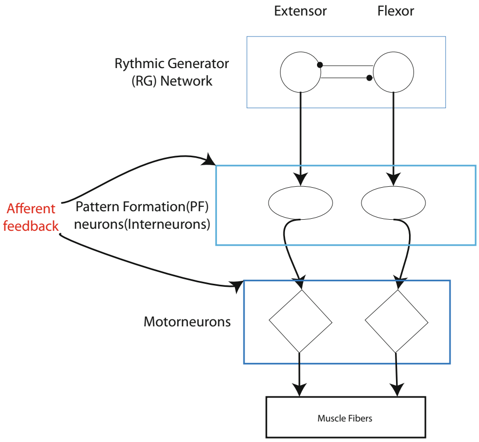

Extensor
Flexor
Rythmic Generator (RG) Network
Afferent feedback
Pattern Formation(PF) neurons(Interneurons)
Motorneurons
Muscle Fibers

Fig. 3. The locomotor CPG consists of a rhythm generator (RG) and pattern formation (PF) neurons. The RG defines the locomotor rhythm and the durations of flexor and extensor phases and controls the activity of the PF neurons. The PF neurons represent the interneurons each of which provides excitation to multiple synergist motoneurons. Afferent feedback and spontaneous perturbations may affect the CPG either at the level of the RG, producing alterations (e.g. phase shifting or resetting) of the locomotor rhythm, or at the level of PF, altering the level of motoneuron activation and/or the timing of phase transitions without shifting the phase of (or resetting) the locomotor rhythm generated by the RG[1].

$$ES = 1 + \frac{1}{a(1 + e^{\theta_{threshold} - \theta_{input}})} \tag{4}$$

$$FS = 1 + \frac{1}{a(1 + e^{\theta_{input} - \theta_{threshold}})} \tag{5}$$

The AES (Anterior Extension Sensor), RFC (Rear Foot Contact), FFC (Front Foot Contact), BAS (Backward Angle Sensor) and FAS (Forward Angle Sensor) neurons' activations are all transformed by sigmoid functions with thresholds but have different use. The AES neuron is used to detect the joint extremity of the hip during the stance phase. When hip reaches the anterior extremity, it activates the extensor of knee to make it contract immediately. The AES neuron has also been proved to be a very necessary sensor neuron in both human and robotic walking[2]. The RFC and FFC neurons can make the foot contact with the ground more naturally. In our simulated robot, we find that if the RFC and FCC neurons are removed from the robot, the heel-strike and heel-raise behaviors which are considered to be the characteristics of humanoid walking will disappear. BAS and FAS are sensor neurons getting data from "the vestibular system" (gyro sensors) in the NAO robot. With these two neurons, the robot can autonomously change its ankle posture in order to keep the body in balance.

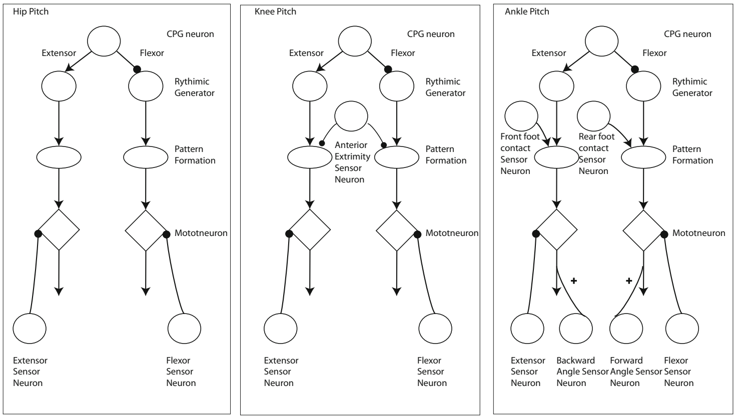


Fig. 4. The neural-mechanical controller used for each joint specifically. Different joints have different senory feedback. The arrow line is the additional connection and the dot-end line is the subtractive connection. The CPG neuron from the spinal cord is further seprated into two anti-phase rythmic generators. The architecture below the RGs is inspired by Amrollah's model[1] and Nassour's work[9] for adaptive walking.

Finally, the importance of using thresholds in different neurons should be highly emphasized. The threshold should be carefully tuned according to their own inputs. Furthermore, the saturation of sigmoid functions should be avoided in all the abovementioned neurons (see Figure.5). We must tune the change rate a of the sigmoid function to an appropriate value in order that at least 90% of the "S" shape can be used.

4 Gait Analysis on the Simulated and Physical Robot

We test the simplified architecture of CPG network on the simulated and physical robot. The simulator used is the NAO simulator which is specifically designed for the NAO robot (visit http://www.alderbaran-robotics.com). In order to make the robot walk, we need to extend the 2D model to 3D walking model by involving a proper roll motion. For the NAO robot, it has two roll joints for hip and ankle respectively. We use two approaches to model it: one is the sensor-driven without RGs and the other is by RGs (See also Figure.7).

4.1 The Two Different Walking Gaits

Firstly, we implement the sensor-driven roll motion and the robot generates one walking gait (see Figure.8). This walking gait has very interesting properties: a. when the CPGs are oscillating slowly, the amplitude of each joint should be increased in order to make the robot be able to walk including pitch and roll joints. b. when you change the stance and swing durations and keep the $w_{stance} + w_{swing}$ equal to a constant, the walking gets faster.

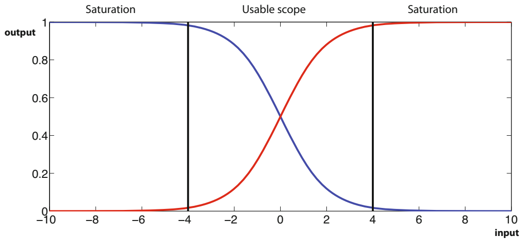


Fig. 5. Sigmoid function with its usable scope and saturation ($\theta_{threshold} = 0$ in equation 4 or 5)

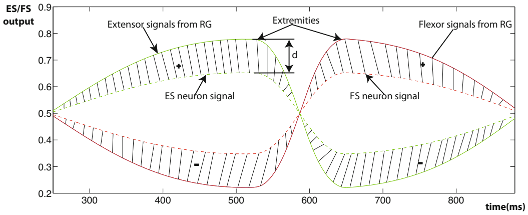


Fig. 6. This figure shows how the ES and FS neurons work together to limit the joint value when it gets to extremity (here we show the hip neurons as an example). When the RG signal is supposed to make joint move to its maximum value, the ES or FS neurons generate maximum value as well to limit this movement. Actually, according to different applications, the limited distance d could be tuned to different values if the change rate a of ES and FS neurons is changed.

We could clearly see the roll motion plays a very important role in this walking behavior. This is the difference between 2D and 3D modelling of walking behavior. The robot's center of gravity moves between left and right legs so that the swing leg is able to swing swiftly. Actually, when we implement this behavior on the physical robot, all the parameters need to be retuned to adapt to the physical body's dynamics which is quite distinct from the simulated robot which can even run in the simulator.

Subsequently, we use the generic roll motion with RGs to replace the sensor-driven roll motion. Interestingly, a entirely new walking gait emerges (See Figure.9). The big difference of this gait compared to the previous one is how the roll motion synchronized with the pitch motion. The former is more similar to a running gait as when the body dynamics change very fast, the stance leg will lose contact with the ground ahead of time. However, the latter does not have this characteristic. Similary, this walking gait also has the same properties (a and b) as the first one.

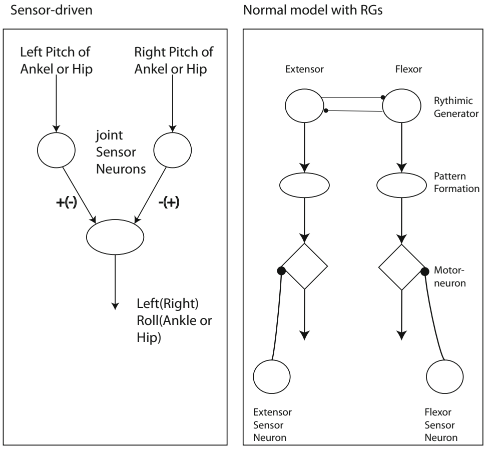


Fig. 7. Left: the sensor-driven roll motion. The two sensor neurons obtain the real-time values of correspondent pitch joints to form roll motion (hip or ankle). The left and right roll joint values should be opposite to each other. Right: use two anti-phase RGs for each roll joint (hip or ankle) and these two RGs are synchronized with the CPG network of pitch motion. Similarly, the left and right roll's values are opposite.

How can we change the robot's walking speed? For both of these two walking gaits, either change the frequency of the CPGs or change the stance duration. The higher the CPG frequency is, the faster the walking is. The shorter the stance duration is, the faster the walking is too. Normally, the frequency is tuned to a certain value which suits the body dynamics of the robot and the stance phase is shorten to make the robot walk faster. In all the above implementation, the stance and swing phase are numerously tuned. The best performance is $w_{stance} = 3w_{swing}$.

4.2 The Ankle Joint with Sensor Neurons

As abovementioned in the third part, the FFC and RFC neurons are important to form nature humanoid walking behaviors. The heel-strike and heel-raise , these two characteristics of human walking occur if we add a strong sensory feedback on the ankle from FFC and RFC (Figure.10). In the previous implementation, decreasing the influence of FFC and RFC on the ankle is necessary because of the NAO robot's disproportionately big feet. Therefore, if the heel-strike and heel-raise are too obvious, it will distablize the walking gait.

We compare the output of MN without any sensory feedback, with sensory feedback from gyro sensors and with strong sensory feedback from both of gyro sensors and foot contact sensors and we can clearly see the FFC and RFC can totally reshape the ankle CPG and make it entirely adaptive to the robot's body dynamics (Figure.11).

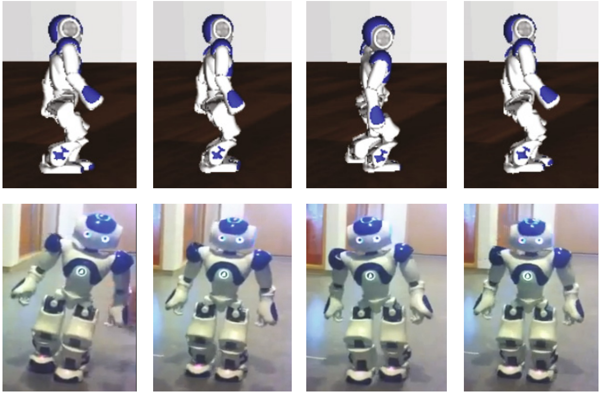


Fig. 8. Upper: walking gait of the simulated NAO robot Below: walking gait of the physical NAO robot. This walking gait is with sensor-driven roll motion.

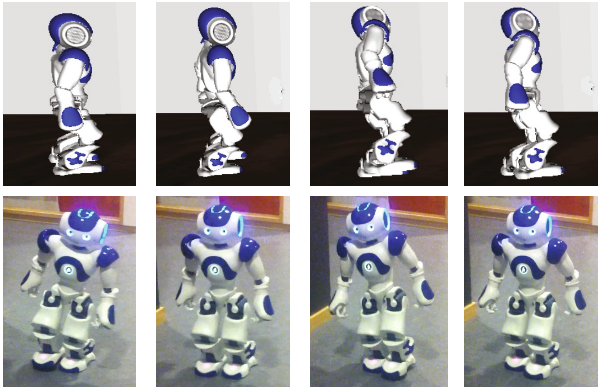


Fig. 9. Upper: walking gait of the simulated NAO robot Below: walking gait of the physical NAO robot. This walking gait is performed with a full architecture of roll motion based on RG-PF-MN mechanism.

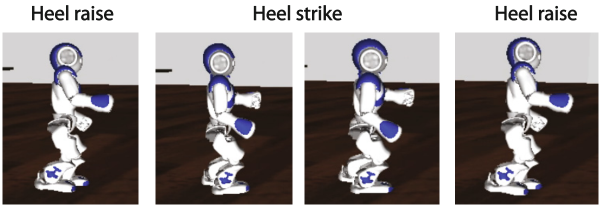


Fig. 10. The walking gait with strong sensory feedback from FFC and RFC and the appearance of heel-strike and heel-raise behaviors

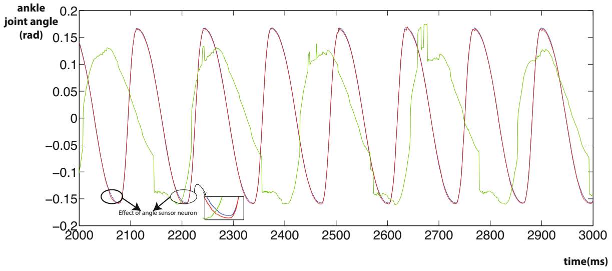


Fig. 11. The ankle dynamics during walking: the red line is the output without any sensory feedback. The blue line (almost overlapped with red line) is the output with sensory feedback from BAS and FAS (as for the influence of sensory feedback, please see the zoomed-out snapshot). The green line is the output totally reshaped by FFC and RFC neurons after the sensory feedback integration.

5 Conclusions and Future Work

In this work, we propose a relatively complete CPG architecture inspired from neurophysiological structure. Compared to the previous work on modelling early infant walking[5], this model provides specific involvement of sensory feedback which can increase adaptivity of walking. With respect to the walking gait, our model generates two kinds of walking behaviors based on tuning the parameters of IN and MN respectively. Both of this two gaits are formed on the basis of two categories of sensory feedback: proprioceptive and exteroceptive feedback. The former is about position feedback of each joint and the latter is more focused on external interaction with the environment. Meanwhile, a very simple mechanism analogue to the vestibular system is used to adjust the posture of ankle so that the stable upright posture can be maintained. Based on all these points, our model captures most of salient sensory information for adaptive walking. Since the visual system which is also very important for posture and balance control is not covered in our system, to some extent, adaptivity of our model is limited.

The oscillatory recurrent neural network is used in this model because not only is it structurally stable but also its synchronization based on group theory is very useful on different robotic platforms[11,3]. Therefore, this model could be considered as a generic model. On the other hand, compared to traditional engineering approaches on modelling humanoid walking, like the ZMP method, the advantage of our model is its simplicity. However, in our model, we need to carefully tune a lot of parameters in IN and MN to get nice walking gaits. This part will be replaced by a learning mechanism in future work. With a learning system based on a dynamic systems approach, the NAO robot forms its own walking behaviors based on the sensory feedback and maintains the limit-cycle attractors of the whole body motion.

References

1. Amrollah, E., Henaff, P.: On the role of sensory feedbacks in rowat–selverston cpg to improve robot legged locomotion. Frontiers in Neuroscience 4 (2010)
2. Geng, T., Porr, B., Worgotter, F.: Fast biped walking with a sensor-driven neuronal controller and real-time online learning. Journal of Robotics Research 25, 243–259 (2006)
3. Ijspeert, A.J.: Central pattern generators for locomotion control in animals and robots: a review. Neural Networks 21(4), 642–653 (2008)
4. Latash, M.L.: Neurophysiological Basis of Movement. Human Kinetics Publishers (1998)
5. Lee, G., Lowe, R., Ziemke, T.: Modelling early infant walking: Testing a generic cpg architecture on the nao humanoid. In: Proceedings of Development and Learning (ICDL) 2011 IEEE International Conference, Frankfurt, Germany (October 2011)
6. Li, C., Lowe, R., Duran, B., Ziemke, T.: Humanoids that crawl: Comparing gait performance of icub and nao using a cpg architecture. In: Computer Science and Automation Engineering (CSAE), Shanghai, China (May 2011)
7. Liu, C., Chen, Q., Wang, D.: Biped robot walking using central pattern generator and genetic algorithm. In: Proceeding of International Conference on Robotics (2010)
8. Nakamura, Y., Mori, T., Sato, M., Ishii, S.: Reinforcement learning for a biped robot based on a cpg-actor-critic method. Neural Netw. 20(6), 723–735 (2007)
9. Nassour, J., Hénaff, P., Ben Ouezdou, F., Cheng, G.: A Study of Adaptive Locomotive Behaviors of a Biped Robot: Patterns Generation and Classification. In: Doncieux, S., Girard, B., Guillot, A., Hallam, J., Meyer, J.-A., Mouret, J.-B. (eds.) SAB 2010. LNCS, vol. 6226, pp. 313–324. Springer, Heidelberg (2010)
10. Orlovskii, G.N., Deliagina, T.G., Grillner, S.: Neuronal control of locomotion: from mollusc to man. Oxford University Press (1999)
11. Righetti, L., Ijspeert, A.: Design methodologies for central pattern generators: an application to crawling humanoids. In: Proceedings of Robotics: Science and Systems, Philadelphia, USA (August 2006)
12. Rutishauser, S., Spröwitz, A., Righetti, L., Ijspeert, A.J.: Passive compliant quadruped robot using central pattern generators for locomotion control. Physica 577.2, 617–639 (2006)
13. Rybak, I.A., Shevtsova, N.A., Lafreniere-Roula, M., McCrea, D.A.: Modelling spinal circuitry involved in locomotor pattern generation: insights from deletions during fictive locomotion. The Journal of Physiology Online 577, 617–639 (2006)
14. Thelen, E., Smith, L.B.: A dynamic systems approach to the development of cognition and action. MIT Press, Boston (1994)
15. Vaal, J., Van Soest, A.J., Hopkins, B.: Modelling the early development of bipedal locomotion: A multidisciplinary approach. Human Movement Science 14(4-5), 609–636 (1995)

Plastic Representation of the Reachable Space for a Humanoid Robot

Marco Antonelli, Beata J. Grzyb, Vicente Castelló, and Angel P. del Pobil

Robotic Intelligence Lab
Universitat Jaume I, Castellón, Spain
{antonell,grzyb,castellv,pobil}@uji.es

Abstract. Reaching a target object requires accurate estimation of the object spatial position and its further transformation into a suitable arm-motor command. In this paper, we propose a framework that provides a robot with a capacity to represent its reachable space in an adaptive way. The location of the target is represented implicitly by both the gaze direction and the angles of arm joints. Two paired neural networks are used to compute the direct and inverse transformations between the arm position and the head position. These networks allow reaching the target either through a ballistic movement or through visually-guided actions. Thanks to the latter skill, the robot can adapt its sensorimotor transformations so as to reflect changes in its body configuration. The proposed framework was implemented on the NAO humanoid robot, and our experimental results provide evidences for its adaptative capabilities.

1 Introduction

Humans live surrounded by objects. Reaching for an object is one of the most common tasks in everyday life. As robots are expected to be active participants in humans' daily life, they also need to have good reaching skills. Moreover, the robots need to be able to constantly learn and autonomously improve their reaching abilities so as to act on unknown objects in new environments or adapt to the changes in their body configurations.

Reaching a target, however, is not an easy task. It requires estimation of the spatial position of the target and its transformation into an apropriate arm motor command. Estimation of the object position is problematic on its own as a three dimensional object is projected into two dimensional surface of camera sensor, which in turn causes the distance to the target to be lost. The common solution is to employ stereopsis to reconstruct the depth of the scene. Human beings, however, are clearly able to perform reaching actions even with a single functioning eye and we are interested in replicating this phenomenon.

Another challenge here is the transformation of the object's spatial location into the arm position that allows reaching the target. These transformations are typically computed analytically by using the known geometric properties of the robotic system provided by the robot's manufacturer or estimated empirically.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 167–176, 2012.

This approach to reaching permits to achieve good performance, but only under the assumption that the parameters of the system are time invariant. In practice, it is not always the case, and the system needs to be re-calibrated periodically in order to keep working correctly. Therefore, it is convenient to develop a framework that continuously adapt the sensorimotor mapping to the constantly changing robot parameters.

In previous works, we proposed a framework for the implicit sensorimotor mapping of the peripersonal space that was implemented on a humanoid torso endowed with a pan-tilt vergence stereo head and two multi-joint arms [3,1]. Instead of using the classical cartesian space, the spatial position of the target was encoded by the gaze direction and by the angular position of the arm joints. Indeed, these variables were implicit because they were directly provided by proprioceptive cues (encoders). This paper presents our reaching framework extended and adapted to work on a monocular robotic setup.

As our previous framework was based only on the depth information provided by stereo cameras, the first objective of this work is to modify the network so as it makes use of distance estimation provided by a monocular camera. Thus, in the proposed adaptation, the target position is represented by the gaze direction that allows bringing the target into the fovea together with the distance to the target. The same position is expressed in terms of the arm posture that allows reaching the target. The direct and inverse transformations between the two frames of reference are learned autonomously by the robot during the exploration of the environment. The results of our computer simulations and robot experiments show that the robot is able to reach correctly the target both by using direct transformation and by visually-guided approach.

Moreover, in this paper we investigate the ability of the system to adapt to the changes in the robot kinematics. Once the robot had been trained to reach the target, its body configuration was changed, that is the position of its elbow joint was rotated about 20 degrees. As this position was assumed to be an invariant configuration of the system, the robot had to autonomously update its sensorimotor maps to reach correctly the target object. The results obtained from our experiments with the robot, showed that the system is able to instantly update its sensorimotor maps to reflect the changes of its body configuration.

The paper is structured as follows. The next section briefly presents the neuroscientific findings that inspired our work and compares our approach to the existing works. The subsequent section describes how the target can be implicitly encoded by the robot sensorimotor maps, which is then followed by the description of the computational model and learning strategies. The next section shows our experimental setup and results obtained from both computer simulations and real robot experiments. We close the paper with the discussion of the results and future work.

2 Background

Our approach to the sensorimotor transformation problem is inspired by neuroscientific findings, mainly concerned with human and primates' brain. Two types of visual processing exist in the brain, that is visual processing to obtain information about the features of objects such as color, size, shape ("vision for perception") in the ventral stream of visual cortex, and visual processing needed to guide movements such as reaching and grasping ("vision for action") in the dorsal stream of visual cortex [9,4]. The main cortical areas related to reaching actions are V6A and MIP [8,5,2], both located in the parietal lobe. Findings in V6A neurons showed neurons that encoded the gaze directions and the distance of the target [6,15]. Moreover, some neurons seemed to be involved in the execution of reaching movements [8]. These findings indicate that V6A is in charge of performing the sensorimotor transformations required for reaching a given target in 3D space.

The radial basis function networks are suitable to model the parietal cortex neurons as they are able to naturally reproduce the gain-field effects often observed in parietal neurons [20]. Moreover, it was suggested that locations of objects in the peripersonal space are coded through the activity of parietal neurons that act as basis functions, and any coordinate frame can be read out from such population coding according to the task requirements [19]. Because of their biological plausibility, and their ability to approximate any kind of non-linear function [17], the direct and inverse transformations in our framework are encoded by two radial basis function networks (RBFNs).

In robotics, even though extensive literature describes the problem of learning eye-hand coordination [10,7,16,14,21], to the best of our knowledge only few papers describes the use of RBF networks [14,21]. Our model differes from these works in various points. For example, Marjanovic et al. learned the transformation only on a surface of the space, in such a way that the target distance was not explicitly taken into account [14]. Sun at al. used a stereo system to compute the cartesian position of the target, while our system employes implicit variables [21]. Moreover, our model allows to learn directly both the inverse and direct transformations between the arm position and the gaze direction. Finally, neither of the cited works show how to update on-line the sensorimotor transformations in a goal-directed behavior.

3 Representation of the Peripersonal Space

In the proposed framework, the spatial position of the target object was maintained by two global frames of reference (f.o.r.). One f.o.r. is head-centered and it consists of a spherical-like coordinate system in which the azimuth and the inclination angles are replaced by the gaze direction, while the radius is the estimated distance of the target.

One important remark should be given about the use of the distance in the RBFN framework. Indeed, the distance is not directly observable by the robot, that is, it is not an implicit variable. However, primates have access to several cues that can be used to estimate the distance, such as stereopsis, familiar size, motion parallax and so on [13]. These cues are implicit and related to the distance,

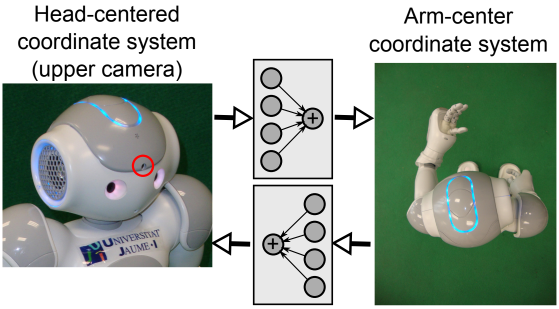


Fig. 1. Computational framework of the sensorimotor integration model. Two transformations allow for converting the head position into arm-motor position and vice-versa.

and could be used in our framework in place of the distance. For example, our previous work, used vergence alone [3]; however, when multiple cues are available, it seems more reliable to integrate the cues before calculating the arm position. Such a computation can be performed by a three layer neural network with reward-mediated learning similarly to what is done in [11]. Thus, in our framework, it is possible to replace the distance with the output of another computation as long as it provides neural activation which is correlated with the distance of the target. In this way, the framework becomes more general and can be used independently from the cues available to estimate the distance.

The arm position also provides the spatial position of the target when the robot is touching the object. In this case, the position is described by the joint angle of the arm, provided by the proprioceptive signals. Usually the arm-centered f.o.r. is redundant in the representation of the position, because many joint configurations can bring the hand to the same spatial position. The implication is that the direct mapping (A→H) between the arm-centered f.o.r. and the head-centered f.o.r. is a single-valued function whereas the inverse one (H→A) is not.

As the main focus of this work is learning the sensorimotor transformations for a humanoid robot, the redundancy problem here was bypassed by simplifying our experimental setup. Therefore, only three joints of the arm, two for the shoulder and one for the elbow were used. In this way, also the inverse transformation became a single-valued function.

4 Encoding the Sensorimotor Associations

As introduced in Section 2 , the sensorimotor associations between the A→H and H→A transformations are maintained by two RBFNs (see Fig. 1).

RBFNs are three-layer feed-forward neural networks whose hidden units($\mathbf{h}$) perform a non-linear transformation of the input data($\mathbf{x}$), whereas the output($\mathbf{y}$) is computed as a linear combination of the hidden units:

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) \cdot \mathbf{W} \tag{1}$$

where $\mathbf{W}$ is the matrix of the weights.

In this work, the hidden units performs a Gaussian activation which is characterized by their centers $\mathbf{c}_i$ and their spread Σ:

$$h_i(\mathbf{x}) = h(||\mathbf{x} - \mathbf{c}_i||) = e^{-(\mathbf{x}-\mathbf{c}_i)^T \Sigma^{-1} (\mathbf{x}-\mathbf{c}_i)} \tag{2}$$

Once the activation of the hidden units is fixed, the learning process can be stated as finding the weights that best approximate the sensorimotor transformation. Given a set of m input-output samples of the target function, the weights of the *j-th* component of the output can be calculated by minimizing the sum of the square error. In this work we use the recursive least square (RLS) algorithm, as proposed in [12].

A new training sample for both maps is obtained when the hand position and the gaze direction are pointing to the same spatial location. The robot autonomously verifies such a condition by checking whether the visual position of the hand is in the center of the visual field (see Fig. 2). If the hand is visible but it is not in the fovea, the robot can gaze the hand to reinforce the head-arm association.

The mapping between the distinct sensorimotor modalities is learned during the interaction with the environment, through gazing and reaching movements. After each performed movement, visual feedback is used to verify the coordination of gaze and arm. At the beginning, the system does not have any previous knowledge of the sensorimotor transformation so random movements are used to begin the exploration of the environment.

Successively, these random movements are suppressed and the system keeps adapting during the goal-directed exploration. In this phase, when the robot fails to reach the target with a ballistic movement (H→A), it starts to use vision to guide the arm movements. This can be done by locally inverting the transformation A→H (for the details please refer to [21]) to calculate the increment of the arm position that is necessary to approximate the target. While the robot is reaching for the target, it tracts its own hand to update its sensorimotor transformations.

5 Experimental Framework

5.1 Robotic Setup

Aldebaran's commercially available humanoid robot NAO was used as platform for testing the proposed framework. The robot is provided with 25 degrees of freedom (d.o.f.s) among which two are placed in the head (pan and tilt) and five in each arm. In this work, we have used three d.o.f.s for the arm and just the upper camera for the vision system.

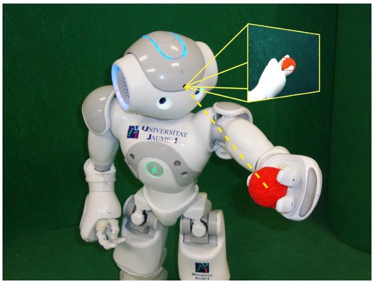


Fig. 2. Association between the oculomotor and arm-motor signals

5.2 Exploration-Based Learning

Learning of the sensorimotor transformations is essentially approximating the function through training samples of the form $(d, \theta_{head}, \theta_{arm})i = 1, 2, ..., n$, where d is the distance from the camera to the robot's hand, $\theta_{head}, \theta_{arm}$ are the joint angles of the head and arm, respectively, and n is the size of the training set. Such a training set was generated by random movements of the arm, while the robot was gazing at the hand. Herein, a visual marker (a ball) was used to facilitate the recognition of the hand in a visual field of view. Subsequently, the visual position of the hand was converted into a head movement in order to foveate the hand. The distance to the hand was computed using the familiar size of the ball. That is, knowing the physical size of an object ($S_{physical}$), its absolute depth (d) was calculated by using the following equation: $d = f \times S_{physical}/S_{observed}$ where $S_{observed}$ is the size of the object observed in the image, while f is the focal length. Both $S_{observed}$ and f are expressed in terms of pixels.

The structure and parameters of the radial basis function networks were chosen using a heuristic search on a simulated model of the robot. We decided to employ fixed centers, uniformly distributed on a lattice in the input space. We employed Gaussian receptive fields, where the matrix Σ was a diagonal matrix $\sigma\mathbf{I}$. The input space of A→H was the shoulder(1,2)-elbow space normalized between 0 and 1, the lattice was composed of 7x7x7 neurons and σ was set to 0.3. The input space of H→A was the pan-tilt-distance space normalized between 0 and 1, the lattice was composed of 7x7x7 neurons and σ was set to 0.28. In this work weights of each network were learned using the recursive least square algorithm on the training samples.

After the exploration process, the networks were tested on the acquired sample points using K-Fold cross validation with K set to 5. The error was calculated as Euclidean distance in the cartesian space between sampled and computed values. This was done using the kinematic model of the robot, which was built using the parameters provided by the manufacturer. The performances of the networks are reported in Table 1.

The transformation of the head-centered to the arm-centered f.o.r. performed worse than the other transformation and, in general, seemed more difficult to

Table 1. Performances of the RBFNs using the K-Fold cross validation (K=5). Mean error and standard deviation ($\mu \pm \sigma$) are expressed in mm in the cartesian space.

Transf.	N. points	K=1 $\mu \pm \sigma$	K=2 $\mu \pm \sigma$	K=3 $\mu \pm \sigma$	K=4 $\mu \pm \sigma$	K=5 $\mu \pm \sigma$
A→H	1458	3.8 ± 2.4	3.9 ± 2.6	4.2 ± 3.2	3.9 ± 2.8	4.2 ± 3.2
H→A	1458	5.0 ± 3.5	5.0 ± 3.2	5.3 ± 3.9	5.0 ± 4.3	5.4 ± 4.3

approximate. Nevertheless, in each case, the magnitude of the error was small enough to allow the robot to grasp the target in most cases (see next session).

5.3 Grasping Task

The performance of the system was tested on a grasping task. The robot had to localize and to grasp a red ball. The ball was placed on two lattices of 3 by 3 points that covered a region of 5 cm by 8 cm (x,y) on the left side of the robot (see Fig. 3). The two lattices were placed at different altitudes. Each movement of the arm was initiated from a safe position that allowed reaching the ball with a ballistic movement without any collision. During training of the H↔A transformations, the robot was gazing at the hand, so we expected that a correct arm movement would bring the center of the hand near the target. For each location of the ball, the robot had to gaze at it and to calculate the arm position by means of the H→A transformation. After the training, the robot grasped correctly the ball for every position on both lattices.

5.4 Goal-Directed Learning

Until now we have demonstrated the capability of the system to encode the sensorimotor transformations. The next step is to demonstrate the plasticity of the system for updating its internal representation to the changes of the body parameters. For this purpose, we changed the body configuration by modifying the position of the elbow roll motor some 20°. The position of the motor is not accesible for the RBFNs, thus the networks require to be adapted to the new configuration of the joint. Indeed, with this new configuration the robot failed to grasp the ball in every position, with a mean error of about $3.2cm$ (see Fig.3).

However, when the robot try to grasp the ball, it can recognize the failure through its vision, by checking if the hand position and the ball position are the same. If it is not the case, the system can thus multiply the distance between the hand and the ball (expressed in the head centered f.o.r.) by the pseudo-inverse of the jacobian of A→H to obtain an increment of the arm position. In this way, the robot produces a sequence of visually-guided arm movements until the target is grasped. At each arm movement, the robot looks at its hand (using visual feed-back) and updates both the A→H and H→A transformations. After, three visually-guided executions of the grasping task, the robot was able to correct its sensorimotor transformations and to perform correctly the ballistic grasping (see Fig. 3).

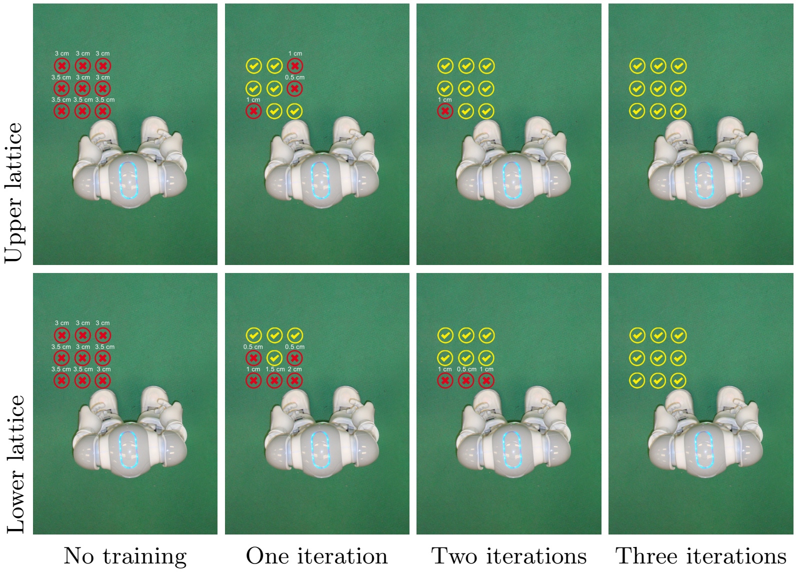


Fig. 3. Experimental setup. After changing the elbow roll position of 20 degrees the robot was tested in a ballistic grasping task (A→H). The target ball was put on two lattices at different hight. The figure shows the grasping error without the on-line adaptation and after one, two and three goal-directed training sessions.

6 Discussion and Future Work

This work was focused on the encoding of the visuomotor transformations for a reaching behavior. The RBFNs were trained with the real data collected while the robot was gazing its hand. The real data, however, are usually quite noisy, which has an impact on the learning process of the neural networks and its performance afterwards. The overall error of the direct transformation (A→H), that is a transformation from arm to eye position was much smaller that the error of the inverse transformation (H→A), that was the transformation from eye to arm position. This can result from the uniform distribution of the centers that, for the H→A transformations, is not so efficient as for the A→H ones. Thus, regularization algorithms [18] that adjust the centers and the spread of the neural activation can improve the performances of the algorithm.

Experimetal results showed that the robot is able to update its performance in the goal-directed behavior by explointing visual feedback to correct the trajectory of the arm. It is done by inverting the forward model that converts the arm position into head position [21].

In the currently implemented framework, the distance was calculated using the familiar size of the object. Such a distance, however, can be estimated by other cues, e.g. motion parallax, kinetic depth effect and so on, which can be combined in the spirit of the Bayesian theorem in order to obtain a reliable distance estimation. Moreover, more implicit distance observations can be used directly as input to our RBFNs. Thus, our future work will focus on the integration of the proposed sensorimotor framework with another model that implicitly encodes the perceived distance.

This work is part of a larger framework that is inspired by infant development. The final goal is to provide the robot with a coherent near and far space representation. The visuomotor knowledge of the peripersonal and extrapersonal space should be built in a dynamical way, through the active interaction with the environment, in a similar fashion as infants do. Following this approach, the robot has to be able to keep learning during its normal behavior, by interacting with the world and continually update its representation of the world itself. Moreover, the learning process should be self-supervised in order to avoid the need of an external teacher. That is, the robot should be able to improve its capabilities by observing the outcome of its actions.

7 Conclusions

This paper presented a framework for sensorimotor transformations that is inspired by neuroscientific findings. The plausibility of our framework was tested with the NAO humanoid robot. The proposed representations of the space are plastic, indeed the robot was able to update and to improve its performance during interaction with the environment. Moreover, the adaptation of our framework on the NAO robot provides further support for the extendability and generality of our approach.

Acknowledgements. This research was partly supported by WCU (World Class University) program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (Grant No. R31-2008-000-10062-0), by Ministerio de Ciencia e Innovación (FPU grant AP2007-02565, FPI grant BES-2009-027151, DPI2011-27846), by Generalitat Valenciana (PROMETEO/2009/052) and by Fundaciò Caixa Castello-Bancaixa (P1-1B2011-54).

References

1. Antonelli, M., Chinellato, E., del Pobil, A.P.: Implicit mapping of the peripersonal space of a humanoid robot. In: IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain, pp. 1–8 (February 2011)
2. Caminiti, R., Ferraina, S., Mayer, A.B.: Visuomotor transformations: early cortical mechanisms of reaching. Current Opinion in Neurobiology 8(6), 753–761 (1998)
3. Chinellato, E., Antonelli, M., Grzyb, B., del Pobil, A.P.: Implicit sensorimotor mapping of the peripersonal space by gazing and reaching. IEEE Transactions on Autonomous Mental Development 3, 45–53 (2011)
4. Chinellato, E., del Pobil, A.P.: The neuroscience of vision-based grasping: a functional review for computational modeling and bio-inspired robotics. Journal of Integrative Neuroscience 8(2), 223–254 (2009)

5. Dechent, P., Frahm, J.: Characterization of the human visual V6 complex by functional magnetic resonance imaging. European Journal of Neuroscience 17(10), 2201–2211 (2003)
6. Fattori, P., Kutz, D., Breveglieri, R., Marzocchi, N., Galletti, C.: Spatial tuning of reaching activity in the medial parieto-occipital cortex (area V6A) of macaque monkey. European Journal of Neuroscience 22(4), 956–972 (2005)
7. Fuke, S., Ogino, M., Asada, M.: Acquisition of the head-centered peri-personal spatial representation found in vip neuron. IEEE Transactions on Autonomous Mental Development 1(2), 131–140 (2009)
8. Galletti, C., Kutz, D., Gamberini, M., Breveglieri, R., Fattori, P.: Role of the medial parieto-occipital cortex in the control of reaching and grasping movements. Experimental Brain Research 153(2), 158–170 (2003)
9. Goodale, M., Westwood, D.: An evolving view of duplex vision: separate but interacting cortical pathways for perception and action. Current Opinion in Neurobiology 14(2), 203–211 (2004)
10. Jones, M., Vernon, D.: Using neural networks to learn hand-eye co-ordination. Neural Computing and Applications 2(1), 2–12 (1994)
11. Karaoguz, C., Weisswange, T.H., Rodemann, T., Wrede, B., Rothkopf, C.A.: Reward-based learning of optimal cue integration in audio and visual depth estimation. In: The 15th International Conference on Advanced Robotics, Tallinn, Estonia (2011)
12. Karayiannis, N.B., Venetsanopoulos, A.N.: Fast learning algorithms for neural networks. IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing 39(7), 1–22 (1992)
13. Landy, M.S., Maloney, L.T., Johnston, E.B., Young, M.: Measurement and modeling of depth cue combination: in defense of weak fusion. Vision Research 35(3), 389–412 (1995)
14. Marjanovic, M., Scassellati, B., Williamson, M.: Self-taught visually guided pointing for a humanoid robot. In: From Animals to Animats 4: Proc. Fourth Int l Conf. Simulation of Adaptive Behavior, pp. 35—44 (1996)
15. Marzocchi, N., Breveglieri, R., Galletti, C., Fattori, P.: Reaching activity in parietal area V6A of macaque: eye influence on arm activity or retinocentric coding of reaching movements? European Journal of Neuroscience 27(3), 775–789 (2008)
16. Nori, F., Natale, L., Sandini, G., Metta, G.: Autonomous learning of 3d reaching in a humanoid robot. In: IEEE/RSJ IROS, International Conference on Intelligent Robots and Systems, pp. 1142–1147 (2007)
17. Park, J., Sandberg, I.W.: Universal approximation using radial-basis-function networks. Neural Comput. 3(2), 246–257 (1991)
18. Poggio, T., Girosi, F.: Regularization algorithms for learning that are equivalent to multilayer networks. Science 247(4945), 978–982 (1990)
19. Pouget, A., Sejnowski, T.J.: A new view of hemineglect based on the response properties of parietal neurones. Philosophical Transactions of the Royal Society B: Biological Sciences 352(1360), 1449–1459 (1997)
20. Salinas, E., Thier, P.: Gain modulation: a major computational principle of the central nervous system. Neuron 27(1), 15–21 (2000)
21. Sun, G., Scassellati, B.: A fast and efficient model for learning to reach. International Journal of Humanoid Robotics 2(4), 391–414 (2005)

Towards Behavioral Consistency in Neuroevolution

Charles Ollion and Stéphane Doncieux

ISIR, UPMC

Abstract. To survive in its environment, an animat must have a behavior that is not too disturbed by noise or any other distractor. Its behavior is supposed to be relatively unchanged when tested on similar situations. Evolving controllers that are robust and generalize well over similar contexts remains a challenge for several reasons. One of them comes from the evaluation: how to check a controller for such properties? The fitness may evaluate a distance towards a behavior known to be robust, but such an example is not always available. An alternative is to test the behavior in multiple conditions, actually as many as possible, to avoid overfitting, but this significantly slows down the search process. This issue is expected to become even more critical when evolving behaviors of increasing complexity. To tackle this issue, we propose to formulate it as a problem of behavioral consistency in different contexts. We then propose a fitness objective aimed at explicitly rewarding behavioral consistency. Its principle is to define different sets of contexts and compare the evolved system behavior on each of them. The fitness function thus defined aims at rewarding individuals that exhibit the expected consistency. We apply it to the evolution of two simple computational neuroscience models.

1 Introduction

While the use of evolutionary robotics has generated encouraging results [1], evolving neural networks for complex tasks still faces evolvability issues [2]. What gradient a fitness function should create? What should a fitness function reward? As of today, no straightforward methodology may help in designing a fitness function in a context of neuro-evolution. Several classifications have been made to take into account the features of a fitness function when comparing results. Floreano and Urzelai [3] thus proposed the *fitness space*, a framework for describing fitness functions in order to qualitatively compare them. Nelson et al [1] have made a review of the fitness functions used in the context of evolutionary robotics and classified them according to the degree of a priori knowledge that they include. If such work review the fitness functions used up to now, they are not aimed at proposing a method for fitness design. This question may be, at least for a part, problem specific, but we hypothesize here that there is a selection pressure that may be common to a lot of neuro-evolution problems and we propose a simple method to design it.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 177–186, 2012.

Neuro-evolution methods aim at generating neural networks that will exhibit a particular behavior in response to the inputs they receive. In the following, we focus on neural networks aimed at controlling the behavior of an animat (or subparts of such a controller). Our hypothesis is that a specific feature is explicitly sought when evolving neural networks in this case: the consistency of the neural network behavior in different contexts. A context will be refered as a particular setting, or environment, in which the animat is. Even if the exact behavior to be generated is not known, the experimenter may know in which contexts he expects a similar behavior and in which contexts he expects different behaviors. An animat moving in its environment should not be too sensitive to noise or other distractors and should behave the same in similar contexts. When designing a neural network with evolutionary methods, these properties are not guaranteed and it is common to observe neural networks that behave well only on the contexts used for evaluation, leading to a lack of generalization [4]. Ideally, such a consistency should emerge from the search process without being explicitly rewarded. However it would require to evaluate the behavior of the network in many different contexts so that this property has an impact on the fitness function —even in this case, the consistency is not guaranteed. We propose a method to design fitness functions that explicitly reward the consistency of neural network behavior. The approach consists in testing the network behavior in different contexts. The fitness function evaluates then how close or how different behaviors are, depending on what the experimenter expects.

Two models from computational neuroscience have been considered to test the approach: an attention selection model and an action selection model. Both represent functions that are expected to be useful for an animat to survive in its environment. The motivation behind this choice is the knowledge of one particular and efficient behavior from the litterature. Results generated with the consistency objective are thus compared to results generated with a more classic fitness function that rewards individuals close to this particular behavior. While requiring a less accurate knowledge, the consistency objective revealed to generate efficient solutions, some of them being original and different from the known behavior; all respecting the constraints.

2 Method

The consistency objective is based on an evaluation of behavioral consistency in different contexts. It relies on simulating one individual on a set S of different contexts, as shown on Figure 1. The consistency of the individual is then evaluated by comparing the behaviors (outputs). We denote $o_i(t)$ the simulated output of context i after t time-steps. Depending on the problem considered, the experimenter defines different contexts and different consistency constraints between contexts. Three possible constraints will be used in the following experiments:

- output of context i is exactly the same as output of context j: $o_i(t) = o_j(t)$
- output of context i is different from output of context j: $o_i(t) \neq o_j(t)$

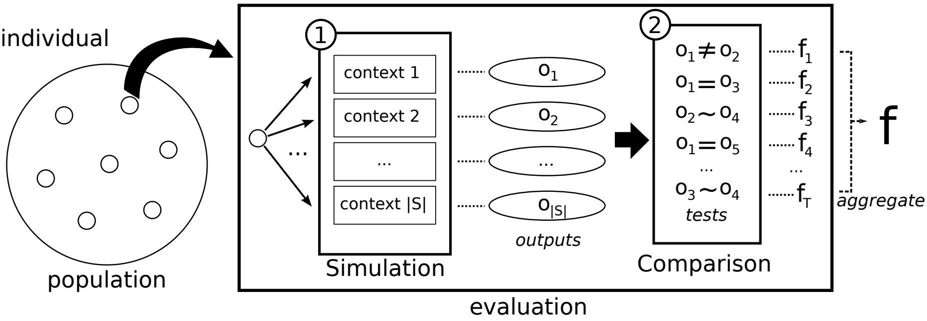


Fig. 1. One individual is evaluated in three steps: 1) simulation over a set of predefined contexts. 2) test of user-defined constraints satisfaction. The level of satisfaction of the constraint is f_i. 3) Aggregation of constraints into the fitness of the individual f.

– output of context i has similar properties as context j: $o_i(t) \sim o_j(t)$. The definition of similarity is specific to each experiment, but typically means equivalent to within a translation.

The constraints are computed in the following way:
– $o_i(t) \neq o_j(t)$: $f_t = d(o_i(t), o_j(t))$
– $o_i(t) = o_j(t)$: $f_t = 1 - d(o_i(t), o_j(t))$
– $o_i(t) \sim o_j(t)$: $f_t = 1 - d_s(o_i(t), o_j(t))$
The normalized distance d denotes the difference between outputs behaviors. d_s is a similarity distance that usually describes how close the shapes of outputs are. d and d_s are specific to the experimental setup.

The final assessment of the quality of an individual is then computed by aggregating the fitness terms. For the sake of simplicity, the simplest aggregation is used:

$$f(x) = \frac{1}{T}\sum_{t}^{T} f_t \tag{1}$$

In short, building the consistency fitness objective involves three steps:

- Defining a collection of different contexts
- Defining constraints between outputs of contexts
- Defining how to compare outputs with distances d and d_s

It should be noticed here that although some knowledge is required, the exact behavior does not need to be known.

3 Attention Selection

3.1 Experimental Setup

The first experiment tackled here is inspired by the work of Quinton [5]. The goal is to use Continuous Neural Field Theory [6] to model attention selection.

The model is taken from [7] in which the neural field is known to output a robust neural activity able to track perceptual information in noisy contexts, even in the presence of distractors. In Quinton's work, the parameters of the neural field are optimized using an evolutionary algorithm.

The neural field is represented by a two dimensional map, and the potential at position vector $\mathbf{x}$ and time t is $u(\mathbf{x}, t)$, with $\mathbf{x}$ in $[-0.5, 0.5]^2$. The field is stimulated by perceptual input $s(\mathbf{x}, t)$, and lateral connections. The dynamics of the neural field follows the equation taken from [5]:

$$\tau \frac{\partial u(\mathbf{x}, t)}{\partial t} = -u(\mathbf{x}, t) + \int_{\mathbf{x}'} u(\mathbf{x}, t) w(\mathbf{x}, \mathbf{x}') d\mathbf{x} + s(\mathbf{x}, t) + h \tag{2}$$

h is the resting potential, set to 0 at first. The lateral connection weights follow the equation:

$$w(\mathbf{x}, \mathbf{x}') = Ae^{-\frac{|\mathbf{x}-\mathbf{x}'|^2}{a^2}} - Be^{-\frac{|\mathbf{x}-\mathbf{x}'|^2}{b^2}} \tag{3}$$

The lateral connection parameters A, B, a and b are evolved, as well as τ the inertial parameter of the dynamics. As in Quinton's work, the parameters are constrained in order to obtain a "mexican hat" like lateral connectivity: $A > B$ and $b > a$. In order to simulate the dynamics, the map is discretized onto a grid of 50×50 units. The dynamics of the neural fields are simulated on different contexts, in which the inputs of the neural field $s(\mathbf{x}, t)$ differ, as depicted in figure 2. The first and second contexts A and B correspond to constant inputs.

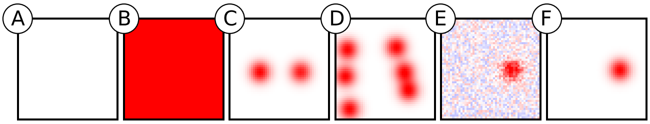


Fig. 2. Typical inputs (contexts) from left to right: A: empty B: full C: competition D: distractors E: noise F: simple

C, D, E are 3 contexts which correspond to Quinton's scenarios. In context C, a static input bubble is in competition with a second one of variable intensity. In contexts D and E, a rotating input bubble must be tracked, in presence of distractors (D) or noise (E). Finally, context F has a single static input bubble and no distractors. The details of those contexts can be found in [5].

Control Experiment. The fitness used in the original experiment is used as a control fitness. It is based on the three contexts C, D and E, and is described in [5]. It assumes the knowledge of the position and shape of the output at each time-step.

Context-Based Experiment. In order to measure the bias added by the choice of contexts, various contexts are used to compute the fitness. The fitness

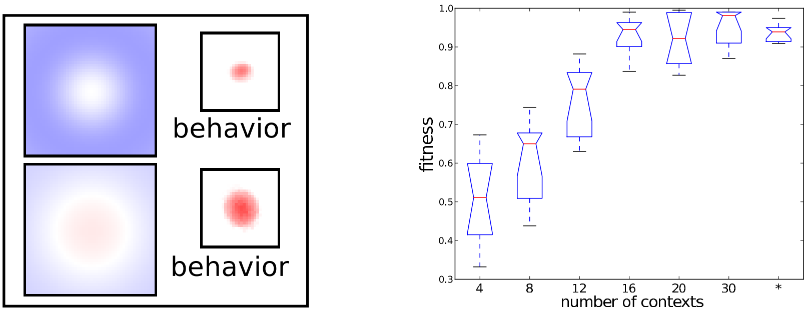


Fig. 3. (Left) Two resulting behaviors obtained with the context-based fitness. Left is the lateral weight profile and right the corresponding output activity. (Right) Influence of the number of contexts on the context-based fitness with 30 contexts. $*$ represents parameters obtained with the fitness of the original experiment.

includes up to $|S| = 30$ contexts. The first two correspond to contexts with a totally activated/deactivated neural field. The others are chosen randomly among C, D, E, or F —each run— to reduce any bias in the result statistics. In contexts C, D, E, or F, the initial position of the input bubble, the number of distractors or the noise level can vary. Four sets of contexts are defined in the following way: the first two contexts are in S_{const}, all others in S_{nconst}. Additional contexts derived from D and E have their input moving, therefore they should not converge towards a fixed output and are then in S_{nconv}. On the other hand, contexts derived from C and F are in S_{conv}. From those sets, the following constraints are defined:

- The output behavior of contexts from S_{nconst} should be different from constant behavior of S_{const} contexts: $\forall i \in S_{nconst}, j \in S_{const},\ o_i(t) \neq o_j(t)$
- The output of S_{nconst} should exhibit consistency in time: $\forall i \in S_{nconst}, \exists \epsilon_t\, \forall t, t' > \epsilon_t\ o_i(t) \sim o_i(t')$
- The output of S_{conv} should stabilize: $\forall i \in S_{conv}, \exists \epsilon_t\, \forall t, t' > \epsilon_t\ o_i(t) = o_i(t')$
- The output of S_{nconv} should be different over time: $\forall i \in S_{nconv}, \forall t > \epsilon_t, \exists t' > t, o_i(t) \neq o_i(t')$
- The outputs of different contexts of S_{nconst} should be similar: $\forall i, j \in S_{nconst},\ o_i(t) \sim o_j(t)$

Finally, the distance between behaviors is computed in the following way:

$$d(o_i(t), o_j(t)) = \int_{\mathbf{x}} ||o_i(\mathbf{x}, t) - o_j(\mathbf{x}, t)|| \tag{4}$$

The similarity distance computes the same distance, after aligning the outputs:

$$d_s(o_i(t), o_j(t)) = \min_{d} \int_{\mathbf{x},\mathbf{x}'} ||o_i(\mathbf{x}, t) - o_j(\mathbf{x} - \mathbf{d}, t)|| \tag{5}$$

d corresponds to the shift between the center of activities between the two outputs. Details of this computation can be found in [7]

Each individual is evaluated during 20 time-steps on each context. In order to avoid initial chaotic dynamics, the first 5 time-steps of the resulting behavior are discarded. Each experiment —with various numbers of contexts— is run 10 times, with a population size of 20 and the number of generations is 30 (same values as in Quinton's experiment). The evolutionary algorithm used is NSGA-II [8].

3.2 Results

The consistency objective with maximum number of contexts provides results with the expected qualitative behavior, in all 10 runs. This means that the output signal (a bubble of activity) is quickly able to follow the input with a signal consistent in time and shape, and is not damaged by noise or distractors. The evolved systems exhibit then the expected function. Furthermore, different behaviors have been discovered, as shown in Figure 3, left. This highlights an interesting property of the proposed method: it looks for behaviors respecting the given constraints, no matter how they manage to do it.

The number of contexts used greatly influences the effectiveness of the consistency objective. In order to study this influence, the individuals obtained with k contexts are tested on the fitness based on 30 contexts. Additionally, an individual obtained with Quinton's fitness was tested on the 30-context-based fitness. The graph (Figure 3 right) shows that:

- not surprisingly, individuals obtained with Quinton's fitness perform well on the context objective. It should be underlined here that the reverse is generally not true as Quinton's fitness looks for a particular shape that is not the unique solution respecting the constraints as mentioned above;
- the number of contexts needs to be high enough for the objective to be effective (around 16), but there is little difference in performance if the number of contexts is over 16 ($p > 0.1$). The difference between 4, 8, 12 and 16 contexts is significant ($p < 0.01$ for each).

4 Action Selection

This section describes the second experiment, based on a neuroevolution experiment on basal ganglia [9]. The goal of this experiment is to evolve a neural network able to perform action selection, a cognitive function supposed to be performed by the basal ganglia. The search space is much larger than in the previous experiment, as the structure and parameters of the neural network are evolved. Action selection in the brain is the problem of choosing an action, given external and internal sensory information. The focus is on the process of selection of a single action among conflicting ones, also known as a winner-takes-all (WTA) circuit. This section first describes the original experiment and fitness, and then the definition of contexts to build the new fitness function.

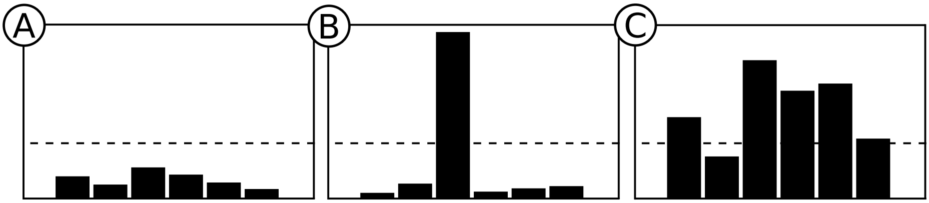


Fig. 4. Typical contexts from left to right (Input of the action selection model): A: empty or low noise B: easy selection C: competition

4.1 Experimental Setup

Encoding. In order to build an action selection model that generalizes to different number of action channels, Mouret et al. used a map-based encoding [9] in which parameters and structure of a neural network are evolved. The neural network is initially a feed-forward network, and the structure is modified through evolution by the addition of new neurons or maps of N neurons, and connections between them. Furthermore, connections between maps of neurons have additional evolved parameters that determine the nature of a connection: a 1-1 connection scheme connects each neuron of the input map to a single one in the output map, while a 1-all connection scheme connects one neuron to all neurons in the output map.

This experiment uses this map based encoding, with a modification: the connections between maps have an additional evolved parameter: an offset o. For instance, a 1-1 connection between two maps of size N with offset o connects neurons as follows: neuron i of the input map is connected to neuron $i + o \, modulo \, N$ of the output map.

Control Experiment. The authors of the original experiment expect to build a model that reproduces the functioning of basal ganglia in the brain. The fitness function assumes full knowledge of the output and is described in [10]. At rest the output of the basal ganglia is active, and represents inhibition to the target. The selected action will then be the channel where inhibition is removed. The neural networks have one input map and one output map of N neurons, N corresponding to the number of channels. Over a collection of $K = 500$ random inputs (inputs range from 0 to 1), the most activated input neuron has to be selected, which means that the corresponding output should be close to zero while others should be close to one. It rewards individuals that desinhibit the selected channel and inhibit all others.

As this setup is more challenging than the previous one due to the complex search space, a behavioral diversity helper objective is added, in a multi-objective scheme [9].

Consistency Objective. The goal of the consistency objective is to evolve networks that perform action selection in any possible way, not only in a biologically plausible way.

Three types of contexts are displayed in Figure 4. Context type A corresponds to a very weak noise input. Context B corresponds to a very simple action selection: one channel is set to 1.0 while the others are set to 0 plus a uniform weak noise. C represents randomly generated contexts with random inputs.

Like in the previous experiment, we define 2 sets of contexts in the following way: S_{const} for context A, S_{nconst} for all others.

- All the outputs should stabilize: $\forall i \in S, \exists \epsilon_t \, \forall t, t' > \epsilon_t \, o_i(t) = o_i(t')$
- The output behavior of contexts from S_{nconst} should be different from constant behavior of S_{const} contexts: $\forall i \in S_{nconst}, j \in S_{const}$, $o_i(t) \neq o_j(t)$
- The outputs of two different contexts of S_{nconst} should have similar behaviors: $\forall i, j \in S_{nconst}$, $o_i(t) \sim o_j(t)$
- The outputs of two different contexts in S_{nconst} which select the same channel should be exactly the same: $\forall i, j \in S_{nconst}, max_i = max_j \Rightarrow o_i(t) = o_j(t)$
- The outputs of two different contexts in S_{nconst} which select different channels should be different: $\forall i, j \in S_{nconst}, max_i \neq max_j \Rightarrow o_i(t) \neq o_j(t)$

Finally the distances to compute difference and similarity are defined as follows:

$$\begin{aligned} d(o_i(t), o_j(t)) &= \textstyle\sum_k ||o_i(k,t) - o_j(k,t)|| \\ d_s(o_i(t), o_j(t)) &= \textstyle\sum_k ||o_i(k,t) - o_j(k-\delta,t)|| \end{aligned}$$

where δ is the shift between centers of activity of the two outputs, computed in a similar fashion as in the first experiment.

The population size is 200 and the algorithm stops after 1000 generations. NSGA-II algorithm is used [8], and the source code is available online at `http://pages.isir.upmc.fr/evorob_db`

4.2 Results

Concerning action selection, the consistency objective was able to evolve successfully two main categories of solutions. Both of them realize action selection by outputting a coherent response for any output, and making a single output stand out from others. The first one (Figure 5, top), realizes the most intuitive action selection, and its corresponding neural network obtained has two internal maps of neurons and excitatory recurrent connections. The second category (Figure 5, bottom) of behaviors are similar to the Mouret et al. results, except for a possible shift in the output channel index. The corresponding neural network has two internal maps, but more connections, both excitatory and inhibitory.

The number of contexts have exactly the same influence as in the previous experiment: with a low number of contexts, the performance is not reliable, while with at least 15 contexts, the performance stabilizes. The difference between 15 and 30 contexts is not statistically significant (p-value > 0.1 Mann-Withney U test). The solutions obtained with the original fitness perform very well when evaluated with consistency objective.

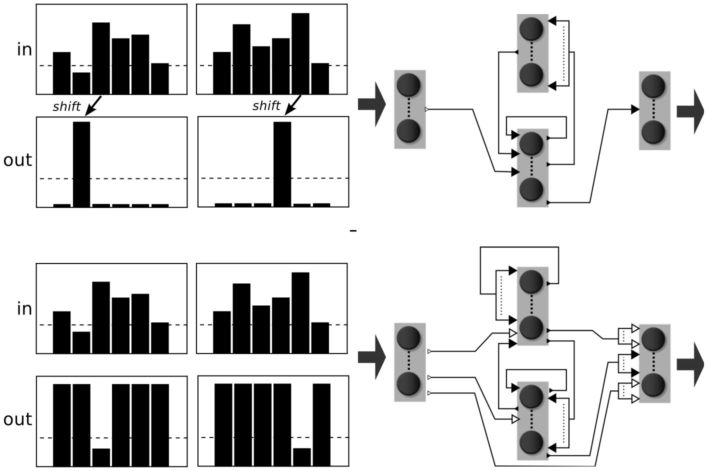


Fig. 5. (Left) 2 behaviors generated with the consistency objective. The graphs show a random input (in), and the output of the evolved model (out). Consistency objective fitness: 0.94 (top) and 0.96 (bottom). **(Right)** corresponding neural networks. Parameters and offsets are not displayed.

5 Conclusion

Properties such as cognitive abilities, robustness to noise, or generalization rarely emerge with fitness functions that only reward the completion of a task. Many evaluations over a lot of different contexts are required for those properties to emerge, and there is no guarantee that such properties actually emerge.

The consistency method can be used to explicitly drive the evolutionary search to the emergence of such properties. While it requires *a priori* knowledge on the expected property, a successful run ensures the emergence of this property, with a limited number of evaluations. The method is shown to successfully build two properties: attention selection and action selection.

Furthermore, the consistency method does not drive the evolutionary search to an explicit behavior. This means that the exact knowledge of a behavior presenting the desired property is not required. In addition, this does not restrain the evolutionary search to one particular solution, leading to the emergence of the property in many different ways.

It is important to note that even if the knowledge of a behavior is not required, the experimenter includes knowledge in building the different contexts and comparison between contexts. While the design of these contexts is straightforward in simple cases, one can expect more challenges for difficult properties to emerge.

The method is based on selection pressure rather than encoding, thus it could potentially be applied to any evolutionary algorithm. Future works include the application of the Consistency objective to the evolution of more complex computational neuroscience models. Furthermore, it could be used as a helper objective in a multi-objective scheme, alongside other objectives, such as a goal oriented objective, behavioral diversity or novelty objectives.

Acknowledgments. This work was funded by the ANR EvoNeuro project, ANR-09-EMER-005-01.

References

1. Nelson, A.L., Barlow, G.J., Doitsidis, L.: Fitness functions in evolutionary robotics: A survey and analysis. Robotics and Autonomous Systems 57(4) (April 2009)
2. Blynel, J., Floreano, D.: Exploring the T-Maze: Evolving Learning-Like Robot Behaviors Using CTRNNs. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J.R., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) EvoWorkshops 2003. LNCS, vol. 2611, pp. 593–604. Springer, Heidelberg (2003)
3. Floreano, D., Urzelai, J.: Evolutionary robots with on-line self-organization and behavioral fitness. Neural Networks 13(4), 431–443 (2000)
4. Pinville, T., Koos, S., Mouret, J.B., Doncieux, S.: How to Promote Generalisation in Evolutionary Robotics: the ProGAb Approach Formalising the Generalisation Ability. In: GECCO 2011: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pp. 259–266 (2011)
5. Quinton, J.C.: Exploring and Optimizing Dynamic Neural Fields Parameters Using Genetic Algorithms. Statistics, 0–6 (2010)
6. Amari, S.I.: Dynamics of pattern formation in lateral-inhibition type neural fields. Biological Cybernetics 87, 77–87 (1977)
7. Rougier, N.P., Vitay, J.: Emergence of attention within a neural population. The Official Journal of the International Neural Network Society 19(5) (June 2006)
8. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
9. Mouret, J.B., Doncieux, S., Girard, B.: Importing the computational neuroscience toolbox into neuro-evolution—application to basal ganglia. In: Proceedings of GECCO 2010, pp. 587–594 (2010)
10. Liénard, J., Guillot, A., Girard, B.: Multi-objective Evolutionary Algorithms to Investigate Neurocomputational Issues: The Case Study of Basal Ganglia Models. In: Doncieux, S., Girard, B., Guillot, A., Hallam, J., Meyer, J.-A., Mouret, J.-B. (eds.) SAB 2010. LNCS, vol. 6226, pp. 597–606. Springer, Heidelberg (2010)

Evolving Variants of Neuro-Control Using Constraint Masks

Christian Rempis and Frank Pasemann

University of Osnabrück, Institute of Cognitive Science,
Osnabrück, Germany
{christian.rempis,frank.pasemann}@uni-osnabrueck.de
http://ikw.uni-osnabrueck.de/~neurokybernetik/

Abstract. The search for variants of effective neural behavior is a major requirement for the identification of novel neuro-dynamical control principles. Evolutionary algorithms are successfully used to search for such controllers. But neuro-evolution tends to find similar, well performing solutions when run multiple times, instead of many, perhaps also weaker performing, but neuro-dynamically highly interesting variants. Furthermore, variants only develop by chance, so that a systematic exploration of different neural control strategies is difficult. With the ICONE method the search space can be shaped by so-called *constraint masks* (CM) to bias the evolving networks towards specific configurations. On the basis of an animat experiment we demonstrate that the number of evolved distinct variants can be significantly increased using different CMs.

Keywords: Neuro-Evolution, Search Space Restriction, Variation Exploration.

1 Introduction

In the field of neurorobotics and evolutionary robotics [3] one of the research goals is to identify dynamical and organizational principles of neuro-control that lead to reasonable behaviors of animats [11]. Often, results do not only provide practical guidelines on how a specific neuro-controller can be realized, but may also give insights and suggestions for the examination of nervous systems of living organisms.

To find neuro-controllers that can be analyzed with respect to their interesting dynamical properties, the main approach is neuro-evolution [2], the application of evolutionary algorithms to neural networks. This promising approach, though, only works well for comparably small networks, because with an increasing network size, the search space explodes exponentially. This is called the *scaling problem* of neuro-evolution [5]. A second problem of many evolution experiments is that a small set of possible solutions often *dominates* other valid control structures. Such dominating controllers are usually more likely to evolve or simply perform better, so that weaker performing or less likely controllers only have a marginal chance to evolve. This is a major obstacle for experiments

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 187–197, 2012.

having the goal to find *variants* of neuro-controllers in order to identify *different* approaches with interesting novel properties to solve a problem, if necessary also including weaker solutions. So the goal here is primarily the identification of different neural structures and neuro-dynamical control strategies applicable to a certain problem *domain*, in contrast to the usual focus on finding an optimal solution. An experiment in this context is merely a representative sample for an *entire problem domain* and not a specific problem that should be solved optimally. This is why an optimal solution for a particular experimental setting is not more interesting than other, in that particular setting possibly weaker performing solution strategies: All identified solutions are essentially interesting because of their potential to be *(re-)used* in other experiments, in which the formerly optimal controller may be less capable than some of the identified alternatives.

Being able to identify as many different solutions as possible on the basis of a single experimental setting also reduces the effort of the experiment design and allows less elaborate, simpler experiments to be used.

A number of approaches have been proposed to tackle these problems, including new evolution methods for larger networks (e.g. HyperNEAT [1], NEATfields [7]) and measures to increase the diversity in the population (e.g. niching [10][17], novelty search [9], behavioral distance [12]). However, these methods focus primarily on the autonomous, random discovery of regularities or differences, which makes it difficult to *systematically* search for specific, knowledge-driven variants of neural control.

In this sense, more control over the evolving networks is given with *shaping* approaches [4], in which domain knowledge is usually applied to break down a difficult task into simpler subtasks and to guide the evolution in incremental steps by changing the evolution operator settings, the fitness function and the evaluation method over the course of successive evolutions. This approach can also be used to search for different solutions, simply by evolving each single subtask multiple times with different settings, each time with the potential of having different dominating solutions to evolve successfully. With this strategy, solutions dominated by stronger solutions in one experiment may become dominant in others and hence get a chance to evolve.

2 Network Shaping

A related, rarely used approach is search space shaping at the network level. In this context, we call this approach *network shaping*. Hereby, the initial network characteristics are systematically varied between experiments to bias the search towards different network topologies and parameter domains. This, for instance, can be done by initially providing structures to start with, by changing the location of neurons [1][13], by choosing locations for network assembly programs [8] or by excluding selected network elements from evolution [6].

A method that allows a much stronger and more general definition of the desired network topologies, and that therefore is optimally suited for the network shaping approach, is the ICONE method, presented in the next section. With this method very complex and specific network topologies can be described and enforced on the network level in a simple and intuitive way. Accordingly, the associated search space for a series of otherwise identical experiments can be biased systematically and with any degree of specificity into different directions, so that the evolution of different solutions to a problem becomes much more likely. This focus on certain search spaces may be realized with very specific settings, merely to confirm assumed solutions, as well as by using quite loose specifications to allow also surprising, unexpected solutions.

In this contribution, we show how network shaping with the ICONE method can be used to systematically search for neuro-controller variants, including network configurations that otherwise would have been unlikely to evolve.

3 Interactively Constrained Neuro-Evolution

The **I**nteractively **Co**nstrained **N**euro-**E**volution method (ICONE) [14][15] has been developed to cope with the larger search spaces of the upcoming class of animats with a rich sensorimotor equipment. Such animats will have a comparably large number of neurons and synapses, that brings most evolution algorithms to their limits.

The ICONE method is a universal, flexible technique to restrict and shape the search space based on domain knowledge. With ICONE, all peculiarities of an animat and its behavior, that are known in advance, can be used to bias the search towards very specific, smaller subspaces of the overall search space, in which evolution is feasible again. In the following, only the basics required for understanding the network shaping approach are explained here in detail.

The search space restriction of the ICONE method is achieved by providing so-called *constraint masks* (CM) that define the valid search space for an experiment. During evolution, only networks within this search space can evolve. A CM is formed by manually choosing groups of neurons (e.g. neuro-modules) that correspond to the neurons' roles, mutual relations, logical aspects or to their associated location on the animat. Every neuron can belong to multiple groups simultaneously. In a second step, these groups can be equipped with so-called *functional constraints* to define the actual CM. Constraints operate within the scope of their associated neuron-group and enforce their limitations and requirements by *actively changing* the subnetworks with appropriate operators. So, if mutations violate constraints, then the constraint operators repair the damage and make the network compliant with the CM again. Accordingly, functional constraints induce strong dependencies between parameters in the network, so the dependent parameters are not part of the search space any more. With these constraints, any topological and organizational property, limitation or requirement can be expressed and enforced, reducing the search space by excluding all inconsistent network configurations. During evolution, any mutation still can take place, but due to the repair mechanisms of the functional constraints all mutations nevertheless lead to valid networks within the defined search space.

The current reference implementation of ICONE [16] with its graphical tools provides many standard constraints, like cloning of subnetworks, different kinds of symmetries, synaptic pathways, connectivity pattern, weight and bias restrictions, complex weight dependencies, and many more [14]. Additionally, very specific *custom* constraints can be implemented as plug-ins to extend the method by less common constraints. This allows a simple and powerful description of the desired network topologies and an intuitive definition of CMs.

A CM, by reducing the search space, also biases evolution towards a specific network topology. Hereby, more restrictive CMs lead to a stronger guidance and more predetermined solutions. Therefore, defining a CM is always a trade-off between the dimensionality of the search space and the degree of indeterminacy of the solution. The definition of CMs hereby is not a trivial task. Detailed descriptions of examples can be found in [14][16].

4 Experiment for the Exploration of Controller Variants

To demonstrate the exploration of controller variants with network shaping we choose an experiment with a non-trivial animat having a sufficiently large number of sensor and motor neurons to allow many different solutions for behavior control. This animat consists of a closed chain of connected plates. Each plate is equipped with a servo motor to control the joint angle towards its successor plate. The motor provides two motor neurons, one to control the desired angle and one to specify the maximal torque applicable to reach the desired position (Fig. 1a). Furthermore, each plate carries a full set of sensors, including an angular sensor for the joint, a force sensor to detect pressure on the plate, acceleration

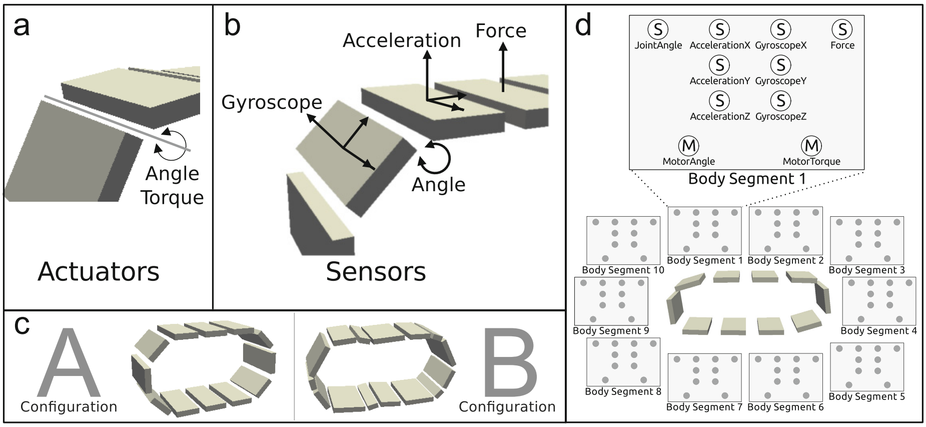


Fig. 1. The closed-chain animat and its motor and sensor equipment. All motors and sensors shown in (a) and (b) are available on every body segment. The animat can be configured with a variable number of segments with different sizes. The two animat configurations used in the experiment are shown in (c). (d) shows the minimal network (80 sensor neurons, 20 motor neurons) for a 10-segment animat with a grouping of the neurons according to their position on the animat.

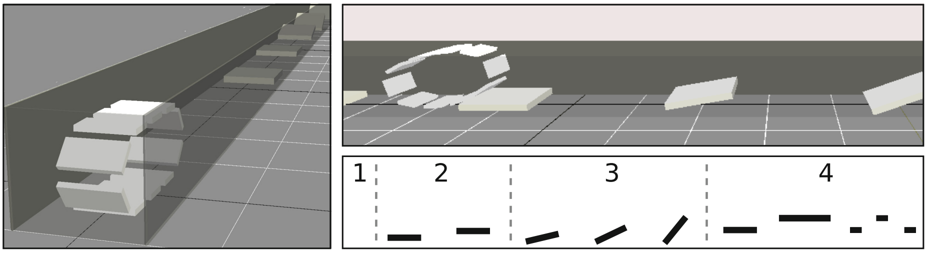


Fig. 2. The closed-chain animat in the obstacle course. The four obstacle zones of the course are roughly sketched in the lower right figure.

sensors and a gyroscope sensor (Fig. 1b). This *closed-chain animat* comes in two versions with differing complexity. Configuration (A) consists of 10 equally sized plates (Fig. 1c left). Configuration (B) has 15 plates in three different sizes that divide the animat in 5 equal triplets (Fig. 1c right).

In the experiment the animat has to follow a straight obstacle course (Fig. 2) by moving forwards over different types of obstacles. For this, it has to develop a coordinated motion of all body segments. The course (Fig. 2) is bounded by walls to ensure that the animat cannot topple over or diverge from the path. The path is separated into four zones. Zone (1) is obstacle free to allow the development of a forwards locomotion first. Zone (2) comprises some low, horizontal obstacles that are easy to pass, followed in zone (3) by a sequence of ramps with increasing steepness. Zone (4) then provides higher, more diverse hurdles.

The experiment, however, aims at finding variants of neuro-controllers for that task, exploiting different uses of the sensors, motor control approaches and network organizations. First, each animat configuration is evolved multiple times with a general CM that constrains the evolution only minimally, so that on one hand many valid controller variants are possible, and on the other hand the search space is small enough for the evolution to succeed. Without these *minimal* CMs bootstrapping the evolution becomes too difficult, because fitness is only gained for a coordinated movement of all body parts. The probability to 'freely' evolve such a coordination is is very low. These results demonstrate the diversity of the best solutions for these least constrained cases.

In the second step, a series of experiments with more restrictive CMs is performed. Each experiment focuses on different domains of the search space using specific CMs and is run multiple times to search for controller variants. All hereby involved search spaces are subspaces of the first two experiments. Therefore, in principle, all controllers found with the stronger constraints could also emerge in the two least constrained cases. We then compare the evolved solutions to investigate whether more restrictive CMs lead to controller variants not appearing in the results for the minimally restricted cases and whether the identified solutions also differ for each CM. This would indicate that the successive application of different CMs on the same experiment generates more controller variants than the evolution that is applied exclusively on the least constrained search space.

Fitness Function. In all experiments, the fitness f at time step t is given by

$$f(t) = (1-\delta)g(t) + \delta \sum_{j=0}^{t-1} g(j)\,, \quad \delta \in [0,1] \tag{1}$$

$$g(t) = g(t-1) + \begin{cases} 0.1 & \text{if } max_t > max_{t-1} \\ -0.1 & \text{if } max_t < max_{t-1} \\ 0.0 & \text{else} \end{cases}, \quad g(0) = 0\,, \tag{2}$$

that is, $f(t)$ is the sum of all increments g up to the current step. The ratio between current increment $g(t)$ and the increment history can be adjusted with parameter δ; $g(t)$ is based on the positions of the animat's body parts; max_i denotes the index of the body part with the largest distance from the origin (in positive x axis) at time step t. If that index is higher (lower) than the previously furthest body part, then $g(t)$ is increased (decreased). Hereby, index 0 is considered to be the valid successor of the highest index, closing the chain. In short, the faster *and* further an agent gets, the higher its fitness becomes.

Evolution. The experiments use the physical simulator of the NERD Toolkit [16] software. During evolution, each neuro-controller is evaluated multiple times with slightly randomized positions of the animat and the obstacles. The fitness of a neuro-controller is defined as the mean of all tests performed in this way. The settings of the evolution operators are similar for all evolution experiments. Three sets of evolution settings are provided and automatically used at certain generations. The first set is only used to generate a very large initial generation with many different individuals. The second set is for the main evolution hereafter. The third set is used starting with the 150th generation to allow more fine-tuning of weights and less structural changes. The settings of the three sets are given in figure 3. In the context of this experiment, not all features provided by the ICONE method – such as the interactivity and the modular crossover – are used. This avoids that the results are influenced by interventions of the experimenter. Further comments on the motivation for these parameter settings and descriptions of the mutation operators can be found in [14].

Population Size 500/50/50 Simulation Steps per Trial 500/1500/3000 Trials per Evaluation 1/3/6 Modular Crossover disabled	Remove Neuron - Probability 0.002 - Number of Attempts 1	Add Neuron - Probability 0.01/0.002 - Number of Attempts 2/2/2
	Remove Synapse - Probability 0.005 - Number of Attempts 5	Add Synapse - Probability 0.1/0.01 - Number of Attempts 10/2/1
	Remove Bias - Probability 0.004 - Number of Attempts 2	Add Bias - Probability 0.02/0.005 - Number of Attempts 5/2/1
Tournament Selection - Tournament Size 8/4/3 - Keep Best Parents 10/1/1	Change Bias - Probability 0.05/0.02 - Deviation 0.5/0.1/0.1	Change Weight - Probability 0.2/0.05 - Deviation 0.2/0.1/0.05

Fig. 3. The settings of the evolution operators. Parameters of different sets are separated by a slash. Parameters not specifying a second or third setting keep the last setting also for the next set(s).

5 Constraint Masks

General Constraint Masks. The general CMs used for the two first, minimally constrained evolutions (M1, M2, see table 1) already restrict the search space significantly to allow successful evolutions at all. In both cases, the networks are divided into neuron groups, each containing all motor and sensor neurons of a single body segment (Fig. 1d). In configuration (A), a single group (master group) is allowed to evolve freely, whereas all other groups are constrained with a *clone* constraint to enforce a structure identical to that of the master. Configuration (B) uses three master groups that represent the three consecutive body segments with differing size (the three *segment types* forming a triplet). All other groups here also use a clone constraint that enforces the structure to be identical with that of their corresponding master module, i.e. with the one

Table 1. Descriptions of the CMs for network shaping. The first column specifies the experiment, the second the used animat configuration (A) or (B), and the third the sensors allowed by the CM. M1 and M2 refer to the minimally constrained evolutions. The last column gives a description of the *additional* constraints focusing on the evolution of specific controller variations (V1 - V8).

E	**C**	**Sensors**	**Constraint Mask in Addition to Default Mask**
M1	A	All	-
M2	B	All	-
V1	A	Angle	Enforces the usage of neighboring groups and prevents the use of the own sensor. Additionally, the motor angle neuron is fixed and the behavior has to be realized using the motor torque neuron only.
V2	A	Angle	Allows only connections between every second neighbor, hereby defining two neuro-dynamically independent rings of groups.
V3	A	Accel.	-
V4	B	All	In addition to the angle sensor, each of the three differently sized segments uses only *one* of the other sensors (acceleration, gyroscope, force). So, each group of a triplet now has a different set of sensors.
V5	B	Gyro	A single group of each triplet now has a gyroscope sensor. Accordingly, there are only 5 gyroscope sensor sets (x, y, z) to control all 15 segments. A special constraint ensures, that only networks evolve in which all motor neurons are influenced by (an arbitrarily long chain of) synapses coming from the gyroscope sensors.
V6	A	Angle Accel.	The master group is forced to use a neural oscillator with frequency control between the sensors and the motors. Therefore, the oscillators have to be connected suitably and a strategy for an effective synchronization has to be found.
V7	A	None	The network is equipped with a structure that produces an activity pulse that passes through all groups. The frequency of the pulse can be evolved, as well as a strategy to use it to generate locomotion.
V8	A	Accel.	The master group is forced to develop an excitatory feed-forward structure between the sensors and the motors, whereas signals from the direct neighbors can influence that structure with inhibition only.

that has the same segment type. In both configurations, the master groups are constrained to grow only local synapses within the same group. For connections between the groups, a *connection symmetry* constraint is used, that enforces a rotation symmetry in the ring of groups. So, if a connection from group 1 to 5 is added in (A), then this constraint adds connections from group 2 to 6, 3 to 7, 4 to 8 and so forth. In (B), it adds connections only for groups of the same type, so from group 4 to 8, 7 to 11 and 10 to 14. Consequently, all groups (of the same type) provide identical subnetworks and a rotation symmetric interconnection pattern. These CMs reduce the search space to the feasible size of a single (or three in the case of (B)) joint controller(s).

Special Constraint Masks. The used CMs to bias the search space further (V1 - V8) are described in table 1. They are chosen exemplarily to demonstrate the various ways the CMs can describe specific approaches. Of course, a systematic exploration is also possible by systematically enforcing the use of interesting combinations of sensors, actuators, coordination heuristics and functional structures. A major impact on the controller strategy certainly has the choice of the sensors. Here, CMs can not only prevent certain sensors from being used (V3), but also allow the definition of specific uses of these sensors. Experiment V1, for instance, only allows the use of sensors of neighboring groups, and V5 enforces all motor neurons of a group triplet to be influenced by a gyroscope sensor. A second variation worth looking at are different motor configurations. With CMs, one type of the motor neurons (angle or torque) may be fixed (as in V1), so that the behavior has to be realized with the remaining motor neurons. A fixed torque neuron, for instance, requires a control based on angle settings, whereas a fixed motor angle neuron requires a control of the torque. In a closed-chain animat it is also interesting to investigate the role of communication between the body segments, for instance how different communication pathways change the control strategy (V1 - V3). And finally, it is interesting to test hypotheses for control, as is done exemplarily in V6 - V8: the use of oscillators, an activation pulse and a feed-forward structure with lateral inhibition. This can be achieved by constraining the networks to use very specific given structures or to organize in specific ways.

6 Results

Figure 4 shows an overview of the results. The upper line gives the number of identified distinct solutions evolved during all evolution runs of each experiment. To classify controllers into solution classes, controllers have been grouped by their observed behavior, by their internal sensor usage and (where possible) by their identified underlying neuro-dynamical properties[1]. The fitness of the best evolved controller (2nd row) and the average of all runs (3rd row) for each experiment can be used to compare the performance of controller variants. The fitness has been

[1] Due to the sheer mass of controllers, only a limited number of interesting controllers have been analyzed in detail [14].

	M1	M2	V1	V2	V3	V4	V5	V6	V7	V8
Distinct Variations	> 8	> 5	> 3	> 4	> 3	> 5	> 4	> 3	> 2	> 4
Best Fitness	11926	7824	5201	5496	3804	5584	3888	7245	6920	2214
Average Fitness	5811	3601	2284	2261	1352	2977	1890	3391	4240	1385
Total Evolution Runs	59	24	65	34	23	33	19	26	34	65

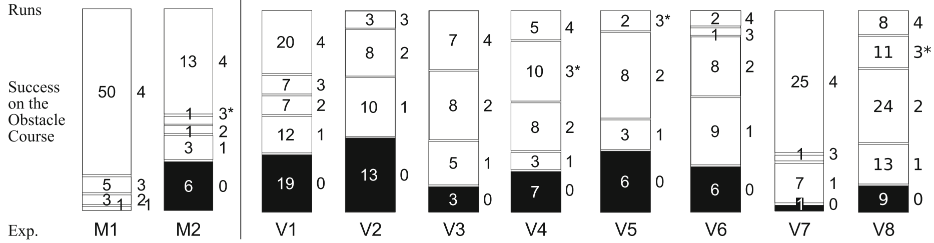


Fig. 4. Results of the evolution experiments, showing the number of distinct variations per experiment (only identified variants based on a rough analysis; more variants may be found with a deeper analysis), the best fitness per experiment and the capability of controllers to overcome the obstacles. The zones of the obstacle course are given as numbers (1 - 4) next to each column. 0 hereby indicates a total failure and 3* refers to the third zone *excluding* the steepest ramp.

normalized by rerunning the best network per experiment to obtain the mean of 10 tries with 6000 simulation steps each. A different perspective on the performance can be read from the columns, which show the capability of the best controller of each evolution run to overcome the obstacle zones.

Analysis. As a first observation, each experiment provides its own set of solution variants (including a common, dominant solution and some less likely, differing solutions) that are in most cases different from the ones found in the other experiments. Therefore, the number of identified variants is indeed much higher (≈ 40) compared to the more general search spaces only (M1: ≈ 8 and M2: ≈ 5). Videos and networks of these controllers can be found at our homepage[2].

The high fitness of the evolved controllers in M1 and M2 indicates that with these configurations highly capable controllers (passing all obstacles very fast) have been found that presumably dominated other, weaker performing solutions (slower or incapable of overcoming certain obstacles). This suggests that the lower performing solutions found in experiments V1 - V8 have been suppressed in M1 and M2, where they, in principle, are also possible. Furthermore, the variants evolved with M1 and M2 are very alike: Most solutions rely on the gyroscope and the force sensors and differ only slightly in their control structure. Controllers using other sensors became extinct due to the very likely occurrence of that successful approach. Because these controllers do not require a sophisticated communication between the segments, only trivial connectivity pattern evolved.

More interesting structures only developed using the stronger CMs. The identified variants hereby do not only express differences in the observable behavior

[2] nerd.x-bot.org/closed-chain-animat

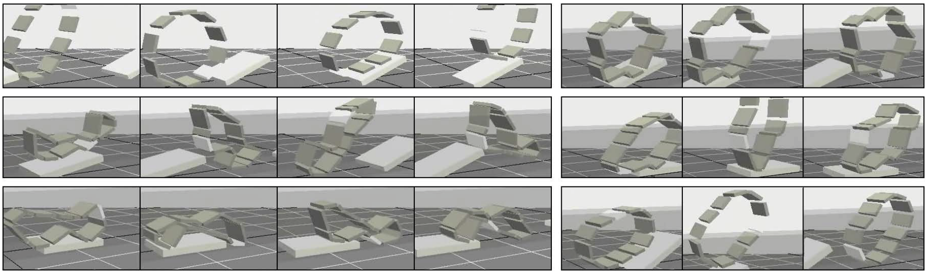


Fig. 5. Time-series of six examples with considerable behavior differences

(Fig. 5), such as elongated or wheel-shaped rolling, organic looking motions, behavior switching at obstacles (e.g. leaning forwards, repeated approaching, backtracking with run-ups), coexisting behaviors (depending on starting conditions), the emergence of 'virtual limbs' (e.g. to push the agent with a periodically forming 'tail') and intermittent forwards-catapulting, to list a few of the more interesting ones. In addition – and more importantly – the solutions also show different neuro-dynamical and structural approaches: Controllers evolved complex communication and coordination strategies between the body segments, various sensor combinations have been successfully used to generate locomotion and the motors have been controlled with different strategies (e.g. joint control, stiffness control or both). This also involved different numbers, distributions and kinds of 'active zones' over the body, in which the joints are actively driven (e.g. bent, stretched, positioned), in contrast to 'passive zones' with a low motor torque. And also some more 'artificial' approaches could be successfully tested with various results, such as driving the locomotion with neural oscillators.

This overview shows that network shaping with CMs is helpful to protect *specific* configurations, to increase the number of *different* solutions and to find also controllers that otherwise would be suppressed by more dominant solutions.

7 Discussion and Conclusion

We have shown that variations of neuro-controllers can be evolved by biasing the search space with CMs towards specific solution approaches. The empirical results show, that the number of variants can be significantly increased compared to evolutions without such specific masks. Hereby, the CMs can not only be used to *systematically* choose combinations of motors and sensors, but also to apply any kind of domain knowledge to the evolving population, so that even very specific network organizations and control hypotheses can be explicitly approached. Although many resulting controllers are not optimal, they often are interesting because of the (novel) neuro-dynamic control principles they provide. Solutions with a lower performance are, without other techniques to maintain a strong diversity in the population (e.g. niching) likely to become extinct early and to be dominated by better solutions. In such cases and when trying to find network solutions which are less likely to evolve, network shaping – as has been shown with the ICONE CMs – is a powerful technique to find suitable controllers.

Acknowledgements. This work was supported by EU-Project ICT – 214856. We thank Tanya Beck for proof-reading and the reviewers for their advice.

References

1. D'Ambrosio, D.B., Stanley, K.O.: A novel generative encoding for exploiting neural network sensor and output geometry. In: Lipson, H. (ed.) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007), pp. 974–981 (2007)
2. Floreano, D., Dürr, P., Mattiussi, C.: Neuroevolution: from architectures to learning. Evolutionary Intelligence 1(1), 47–62 (2008)
3. Floreano, D., Husbands, P., Nolfi, S.: Evolutionary robotics. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, pp. 1423–1451. Springer (2008)
4. Gomez, F.J.: Robust Non-Linear Control through Neuroevolution. PhD thesis, The University of Texas at Austin (2003)
5. Hornby, G., Lipson, H., Pollack, J.: Generative representations for the automated design of modular physical robots. IEEE Transactions on Robotics and Automation 19, 703–719 (2003)
6. Hülse, M., Wischmann, S., Pasemann, F.: Structure and function of evolved neurocontrollers for autonomous robots. Connection Science 16(4), 249–266 (2004)
7. Inden, B., Jin, Y., Haschke, R., Ritter, H.: Evolving neural fields for problems with large input and output spaces. Neural Networks 28, 24–39 (2012)
8. Kodjabachian, J., Meyer, J.: Evolution and development of neural controllers for locomotion, gradient-following, and obstacle-avoidance in artificial insects. IEEE Transactions on Neural Networks 9(5), 796–812 (1998)
9. Lehman, J., Stanley, K.: Abandoning objectives: Evolution through the search for novelty alone. Evolutionary Computation 19(2), 189–223 (2011)
10. Mahfoud, S.W.: Niching methods for genetic algorithms. PhD Thesis. Department of Computer Science, University of Illinois at Urbana-Champaign (1995)
11. Meyer, J., Guillot, A.: Simulation of adaptive behavior in animats: Review and prospect. In: Meyer, J., Wilson, S. (eds.) From Animals to Animats 1, pp. 2–14 (1991)
12. Mouret, J., Doncieux, S.: Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In: Proceedings of the Eleventh Congress on Evolutionary Computation (CEC 2009), pp. 1161–1168 (2009)
13. Nolfi, S., Parisi, D.: Growing neural networks. Tech. Rep. PCIA-91-15, Institute of Psychology (1991)
14. Rempis, C.: Evolving Complex Neuro-Controllers with Interactively Constrained Neuro-Evolution. PhD thesis, to appear: University of Osnabrueck (2012)
15. Rempis, C., Pasemann, F.: An Interactively Constrained Neuro-Evolution Approach for Behavior Control of Complex Robots. In: Chiong, R., Weise, T., Michalewicz, Z. (eds.) Variants of Evolutionary Algorithms for Real-World Applications, vol. 87, pp. 305–341. Springer, Heidelberg (2012)
16. Rempis, C., Thomas, V., Bachmann, F., Pasemann, F.: NERD Neurodynamics and Evolutionary Robotics Development Kit. In: Ando, N., Balakirsky, S., Hemker, T., Reggiani, M., von Stryk, O. (eds.) SIMPAR 2010. LNCS (LNAI), vol. 6472, pp. 121–132. Springer, Heidelberg (2010)
17. Sareni, B., Krahenbuhl, L.: Fitness sharing and niching methods revisited. IEEE Transactions on Evolutionary Computation 2(3), 97–106 (1998)

The Search for Beauty: Evolution of Minimal Cognition in an Animat Controlled by a Gene Regulatory Network and Powered by a Metabolic System

Borys Wróbel[1,2,3], Michał Joachimczak[1],
Alberto Montebelli[4], and Robert Lowe[4]

[1] Systems Modeling Laboratory, IO PAN, Sopot, Poland
[2] Evolutionary Systems Laboratory, Uniwersytet im. Adama Mickiewicza, Poznań, Poland
[3] Institut für Neuroinformatik, Universität & ETH Zürich, Switzerland
[4] Cognition & Interaction Lab, Högskolan i Skövde, Sweden
http://www.evosys.org

Abstract. We have created a model of a hybrid system in which a gene regulatory network (GRN) controls the search for resources (*fuel/food* and *water*) necessary to allow an artificial metabolic system (simulated microbial fuel cell) to produce energy. We explore the behaviour of simple animats in a two-dimensional simulated environment requiring minimal cognition. In our system control evolves in a biologically-realistic manner under tight energy constraints. We use a model of GRN in which there is no limit on the size of the network, and the concentration of regulatory substances (transcriptional factors, TFs) change in a continuous fashion. Externally driven concentrations of selected TFs provide the sensory information to the animat, while the concentration of selected internally produced TFs is interpreted as the signal for actuators. We use a genetic algorithm to obtain diverse evolved strategies in ecologically grounded animats with *motivational autonomy*, even though they lack a dedicated motivational circuit. There are three motivations (or *drives*) in the system: thirst, hunger, and reproduction. The animats need to search for *food* and *water*, but also to perform *work*. Because the value of such *work* is arbitrary (in the eye of the beholder), but affects the chances of reproduction, we suggest that the term *beauty* is more appropriate, and we name the task the *Search for Beauty*. The results obtained provide a step towards realizing a biologically realistic system with respect to: the way the control is exercised, the way it evolves, and the way the metabolism provides energy.

Keywords: minimal cognition, gene regulatory network, chemotaxis, microbial fuel cell, artificial metabolism, genetic algorithm.

1 Introduction

The importance of bodily variables essential to metabolic functioning and internal and behavioural homeostatic regulation has been appreciated since the

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 198–208, 2012.

cybernetics movement of the 1940-60s (cf. [1]) and has inspired research into self-sustaining robots required to trade off *work* with refueling needs [2, 3]) Microbial fuel cells (MFCs) comprise a form of artificial metabolism whose essential metabolic variables consist of chemical energy available to the bacteria in the anodic chamber and the hydration level of the cathode (Fig. 1,left). Bacteria convert the chemical energy into electric energy, which can be then made available to the animat. Chemical energy is provided in the form of *substrate*, which can be refined or unrefined biomass collected from the environment (cf. [4]). MFCs have already been deployed in the context of energy autonomous robotic agents ([4]), i.e. agents that are capable of refueling themselves whilst being flexible regarding the source of fuel (the *substrate*). Furthermore, work in simulation has indicated that motivationally autonomous (cf. [2]) robotic agents (MFC-powered simulated robots with minimal need-constrained action selection) may be imbued with minimal cognitive capacities such as anticipation and opportunism (e.g., [5, 6]).

In this paper we create a hybrid system in which artificial metabolism (an MFC) provides energy to the animats whose behaviour is controlled by a regulatory network inspired by the networks which are the basis of the control of all living cells. Our Artificial Life platform, GReaNs (for Gene Regulatory evolving artificial Networks) was previously used by two of us to model evolution of chemotaxis in unicellular animats [7] and evolution of soft-bodied multicellular animats [8], and was developed originally for research on artificial multicellular development [9, 10]. The model of a regulatory network in GReaNs is similar to the models used by Eggenberger Hotz [11] and other researchers (e.g.,[12, 13]) in the field of Artificial Embryology. We have previously demonstrated high evolvabilty of GReaNs in signal processing tasks [14], and the dynamical properties of other models of gene regulatory networks (GRNs) were investigated by other authors [15, 16]. We were not the first to use GRNs to control animat behaviour (e.g., [17–19] considered wall and light following, and obstacle avoidance).

In a previous model of a hybrid (symbiotic) system – robot and MFC – proposed by two of us [20], the generated energy is stored in the capacitor (C; Fig. 1, right), added externally to the fuel cell in order to comply with the electric constraints dictated by physical robots. Functioning of the fuel cell depends on the balance of the levels of hydration and substrate. Water and substrate need to be replenished to maintain the electrochemical process, in accordance with appropriate agent behavior. In the work reported here, animat behavior enables this replenishment, while the functioning of the MFC may mould cognitive-behavioural capacities at a level of grounding not investigated previously, i.e. using GRNs. The level of integration between the animat and the environment that GReaNs promote in combination with the energy constrained dynamics demonstrated by MFC-powered robots promises much for exploring emergent cognitive phenomena in animats.

The platform used in this paper has four biologically inspired elements: metabolism (MFC), control (GRN), evolution (a genetic algorithm), and a model of a unicellular animat which interacts with its environment in a physically-realistic

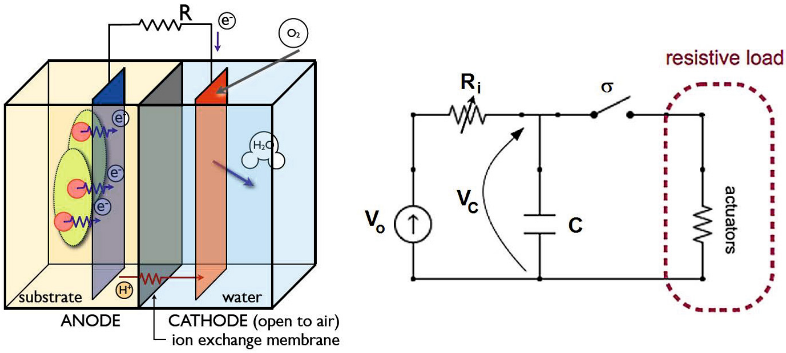


Fig. 1. The oxygen-diffusion cathode microbial fuel cell and the wiring diagram of the animat. See text for details.

fashion. This interaction is governed by simple simulated physics (which includes Newtonian laws). The way the animat senses chemicals in the environment is inspired by the mechanisms in unicellular eukaryotic organisms (for a short review, see [21]). Finally, the animats need to trade off *work* with refueling needs in order to produce progeny. Because the value of such *work* is arbitrary (in the eye of the beholder), we suggest that *beauty* is more appropriate, and name the task the *Search for Beauty*. Another term that might capture the urge to perform *work*, with rich bio-philosophical connotations, is *striving*.

In section 2 (Model), we will first provide a brief description of the elements of the platform. In section 3 (Results), we will analyse the behaviours of animats which were evolved to perform a simple cognitive task: search for resources and *beauty*. Importantly, the information about the state of the MFC (hydration of the cathode, amount of substrate in the anodic chamber) is *not* provided to the GRN in the simulations described in this paper. Finally, in section 4, we conclude with some remarks on the implications of our results for the research in the field of adaptive behaviour.

2 Model

The model for the evolution of the linear genome encoding a GRN used in this paper is essentially the same as the one used in [7, 8, 10]. The model of the circular unicellular animat [7] with sensors at the front and two actuators at the back (Fig. 2) is modified here to include a novel energy source in GReaNs: a simulated MFC [20]. We provide here a brief description of the whole system.

2.1 Linear Genomes, Artificial Gene Regulatory Networks, and Gene Regulation

GRNs are specified by linear genomes and have *internal* and *external* nodes. A genome consists of a list of *genetic elements* (Fig. 2) of three types: E, P,

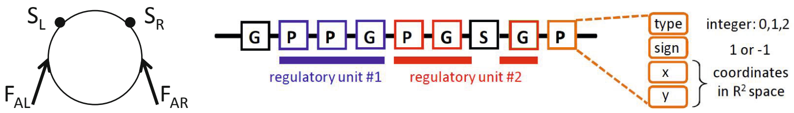


Fig. 2. The animat and its genome. See text for details.

or G. Each element is by itself a list of numbers: type (0 for E, 1 for P, or 2 for G), sign (-1 or 1), and two coordinates (real numbers). The genome is parsed sequentially to construct a GRN. First, all E elements (for *external*) are assigned to external nodes (inputs and outputs), in the order in which they appear in the genome. After E elements are assigned, each sequential group of P elements (*promoters*) followed by a sequential group of G elements (*genes*, which code for *transcriptional factors*, TFs) is interpreted as a regulatory unit. These units correspond to the internal nodes in the GRN. E elements assigned to outputs can be seen as promoters hard-wired to a product with a specific function (controlling an actuator). Elements assigned to inputs can be seen as coding for TFs whose concentration is determined externally to the cell (e.g., by the activity of sensors). An important feature of our model is that there is no limit on the number of genetic elements in the genome, and thus on the number of nodes in the GRN, and no limit on the number of links between nodes. The number of E elements is also not limited, but only nine are assigned, two to outputs (two actuators), seven to inputs (two per each of three resources, one to a TF whose concentration is kept constant, equal to 1). Superfluous E elements in the genome are ignored.

A link between the nodes is formed if a TF has *affinity* to a promoter. Direct links between external nodes are not permitted. *Affinity* depends on the coordinates. In this paper, each element has two coordinates and thus corresponds to a point in an abstract 2D space (not to be confused with the 2D environment in which animats move). *Affinity* is determined by Euclidean distance between points (with a threshold to prevent full connectivity). The concentrations of TFs (real values in the interval $[0, 1]$) change in each simulation step. The concentrations of all TFs belonging to one regulatory unit are the same, and depend on the sum of the activation of the promoters of this unit. Activation of a promoter is the sum of the concentration of each TF that has affinity to this promoter, weighted by this affinity, taking the sign of the two elements into consideration (so regulation is inhibitory when the signs differ, and excitatory otherwise). The sum of the activation is used as an argument of a sigmoid function which produces values in the interval $(-1, 1)$. The current concentration of the TFs coded by the unit are subtracted from the value of the sigmoid function, and this end result is interpreted as the rate of synthesis/degradation to determine TF concentrations in the next step (using Euler integration). In other words, all products degrade exponentially over time unless the synthesis rate is above the intrinsic degradation rate.

2.2 Animats Powered by the Microbial Fuel Cell and Their Environment

The energy produced by the MFC and made available to the animat is a function of the stored electric potential difference across the capacitor ($\varepsilon = \frac{CV_C^2}{2}$). The voltage across the capacitor is updated with the Euler integration method in each step of the simulation of GRN activity and animat movement using the equation $\frac{dV_C}{dt} = \frac{V_o - V_C}{CR_i}$ ($C = 0.0282$). V_o, the electromotive force of the MFC, and R_i, its internal resistance, depend on the level of substrate (s) provided to the anodic chamber, and on the level of hydration (h) of the cathode: $V_o = V_{o,ref}s + V_{o,min}(1-s) - V_{o,hyd}\frac{1-h}{1-h_{lim}}$, $R_i = R_{i,ref}s + R_{i,max}(1-s) + R_{i,hyd}\frac{1-h}{1-h_{lim}}$ (voltages: $V_{o,ref} = 3.2$, $V_{o,min} = 2.8$, $V_{o,hyd} = 0.18$; resistances: $R_{i,max} = 3200$, $R_{i,ref} = 550$, $R_{i,hyd} = 600$; $h_{lim} = 0.19$ is the asymptotic hydration level, see below).

The level of substrate in the anodic chamber changes linearly with time: $s = 1 - \frac{t_s}{\tau_s}$ (t_s is the time from the last replenishment with substrate, $\tau_s = 6000$, thus the rate of substrate consumption is 10 times higher than in [20]; without this modification there is no pressure for substrate replenishment). The level of hydration is modelled as: $h = h_{lim} + h_{pos}\frac{1}{1+e^{\gamma_{pos}(t_h - \tau_{pos})}} - h_{neg}\frac{1}{1+e^{\gamma_{neg}(t_h - \tau_{neg})}}$ (t_h is the time from the last replenishment with water, $h_{pos} = 1.52$, $h_{neg} = 0.68$, $\gamma_{pos} = 0.0055$, $\gamma_{neg} = 0.031$, $\tau_{neg} = 710$, $\tau_{neg} = 600$).

There are three types of resource particles in the environment: *beauty*, *food*, and *water*. The number of *beauty* particles collected determines directly the fitness of the animat, but does not affect the MFC. There is a scalar field of scent for each resource. The scent coming from all particles of a given resource is summed. When a particle is consumed by the animat, the corresponding field changes instantaneously. If it is *water* (or *food*), t_h (or t_s) is set to 0 thus simulating the rehydration of the cathode (or the replenishment of the substrate in the anodic chamber). The scent coming from a given particle decreases with the Euclidean distance (d_{Euc}) from this particle (as $\frac{1}{1+0.2d_{Euc}}$), and reaches maximum (1) at zero distance. The activation of the sensor (S_L and S_R, Fig. 2) is equal to the value of scent at the sensor's location. In order to forage efficiently, the animat has to detect the concentration at a given location and the direction in which it changes (the gradient). The sensory information is provided to the GRN using two TFs per resource, encoded by two E genetic elements. The concentration of one TF depends on the average activation of both sensors ($\frac{2}{1+e^{-\gamma_{avg}(S_R+S_L)}} - 1$), and of one TF on the difference in their activation ($\frac{1}{1+e^{-\gamma_{dif}(S_R-S_L)}}$). In other words, the concentration of the latter TF is 0.5 when the activation of the left and right sensor is the same, and it decreases towards 0 (or increases towards 1) when the right-left difference decreases (or increases). The steepness of the sigmoid functions is set to amplify small differences or to allow for a dynamic response even when the animat is close to several particles ($\gamma_{dif} = 10$, $\gamma_{avg} = 0.5$).

The thrust forces (F_{AL} and F_{AR}, Fig. 2) generated by the actuators are proportional to the concentration of a TF associated with the corresponding output. The directions of the forces are such that when the activations of the actuators differ, the animat turns, but even when only one actuator is active, the animat moves in a loop rather than turning on the spot. When a moving animat deactivates both actuators, the motion continues due to inertia until it is brought to a stop due to fluid drag (proportional to squared velocity). This drag also imposes a maximum velocity possible.

Activation of actuators entails draining the MFC capacitor (Fig. 1). When V_c drops below a certain threshold (2.03), the switch (marked with "σ" in Fig. 1) opens and the distribution of energy (and thus actuation) stops while the capacitor recharges. The switch closes when V_c exceeds the upper threshold (2.90).

2.3 Evolving Efficient Controllers for Foraging in 2D Environment

Each evolutionary runs is 1000 generations of a genetic algorithm with a constant population size of 100 individuals and binary tournament selection (draw two, keep the better one). The genomes of the animats in the initial population have nine E elements and five randomly created regulatory units. Coordinates in genetic elements are randomized by drawing a random direction and a random distance from $(0, 0)$ using a uniform distribution. Genetic operators are: changes of element type, sign, coordinates (so that the associated point in the abstract 2D space is moved in a random direction by a distance drawn from a Gaussian distribution), and duplications and deletions of a random number of elements (drawn from a geometric distribution) at random locations in the genome. The probabilities of deletions and duplications were equal.

Before the trial, 20 *beauty*, 40 *food*, and 40 *water* particles are placed at random positions in the environment (drawn from a uniform distribution centred on the initial position of the animat) to create a *random map*. The space is open (there are no boundaries). The initial direction of the animat is random. The coordinates of the animat and food particles are represented as real numbers. The equations that govern the GRN and MFC are integrated for a specific number of time steps (7000). The genetic algorithm aims to minimize the average value of the fitness function over 10 random maps. The fitness function is: $f_{fit} = 0.8 b_{dir}(1 - \frac{c_{bea}}{20})$, where (c_{bea}) is the amount of *beauty* particles collected (out of 20 on the map), while the value of b_{dir} is 1 if the animat makes at least one turn to the right and one to the left during a trial (then f_{fit} is lowered by 20%), or 1.25 otherwise (no reward). This reward promotes escaping a local optimum (a hill in the fitness landscape) which consists of moving in a loop (finding particles by chance even without any control, or with control of one actuator to tighten the loop when the scent increases [7]).

3 Results

We have performed 100 independent evolutionary runs for 1000 generations, and picked one best individual from each of the 100 final populations thus obtained. These best individuals were sorted by fitness, and 20 "best-of-the-best" were chosen for further analysis. We have re-evaluated each of these 20 individuals

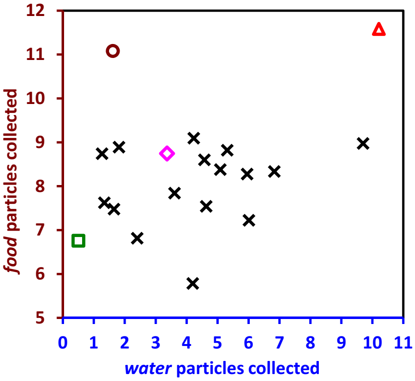


Fig. 3. Diversity of animat strategy in maintaining MFC viability. The location of the points correspond to the average number of *water* and *food* particles during 7000 time steps on 1000 random maps with 40 particles of *water*, 40 of *food*, and 20 of *beauty*. The strategy of the red animat has been marked by a triangle, of the fuchsia animat with a diamond, of the green with a square, and of the maroon with a circle.

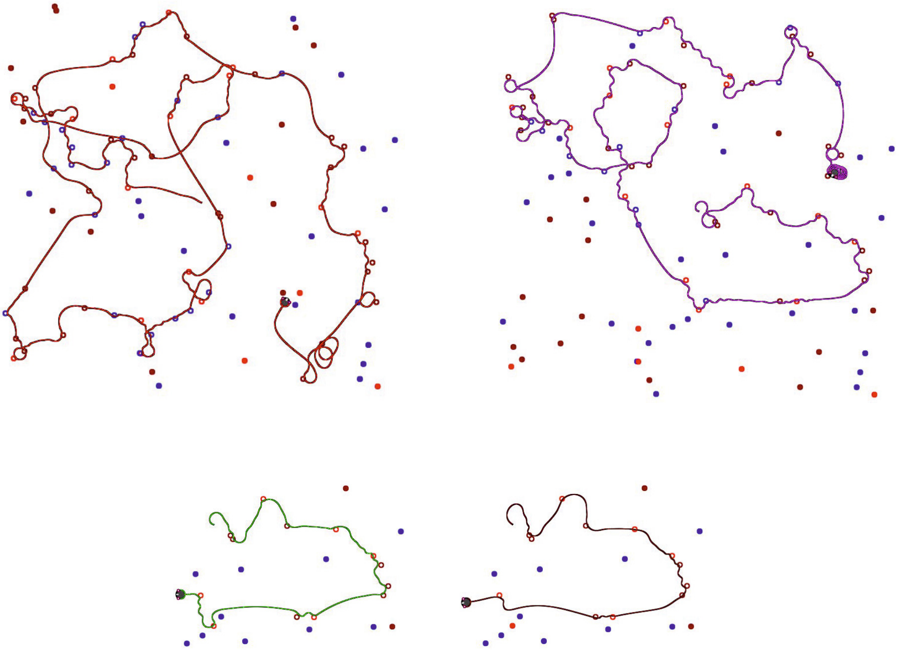


Fig. 4. Example trajectories of the evolved animats. The whole area containing particles is shown for the top two (left: red animat, right: fuchsia), only a fragment for the bottom two (left: green, right: maroon). The simulation was run until the MFC stopped its activity due to dehydration and/or lack of substrate. 40 particles of *water* (blue circles), 40 of *food* (maroon circles), and 20 of *beauty* (red circles) were placed initially, consumed particles are represented as empty circles.

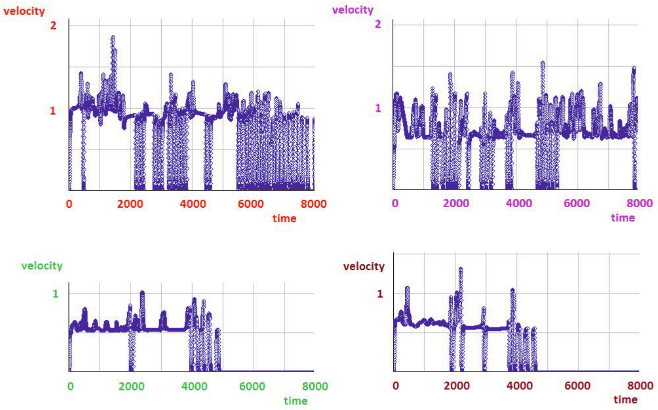


Fig. 5. Velocity of the animats. The graphs show the first 8000 simulation time steps on the same map as in Fig. 4, in the same order: red (top left) and fuchsia (top right), green (bottom left), maroon (bottom right).

over 1000 random maps (instead of 10 used during the runs). The individuals collected about the same number of *beauty* particles in 7000 simulation steps (the average over 1000 trials ranged from 5.40 to 7.80, out of 20 available), differed in the strategy used to maintain the viability of the MFC, i.e., in the number of *water* and *food* particles collected (Fig. 3). The reason why this diversity is possible is that the artificial metabolism model used here (MFC) allows for some energy production when hydration is low provided the substrate level is high. Biological metabolism is similar: water is one of the products of the metabolism of carbohydrates and lipids.

To describe four of these strategies, we will use colour codes for the animats (as in Fig. 3): red for the animat that collected the most particles of all three types (*beauty*: 7.80; *water*: 10.21, *food*: 11.58), fuchsia for the second best in the search for *beauty* (7.04, *water*: 3.37, *food*: 8.75), maroon for the second best animat in *food* collected (11.08, *water*: 1.61), and green for the animat that collected the smallest average number of all particles over 1000 trials (*water*: 0.50, *food*: 6.76, *beauty*: 5.68).

Although on some maps a particular animat may be unsuccessful (Fig. 4), all four animats collected a similar average number of *beauty* particles over 1000 maps when the simulation time was extended to 21000 steps (red animat: 12.86, fuchsia: 11.95, green: 13.79, maroon: 12.68). Higher consumption of provisional resources allows the red animat to maintain a higher velocity than fuchsia, and much higher than green and maroon (Fig. 5), highlighting the fact that although it is possible to maintain low energy production when the cathode is dehydrated, this situation has its consequences for viability (moving too fast results in draining the capacitor and stopping, which may lead to death if energy production is too low to bring V_c over the upper threshold). It is possible to observe convergent evolution of some elements of the strategies, for example, all animats display left-right "sweeping" movements when the scent is high, this prevents missing the target by a short margin. Moreover, despite differences in strategies, one can easily observe some similarity in individual trajectories over some maps (Fig. 4).

4 Conclusions and Future Work

Our results provide a preliminary glimpse on the diversity of behaviours that evolve in agents endowed with a biologically-inspired control and an artificial metabolism, and which interact with a simple environment in a physically realistic fashion. These behaviours can be viewed as a form of *minimal cognition* [22]. The evolved animats can be considered to be energetically autonomous (they can extract from the environment the resources necessary to maintain their essential variables), *ecologically grounded*, and to have evolved limited *motivational autonomy*. The set-up explored in this paper is an example of a *three resource problem* (an extension of the *two resource problem* of [2]), in which provisional resources (here: *water* and *food*) are necessary to execute *work*. Because this *work* is not necessary for bare survival, but influences the chance of producing progeny, we name is *beauty*.

The environmental set-up for the *Search for Beauty* used in this paper provides the possibility for the evolution of three motivations: thirst, hunger, and reproduction. We use the term *motivations* in a broad, evolutionary sense, because we did not provide the animats with any motivational circuit. However, there are indirect connections between control and metabolism in our system. On one hand, metabolism imposes constraints on the control, because if there is not enough energy to allow for movement, the animat stops (and, in principle, a GRN can detect this state). On the other hand, appropriate behaviour maintains a stable metabolism. Because the connections in the GRN can be recurrent, our set-up is open to control that depends on *memory* rather than *motivation* in the strict sense of the word [23]. GReaNs can be used for tasks which require memory [14]), but the *Search for Beauty* does not: the animats do not seem to have stronger preference for *food* right after collecting *water* or vice versa (Fig. 4). Because the *Search for Beauty* can be nonetheless efficient, it is doubtful if any significant increase of efficiency in this particular environmental set-up could come from endowing animats with internal sensors of the hydration or substrate level in the MFC, or level of stored energy in the capacitor. A more promising approach is to walk along the line of further *ecological grounding*, for example, by introducing an indirect penalty for overconsumption of resources (perhaps such overeating/drinking could result in increased drag), variability in the availability of resources, or their patchy distribution [23]. Another possible means of exploring the complexity and richness of the hybrid system presented in this paper would be via providing an incentive for higher velocity, perhaps in a situation of direct competition for resources or in a predator/prey set-up. We plan to explore these directions in our future work.

Acknowledgements. Computational resources were provided by the Polish Ministry of Science and Education (project N519 384236, N303 291234), the Tri-City Academic Computer Centre (TASK), and the Interdisciplinary Centre for Molecular and Mathematical Modeling (ICM, University of Warsaw; project G33-8).

References

1. Ashby, W.R.: Principles of the self-organizing dynamic system. In: Principles of Self-Organization: Transactions of the University of Illinois Symposium, vol. 37, pp. 125–128 (1960)
2. McFarland, D., Spier, E.: Basic cycles, utility and opportunism in self-sufficient robots. Robot Auton. Syst. 20, 179–190 (1997)
3. Avila-García, O., Cañamero, L.: Hormonal modulation of perception in motivation-based action selection architectures. In: From Animals to Animats 8: Proceedings of the 8th International Conference on Simulation of Adaptive Behavior, pp. 243–252. MIT Press (2004)
4. Melhuish, C., Ieropoulos, I., Greenman, J., Horsfield, I.: Energetically autonomous robots: food for thought. Auton. Robot 21 (2006)
5. Montebelli, A., Lowe, R., Ieropoulos, I., Melhuish, C., Greenman, J., Ziemke, T.: Microbial fuel cell driven behavioral dynamics in robot simulations. In: Artificial Life XII: Proceedings of the 12th International Conference on the Simulation and Synthesis of Living Systems, pp. 749–756. MIT Press (2010)
6. Lowe, R., Montebelli, A., Ieropoulos, I., Greenman, J., Melhuish, C., Ziemke, T.: Grounding motivation in energy autonomy: A study of artificial metabolism constrained robot dynamics. In: Artificial Life XII: Proceedings of the 12th International Conference on the Simulation and Synthesis of Living Systems, pp. 725–732. MIT Press (2010)
7. Joachimczak, M., Wróbel, B.: Evolving gene regulatory networks for real time control of foraging behaviours. In: Artificial Life XII: Proceedings of the 12th International Conference on the Simulation and Synthesis of Living Systems, pp. 348–355. MIT Press (2010)
8. Joachimczak, M., Wróbel, B.: Co-evolution of morphology and control of soft-bodied multicellular animats. In: GECCO 2012: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation. ACM (in press, 2012)
9. Joachimczak, M., Wróbel, B.: Evo-devo *in silico*: a model of a gene network regulating multicellular development in 3D space with artificial physics. In: Artificial Life XI: Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems, pp. 297–304. MIT Press (2008)
10. Joachimczak, M., Wróbel, B.: Evolution of the Morphology and Patterning of Artificial Embryos: Scaling the Tricolour Problem to the Third Dimension. In: Kampis, G., Karsai, I., Szathmáry, E. (eds.) ECAL 2009, Part I. LNCS, vol. 5777, pp. 35–43. Springer, Heidelberg (2011)
11. Eggenberger Hotz, P.: Evolving morphologies of simulated 3D organisms based on differential gene expression. In: Proceedings of the 4th European Conference on Artificial Life, ECAL 1997, pp. 205–213. MIT Press (1997)
12. Andersen, T., Newman, R., Otter, T.: Shape homeostasis in virtual embryos. Artif. Life 15, 161–183 (2009)

13. Schramm, L., Sendhoff, B.: An animat's cell doctrine. In: Advances in Artificial Life, ECAL 2011: Proceedings of the 11th European Conference on the Synthesis and Simulation of Living Systems, pp. 739–746. MIT Press (2011)
14. Joachimczak, M., Wróbel, B.: Processing signals with evolving artificial gene regulatory networks. In: Artificial Life XII: Proceedings of the 12th International Conference on the Simulation and Synthesis of Living Systems, pp. 203–210. MIT Press (2010)
15. Kuo, P.D., Leier, A., Banzhaf, W.: Evolving Dynamics in an Artificial Regulatory Network Model. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 571–580. Springer, Heidelberg (2004)
16. Knabe, J.F., Nehaniv, C.L., Schilstra, M.J., Quick, T.: Evolving biological clocks using genetic regulatory networks. In: Artificial Life X: Proceedings of the 10th International Conference on the Simulation and Synthesis of Living Systems, pp. 15–21. MIT Press (2006)
17. Bentley, P.J.: Adaptive fractal gene regulatory networks for robot control. In: Workshop on Regeneration and Learning in Developmental Systems in the Genetic and Evolutionary Computation Conference (GECCO 2004) (2004)
18. Taylor, T.: A genetic regulatory network-inspired real-time controller for a group of underwater robots. In: Proceedings of the Eighth Conference on Intelligent Autonomous Systems (IAS-8), pp. 403–412 (2004)
19. Quick, T., Nehaniv, C.L., Dautenhahn, K., Roberts, G.: Evolving Embodied Genetic Regulatory Network-Driven Control Systems. In: Banzhaf, W., Ziegler, J., Christaller, T., Dittrich, P., Kim, J.T. (eds.) ECAL 2003. LNCS (LNAI), vol. 2801, pp. 266–277. Springer, Heidelberg (2003)
20. Montebelli, A., Lowe, R., Ziemke, T.: Towards metabolic robotics: insights from modeling embodied cognition in a bio-mechatronic symbiont. Artif. Life (in press, 2012)
21. Bagorda, A., Parent, C.A.: Eukaryotic chemotaxis at a glance. J. Cell Sci. 121, 2621–2624 (2008)
22. Beer, R.: Toward the evolution of dynamical neural networks for minimally cognitive behavior. In: From Animals to Animats 4: International Conference on Simulation of Adaptive Behavior, pp. 421–429 (1996)
23. Saglimbeni, F., Parisi, D.: Input from the External Environment and Input from within the Body. In: Kampis, G., Karsai, I., Szathmáry, E. (eds.) ECAL 2009, Part I. LNCS, vol. 5777, pp. 148–155. Springer, Heidelberg (2011)

Evolving Reactive Controller for a Modular Robot: Benefits of the Property of State-Switching in Fractal Gene Regulatory Networks

Payam Zahadat, Thomas Schmickl, and Karl Crailsheim

Artificial Life Lab of the Department of Zoology,
Universitätsplatz 2, Karl-Franzens University Graz, 8010 Graz, Austria
payam.zahadat@uni-graz.at

Abstract. In this paper, we study Fractal Gene Regulatory Networks (FGRNs) evolved as local controllers for a modular robot in snake topology that reacts adaptively to environment. The task is to have the robot moving in a specific direction until it reaches a randomly placed target-zone and stays there. We point to a characteristic of FGRN model, namely "state-switching property" and demonstrate it as a beneficial property in evolving reactive controllers.

Keywords: Gene regulatory network, reactive controller, modular robot.

1 Introduction

A real world task for a robot usually consists of several parts where each part demands for a specific behavior. The robot needs to sense environment and react properly by regulating its behavior. In evolutionary robotics, evolvability for reactive controllers which are able to switch between different behaviors based on the environmental signals seems to be an important issue to consider. This paper demonstrates evolvability of Fractal Gene Regulatory Networks (FGRNs) as decentralized reactive controllers for a modular robot that adaptively reacts to environment. The paper points to a special characteristic of FGRN we will refer to as "state-switching property" as a potential strength of the method making it suitable for evolving reactive controllers.

A modular robot is made up from a number of mechanically coupled modules where each module is typically controlled by its own local controller. A local controller can be a computational Gene Regulatory Network (GRN) that is a network of genes which produce outputs during their interactions. GRN-like systems are previously used to control different modular robots in relatively limited tasks [17,14,7]. Each module of the robot is controlled by a single GRN as a control unit. All the GRN units are genetically identical but runs in parallel and behave variously based on the signals receiving from local sensors. Models of GRNs are inspired by interactions happening inside of a biological cell. Unlike the standard Evolutionary Computational (EC) systems, in a GRN system,

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 209–218, 2012.

phenotypes are indirectly generated from genotypes. A mediatory process of development is required to generate the phenotype through interactions between the genes and between the genes and the environment. Based on the inspiration source of the GRN models the designers anticipate to reach acceptable levels of evolvability for their systems. Many different variants of GRNs are defined by researchers in the field, e.g. [13,1,5,4]. Different GRNs are various in defining the methods of interactions and encoding representations for their genomes which influence evolvability of the system by affecting the shape of the fitness landscape. Apart from the explanatory definition of the interactions during developmental process and representations, a GRN model is eventually a computational network. The model implicitly defines a potentially complex network of nodes and equations that determine the output of each node. Taking a closer look into a particular GRN model might be helpful to get a better understanding of it and identifying the causes of strengths or weaknesses of the model. It may also lead to design new versions of GRNs with improved capabilities.

FGRN [3] is a variant of computational GRNs. It has been successfully evolved for different tasks such as producing patterns [3], controlling single robots [2], motion planning [18], pole-balancing [9], and controling modular robots [17]. Dynamics of an FGRN system can be described as several conditional sets of differential equations which are implicitly encoded in the genotype [19]. Interestingly, each set is connected to a particular internal state of the system and by changing the state, a different set of equations is triggered to describe dynamics of the system. This property is named "state-switching" and we suspect it beneficial in evolving solutions for some types of problems such as reactive tasks, in contrast with having a single set of differential equations as utilized in other GRN models.

2 Fractal Gene Regulatory Network

FGRN is inspired by biological cells [10]. In a biological cell, a number of genes (genome) encode proteins and the conditions of producing them. Proteins are means of interactions between the genes and also between the genome and the environment. These lifelong interactions drive the development and behaviors of the cell. The rate of production of a protein is based on the current protein content of the cell and environmental stimulants. For a particular gene, if a sufficient amount of specific set of proteins exists, the encoded protein is produced (or suppressed).

FGRN model is described in detail in [3,18]. Here we give a very short summary stressing the points which are more important for this paper. An FGRN system contains "fractal proteins" and a set of genes which encode them. Fractal proteins are introduced as an abstraction of the interaction substance. A fractal protein consists of two parts: shape, and concentration level. The protein shape defines how the protein interacts in the system. The concentration level represents the current amount of the protein and is between zero and a maximum value. A level of zero means the protein doesn't exist at the moment.

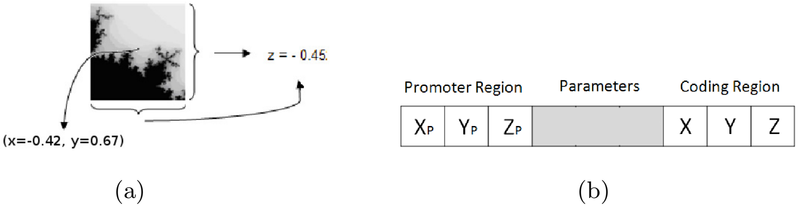


Fig. 1. An example fractal protein shape (a) and a typical gene (b)

A protein shape is a square window on the Mandelbrot fractal set and is encoded in a gene by three real values (x, y, z) (see Fig. 1(a)). By changing these values, the window reaches different locations and scales on the fractal set.

Genes in the FGRN system are of different types: environmental, receptor, regulatory, and behavioral. The genes belonging to the first three types encode the shape of proteins of the same type. The production rate of an environmental protein is determined by an environmental stimulant. This way, environmental information is brought into the system. Receptor proteins are responsible for filtering out parts of the environmental proteins. The production rate of regulatory proteins are regulated by a combination of proteins inside the system, namely a “protein compound”. The behavioral genes generate outputs of the FGRN system.

All the environmental and regulatory proteins which are currently present inside the system merge together to a protein compound. The protein compound interacts with the regulatory and behavioral genes and regulates the concentration level of regulatory proteins and the output of behavioral genes.

A typical gene is a sequence of numbers. It consists of a promoter region, a coding region, and a set of parameters (Fig. 1(b)). The coding region encodes the shape of the protein producible by the gene. The promoter region encodes a protein shape as well. The present protein compound in the system matches against this shape and together with the parameter set of the gene, a value is computed. This value is used to determine the new concentration level of the gene in case of the regulatory genes and the output of the gene in case of the behavioral genes.

The lifetime of an FGRN system consists of several repetitions of the following steps: First, parts of the environmental proteins are filtered out by receptor proteins and the concentration levels of the rest are updated based on the environmental stimulants. Then the protein compound is computed from the present regulatory and environmental proteins. Then the new concentration levels of the regulatory proteins and the outputs of the system are produced based on the protein compound and the corresponding genes.

2.1 State-Switching Property

Here we aim to drive attention to an important characteristic of FGRN model we call “state-switching property”. In order to do that, we leave the conventional descriptive viewpoint of the model as an ongoing interactions between fractal proteins and genes. Instead, we view the system as several fixed conditional sets of differential equations (e.g. Table 1).

As mentioned before, a protein compound is a representation for a combination of the proteins currently present in the system. During lifetime of a system, the concentration levels of proteins may change. The proteins may vanish (their concentration reaches zero) or appear anew to the system (their concentration raises from zero). This means that the set of present proteins may change during the lifetime and consequently the protein compound can get different shapes.

We consider a set of present proteins as a particular state of the system. Since every single protein shape is encoded in the genome and doesn't change during lifetime, for every particular state of the system there is a fixed shape for the protein compound and it is computable solely based on the genome.

A new value for concentration level of a regulatory gene (or the output of a behavioral gene) is computed based on the promoter and parameters of that gene, and the present protein compound. A matching operation is performed between the shape of the compound and the protein shape encoded in the promoter region of the gene. From this matching operation, a contribution degree is computed and assigned to each protein which has participated in forming the protein compound. This contribution degree is a coefficient in a differential equation describing the new concentration level or output of the gene in the current state of the system. Therefore, for every particular state of the system a set of differential equations are defined. The variables of these equations are the concentration levels of the present proteins. In every step, the new outputs and regulatory concentration levels are computed based on the current set of the differential equations. If a change in the concentration levels leads to switching between the states of the system, a different set of differential equations is triggered in the next step. Note that these different conditional sets of equations are implicitly encoded in the FGRN genome and evolve during generations.

A detailed description about extracting the equational representation of the system from both the genome and definition of interactions in FGRN model is explained in [19]. An example description of a simple FGRN is represented in Table 1. The table describes a system with four different states (conditional statements) where a set of differential equations is associated with each state.

3 Investigated Scenario

In this work, populations of FGRN genomes are evolved as local controllers for a modular snake robot to perform a reactive locomotion task. The robot is made up of three homogenous modules and is supposed to locomote in a specific direction until it reaches a target zone at a random distance. The target zone is realized by a light source and the sensors of modules perceive the light only within an area with a threshold distance from the source. This area is considered the target zone and the controllers should react to that by preventing the robot from exiting the zone.

Table 1. An example of a simple FGRN as several conditional sets of differential equations. This system consists of four states. S represents current state as the set of present proteins. $P1$ and $P2$ correspond to an environmental and a regulatory protein respectively and $p1$ and $p2$ are their corresponding concentration levels. out is the output of the system.

condition (state)	equation set
if $S = \{P_1, P_2\}$	$p_2 \leftarrow 0.8p_2 - (0.2p_1 + 0.5p_2) \times tanh(0.6p_1 + 1.5p_2 - 3.6) - 0.2$ $out \leftarrow 0.15p_1 + 0.2p_2 + 4$
if $S = \{P_1\}$	$p_2 \leftarrow 0.4p_1 + 0.5$ $out \leftarrow 0.32p_1 + 2$
if $S = \{P_2\}$	$p_2 \leftarrow 0.8p_2 - 0.25p_2 \times tanh(0.5p_2 - 0.4) - 0.2$ $out \leftarrow 0$
if $S = \{\}$	$p_2 \leftarrow 0.4$ $out \leftarrow 0.25$

There is no explicit communication between the modules. The modules are controlled independently by genetically identical controllers. Since local sensors provide the input values for the modules, the outputs which are generated by the controllers may differ and lead to various behaviors for different modules.

The experiments are performed in Symbricator3D [16]. Symbricator3D is a simulator designed for the projects SYMBRION and REPLICATOR [15,12] and uses the design of the prototype in [8] (Fig. 2). It is based on the game engine Delta-3D and uses the Open Dynamics Engine for the simulation of dynamics.

For the current experiments, three Symbricator3D modules are connected to each other to form a short snake. Each module is supplied by two proximity sensors at the front and rear faces which are implemented in the main simulator. In addition, we utilize an ideal luminance sensor concerning the target zone. The sensor provides a zero value when the light source is farther than a threshold. Otherwise the value is inversely proportional to the distance from the source.

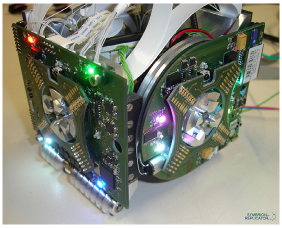


Fig. 2. A prototype of the module hardware

Since three sensors are used for a module, four types of environmental genes are defined, one for each sensor and another one provides a maternal protein which is always produced. A value received from a sensor is normalized and determines the concentration of the proteins encoded by the genes of the associated type. Every robotic module has an actuator which is a central hinge. All the behavioral genes are associated with this actuator. The total value produced

Table 2. Evolutionary settings and the initial number of each type of genes

population size	#generations	#parents for crossover	mutation probability
30	25	12	1%
#receptor-genes	#evironmental genes	#regulatory genes	#behavioral genes
2	4	2	2

by the behavioral genes are considered to be the controller output. The output is scaled into the appropriate range and fed into the hinge making it to approach a specific angle.

3.1 Evolving for the Reactive Behavior

A variant of a Genetic Algorithms is used in this experiment (See [3] for details). The initialization and evolutionary settings are represented in Table 2. A population of 30 randomly generated FGRNs is pre-evolved for locomotion in the right direction and this population is then evolved for the full behavior.

Fitness is computed based on two independent evaluations of a genome. For every evaluation, the target is set in a pre-specified distance from the robot. The two distances are different and fixed. The fitness is computed at the end of each evaluation period and the total fitness is the minimum of the two.

The reason for having two evaluations for each genome initiates from an evolutionary trap of the task. Having only one evaluation may end up to non-reactive controllers with high fitness evolved from exploiting the particular evaluation's settings, e.g. distance from the target zone and length of the evaluation period. For example, if a simple non-reactive controller makes locomotion with a particular speed, the robot happens to be at the target zone at the end of the evaluation period that leads to a high fitness.

Due to high computational costs in the simulator, the number of evaluations within reasonable time is limited. We set the number of evaluations of each genome to two which keeps the computational costs relatively low and still avoids the evolutionary trap. The fitness function of an evaluation is defined as follows:

$$fitness = \begin{cases} \alpha \times \frac{d_{TH}}{d_{current}} + g(speed_{out}, speed_{in}) & \text{if } d_{current} < d_{TH} \\ g(speed_{out}, speed_{in}) & \text{else} \end{cases} \tag{1}$$

where $d_{current}$ is the current distance from the center of the target zone, d_{TH} is the threshold distance of visibility, $speed_{out}$ and $speed_{in}$ are the speeds of locomotion outside and inside the target zone. α is a coefficient, and g is a function scoring for higher ratio of speeds outside to inside the zone.

3.2 Investigating the State-Switching Property

Two experiments with different FGRN setups are performed. In the first experiment, the standard FGRN setup is implemented. In the second setup, the state-switching property is suppressed by altering the model such that a single state and consequently a single set of equations are always triggered during runtime. In order to do that, all proteins of the system are counted as present

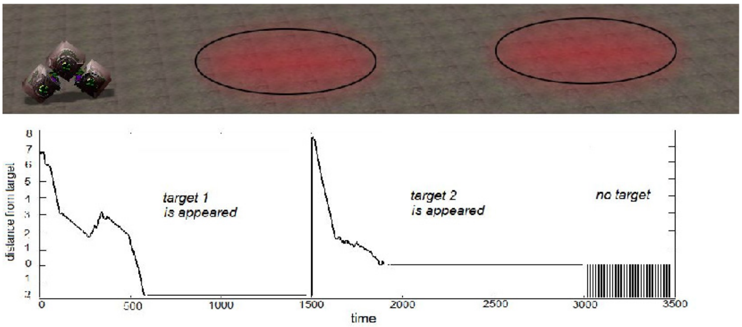


Fig. 3. Two consecutive random targets appear and disappear one after the other (top), and distance of the robot from the existing target during locomotion (bottom)

Table 3. Experimental settings and results

target distances during evolution	9	19			
target distances during reactivity evaluation	8	12	16	20	24 (units)

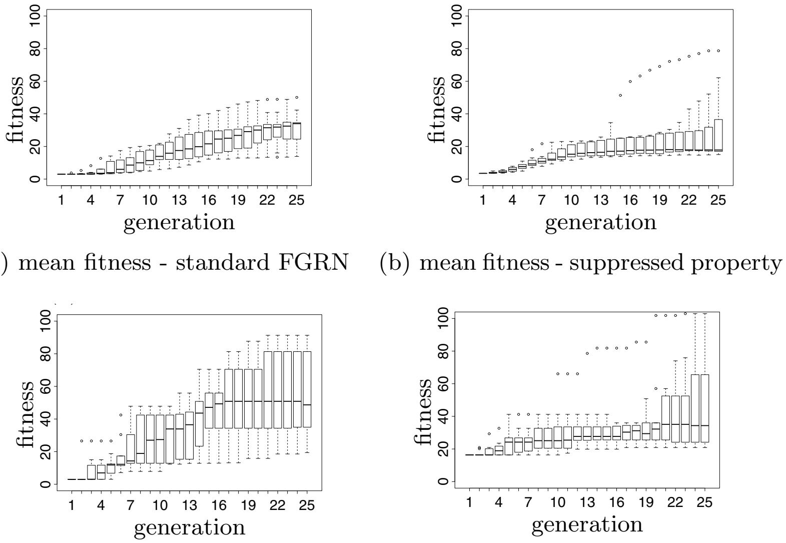
	duration of evaluation period		
	during evolution	during reactivity evaluation	**successful runs**
state-switching on	600	3000	**80%**
state-switching off	1450	7000 (ticks)	**50%**
target visibility threshold 4 (units)			

proteins which determine the state of the system. Each experiment is repeated for 10 independent runs evolving for the full behavior in 25 generations. Every run starts from a population that is pre-evolved for locomotion.

Table 3 represents the settings including the duration of evaluation periods and the distances of target zones for both setups (state-switching on/off). Due to the various speeds of locomotion achieved by the pre-evolved populations of each setup, the durations of evaluation periods are different. The evolutionary progress of the evaluated fitness of the two setups are represented in Fig. 4 for the 10 independent evolutionary runs. As it is demonstrated in the figure, the setup with state-switching property (standard FGRN) performs better than the setup with the suppressed state-switching in terms of the evaluated fitness. Performing a Wilcoxon rank-sum test showed a difference between the two setups with a significance of 0.3 for both mean and best fitnesses in the last generation.

In order to evaluate the ability of the evolved controllers to adaptively react to their environment, the behavior of the robot achieved by the evolved controller in each run is observed against a set of additional differently-positioned targets. For each target, the robot starts from its initial position and the behavior is observed for a sufficiently long evaluation period. The distances of the targets and durations of evaluation periods are presented in Table 3.

The observations indicate that the robots of all the 10 runs of each setup react to the target zone by changing their movement when the zone is reached. For the setup with state-switching, in 8 runs out of the 10, the robot manages to successfully stay in the zone until the end of the evaluation period for all



(a) mean fitness - standard FGRN (b) mean fitness - suppressed property

(c) best fitness - standard FGRN (d) best fitness - suppressed property

Fig. 4. Progress of the fitness values during evolution

the five different targets. In some cases, the robot stands still at some point of the target zone, lying down on the ground or bending. In the other cases, the robot moves slowly back and forth in the way that it stays inside the zone. For the setup with suppressed state-switching, in 5 runs out of the 10, the robot manages to stay in the target zone for all the targets. Table 3 summarizes the settings and the results of the two investigated setups. As it is represented in the table, the standard FGRN setup shows a higher performance than the setup with the suppressed state-switching indicating the importance of the state-switching property in the FGRN system.

3.3 Observing a Typical Evolved Controller

In order to look at the potential behaviors achieved as side-effects of evolution, a typical controller evolved in the standard FGRN setup is chosen from an arbitrary run. The behavior of the robot is observed in two observation settings.

In the first setting, two consecutive targets are used (Fig. 3). The first target is positioned at a random distance from the robot and stays there for a long period of time (1500 sec). Then the target disappears and a new target appears farther at another random distance. For the observed controller, the robot reaches the first target zone and stays there as long as the target exists. Then, it continues the locomotion and reaches the second target and stays there. When the second target disappears as well (3000 sec), the robot starts to move again.

In the second observation setting, the robot is initially positioned in the target zone while the center of the zone is situated behind it. In this case, the robot moves backwards for a short distance. Then the target starts to move backwards in a slow speed. As a response to that, the robot also moves slowly backwards following the target[1]. Although these types of behaviors which are considered as side effects of an evolutionary run might be just random, they are achieved for free and still seem to be interesting.

4 Conclusion

This paper reported an application of FGRN system in controlling a modular snake-shaped robot in a reactive task. The task demands for appropriate changes in the robot behavior in response to particular environmental changes. In previous reactive controllers for multi-modular robots, e.g. HyperNEAT [6] and Central Pattern Generators (CPG) [11], or single robots ,e.g. Neural Networks (ANN) [20], the proper behavior is achieved by modulating an input part of an ANN to switch between the ordinary and special behaviors based on environmental signals. A relatively similar mechanism, namely "state-switching property", is highlighted here as a characteristic exclusive for FGRN model in contrast with other GRN models [13,1,5,4]. As demonstrated in the paper, this capability of implicitly switching between internal states is beneficial when FGRNs are evolved for reactive tasks. The FGRN systems were evolved for several independent evolutionary runs and post-evaluations of the achieved controllers demonstrated successful reactive behaviors for the majority of the runs. In addition, the behavior of an arbitrary evolved controller was observed in new environmental settings and interesting behaviors as side-effects of evolutionary process were detected. The evolvability of the model for adaptive reactions of higher complexity will be studied in future. In order to investigate the effect of "state-switching property" of FGRNs in evolving reactive controllers, an altered version of the model was implemented such that the state-switching property is suppressed. The altered FGRN was evolved for the same task and the comparison of the resulted controllers indicated the significance of "state-switching property" in achieving the successful solutions for the investigated task. In the next step of this study, inclusion of this characteristic in other GRN models will be investigated in order to achieve potential improvements in GRNs.

Acknowledgments. The authors thank Heiko Hamann and Jürgen Stradner for their helps and comments during this work. This work is supported by: EU-IST-FET project 'SYMBRION', no. 216342; EU-ICT project 'REPLICATOR', no. 216240; Austrian Federal Ministry of Science and Research (BM.W_F); EU-ICT 'CoCoRo', no. 270382.

References

1. Banzhaf, W.: Artificial regulatory networks and genetic programming. In: Genetic Programming Theory and Practice, pp. 43–62. Kluwer (2003)
2. Bentley, P.J.: Adaptive fractal gene regulatory networks for robot control. In: Workshop on Regeneration and Learning in Developmental Systems in the Genetic and Evolutionary Computation Conference, GECCO 2004 (2004)

[1] http://payam.zahadat.com/pub/reactiveController1.mpeg

3. Bentley, P.J.: Fractal proteins. J. Genet. Program Evol. Mach. (5), 71–101 (2004)
4. Bongard, J.C., Pfeifer, R.: Repeated structure and dissociation of genotypic and phenotypic complexity in artificial ontogeny, pp. 829–836. Morgan Kaufmann (2001)
5. Eggenberger, P.: Evolving morphologies of simulated 3d organisms based on differential gene expression. In: Proceedings of the Fourth European Conference on Artificial Life, pp. 205–213. MIT Press (1997)
6. Haasdijk, E., Rusu, A.A., Eiben, A.E.: HyperNEAT for Locomotion Control in Modular Robots. In: Tempesti, G., Tyrrell, A.M., Miller, J.F. (eds.) ICES 2010. LNCS, vol. 6274, pp. 169–180. Springer, Heidelberg (2010)
7. Hamann, H., Stradner, J., Schmickl, T., Crailsheim, K.: Artificial hormone reaction networks: Towards higher evolvability in evolutionary multi-modular robotics. In: Proc. of the ALife XII Conference, pp. 773–780 (2010)
8. Harada, K., Corradi, P., Popesku, S., Liedke, J.: Reconfigurable heterogeneous mechanical modules. In: Levi, P., Kernbach, S. (eds.) Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution, Springer (2010)
9. Krohn, J., Gorse, D.: Fractal Gene Regulatory Networks for Control of Nonlinear Systems. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN XI, Part II. LNCS, vol. 6239, pp. 209–218. Springer, Heidelberg (2010)
10. Lodish, H., Berk, A., Zipursky, L.S., Matsudaira, P., Baltimore, D., Darnell, J.E.: Molecular Cell Biology, 5th edn. W.H. Freeman and Company, New York (2003)
11. Manoonpong, P., Pasemann, F., Roth, H.: Modular reactive neurocontrol for biologically-inspired walking machines. The International Journal of Robotics Research 26, 301–331 (2007)
12. REPLICATOR: Project website (2011), http://www.replicators.eu
13. Roggen, D., Federici, D.: Multi-cellular Development: Is There Scalability and Robustness to Gain? In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 391–400. Springer, Heidelberg (2004)
14. Schmickl, T., Hamann, H., Crailsheim, K.: Modelling a hormone-inspired controller for individual- and multi-modular robotic systems. Mathematical and Computer Modelling of Dynamical Systems 17(3), 221–242 (2011)
15. SYMBRION: Project website (2011), http://www.symbrion.eu
16. Winkler, L., Wörn, H.: Symbricator3D – A Distributed Simulation Environment for Modular Robots. In: Xie, M., Xiong, Y., Xiong, C., Liu, H., Hu, Z. (eds.) ICIRA 2009. LNCS, vol. 5928, pp. 1266–1277. Springer, Heidelberg (2009)
17. Zahadat, P., Christensen, D.J., Schultz, U.P., Katebi, S., Stoy, K.: Fractal Gene Regulatory Networks for Robust Locomotion Control of Modular Robots. In: Doncieux, S., Girard, B., Guillot, A., Hallam, J., Meyer, J.-A., Mouret, J.-B. (eds.) SAB 2010. LNCS, vol. 6226, pp. 544–554. Springer, Heidelberg (2010)
18. Zahadat, P., Katebi, S.D.: Tartarus and fractal gene regulatory networks with inputs. Advances in Complex Systems (ACS) 11(06), 803–829 (2008)
19. Zahadat, P., Støy, K.: An alternative representation of fractal gene regulatory networks facilitating analysis and interpretation. Annals of Mathematics and Artificial Intelligence (submitted)
20. Ziemke, T., Thieme, M.: Neuromodulation of Reactive Sensorimotor Mappings as a Short-Term Memory Mechanism in Delayed Response Tasks. Adaptive Behavior 10(3-4), 185–199 (2002)

Adaptation and Genomic Evolution in EcoSim

Marwa Khater and Robin Gras

School of Computer Science, University of Windsor
ON, Canada
{khater,rgras}@uwindsor.ca

Abstract. Artificial life evolutionary systems facilitate addressing lots of fundamental questions in evolutionary genetics. Behavioral adaptation requires long term evolution with continuous emergence of new traits, governed by natural selection. We model organism's genomes coding for their behavioral model and represented by fuzzy cognitive maps (FCM), in an individual-based evolutionary ecosystem simulation (EcoSim). Our system allows the emergence of new traits and disappearing of others, throughout a course of evolution. In this paper we show how continuous adaptation to a changing environment affects genomic structure and genetic diversity. We adopted the notion of Shannon entropy as a measure of genetic diversity. We emphasized the difference in genetic diversity between EcoSim and its neutral model (a partially randomized version of EcoSim). In addition, we studied the effect that genetic diversity has on species fitness and we showed how they correlate with each other. We used Random Forest to build a classifier to further validate our findings, along with some meaningful rule extraction.

Keywords: artificial life modeling, individual-based modeling, genetic diversity, entropy, fitness.

1 Introduction

Charles Darwin's theory of adaptation through natural selection came to be widely seen as the primary explanation of the process of evolution and forms the basis of modern evolutionary theory. Darwin's principle of natural selection relies on a number of propositions. The individuals in a population are not identical but vary in certain traits. This variation, at least partly, is heritable. Individuals vary in the number and the quality of their descendants, depending on the interactions of the individual's trait and its environment. Populations with these characteristics may become more adapted to their environment over generations. The key to adaptation by natural selection is the effect of a multitude of small but cumulative changes. While most of these changes are random, the majority of those that are preserved are not damaging to the fitness of individuals. Instead these variations may turn out to be somehow beneficial to the reproductive success of their carrier. From a genetic perspective, the combination of mutation and natural selection, enforce the emergence of new traits and disappearance of others. These continuous genetic changes help preserve genetic diversity.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 219–229, 2012.

Genetic diversity serves as a way for populations to adapt to changing environments. With more variations, it is more likely that some individuals in a population will possess variations of alleles that are suited for their current environment. Those individuals are more likely to survive to produce offspring bearing those alleles. These alleles will propagate through the population over many generations because of the success of these individuals. In summary, genetic diversity strengthens a population by increasing the likelihood that at least some of the individuals will be able to survive major disturbances. Many biological studies showed that a decrease in population genetic diversity can be associated with a decline in population fitness [1] [2] [3]. Because overall population diversity seems to affect both short-term individual fitness and long-term population adaptive capacity, there is a need to develop an empirical quantitative understanding of the relationship between population genetic diversity and population viability.

Like in many disciplines; simulation modeling played a great role in studying evolutionary processes. Many biological studies that require data of hundreds of years can be obtained by simulation modeling that produces results in a matter of a few hours or days depending on the computational cost of each system. In this paper we show how individuals in EcoSim [4], an evolutionary predator-prey ecosystem simulation, follow the Darwinian evolutionary process through natural selection. We show how genetic evolution and diversity governs the adaptation process. We show that EcoSim's individuals adapt to their changing environment by comparing their behavior with a neutral model - a partially randomized version of EcoSim. We use the Shannon entropy, which is a measure of unpredictability and disorder from Information theory, as a measure of genetic diversity and present the difference in entropy between EcoSim and the neutral model to emphasize the adaptive characteristics of EcoSim. Furthermore, we investigate the relationship between genetic diversity and species fitness and present the correlations found between these two measures in EcoSim. The rest of the paper is organized as follows: A brief description of EcoSim and its neutral model is presented in Section 2 and 3 respectively. Section 4 depicts the details of the entropy as a genetic diversity measure followed by a comparison between EcoSim and its neutral model, in terms of entropy as a genetic diversity, in 5. The correlation results between entropy and fitness are presented in Section 6, followed by building a classifier for inference in Section 7. A summed up conclusion is presented in Section 8.

2 The EcoSim Model

In order to investigate several open theoretical ecology questions we have designed the individual-based evolving predator-prey ecosystem simulation platform EcoSim, introduced by Gras et al. [4] [1]. Our objective is to study how individual and local events can affect high level mechanisms such as community formation, speciation and evolution. EcoSim uses Fuzzy Cognitive Map as a behavior model [5] which allows a combination of compactness with a very low

[1] http://sites.google.com/site/ecosimgroup/research/ecosystem-simulation

computational requirement while having the capacity to represent complex high level notions. The complex adaptive agents (or individuals) of this simulation are either prey or predators which act in a dynamic 2D environment of 1000 x 1000 cells. Each individual possesses several physical characteristics including age, minimum age for breeding, speed, vision distance, levels of energy, and the amount of energy transmitted to the offspring. Preys consume grass, and predators predate on prey individuals. Grass distribution is dynamic, as it diffuses in the world and disappears when consumed by preys. An individual consumes some energy each time it performs an action such as evasion, search for food, eating or breeding. Each individual performs one action during a time step based on its perception of the environment. Fuzzy Cognitive Map (FCM) [5] is used to model the individual's behavior and to compute the next action to be performed. The individual's FCM is coded in its genome and therefore subjected to evolution. A typical run lasts tens of thousands of time steps, during which more than a billion of agents are born and several thousands of species are generated, allowing evolutionary processes to take place and new behaviors to emerge to adapt to a constantly changing environment. Our simulation embodies species as a set of individuals sharing similar genomes [6]. Indeed, every member of a species has a genome that is within a threshold distance away from the species genome - an average of the FCMs of its members. To model the process of speciation, EcoSim allows splitting of a species into two sister species. The splitting mechanism produces two clusters of individuals with high intra-cluster similarity and strong inter-cluster dissimilarity. It is worth noting that the speciation mechanism is only a labeling process: the information about species membership is not used for any purpose during the simulation but only for post-processing analysis of the results.

Formally an FCM is a graph which contains a set of nodes, each node being a concept, and a set of edges, each edge representing the influence of one concept on another. In each FCM, three kinds of concepts are defined: sensitive (such as distance to foe or food, amount of energy, etc), internal (fear, hunger, curiosity, satisfaction, etc) and motor (evasion, socialization, exploration, breeding, etc.). We use a FCM to model an agent's behavior (structure of the graph) and to compute the next action of the agent (i.e. through the dynamics of the map). The FCM serves as a genome for each individual. The genome length is maximum 390 sites, where each site corresponds to an edge between two concepts of the FCM. The FCM allows the formation of new edges and disappearing of others through the evolutionary process. During a breeding event the FCMs of two parents are combined and transmitted to their offspring after the possible addition of some mutations which is similar to the genetic process of recombination. The behavior model of each individual is therefore unique.

3 The Neutral Model

In order to study the effect of adaptation on evolution, we built a neutral shadow [7] of EcoSim. All selection processes and behaviors in the neutral shadow for the predator/prey are random, which eliminates natural selection from this model.

In terms of the behavioral model of this version, all the actions such as eating, hunting (for predators), socializing, searching for food and escaping (for prey) are removed. The only two actions any individual can take are reproduction and movement. Unlike in the EcoSim, in the neutral model there is no necessity for the individuals to have genetic similarity to reproduce. Instead, in the neutral model the reproduction action is done by randomly choosing any 2 individuals in the world. The statistics of genetic operations (mutation rates and crossover) are the same as EcoSim. In EcoSim, individuals choose to reproduce according to their internal state, suitable environmental conditions and behavior model but not in the neutral model. To preserve population dynamics in neutral model similar to that of EcoSim, the Lotka-Volterra computational model [8] is used. This model controls the number of births and deaths at each time step. In addition, death of individuals and pairs of parents for reproduction are randomly selected. In this way a similarity in population sizes between the neutral shadow and EcoSim is preserved. Finally, the movements in the neutral model are random, but the distribution of distances is kept the same as in EcoSim.

The crucial property of EcoSim neutral shadow is that its evolutionary dynamics are identical to EcoSim except that neither the presence not the frequency of a genotype can be explained by its adaptive significance. This is because all selection in the neutral model is random, so no genotype has any dominance over any other. In other words, although gene states are subject to the same variation as in EcoSim, they have no evolutionary fitness consequences or effects. In addition, changes in the environment have no effect on individuals in the neutral model. Consequently, the process of natural selection is considered to be eliminated in this neutral model.

4 Entropy as a Measure of Genetic Diversity

Depending on the specific problem or representation being used, ranging from biological domain to genetic programming, numerous diversity measures and methods exist. Foe example, Sherwin [9] has shown the efficancy of Shannon entropy in measuring diversity in ecological community and genetics. He has also highlighted the advantages of using entropy based genetic diversity measures, and surveyed these diversity measures. A close relationship between biological concepts of Darwinian fitness and information-theoretic measures such as Shannon entropy or mutual information, was found [10]. Shannon Information theory [11] defines uncertainty (entropy) as the number of bits needed to fully specify a situation, given a set of probabilities. These probabilities can be estimated by simply counting the abundance of each genotype (site) in the population. The per-site entropy of an ensemble of sequences X, in which genotype s_i occurs with probability p_i is calculated as

$$H(X) = -\Sigma p_i \log_2(p_i) \tag{1}$$

where the sum goes over all different genotypes i in X. Next, the entropy content of the whole sequence (genome) is approximated by summing the per-site entropy over all sites in the sequence. This is only an approximation because it ignores interactions between sites (i.e. epistasis). We do not have a fixed set of alleles but they are discreet values that change over time in the simulation. The lower the entropy, the less diverse are the genomes of a population and vice versa. There is a limit in the desired values of entropy in EcoSim. When it approaches its maximum(corresponding to an uniform distribution of all genotypes) it indicates a completely uniform population close to randomness. On the other hand very low entropy (close to 0) means that there is too much similarity between individual genomes, and means that individulas need to diverge more in order to adapt to a dynamic environment. A good balance between learning from the environment (low genetic diversity) and increasing the diversity (high genetic diversity) should be met in order to ensure the well being of species.

5 Evolution in EcoSim verses Neutral Model

The FCM of each individual plays the role of its genome and has a maximum size of 390 sites. Every site is a real discreet number which measures the level of influence from one concept to another. Initially all prey and predator individuals are given the same values for their genome respectively. Time step after another, as more individuals are created, changes in the FCM occur due to the formation of new edges, removal of existing ones and changes in the weight associated to edges. We neglect the first couple of thousand of time steps in our calculations to overcome any misleading results due to the initial similarity between individual genomes. In each time step we have a value of entropy of all existing preys species, along with the entropy of the entire population of prey. We also calculated the fitness for every species as the average fitness of its individuals. We define fitness of an individual as the age of death of the individual plus the sum of the age of death of its entire direct offspring. Accordingly, the fitness value mirrors the individual's capability to survive longer and produce high number of strong adaptive offspring.

The information contained within a genome determines how the organism behaves in its current environment. Thus, this information determines the capability of the organism to reproduce and transmit its genome. The environment changes from one place to another and from one time step to the next. Individuals that evolve in different parts of the world have different information about the environment they evolve in stored in their genome. Furthermore, as we model a predator-prey system, we also have co-evolution. The strategies (behavior) of each kind of individula(predator/prey) are continuously changing as they try to adapt to the other kind. The more the individuals try to learn the more the environment changes and the more there is still something different to learn. This fact drives the individuals to keep learning and continuously try to come up with survival strategies that helps them adapt to their changing environment. This is the reason behind the fluctuations we see in the EcoSim entropy curves see Fig.1. On the other hand the neutral model shows much more steadiness in the entropy values. Under highly random conditions and when natural selection

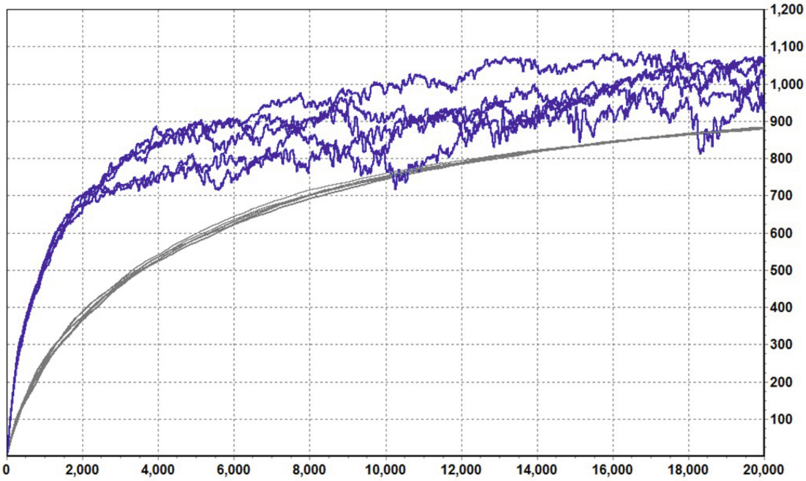


Fig. 1. Global Entropy for 10 different runs of the simulation. Top 5 curves are for EcoSim and lower 5 for Neutral Model.

is eliminated, the genomic structure shows neither learning nor adaptation to the surrounding environment. These results show that entropy changes through the course of evolution. The EcoSim simulation gave us the chance to acquire data for thousands of generations and to study the performance of entropy as a genetic diversity measure.

6 Measuring Correlation between Entropy and Fitness

In order to further emphasize the importance of genetic diversity to adaptation and thus the well being of individuals, we were encouraged to study the effect that genetic diversity has on fitness. EcoSim gives us the chance to study the relation between species genetic diversity and species fitness without the limits in environmental conditions and time scales found in biological studies [2] [3] [12], but in highly variable environments and across evolutionary time. There are many factors affecting genetic diversity and fitness, and the correlation between them. At every time step we calculate the entropy and the fitness for all existing species. In order to investigate their possible correlations, we first begin by calculating the Spearman's cross correlation [13], between entropy and fitness of all prey species. A perfect Spearman correlation of +1 or -1 is attained when each of the variables is a perfect monotone function of the other; a value close to zero means that there is no correlation.

In our evolutionary ecosystem the effect of entropy on fitness is not immediate. A time shift between the variation in entropy and its effect on fitness is therefore expected. Also, because we did not determine which attribute is the cause of the other we calculate the correlation in both shift directions. We computed the Spearman correlation coefficient, between these two time series for every possible

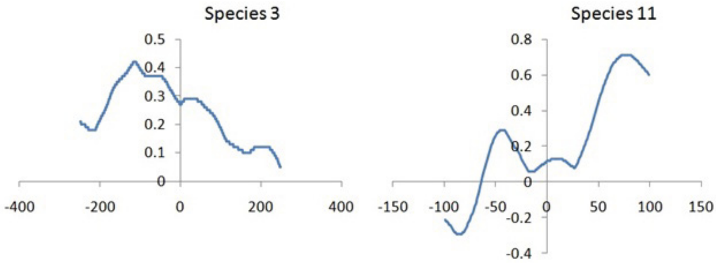


Fig. 2. Different species correlation values between entropy and fitness. x-axis represents the different time shifts. Y-axis represents the correlation values.

shift between -s and +s time steps. Thus, we correlated the entropy at time t with fitness at time t + S where S ranges from -s to +s. We've presented the cross-correlation charts for some prey species in Fig.2. The x-axis in these charts represents the different shifts for the time series. The y-axis represents the cross-correlation value at the corresponding shift. From the figure we see that not only different species have different cross-correlation values, but also the same species correlated differently based on the time shift.

It should be noted that the dynamic environment, co-evolution and changing parameters with time, all affect species behavior. Thus, correlation values for the same species might vary through the course of evolution. This presents a feasible problem to study in our model but not in biological experiments. This fact encouraged us to add a time frame to the two series and measure correlation within the specific time frame. Consequently, we split these time series into sliding windows. Within each window we calculated all possible correlations with different shifts ±s. Then we chose the highest correlation value (whether positive or negative) and assigned it to the corresponding species instance of the time series.

In order to examine the possible correlation values between species entropy and fitness at every time step we usde data collected from 5 different runs of the EcoSim simulation, each running with the same initial conditions. Each replicate ran for 16,000 time steps and generated 110,000 instances (an instance corresponding to one given species at give one time step) in average. For each instance we calculated the Spearman's cross correlation between entropy and fitness for the corresponding window. We assigned three different classes to the correlation values. Correlations with values between -0.5 and 0.5 are class WEAK CORR which shows the situation where there is either no or weak correlation. Correlation values above 0.5 are high positive (HIGHP) and correlation values below -0.5 are high negative (HIGHN) respectively. Different shift values and sliding windows ranges have been examined and previously presented in [14]. We choose±25 as a shift value based on the analysis of which shift leads to the highest correlations and a sliding windows of 200. In an average of 5 different runs of the simulation 26.8% of instances had HIGHP correlation, 38.4% of instances had HIGHN correlation and 34.7% of instances had WEEK correlation.

Although there are many factors that might affect fitness besides entropy, we managed to find strong correlation between entropy and fitness for all prey species. We observed high values for both negative and positive correlations. These results support the claim of the great influence the genetic diversity has on the well being of species. High positive correlation values mean that an increase in the genetic diversity, results in an increase in species fitness. However, there are many ways to interpret these results. A newly forming species with a small population would gradually tend to increase its genetic diversity and will therefore correlate positively with fitness. Also, since individuals in EcoSim adapt to a constantly changing environment these adaptations could be mirrored in the increase of individuals' genome similarity (and thus a decrease in entropy), as new behaviors diffuse in the population. Conversely negative correlations imply that a species decreases diversity in order to reach stability by learning from its environment and adapting.

7 Building a Random Forest Classifier for Inference and Rule Learning

Our motivation to validate these results and further investigate the reason behind these correlation values encouraged us to build a classifier. The purpose of building this classifier is first to see if some specific species properties can predict the current evolutionary behavior of the species, that is if it is learning from the environment or increasing its diversity to be able to react to a future change in the environment. It can also help to understand what factors and conditions affect the evolutionary of behavior. The Random Forest [15] technique includes an ensemble of decision trees and incorporates feature selection and interactions within the learning process. It is nonparametric, efficient, and has high prediction accuracy for many types of data including high dimensional ones. We chose features from both individual's internal and physical concepts, such as average energy level, reproduction rate, population size, speed of individuals, spatial dispersal and others, to predict the class correlation value between genetic diversity and fitness. All together we chose 15 features that best described internal and physical properties of any species and verified if they could predict the class correlation variable. To increase the quality of the classifier we used feature selection [16] in order to extract the most important features from the above list. This step provided more semantics about which features most influence the value of correlation. The best chosen features were population size, entropy, fitness, spatial dispersal, average age of the individuals and number of failed reproductions. We used the Random Forest classifier implemented in the weka environment [17]. We split instances for every run into two sets: train and test and used 10 fold cross validation. The average of 5 classifiers testing accuracies representing 5 different simulation runs was 96.7%. The high classification accuracy validates our use of entropy as a measure of genetic diversity and its high correlation with fitness. It also shows that there exist specific conditions of the species that lead to a positive or negative correlation between fitness and genetic diversity.

The model generated by the Random Forest can be challenging to interpret. To by-pass this limitation we use the JRip rule learner [18] to extract more semantics from the prediction model and gain more insight about the conditions affecting correlation between genetic diversity and fitness. JRip implements a propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER), which is an optimized version of IREP. Different IF THEN rules are learned from JRip to predict the three correlation classes. In 5 different runs 19 rules were discovered in average with average accuracy of 76% using 10 fold cross validation see Table 1. We were mainly interested in studying the rules that predict the HIGHP and HIGHN classes and we present some of these rules having the highest number of instances.

Table 1. JRip rule learner accuracies and number of produced rules for five different runs of the simulation

Run	Train accuracy	Test accuracy	No. of rules
Run 1	76%	75.6%	24
Run 2	71.7%	72%	23
Run 3	75%	75.8%	24
Run 4	79%	78.8%	7
Run 5	75.5%	76.1%	18
Average	75.4%	76%	19

- IF number of individuals is low, AND fitness is low, AND entropy is low, AND failed reproduction is high THEN correlation is HIGHP.

- IF number of individulas is low, AND age is high, AND fitness is low, AND entropy is low THEN correlation is HIGHP.

- IF fitness is low, AND age is medium, AND spatial dispersal is low THEN correlation is HIGHP.

- IF number of individuals is high, AND age is high, AND entropy is high, AND spatial dispersal is high THEN correlation is HIGHN.

- IF spatial dispersal is high, AND number of individulas is high, AND age is medium, AND entropy is medium, AND fitness is high THEN correlation is HIGHN.

- IF failed reproduction is low, AND entropy is high, AND number of individuals is high THEN correlation is HIGHN.

The other discovered rules were also similar. In general, we found that a low number of individuals associated with a low entropy, low fitness and low spatial dispersal led to a high positive correlation between entropy and fitness. Small species tended to increase their genetic diversity in order to increase their fitness. On the other hand, a high number of individuals associated with a high entropy, high fitness and high average age led to high negative correlations between entropy and fitness. Large species in terms of population size tended to move towards lower genetic diversity as individuals learned common survival strategies that tended to increase their fitness.

8 Conclusion

We showed how the evolutionary process implemented in EcoSim affects the behavioral model of the individuals as they adapt to a changing environment. To emphasize the capability of EcoSim to model evolutionary behavioral adaptation we compared it to a partially random version focusing on genetic diversity. We showed how entropy used to measure genetic diversity, behaves differently in both systems. The fluctuation in entropy curves for EcoSim showed how individuals try to learn and adapt to their environment. On the other hand the neutral model showed more steadiness in the curves due to more randomness and elimination of natural selection process. Furthermore, we presented high correlation values between species fitness and genetic diversity which strongly indicates how genetic diversity affects the well being of the species. A validation step was performed with the use of machine learning techniques. A random forest classifier was built to predict the correlation values based on internal and physical properties of species used as features. The rules discovered from the rule learner, which seem to be biologically pertinent, gave us more understanding about the conditions affecting the values of correlation between genetic diversity and fitness.

Acknowledgments. This work is supported by the NSERC grant ORGPIN 341854, the CRC grant 950-2-3617 and the CFI grant 203617 and is made possible by the facilities of the Shared Hierarchical Academic Research Computing Network (SHARCNET:www.sharcnet.ca).

References

1. Reed, D., Frankham, R.: Correlation between fitness and genetic diversity. Conserv. Biology 17, 230–237 (2003)
2. Markert, J., Champlin, D., Gutjahr-Gobell, R., Grear, J., Kuhn, A., McGreevy, T., Roth, A., Bagley, M., Nacci, D.: Population genetic diversity and fitness in multiple environments. BMC Evolutionary Biology 10 (2010) 1471–2148–10–205
3. Vandewoestijne, S., Schtickzelle, N., Baguette, M.: Positive correlation between genetic diversity and fitness in a large, well-connected metapopulation. BMC Biology 6 (2008) 1741–7007–6–46
4. Gras, R., Devaurs, D., Wozniak, A., Aspinall, A.: An individual-based evolving predator-prey ecosystem simulation using fuzzy cognitive map as behavior model. Artificial Life 15(4), 423–463 (2009)
5. Kosko, B.: Fuzzy cognitive maps. Int. Jornal of Man-Machine Studies, 65–75 (1986)

6. Aspinall, A., Gras, R.: K-Means Clustering as a Speciation Mechanism within an Individual-Based Evolving Predator-Prey Ecosystem Simulation. In: An, A., Lingras, P., Petty, S., Huang, R. (eds.) AMT 2010. LNCS, vol. 6335, pp. 318–329. Springer, Heidelberg (2010)
7. Bedau, M.A., Snyder, E., Packard, N.H.: A classification of longterm evolutionary dynamics. In: Proc. of Art. Life VI, pp. 228–237. MIT Press (1998)
8. Volterra, V.: Variations and fluctuations of the number of individulas in animal species living together. Animal Ecology 3, 409–448 (1931)
9. Sherwin, W.B.: Entropy and information approaches to genetic diversity and its expression: Genomic geography. Entropy 12, 1765–1798 (2010)
10. Bergstrom, C., Lachmann, M.: Shannon information and biological fitness. In: IEEE Information Theory Workshop, pp. 50–54 (2004)
11. Shannon, C.: A mathematical theory of communication. Bell Systems Technical Journal, 379–423 (1948)
12. Oostermeijer, J., van Eijck, M., den Nijs, J.: Offspring fitness in relation to population size and genetic variation in the rare perennial plant species gentiana pneumonanthe (gentianceae). Oecologia 97, 289–296 (1994)
13. Siegel, S.: Nonparametric Statistics for the Behavioral Sciences. McGraw-Hill, New York (1956)
14. Khater, M., Salehi, E., Gras, R.: Correlation between Genetic Diversity and Fitness in a Predator-Prey Ecosystem Simulation. In: Wang, D., Reynolds, M. (eds.) AI 2011. LNCS, vol. 7106, pp. 422–431. Springer, Heidelberg (2011)
15. Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)
16. Yang, Q., Salehi, E., Gras, R.: Using Feature Selection Approaches to Find the Dependent Features. In: Rutkowski, L., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2010. LNCS (LNAI), vol. 6113, pp. 487–494. Springer, Heidelberg (2010)
17. Witten, I., Frank, E.: Data Mining- Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, USA (2000)
18. Cohen, W.: Fast effective rule induction. In: 12th International Conference on Machine Learning, pp. 115–123 (1995)

Degeneration of a von Neumann Self-reproducer into a Self-copier within the Avida World

Tomonori Hasegawa and Barry McMullin

The Rince Institute, Dublin City University, Ireland
tomonori.hasegawa2@mail.dcu.ie, barry.mcmullin@dcu.ie

Abstract. The theory of machine self-reproduction formalised by John von Neumann illustrates the real living organisms' self-reproduction equipped with *genotype* and *phenotype*. However, within such a simulated world as *Avida*, this particular style of self-reproduction has not been previously studied. In an attempt to characterise the von Neumann style self-reproducer in a computational system, we have implemented a novel seed program that self-reproduces using von Neumann's architecture. We expected that distinctly different evolutionary dynamics of organisms in the system would be observed, specifically including the possibility of mutationally altered genotype-phenotype mapping. However, what we have observed is degenerative displacement by self-copiers, which are conventional self-reproducers in the system. The mutational easiness of this degeneration was not anticipated, although we knew the selective advantage that such self-copiers intrinsically would have in the system.

Keywords: von Neumann, self-reproduction, Avida, artificial life, genotype-phenotype mapping, evolutionary growth of complexity.

1 Introduction

1.1 Von Neumann's Architecture of Self-reproduction

The abstract architecture of machine self-reproduction formalised by von Neumann by around 1948 [9,2], comprises two components characterised by active and passive roles. The active machinery is subdivided into four conceptually different components: namely, a general purpose constructor, a tape copier, a controller, and ancillary machinery. The former three components (i.e., a general purpose constructor, a tape copier, and a controller) collaboratively engage in, and are directly in charge of, creation of an offspring machine; whereas the ancillary machinery is a part not directly involved in the reproduction process, but provides functionality not directly associated with reproduction. The passive machinery is called a description tape, and it carries information which describes, or encodes, a machine organisation in an arbitrary way.

In this schema, given a machine description, the controller takes the initiative to manoeuvre (i) the constructor to decode the description, according to a

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 230–239, 2012.

specific procedure, and to create an "offspring" machine's active components, as well as (ii) the copier to copy the description into an offspring machine.

This amounts to machine self-reproduction when the description part describes the whole machine itself (both active and passive machineries) and successfully organises an offspring machine, which has the same organisation as its parent and is capable of reproducing itself (see Fig. 1). Offspring machines will repeatedly reproduce themselves in this way, thereby potentially increasing exponentially in number until some resource limit is encountered.

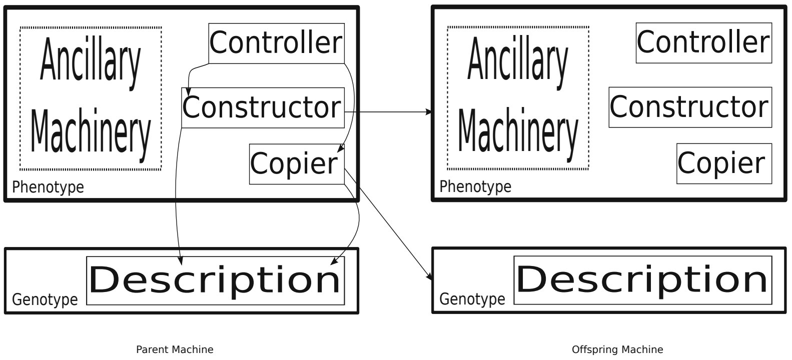


Fig. 1. Von Neumann Architecture of Machine Self-reproduction

There is an alternative architecture for self-reproduction that relies on self-inspection and copying, but lacks any such division of reproductive roles into active or passive components, as opposed to von Neumann's model of self-reproduction. Both of these schemas for self-reproduction are non-trivial in that they potentially support inheritable variation by allowing room for diverse and cumulative mutation. They are therefore distinguished from "naïve" self-reproduction such as growing crystals [10, p. 86], which would not support inheritable variation.

1.2 Genotype-Phenotype Mapping and Its Evolution

A particular significance of von Neumann's architecture of self-reproduction arises where a mutation that occurs in a particular section of the description which encodes the constructor can breed true [3]. In other words, such a mutation will manifest as an offspring machine's constructor that is different from its parent's (which signifies that the genotype-phenotype mapping is also different), and will inherit in the offspring machine's description accordingly. Thus the genotype-phenotype mapping, or the relationship between genotype and phenotype, may exhibit evolvability, bringing about distinctive mutational pathways and resulting evolutionary dynamics which otherwise cannot be achieved.

Behind von Neumann's theory of self-reproduction, there seems to have been inspiration drawn from Schrödinger's "What is Life?" [8], in which the physical basis of mutation and inheritance in self-reproducers is developed noticeably earlier than the discovery of the double helix structure of DNA. The more insights were gained through the subsequent discovery of DNA structure and investigation of genotype-phenotype mapping in living organisms, the further it turned out that von Neumann's architecture in fact already illustrated how the genetic mechanism of decoding and translation is performed in real living organisms.

Whereas, in biology, genotype-phenotype mapping generally refers to the relationship between the genome and its manifestation such as physical or behavioural traits, that in computational systems may as well be described as decoding a sequence of numbers, according to predefined decoding rules, into another sequence of numbers that harbours particular functions and/or properties. In other words, the genotype-phenotype mapping in computers signifies what kind of transformation of numbers is applied to the description.

1.3 The Problem Situation

Epitomised by his series of works centred around Cellular Automata [9,10], von Neumann has theorised and elaborated the implementation of the particular universe, which has been typically represented in a two dimensional grid world. It is noteworthy that von Neumann himself seems to have discounted the possibility that mutation(s) that have occurred in the constructor component will result in a viable offspring machine [10, p. 86], ruling out the potential evolution of genotype-phenotype mapping which such mutation(s) can trigger. While there have been various forms of implementation of von Neumann's architecture on computational systems (see [5] for example), these subsequent attempts have not specifically considered the possible evolution of genotype-phenotype mapping.

Thus, our long term goal is to investigate the question of whether von Neumann's mode of self-reproduction, including the specific possibility for evolving the genotype-phenotype mapping, might ultimately enhance the potential for spontaneous evolutionary growth of complexity. Especially, the present research focuses on preliminary characterisation of this evolution of genotype-phenotype mapping, particularly within a system called *Avida*.

2 The *Avida* World

2.1 System Structure

Avida is a computational evolution system that has been in development by Adami et al. since 1994 [1,4] . Inspired by its predecessors including *Coreworld* [6] and *Tierra* [7], Avida is an abstraction of a typical distributed or cluster computer. Each node comprises a virtual CPU running on memory, and the CPU has components such as several registers and stacks and various heads to work with.

In consequence of such a system architecture, Avida makes a model ecosystem, where digital “organisms” compete with one another for space and CPU time, exhibiting Darwinian evolution. Due to this fact, as an experimental platform, it has been widely used for investigation of general properties of spontaneous evolutionary processes.[1]

Characteristically, those predecessors, Coreworld and Tierra, consist of a single shared memory space for organisms to reside in; however, the Avida world is represented as a finite two dimensional lattice of compute nodes, each node denoting a virtual hardware that comprises a memory space possibly occupied by an executable program with a virtual CPU running on it. In other words, one of the distinctive features of the Avida world is that an organism has a separate variable sized memory, unlike its predecessors that adopt a single fixed-sized memory shared by all organisms.

2.2 Self-reproduction in Avida

A successfully designed organism when seeded into the Avida world (which is often referred to as an *ancestor*) can reproduce itself by running its CPU and executing the program on its memory, the offspring organism replacing another organism that resides in the neighbourhood. In this manner, the population of organisms grow, conceivably filling up the finite nodes of the Avida world, in the way that a colony of bacteria increase in a petri dish; however, unlike such real living organisms, the organisms in the Avida world principally face no limit of nutrients, in the sense that the CPU time necessary to run them is given as much as the experimenter wishes.

During reproduction, the (putative) parent organism allocates memory within it for creation of its offspring program, and when the creation is done, it divides off the offspring program, replacing a node in the neighbourhood, with the parent being reset to a predefined initial setting (i.e., typically, the instruction head and the other heads will position back to the beginning of the memory and the registers and the stacks will be set to zeros), and its CPU then continuing to (re-)run the program. Conventionally, the self-reproduction in the Avida world is achieved by self-copying based on self-inspection, in particular by using an `h-copy` instruction provided in the default instruction set.

The instruction `h-copy` (with `h-` denoting “heads” used in this copying process, namely, the read head and the write head) is a compound instruction that both reads an instruction from a source memory location indicated by the read head, and writes it straight into a destination memory location indicated by the write head. With regard to this, it is worth mentioning that, by design, accidental program alterations can occur, and give rise to inheritable mutations. They are inheritable mutations because, in these standard self-reproducers in the Avida system, the complete memory image of the parent is copied to the offspring, so

[1] In many research works, external fitness is usually applied in order to investigate how the environmental factors affect the evolutionary process; however, in the present paper, we do not enable this as it is not relevant to our immediate purpose.

any change in the memory image of a parent will be copied to the offspring, and then preserved in subsequent generations. As the evolution proceeds, mutations will be accumulated, leading to progressive variation among organisms.

3 Implementation Design

3.1 Design of an Ancestor

Based on the scheme of self-reproduction originally presented by von Neumann, we have designed a novel ancestor to seed the Avida world. This is opposed to such self-copying architecture as provided in the Avida system that lacks genotype-phenotype distinction and presumably leaves no room for the genotype-phenotype mapping to evolve. The novel ancestor may be decomposed into *genome* and *phenome*, corresponding to the passive machinery and the active machinery of von Neumann's self-reproductive architecture, respectively. Thus, *phenome* supports the active process of *decoding* and copying of the description (*genome*), including allocation of memory in the parental memory where the memory image of a prospective offspring is created, and also dividing off that part as an offspring to a neighbouring node. We have included no explicit ancillary machinery in our design of the ancestor, for every instruction of its program commits to the reproductive process.[2]

Decoding. *Decoding*, or translation, is the process of mapping the content of a memory location that is read from a putative parent organism into the content of a memory location in the prospective offspring organism. More specifically, a content of a memory location may be interpreted as an instruction, which is predefined in the form of an *instruction set* in the system configuration setting, or, alternatively, may not be interpreted as an instruction but as uninterpreted data, usually represented as a distinct integer number.

Importantly, we had to enable two kinds if instructions, namely `read` and `write` instructions, into the default instruction set for the decoding purpose mentioned above. The reason is because, unlike the `h-copy` instruction used for copying, these were not activated by default. The instruction set provides a list of numbers, each corresponding to a particular instruction. These numbers compose an organism's program. The set also implies possible resulting number(s) in mutational events.[3]

[2] If, through accumulated mutations, there proves to be any part of the program that does not necessarily engage in the reproductive process, then that part may be regarded as ancillary machinery in von Neumann's terms.

[3] Note that the instruction set sequentially defines what instruction corresponds to what (index) number (i.e., 0, 1,... N-1; N is the size of the set) and is therefore fixed in size; however, this finite set of integers is capable of interpreting any manipulable numbers (integers) as an instruction, in a certain manner where, in the case of a number larger than the set size, the modulo of the number is used as an index number within the set.

Lookup Table. We have implemented the ancestor so that the decoding is performed based on a *lookup table* incorporated in the phenome, on the basis of mapping one number into another. The initial choice of the mapping is quite arbitrary, and due to such arbitrariness, there is more to this mapping than genetic coding in real biology (e.g., codons translated into proteins with physical or chemical constraints). Likewise, the choice of implementation via such a lookup table is arbitrary. Our lookup table has been designed as a list of numbers the same size as the instruction set, with each number respectively signifying what each number in the instruction set translates into.

The decoding mechanism works properly using this lookup table because a read number can be treated as an index number of the list (or, relative location within the list) as well as an instruction. For example, if the parent program reads a number 3 in the genome, it first looks up what is at the index of 3 of the lookup table, and it then goes on to write the looked-up new number, say, 5, into the offspring organism, meaning that the number 3 is decoded into the number 5. Importantly, we have chosen the mapping to be invertible, one-to-one, correspondence, so that with the designed lookup table it is possible to reverse-engineer the ancestor: we can obtain the genome of the ancestor, when given the phenotype (incorporating the lookup table), such that the phenome is the decoded version of the genome, and the genome is the encoded version of the phenome.

3.2 Self-reproduction of the Novel Ancestor

In brief, a typical self-reproduction of this ancestor will be achieved as follows: the parental phenome decodes the parental genome based on the lookup table into the offspring phenome; then, the parental phenome copies the parental genome one instruction after another into the offspring genome (see Fig. 2). The genotype-phenotype mapping of this novel program is analogous to the mapping from a string of words (i.e. genome) into another string of words (i.e. phenome) underlain by the mapping between words defined by the lookup table. Therefore, it is reasonable to expect the genotype-phenotype mapping to be evolvable and infinite in principle, with the lookup table itself and the usage of it being open to change evolutionarily. The ancestor obtained in this way is much larger in size and spends more steps to self-reproduce than the standard shortest (self-copying) ancestor due not only to the more complex procedure of self-reproduction, but also to the primitiveness of the instructions that resembles assembly language, which makes the program structure substantially cumbersome.

Mutation. We selectively allowed only single-point mutation to occur in the experiment, rather than other types of mutation such as multi-point mutation, insertion and deletion. This is because, by doing so, the traceability of mutational pathway becomes straightforward. Single-point mutation is an occasional inheritable change of a memory content in the genome. A change in a memory of the phenome does not inherit, so it is not mutation. If a default design of ancestor in Avida, which is a self-copier, undergoes any changes in its memory image, the changes will be all inheritable and are thus referred to as mutation. On the contrary, however, in our implementation of von Neumann's self-reproduction, not all changes in the memory image, but those in the "genome" part that is selectively caused in a process of copying, are inheritable. Note that mutation in this style of self-reproduction will exhibit delayed manifestation. That is, one generation is needed for a single-point mutation to occur (this is when a parent organism reproduces an offspring), and then another generation is needed for it to get expressed (this is when the offspring reproduces its offspring). This delay, however, does not apply in the case of the Avida default ancestor because when a mutation occurs, it is expressed immediately in the offspring.[4]

With the mutation expressed in the phenome (as opposed to *inherited* in the genome), the organism could be viable and fertile. Because all of our novel ancestor's phenome is concerned with self-reproduction, the expression of the mutation is most likely to affect the reproductive process. It is still possible, however, that somehow the reproductive functions work and succeed to produce an offspring organism that breeds true (i.e., inheriting the mutation as well as preserving the self-reproductive functions). What we are particularly interested in exploring is the possibility that there might occasionally be a viable, fertile, mutations whose expression affects the "general purpose constructor" part in the von Neumann architecture.

4 Experimental Results and Discussion

4.1 Behaviour of the Hand-Designed Ancestor

Our naïve expectation was that the novel ancestor would be able to give rise to a "traditional" Avidian evolutionary process. More specifically, we expected that the ancestor could reproduce reliably enough to increase in population under appropriate rates for the stochastic effects, where the total population growth would be limited by the size of the Avida world. Not only that, mutant organisms capable of breeding true, and still maintaining von Neumann's architecture, were expected to emerge, through at least some stochastic effects applied on the genome, and to undergo selection and evolution accordingly.

When seeded in the Avida world, this novel ancestor turns out to successfully reproduce itself, indicating the implementation of a von Neumann style ancestor incorporating genotype-phenotype mapping in this system is realisable. In other words, the organism decoded and copied its description as designed so as to reproduce itself.

Moreover, with only single-point mutation enabled, we have run Avida simulations repeatedly and singled out runs where takeover of the dominant organisms

[4] Likewise, such a delay does not carry over if there are other ways available to occasionally change the memory content (e.g., "cosmic ray" type of alterations in the memory image, decoding errors etc.), in which any change made in the phenome ends up being temporary.

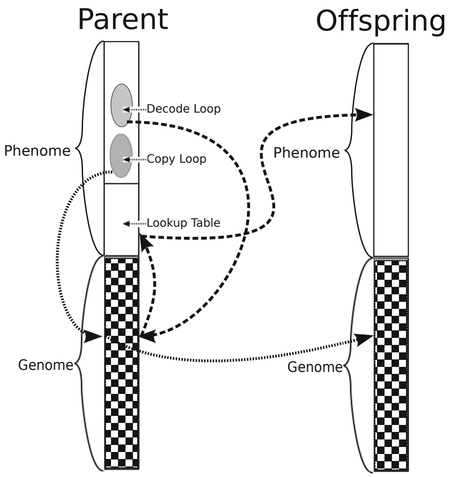


Fig. 2. Designed Self-reproduction of the Novel Ancestor

is observed. For each of these runs, we have traced mutational changes and identified the transitional path from the ancestor to the final dominant strain. Analysis of these established that there are several cases of takeover by self-copiers. In other words, degeneration from the von Neumann style architecture to the self-copying architecture has been observed. This marks the loss of the decomposition into genome and phenome, and also the loss of the mutable or evolvable genotype-phenotype mapping. That is to say, the von Neumann style organism becomes a self-copier that no longer has a division of labour between phenome and genome.

4.2 How the Degeneration Has Occurred

Surprisingly enough, it is found that one step of single-point mutation is sufficient to cause this degenerative transition. These self-copiers self-reproduce only utilising `h-copy`. Although there may possibly be self-copiers that are based on using `read` or `write`, the self-copiers we have obtained are found to self-reproduce without the help of those instructions. The mechanism of the degeneration is as follows: The previous "decode" loop using `read` and `write` is destroyed by a mutational change and starts to malfunction, but the previous "copy" loop using `h-copy` happens to function to overwrite the mistake made by the decode loop and copy the entire memory image, successfully making both ends meet to precisely self-produce in an unanticipated way.

Once a self-copying strain arises, it is not at all surprising that it displaces the von Neumann self-reproducers because it is selectively favoured by the Avida system due to its high reproduction rate; in this particular case, the self-copier, being the same length as the hand-designed ancestor, takes less than half of the

CPU cycles required by that ancestor to self-reproduce.[5] A factor underlying the difference between the necessary CPU cycles is that, for a content to be written into one memory location of the offspring, the decoding needs to execute more instructions than the copying. A self-copier is expected to be faster because it avoids the cost of the decoding.

Regarding other strains, due to the relative largeness of size and complexity of their structures, it is not straightforward to distinguish what style of self-reproduction a particular organism employs. Determining the style of self-reproduction an organism does employ can be rather cumbersome because even when the program of the organism looks the same, the use, or interpretation, of the numbers in the program can be completely different. Therefore, instead of scrutinising how each distinctive organism reproduces itself, we examined the gestation time because, it indicates, even if crudely, the style of self-reproduction.

Out of the organisms obtained throughout the run, there are a number of offspring that self-reproduce taking approximately half of the gestation time of the ancestral organism. It is difficult to be certain that we observed no organism that is viable, self-reproducing, and mutated from the ancestral organism and still preserves von Neumann style self-reproduction; however, in terms of gestation time, there seems to be no such organism. Most of the organisms with genotypic changes have almost half of the gestation time, implying they are not von Neumann style organisms, but highly likely to be self-copiers.

5 Conclusion and Future Work

In our very initial experiments, we did not observe displacement of our hand-designed seed von Neumann architecture self-reproducer by any viable self-reproducer preserving this architecture. More importantly, we certainly did not observe von Neumann style self-reproducers with an evolved genotype-phenotype mapping, corresponding to mutations affecting the "general purpose constructor" of von Neumann's architecture. What surprised us is the remarkably accessible mutational pathway from the ancestral self-reproducer to the self-copier. It had been expected, however, that such adaptation where the self-copiers are more likely to become dominant once they appear given the intrinsic advantage in this particular system.

Further analysis on this mechanism of the degeneration and classification of possible mutational pathways of our hand-designed organism are ongoing. Not only this, one of our next important steps is to eliminate the intrinsic advantage the self-copiers have. Another approach will be to hinder self-copiers to arise in

[5] Regarding the length of the von Neumann style organism, note that it needs to be longer than (at least twice as long as) a corresponding self-copier in the first place. This is because its phenome is crudely comparable to the entire memory image of a self-copier, and its genome is as long as the phenome. Moreover, compared to a (minimal) self-copier, the phenome of the von Neumann style organism needs extra instructions to conduct the decoding and to manage the more complex reproduction cycle and to accommodate the lookup table.

the first place, by re-designing the ancestral organism such that it uses `read` and `write` instructions for both decoding and copying throughout the reproduction, without using the `h-copy` instruction that our hand-designed ancestor adopts for the genome copying purpose. For this approach to observe less degeneration to self-copiers, it will be necessary to exclude `h-copy` from the instruction set so that no mutant organism will be able to employ the instruction. It is likely that such an ancestor will be even larger in size and will require even longer gestation time than the current organism. In any case, by offsetting the intrinsic advantage, we can avoid the degenerative displacement of the von Neumann style self-reproducers and refocus on our original intention to investigate and characterise the evolution of genotype-phenotype mapping.

Resources including the ancestral program and the Avida configuration file we used in the experiment can be downloaded at `http://alife.rince.ie/evosym/sab_2012_th.zip`.

Acknowledgments. This work has been supported by the European Complexity Network (Complexity-NET) through the Irish Research Council for Science and Technology (IRCSET) under the collaborative project EvoSym.

References

1. Adami, C.: Introduction to Artificial Life. Springer (1997)
2. McMullin, B.: John von Neumann and the evolutionary growth of complexity: Looking backward, looking forward. Artificial Life 6(4), 347–361 (2000), `http://dx.doi.org/10.1162/106454600300103674`
3. McMullin, B., Taylor, T., von Kamp, A.: Who needs genomes? (2001), `http://alife.rince.ie/bmcm-cbgi-2001/`
4. Ofria, C., Wilke, C.O.: Avida: A software platform for research in computational evolutionary biology. Artificial Life 10(2), 191–229 (2004), `http://dx.doi.org/10.1162/106454604773563612`
5. Pesavento, U.: An implementation of von neumann's self-reproducing machine. Artificial Life 2(4), 337–354 (1995)
6. Rasmussen, S., Knudsen, C., Feldberg, R., Hindsholm, M.: The coreworld: Emergence and evolution of cooperative structures in a computational chemistry. Physica D: Nonlinear Phenomena 42(1-3), 111–134 (1990), `http://www.sciencedirect.com/science/article/pii/0167278990900706`
7. Ray, T.: An evolutionary approach to synthetic biology: Zen and the art of creating life. Artificial Life 1(1/2), 179–209 (1994), `http://life.ou.edu/pubs/zen/`
8. Schrödinger, E.: What is Life? Cambridge University Press (1944)
9. von Neumann, J.: The general and logical theory of automata. In: Cerebral Mechanisms in Behavior, pp. 1–41 (1951)
10. von Neumann, J.: Theory of self-reproducing automata. University of Illinois Press (1966)

The Emergence of Pathological Constructors When Implementing the von Neumann Architecture for Self-reproduction in Tierra

Declan Baugh and Barry McMullin

The Rince Institute, Dublin City University, Ireland
declan.baugh2@mail.dcu.ie, barry.mcmullin@dcu.ie

Abstract. John von Neumann's architecture for *genetic reproduction* provides an explanation in principle for how arbitrarily complex machines can construct other ("offspring") machines of equal or even greater complexity. We designed a von Neumann style self-reproducing ancestor, within the framework of the Tierra platform, which implements a (mutable) genotype-phenotype mapping during reproduction. However, we have consistently observed a particular phenomenon where what we call *pathological constructors* quickly emerge, which ultimately lead to catastrophic ecosystem collapse. Pathological constructors are creatures which rapidly construct multiple short malfunctioning offspring within their lifetime. Pathological constructors are a hindrance to an ecosystem because their offspring, although sterile, still occupy both memory space and CPU time. When several pathological constructors coincide in time, their production rate can be so high that their non-functional offspring displace the entire population of functional self-reproducing creatures, resulting in ecosystem collapse. We investigate the origin of pathological constructors, and consider how a more mutational robust architecture which is less susceptible to the emergence of these creatures can be created.

Keywords: von Neumann, genetic reproduction, Tierra, artificial life, genotype-phenotype mapping, evolutionary growth of complexity, pathological constructors.

1 Introduction

As early as 1948, John von Neumann had formulated his theory of the evolutionary growth of machine complexity [3,2]. This theory provides a proof-of-principle demonstration that machines can directly, or indirectly, give rise to machines arbitrarily more complex than themselves. This machine architecture is comprised of two specific parts, the phenotype and the genotype.

The phenotype is the functional, active section of the machine, and the genotype is the passive section, committed to information storage. For genetic reproduction, under some arbitrary genotype-phenotype mapping, the genotype

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 240–248, 2012.

must contain an encoded description of the phenotype. Conversely, the phenotype must include the functionality to both decode the genotype and construct an offspring phenotype.

Previous work with evolutionary systems, where the agents are responsible for their own self reproduction, has been based exclusively on machine architectures which reproduce via *template-reproduction*, where there is no division of labour between genotype and phenotype. In this case, self reproduction is performed by self inspection, and no explicit, mutable genotype-phenotype mapping is implemented.

Within the platform of Tierra, we designed an ancestor that reproduces via genetic reproduction, true to the von Neumann architecture. More importantly, this design implemented a mutable genotype-phenotype mapping, where the arbitrary mapping between genotype and phenotype is subject to heritable mutations. We aim to explore if alternative, viable mutational pathways are introduced while implementing this architecture. However, during implementation within the Tierra platform, several unanticipated phenomena emerged, which are examined and documented here.

2 The von Neumann Machine Self-reproduction Architecture

The von Neumann architecture for machine self-reproduction includes a general constructive automaton A, which can construct an arbitrary machine, X, when supplied with a description of that machine, $\phi(X)$. A general copying automaton B can read machine descriptions, $\phi(X)$, and create copies. A control unit C is required to govern the automaton $(A + B)$. When supplied with a description $\phi(X)$, the control unit C first commands B to produce a copy of $\phi(X)$ itself. Upon completion of copying $\phi(X)$, C commands A to construct the described machine X. Finally C will attach the new instances of X and $\phi(X)$ and sever them from the parent automaton $(A + B + C)$, after which there exists the new entity, $X + \phi(X)$ [4]. Now consider the case where we let $X = (A + B + C)$. This system, $(A + B + C) + \phi(A + B + C)$ will proceed to construct an offspring automaton and attach it to the description of itself, $(A+B+C)+\phi(A+B+C)$. The parent and offspring are then identical, hence achieving self-reproduction.

More generally, we can conceive of an arbitrarily complex machine, $(A + B + C+D)+\phi(A+B+C+D)$, where D is some ancillary machinery (which describes all non-reproductive functionality of the automaton). By the previous reasoning, all such machines will also be self-reproductive (with the one restriction that D must not interfere with the operation of $A + B + C$).

In biological terminology, the *phenotype* is responsible for all active behaviour of the creature, including the general constructive automaton, the general copying automaton, the control mechanism, and the ancillary machinery: $(A + B + C + D)$. The *genotype* is a sequence of passive or inert numbers which are not executed, but which represent an encoded description of the phenotype: $\phi(A + B + C + D)$.

Should an accidental alteration occur within the description (genotype), which results in $\phi(A+B+C+D')$, the system $(A+B+C+D)+\phi(A+B+C+D')$ will produce $(A+B+C+D')+\phi(A+B+C+D')$, which is again self reproductive, i.e., this would be an example of a heritable mutation. We can specifically observe here that when a mutation occurs during copying of the genotype, it takes one further generation before this error is expressed within the phenotype [2]. This generation delay in genotype expression of heritable mutations turns out to be critical for the particular results we encountered.

3 The Tierra Platform

Tierra is an artificial life platform where a population of reproducing *creatures* compete with each other for both CPU time and memory space [5].

Generally, Tierra is used to experimentally explore processes of evolutionary and ecological dynamics. Processes such as the dynamics of host-parasite co-evolution and natural selection are open to investigation within the Tierra Platform [5]. However, in order to understand the limitations of Tierran creatures, we must first discuss the basic principles of the platform.

The platform contains a circular core memory, or *soup*, in which the creatures dwell. For certain purposes, Tierra creatures use a so-called *template addressing* mode rather than an absolute or relative numeric address. Templates are complementary patterns of ones and zeros which are positioned at locations within Tierran creatures, such as boundaries between different functional blocks of the creature. For example, a `jump 101` instruction, will cause the instruction pointer to jump to the nearest occurrence of the complimentary pattern `010`. This framework allows creatures to interact with each other, without the use of absolute addressing.

The Tierran instruction set consists of 32 assembler language instructions, each a single word with no arguments. Each memory location within the soup may contain one instruction. There are many *random perturbations* which introduce random errors to the system. *Cosmic rays* (alteration applied to the contents of a random memory location within the soup), *copy errors* (an error occurred when copying the contents of a memory location) and *segment deletions* (upon birth, a random creature segment is deleted) are examples of random perturbations which alter the contents of memory locations within a creature.

Each creature is assigned a CPU which includes four general purpose registers, Ax, Bx, Cx and Dx, an instruction pointer and a circular stack. A *slicer* allocates CPU time to each living creature within the soup and a *reaper* limits the population by reaping old and malfunctioning creatures.

Tierra is distributed in two different packages, Network Tierra and non-Network Tierra. Network Tierra is described as implementing a network-wide biodiversity reserve for digital organisms, allowing creatures to migrate between different nodes connected to the internet. The standard non-network version was used for our experiments.

4 Implementation of the von Neumann Architecture for Machine Self-reproduction within Tierra

Classically, the reproductive mechanism of Tierra creatures relies on self-copying[1], which involves the creature activating a reproduction mechanism, which incrementally copies the contents of each memory location of the parent creature to an available space in memory which will become the offspring. There is no distinction between phenotype and genotype for a self-copying creature, as the entire creature acts as both the template for replication, and the implementation of the reproduction cycle and all other functionality.

In order to achieve genetic reproduction of creatures within the Tierran platform, we devised a method of implementing a mutable genotype-phenotype mapping. The method we chose to implement, involved the inclusion of a *look-up table* as part of the general constructor A. The look-up table provides a method of translating the inert numbers within the memory locations of a parent genotype to functional instructions which can be executed as part of the offspring phenotype. For this particular genotype-phenotype mapping, we chose a 1:1 mapping, where a single number within the genotype is translated to a single instruction within the phenotype, therefore, the look-up table consists of 32 memory locations, each memory location containing a value which corresponds to an instruction within the Tierran instruction set. The look-up table of the seed ancestor will represent a random permutation of the all possible instructions.

We have implemented a seed creature based explicitly on the von Neumann architecture that reproduces via genetic reproduction, Fig. 1. Prior to reproduction, this seed creature must first allocate space for an offspring. During constructing an offspring phenotype, a CPU register, Ax, incrementally steps through each memory location within the parent genotype, and the number stored at this address is inspected. A second register, Bx which initially points to the start of the look-up table, is displaced by the number which was inspected by Ax. The number within the current Bx memory location (which lies within the look-up table), is now written to the offspring phenotype, where it will subsequently function as an instruction. This activity facilitates the mapping of numbers which are stored within the parent genotype, to instructions incorporated in the offspring phenotype. Furthermore, random perturbations within the look-up table facilitate the alteration of the genotype-phenotype mapping. This may have the effect of introducing new mutational pathways for the creature, which was not possible under the previously unaltered look-up table.

Upon construction of the offspring phenotype, the parent's genotype is incrementally copied to the offspring space and the connection between parent and offspring is severed. At this point, the parent loses write access to the offspring's memory block, and a new CPU is created and allocated to the offspring. While copying the genotype, should a genetic error occur which affects the encoded description of the look-up table (or otherwise modify the decoding process), then

[1] Analogous to RNA template replication.

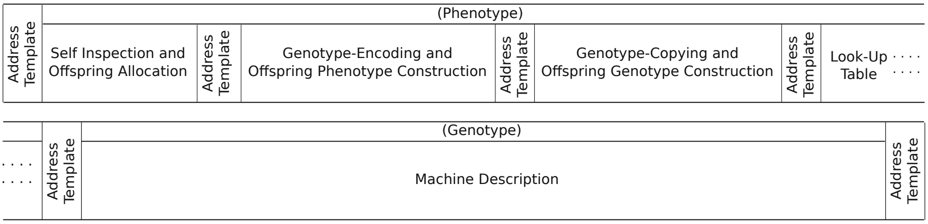


Fig. 1. Von Neumann style ancestor in Tierra

the creature's offspring will incorporate a mutated genotype-phenotype mapping. This is the particular phenomenon which we initially set out to investigate.

5 The Emergence of Pathological Constructors from Genetic Reproducers

When all random perturbations are disabled, our seed creature reproduces effectively and populates the memory to form a stable ecosystem of identical creatures. However, when all random perturbations are switched on, we immediately see a large emergence of pathological constructors, which saturate available CPU time and memory space. Under a series of simulations where each source of random perturbation was individually disabled, the disabling of the segment deletions showed an apparent prevention against the emergence of pathological constructors. When a large segment deletion occurs while copying the genotype from parent to offspring, the resultant creature will typically consist of a functional phenotype, assigned to a partial genotype. This creature continues to rapidly produce offspring, (due to the short genotype), but these offspring are non-functional as they consist of a corrupt phenotype, assigned to a corrupt genotype. When several such pathological constructors coincide in time, their production rate can be so high that their non-functional offspring displace the entire population of functional self reproducing creatures, i.e., ecosystem collapse.

For von Neumann style genetic reproducers, all random perturbations which corrupt the genotype will result in a constructor, which will create at least one functional or non functional offspring. A genotype which experiences a segment deletion will result in a pathological constructor, which can construct many non-functional offspring before it is killed by the reaper.

This analysis concludes that the mechanism which results in ecosystem collapse due to pathological constructors appears to depend critically on both the one generation delay from when a random perturbation occurs in a genotype and when it is expressed in the phenotype, and the inclusion of segment deletions. The combination of these factors results in a high level of ease in which segment deletions can lead to corrupt genotypes, while still leaving a functioning phenotype.

By contrast, in order for a pathological constructor to emerge from a self-copier, relatively much more specific random perturbations must occur upon very specific locations, which will alter, but not corrupt the reproductive functionality. This suggests that the probability of emerging pathological constructors within a population of genetic-reproducers, is much higher than that of a population of self-copiers.

6 The Heterogeneity of the Tierra Platform

A homogeneous universe is an important factor if we wish to observe an accurate representation of evolution within any system. However, the Tierra virtual universe is actually heterogeneous, and we aim to point out how this property can greatly effect the outcome of an evolutionary run, occasionally resulting in ecosystem collapse.

The Tierra circular memory system incorporates a template addressing mode. `Jump` instructions are specified by matching complementary sequences of ones and zeros. However, each memory location is still assigned an absolute underlying, numeric address, and `call` and `return` instructions use this absolute addressing mode to maneuver the instruction pointer throughout memory.

If a CPU executes the `call` instruction, the instruction pointer's current absolute address value is pushed to the stack. When `return` is executed, the instruction pointer jumps to the absolute address corresponding to the value stored in the stack. Upon birth, the stack is initialised with zeros, and a malfunctioning creature may execute `return` before pushing a value to the stack. This will redirect the CPU instruction pointer to the absolute address zero, and execute the residing creature. This absolute addressing mode, employed by `call` and `return`, introduces a level of heterogeneity to the system, where within an ecosystem which includes a number of malfunctioning creatures, address zero is the most profitable location for a creature to exist. In all instances where ecosystem collapse was observed due to the exploitation of pathological constructors, the creature type dominating the soup immediately prior to collapse, was a result of a pathological constructor located at address zero. Typically, in order for pathological constructors to result in ecosystem collapse, a relatively large population of pathological constructors must exist within the soup simultaneously, and one pathological constructor must be located at address zero. The cumulative effects of multiple malfunctioning creatures throughout the soup, relocating their CPU to address zero, may result in ecosystem collapse.

Instances have been observed, where a pathological constructor, located at address zero, creates offspring which immediately execute `return`. This causes the CPU of each offspring it creates to return to the parent, resulting in exponential growth of the number of CPUs captured by the pathological constructor, and furthermore, exponential growth in the number or pathological creatures within the soup. A single instance of such a pathological constructor, located at address zero, may have the effect of completely exploiting the system of resources, quickly resulting in catastrophic ecosystem collapse.

7 The Emergence of Pathological Constructors from Self-copiers

Previously, work with Tierra has been performed exclusively using populations of self-copiers, and the phenomenon of pathological constructors has not been reported explicitly under these circumstances. However, there is evidence that they have been observed in previously documented experiments under both the standard Tierra and Network Tierra distribution.

7.1 Pathological Constructors within Standard Tierra

Simulations experimenting with macro-evolution were performed by Tierra creator, Tom Ray [5], where the CPU time which is allocated to each creature increases as a function of creature length[2]. Upon birth, creatures of greater length are allocated a greater amount of CPU time, pressuring natural selection to favour creatures of longer lengths. Self-reproducing creatures with lengths of up to 10 times that of the standard Tierra ancestor are easily achieved. Under these circumstances the following has been recorded.

"Two communities have been observed to die after long periods. In one community, a chaotic period led to a situation where only a few replicating creatures were left in the soup, and these were producing sterile offspring. When these last replicating creatures died (presumably from an accumulation of mutations) the community was dead" [5].

The direct origin and function of these creatures which caused ecosystem collapse were not investigated and it was simply assumed that "Under these circumstances it is probably difficult for any genotype to breed true, and the genotypes may simply have ‘melted”’ [5], however, this description of ecosystem collapse explicitly correlates to our observations of ecosystem collapse following the emergence of pathological constructors.

7.2 Pathological Constructors within Network Tierra

The Network Tierra ancestor possesses the ability to migrate between different nodes connected to the network. The inclusion of this functionality results in a standard Network Tierra ancestor length of approximately 8 times that of the standard Tierra ancestor. There appears to have been an emergence of pathological constructors within this ecosystem, as an emergence of "small fast creatures that ignore the net" prevented the further evolution of the Network Ancestor [6]. Pathological creatures within this ecosystem would not possess the multi-cellular sensory functionality of the Network ancestor, and hence not possess the ability to migrate to neighbouring nodes. The outcome is that their accumulating non-functional offspring eventually displaces the entire population of functional creatures at each node, ultimately resulting in ecosystem collapse. This problem

[2] Creature length refers to the number of memory locations which a single creature occupies.

was solved by introducing an *Apocalypse event*. This event periodically reaps all creatures on a node. The annihilated node is then re-populated by migrating creatures, and hence allowing effective, self-reproducing creatures to thrive, while reducing the population of defective, non self-reproducing creatures.

While operating Network Tierra upon a single node, the Apocalypse event must be disabled, as there are no connecting nodes to re-populate the soup. Under these conditions, it was found that if selection is switched in favour of creatures of shorter lengths, which has the effect of decreasing the average creature length, the probability of ecosystem collapse is reduced, and longer evolutionary runs were possible.

Assuming that it was indeed pathological constructors which were observed in these experiments, it has become apparent that they have an increasingly damaging effect on ecosystems with increasingly larger ancestor lengths. As ancestor length and reproduction time increases, exploitation due to pathological constructors becomes increasingly damaging as each pathological constructor can construct a greater number of pathological offspring before it is reaped.

This observation suggests that within Tierra, there may be a threshold to the average creature length, above which, the consequences of pathological constructors will eventually result in catastrophic ecosystem collapse. The standard Tierra ancestor is relatively short, so pathological constructors would cause a negligible burden on system resources. However, for experiments where we see an increase in ancestor length and reproduction time, we see documented behaviour which suggests the emergence of pathological constructors.

7.3 Possible Source of Pathological Constructors from Self-copiers

There are many conceivable evolutionary pathways, in which pathological constructors may evolve from a population of self-copiers. For self-copiers, address templates located at either end of a creature allows the copy function to recognise the start and end of the sequence of instructions which is to be copied. For example, a self-copier may include a start and end address template, `111` and `110`. The copy mechanism will search for the nearest occurrence of `111`, and incrementally copy the contents of each memory location until it encounters `110`. A random perturbation may occur within the body of the self-copier, where some pre-existing and unrelated template is converted to `110`. The creature will now construct an incomplete offspring, which has a high probability of being sterile and toxic.

8 Conclusions and Future Work

The highlighted intricate properties of the von Neumann self reproducing automata, implemented in Tierra, suggest that this may not be mutationally robust architecture to support genetic reproduction. A combination of the effects of the segment deletions and the generation delay in expressing random perturbations, contribute to the abundant emergence of pathological constructors. The heterogeneous nature of the Tierra platform, along with the increase in ancestor length and reproduction time, in order to incorporate a genotype-phenotype mapping, amplify the destructive power of pathological constructors. This increases the ecosystem's susceptibility to catastrophic collapse.

It is worth noting that in the typical reproduction cycle of complex (multicellular) biological organisms, most of the decoding of the genotype takes place as development of the offspring, i.e., it is under the direction of the (embryonic) offspring phenotype rather than the parental phenotype [1]. If we incorporate this concept within the von Neumann architecture, where the offspring phenotype is decoded from the offspring genotype (as opposed to the parent genotype which is the case with von Neumann's architecture), then this design may not exhibit the one generation delay from when a random perturbation occurs in a genotype, and when it is expressed in the phenotype. A corrupt genotype will immediately be assigned a corrupt phenotype, and hence will not reproduce. It seems likely that such an architecture, implemented in Tierra, would be more evolutionary stable and much less vulnerable to emergence of pathological constructors.

Acknowledgments. This work has been supported by the European Complexity Network (Complexity-NET) through the Irish Research Council for Science and Technology (IRCSET) under the collaborative project EvoSym. The author would also like to thank the three anonymous reviewers for their valuable comments and suggestions to improve the clarity of this paper.

References

1. Buckley, W.: Computational ontogeny. Biological Theory 3(1), 3–6 (2008)
2. McMullin, B.: John von Neumann and the evolutionary growth of complexity: Looking backward, looking forward. Artificial Life 6(4), 347–361 (2000)
3. von Neumann, J.: The general and logical theory of automata. In: Cerebral Mechanisms in Behaviour, pp. 1–32 (1948)
4. von Neumann, J.: Theory of Self-Reproducing Automata (1966)
5. Ray, T.: An approach to the synthesis of life. Artificial Life II 10, 371–408 (1992)
6. Ray, T.: Technical report on the network tierra experiment (1997), http://life.ou.edu/tierra/netreport/

Appendix 1

Source code to reproduce results in this paper can be accessed at:
http://alife.rince.ie/evosym/sab-2012-db.zip

Automatic Synthesis of Controllers for Real Robots Based on Preprogrammed Behaviors

Miguel Duarte, Sancho Oliveira, and Anders Lyhne Christensen

Instituto de Telecomunicações &
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal
{miguel_duarte,sancho.oliveira,anders.christensen}@iscte.pt

Abstract. We present a novel methodology for the synthesis of behavioral control for real robotic hardware. In our approach, neural controllers decide when different preprogrammed behaviors should be active during task execution. We evaluate our approach in a double T-maze task carried out by an e-puck robot. We compare results obtained in our setup with results obtained in a traditional evolutionary robotics setup where the neural controller has direct control over the robot's actuators. The results show that the combination of preprogrammed and evolved control offers two key benefits over a traditional evolutionary robotics approach: (i) solutions are synthesized faster and achieve a higher performance, and (ii) solutions synthesized in simulation maintain their performance when transferred to real robotic hardware.

1 Introduction

Evolutionary robotics (ER) has been widely researched as a means to synthesize behavioral control for autonomous robots [6]. Artificial evolution has the potential to automate the controller design process without the need for manual and detailed specification of the desired behavior. Artificial neural networks are often used in ER because of their capacity to generalize and to tolerate noise [11] such as that introduced by imperfections in sensors and actuators. Two key issues have prevented evolution from being widely used as an engineering tool for automatic design of behavioral control: (i) bootstrapping the evolutionary process, and (ii) crossing the reality gap, that is, transferring behavioral control from simulation to reality without performance loss. The transition from simulation to real robotic hardware often results in reduced performance or even complete failure due to differences between simulation and the real world [5].

In this study, we propose a novel approach to overcome both bootstrapping issues and to successfully cross the reality gap. We combine artificial evolution with preprogrammed behaviors in order to ensure successful transfer of evolved control from simulation to real robots, while at the same time enable the synthesis of behaviors for relatively complex tasks. We experiment with giving evolution access to sets of simple preprogrammed behaviors, such as *follow wall*, *turn left*, and *turn right*, which can be switched on and off by a neural network

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 249–258, 2012.

that controls the robot. In this way, we marry the benefits of ER, namely automatic synthesis of behavioral control, with the benefits of preprogrammed behaviors that can be hand-optimized for specific sub-tasks and for the real robotic hardware.

We use the double T-maze navigation task [2] and the e-puck robotic platform [16] for our experiments, because both the task and the robotic platform have been widely studied in the past (see Sect. 2 and Sect. 4 for details). The double T-maze task is a delayed response task in which a robot receives stimuli in the form of light flashes and it must respond by making the correct turns in subsequently encountered T-junctions.

The contributions of this paper are as follows: we propose and study a new approach to the synthesis of behavioral control for autonomous robots. The approach is based on a combination of evolutionary computation and (simple) preprogrammed behaviors. We show that in our approach, solutions are synthesized faster and to a higher quality than solutions evolved using a traditional ER approach, and that the behaviors synthesized in simulation can be successfully transferred to real robotic hardware.

2 Background and Related Work

ER emerged as a field in the beginning of the 1990s [18]. Numerous studies followed which demonstrated robots with evolved control systems solving basic tasks in surprisingly simple and elegant ways. However, to date, only relatively simple tasks have been solved using ER such as obstacle avoidance, gait learning, phototaxis, foraging, and so on [17].

Soon after the research into ER began, two main challenges became clear, namely, (i) that the number of evaluations required meant that simulation had to be used extensively, and (ii) that it often is non-trivial to ensure successful transfer of behavior evolved in simulation to real robots. In [14], three complementary approaches to the evolution of control systems for real robots were proposed: "(a) an accurate model of a particular robot-environment dynamics can be built by sampling the real world through the sensors and the actuators of the robot; (b) the performance gap between the behaviors obtained in simulated and real environments may be significantly reduced by introducing a 'conservative' form of noise; (c) if a decrease in performance is observed when the system is transferred to a real environment, successful and robust results can be obtained by continuing the evolutionary process in the real environment for a few generations."

In 1997, Jakobi [10] advocated the use of minimal simulations to ensure the transferability of controllers evolved in simulation. A minimal simulator only implements the specific features of the real world that the experimenter deems necessary for a robot to complete its task. All other features would either not be implemented or be hidden by an envelope of noise. A number of other approaches to overcome the reality gap include performing evolution directly on the target hardware instead of in simulation [7], online adaptation through neural plasticity [8], co-evolution of simulators and controllers [3], and promotion of transferable controllers though multi-objective optimization [12]. Conducting evolution on real robotic hardware is, however, tedious and is not always feasible, especially in complex tasks and/or when many trials are needed to reliably estimate the fitness of an individual. In this paper, we propose a novel approach to the engineering of control systems in which we combine simple preprogrammed behaviors with artificially evolved neural networks. Successful transfer of the control synthesized in simulation to real hardware is guaranteed by specifying a set of behavior primitives that have been tested on real robots.

The topic of behavior modularity in the field of robotics has been studied before. Rodney Brooks proposed the subsumption architecture in the 1980's [4]. Brook's approach is characterized by the decomposition of complicated intelligent behavior into many simple behavior modules, which are in turn organized into layers. The layers and ordered in terms of behavioral complexity: the behaviors on the bottom layers are simpler and have a higher priority than the ones on the higher layers. The more complex behaviors can only execute if none of the layers beneath take control of the robot. The concept of behavioral decomposition has also been studied in the field of ER. Moioli et al. [15] used a homeostatic-inspired GasNet with two different behavior controllers (obstacle avoidance and phototaxis) that were inhibited or activated by the production and secretion of virtual hormones. In [13], logic decision trees and task decomposition have been combined with ER techniques. By evolving different sub-behaviors such as “circle box”, “push box” and “explore”, the authors synthesized a robotic controller that was able to push a box toward a light source.

We use a double T-maze task [2] for our experiments. An example of a double T-maze can be seen in Fig. 1. The T-maze contains three T-junctions. At the start of each experiment, the robot is placed in the “Start zone” and must navigate towards the first junction. On its way, it passes two rows of lights. In each row, one of the lights is activated. The activated light flashes as the robot passes by. The activated light in the first row informs the robot on to which side it must turn in the first T-junction it encounters, while the activated light in the second row informs the robot to which side it must turn in the second T-junction that it encounters. If L1 and R2 are activated, for instance, the robot must make a left turn in the first T-junction and a right turn in the second T-junction so that it reaches exit LR (see Fig. 1), and so on.

Variations of the T-maze task have been used extensively in studies of learning and motivation in animals, neuroscience, and robotics (see [19,20,10] for examples). In robotics, T-mazes have been used to study different neural network models such as diffusing gas networks [9], the online learning capability of continuous time recurrent neural networks [2], and the evolution of transferable controllers [10,12]. However, in the studies where controllers were tested on real hardware, only a single T-maze was used and the mazes were relatively small with respect to the robot. In this study, we synthesize controllers in simulation that enable a real e-puck to solve a relatively large, double T-maze (see Sect. 4).

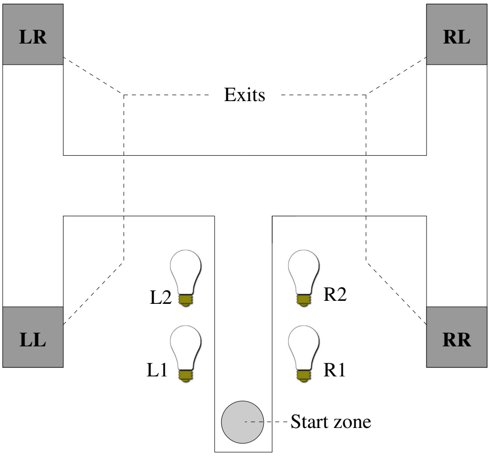


Fig. 1. A double T-maze. A robot is placed in the start zone and must navigate to one of the four exits depending on which lights flash as the robot passes by.

3 Methodology

The main purpose of the proposed methodology is to allow for the synthesis of behavioral control for real robotic hardware. Whereas Jakobi [10] advocated the use of minimal simulations to limit the set of environmental features that a controller can rely on, we suggest limiting the set of actions that a robot can perform. We define a set of behavior primitives based on the robot, its task, and the environment. The behavior primitives are specified in such a way that they have comparable results and performance when executed in simulation and on the real robotic hardware. Conservative noise is added in simulation to promote robustness to the difference that unavoidably exists between the two environments. In this study, the behavior primitives are simple preprogrammed behaviors (*follow wall*, *turn left*, and *turn right*), but they could be more elaborate and even previously synthesized behaviors.

As in a traditional ER setup, the robot's sensory inputs are fed to an artificial neural network. However, instead of controlling the robot's actuators directly, the outputs of the neural network are connected to a "behavior selector" (see Fig. 2). In this study, each output neuron of the neural network corresponds to a single behavior primitive and the primitive which has the highest activation value is executed in a winner-takes-all approach. Some primitives can take more than one control cycle to complete, such as turning 90° left or right. The behavior selector does not execute any other primitive before the previously selected primitive has completed. Alternative behavior selectors could be implemented including selectors that allow for multiple primitives to be executed in parallel. Parallel execution of behavioral primitives could, for instance, allow a robot to communicate at the same time as it executes motor behaviors.

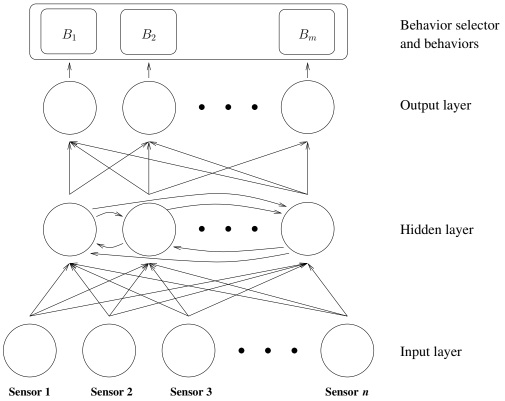


Fig. 2. Example of the controller structure: A continuous-time recurrent neural network [1] receives readings from the robot's sensors. The activation of the neurons in the output layer are fed to the *behavior selector*, which executes one of the behavior primitives based on the activations.

4 Experimental Setup

We used the e-puck [16] for our experiments. The e-puck is a small circular (diameter of 75 mm) mobile robotic platform designed for educational use. For offline synthesis of behavioral control, we use JBotEvolver, an open source, multirobot simulator and neuroevolution framework. The simulator is written in Java and implements 2D differential drive kinematics. Evaluations of controllers can be distributed across multiple computers and different evolutionary runs can be conducted in parallel. The simulator can be downloaded from: `http://sourceforge.net/projects/jbotevolver`.

We built a double T-maze [2] with a size of 2 m × 2 m (see Fig. 3). In the real maze, the states of the flashing lights are controlled by a Lego Mindstorms NXT brick using 4 ultrasonic sensors and 2 motors (see Fig. 3).

We used four of the e-puck's eight infrared proximity sensors: the two front sensors and the two lateral sensors. We collected sensor samples from two different robots. Each sensor was sampled for 10 samples collected are available at `http://home/iscte-iul.pt/~alcen/sab2012`). At the beginning of every simulation trial, we randomly mapped one of the eight collected sets of samples to each of the robot's proximity sensors. Distance-dependent noise was added to the sensor readings in simulation corresponding to the amount of noise measured during the sampling of the sensors. The light sensor was binary: when a light sensor reading deviated sufficiently from an initial set of readings obtained in ambient light conditions, the robot perceived a flash. In simulation, we added Gaussian noise (5%) to the wheel speed.

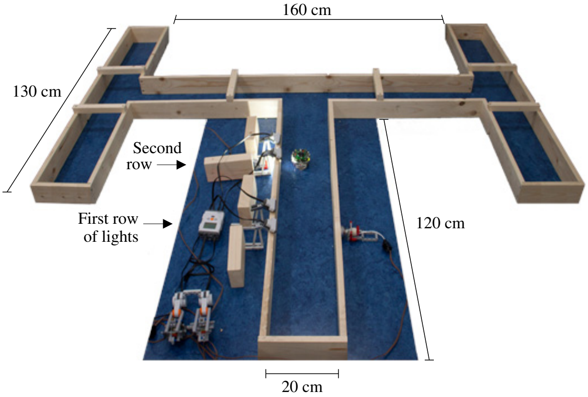


Fig. 3. T-maze with a total size of 2 m × 2 m. The two rows with the lights are located in the central corridor. The first row is at 45 cm and the second row at 83 cm from the start of the maze.

4.1 Controller

The neural controller used in this study is a continuous-time recurrent neural network [1] (see Fig. 2). The input layer of the ANN is composed of 6 neurons: one for each of the four infrared proximity sensors, and one for each of the two light sensors. The readings from the proximity sensors are mapped to distances and then converted to input neuron activations (interval $[0, 1]$). The mapping of readings to distances is done based on the average values for the eight sets of real robot samples. When a light flash is detected, the corresponding input neuron is assigned an activation value of 1.0 for a duration of 15 control cycles.

The hidden layer of the ANN is composed of 10 fully connected neurons. The output layer of the neural network is composed of 3 neurons, one for each of the 3 preprogrammed behaviors available to the network: *turn left*, *turn right*, and *follow wall*. The behavior selector compares the activations of the three output neurons and executes the behavior that corresponds to the neuron with the highest activation. The two *turn* behaviors turn the robot 90°, which takes on average 40 control cycles. During that time, the behavior selector ignores the values of the output neurons in order to allow the turn to complete before executing a new behavior. The *follow wall* behavior moves the robot forward along the closest perceived wall. We limited the speed of the robot to 10 cm/s.

4.2 Evolutionary Algorithm

We train controllers with a simple generational evolutionary algorithm. Each generation is composed of 100 genomes, and each genome corresponds to an ANN with the topology described above. The fitness of a genome is sampled 40 times and the average fitness is computed. Each sample lasts a maximum of 500 control cycles (equivalent to 50 seconds of simulated time). The starting position of the robot is varied up to 5 cm to the left or to the right, and up to 10 cm forward or backward.

The top 5 genomes are selected to populate the next generation using an elitist approach. An offspring is created by applying a Gaussian noise to each gene with a probability of 10%. The 95 mutated offspring and the original 5 genomes constitute the next generation.

The robots are evaluated based on three different outcomes: (i) if they successfully navigate to the correct exit, the assigned fitness is f_1, (ii) if they choose an incorrect exit or colide into a wall, the assigned fitness is f_2, and (iii) if time expires, the assigned fitness is 0. Fitness f_1 and f_2 are defined by:

$$f_1 = 1 + \frac{maxCycles - spentCycles}{maxCycles} \qquad f_2 = \frac{totalDistToExit - distToExit}{3 \cdot totalDistToExit}$$

We ran an additional set of experiments in a traditional ER setup in which the outputs of the neural network controlled the robot's wheels directly. Aside from the difference in the interpretation of the networks output, the experimental setup (network topology, inputs, simulation conditions, and evolutionary parameters) were the same as those described above.

5 Results

We synthesized robotic controllers for a double T-maze task in two different experimental setups: in experimental setup A (Synthesis with Preprogrammed Behaviors), the output of the neural networks activates one of the three possible preprogrammed behaviors, while in experimental setup B (Traditional ER) the output neurons control the wheels of the robot directly.

We conducted 30 evolutionary runs in each of the two experimental setups. Each run lasted 1000 generations. We conducted a post-evaluation of the evolved controllers in which the fitness of every controller was sampled 100 times for each of the 4 possible light configurations. The results are summarized in Figure 4. In experimental setup A, the evolved controllers had an average solve rate of 87%. A solve rate of over 95% was observed in 12 of the 30 controllers. Some of the trials evolved controllers with good solutions as early as the 150th generation.

The solutions produced in different evolutionary runs were similar. The robots learned how to navigate the T-maze correctly, but some of the controllers were not able to use the information from the light flashes to make the correct decisions at the T-junctions, which caused them to navigate to the wrong maze exit.

In experimental setup B, the evolved controllers had an average solve rate of only 42%. The best controller had a solve rate of 88%, and only 12 other controllers were able to correctly solve the T-maze in more than 50% of the samples.

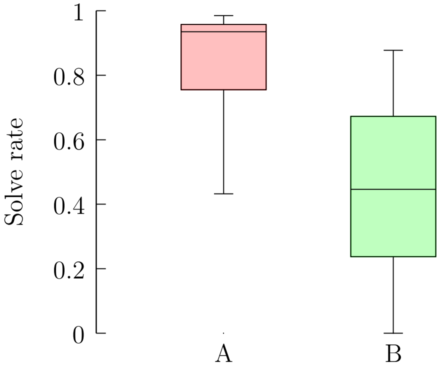


Fig. 4. Summarized results from simulation for setup A and setup B

5.1 Transfer to Real Robotic Hardware

After the evolutionary process had finished, the 5 highest performing controllers synthesized in setup A, and the 5 highest performing controllers synthesized in setup B were tested on a real e-puck. Each controller was tested 16 times, 4 for each light configuration. The results are listed in Table 1.

All of the 5 controllers synthesized based on preprogrammed behaviors were able to successfully cross the reality gap and solve the real maze consistently. The controllers synthesized in run A22 and A25 managed to solve all 16 samples. The remaining 3 controllers sometimes navigated to an incorrect maze exit: A4 and A13 failed 1 out of 16 samples, and A9 failed 2 out of 16 samples.

The controllers from setup B did not display as high a performance as those synthesized in setup A. Partly, this was because their in simulation performance was not as high as the one in experimental setup A. 4 of the 5 controllers transferred correctly, achieving even comparable performance in reality, but the controller from trial B19 only solved 11 out of 16 samples in the real robot experiments. Videos of the experiments can be seen at `http://home/iscte-iul.pt/~alcen/sab2012` .

Table 1. Summary of the real robot results for the five highest performing evolutionary runs of experimental setup A and experimental setup B

Evolutionary run	**A22**	**A9**	**A25**	**A13**	**A4**	**Average**
Solve rate (Simulation)	99%	98%	98%	97%	97%	98%
Solve rate (Real robot)	100%	88%	100%	94%	94%	95%

Evolutionary run	**B11**	**B13**	**B19**	**B16**	**B9**	**Average**
Solve rate (Simulation)	88%	86%	79%	70%	70%	79%
Solve rate (Real robot)	100%	100%	56%	75%	75%	81%

6 Conclusions

In this study, we demonstrated how controllers can be synthesized by combining artificial evolution with simple preprogrammed behaviors. Our results show that the proposed approach found good solutions in fewer generations and achieved higher final fitness scores than in a traditional ER setup in which the neural controller has direct control over the robot's actuators. On real robotic hardware, the performance of the controllers synthesized with our approach was similar to their performance in simulation.

We gave neural controllers three simple preprogrammed behaviors: *follow wall*, *turn left*, and *turn right*. If we had used a different set of preprogrammed behaviors, we would potentially have seen different solutions. The solution space is defined by the set of behaviors to which a neural controller has access. This solution space is smaller than the solution space in a traditional ER setup in which the neural controller has direct control over the robot's actuators. The restricted solution space may exclude the optimal solution(s) for a given robot and task. In our study, the controllers that had direct access to the actuators were able to cut corners and continued to move forward while turning in a T-junction. The controllers synthesized in our approach were limited to the *turn left* and *turn right* behaviors that cause the robot to turn 90° on the spot. Consequently, controllers that had direct access to the robot's actuators were sometimes able to complete the task faster than the controllers that were restricted to a predefined set of preprogrammed behaviors.

While the use of a finite set of predefined behaviors may forestall the synthesis of the theoretically optimal controllers, it opens a number of interesting possibilities. Behaviors can be hand-optimized for a particular robot and for particular sub-tasks. For some sub-tasks, it may be relatively easy to rely on artificial evolution to find a good solution, while for others, such as those that are difficult to simulate with sufficient accuracy, may be more easily solved by manually programming a behavior. Moreover, the predefined behaviors need not be limited to simple behavior primitives, but could be controllers that have previously been synthesized by combining other behaviors and so on. In this way, our approach potentially allows for an incremental and a hierarchical synthesis of behavioral control for real robots.

Acknowledgements. This work was partly supported by Fundation of Science and Technology (FCT) under the grants PTDC/EEACRO/104658/2008, SFRH/BD/76438/2011, and PEst-OE/EEI/LA0008/2011.

References

1. Beer, R.D., Gallagher, J.C.: Evolving dynamical neural networks for adaptive behavior. Adaptive Behavior 1, 91–122 (1992)
2. Blynel, J., Floreano, D.: Exploring the T-Maze: Evolving Learning-Like Robot Behaviors Using CTRNNs. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J.R., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) EvoWorkshops 2003. LNCS, vol. 2611, pp. 593–604. Springer, Heidelberg (2003)

3. Bongard, J., Lipson, H.: Once more unto the breach: Co-evolving a robot and its simulator. In: Proceedings of 9th International Conference on the Simulation and Synthesis of Living Systems, pp. 57–62. MIT Press, Cambridge (2004)
4. Brooks, R.A.: A robust layered control system for a mobile robot. IEEE Journal on Robotics and Automation 2(1), 14–23 (1986)
5. Brooks, R.A.: Artificial life and real robots. In: Proceedings of the First European Conference on Artificial Life, pp. 3–10. MIT Press/Bradford Books, Cambridge, MA (1992)
6. Floreano, D., Keller, L.: Evolution of adaptive behaviour in robots by means of Darwinian selection. PLoS Biology 8, 1–8 (2010)
7. Floreano, D., Mondada, F.: Evolutionary neurocontrollers for autonomous mobile robots. Neural Networks 11(7-8), 1461–1478 (1998)
8. Floreano, D., Urzelai, J.: Evolution of plastic control networks. Autonomous Robots 11(3), 311–317 (2001)
9. Husbands, P.: Evolving Robot Behaviours with Diffusing Gas Networks. In: Husbands, P., Meyer, J.-A. (eds.) EvoROB/EvoRobot 1998. LNCS, vol. 1468, pp. 71–86. Springer, Heidelberg (1998)
10. Jakobi, N., Jakobi, N.: Evolutionary robotics and the radical envelope-of-noise hypothesis. Adaptive Behavior 6, 325–368 (1997)
11. Kam-Chuen, J., Giles, C., Horne, B.: An analysis of noise in recurrent neural networks: convergence and generalization. IEEE Transactions on Neural Networks 7, 1424–1438 (1996)
12. Koos, S., Mouret, J.-B., Doncieux, S.: The transferability approach: Crossing the reality gap in evolutionary robotics. IEEE Transactions on Evolutionary Computation (in press, 2012)
13. Lee, W.-P.: Evolving complex robot behaviors. Inf. Sci. 121(1-2), 1–25 (1999)
14. Miglino, O., Lund, H.H., Nolfi, S.: Evolving mobile robots in simulated and real environments. Artificial Life 2, 417–434 (1996)
15. Moioli, R.C., Vargas, P.A., Zuben, F.J.V., Husbands, P.: Towards the evolution of an artificial homeostatic system. In: IEEE Congress on Evolutionary Computation, pp. 4023–4030 (2008)
16. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Christophe Zufferey, J., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering. In: Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, pp. 59–65 (2009)
17. Nelson, A.L., Barlow, G.J., Doitsidis, L.: Fitness functions in evolutionary robotics: A survey and analysis. Robotics and Autonomous Systems 57(4), 345–370 (2009)
18. Nolfi, S., Floreano, D., Miglino, O., Mondada, F.: How to evolve autonomous robots: Different approaches in evolutionary robotics. In: Proceedings of the 4th International Workshop on Artificial Life, pp. 190–197. MIT Press, Cambridge (1994)
19. Tolman, E.C., Honzik, C.H.: Introduction and removal of reward, and maze performance in rats. University of California Publications in Psychology 4, 257–275 (1930)
20. Torta, A.B.L., Kramer, M.A., Thorn, C., Gibson, D.J., Kubota, Y., Graybiel, A.M., Kopell, N.J.: Dynamic cross-frequency couplings of local field potential oscillations in rat striatum and hippocampus during performance of a t-maze task. Proceedings of the National Academy of Sciences 105(51), 20517–20522 (2008)

Self-organization of Visual Sensor Topologies Based on Spatiotemporal Cross-Correlation

Jonas Ruesch, Ricardo Ferreira, and Alexandre Bernardino

Instituto Superior Técnico, 1049-001 Lisbon, Portugal
{jruesch,ricardo,alex}@isr.ist.utl.pt

Abstract. In living organisms, the morphology of sensory organs and the behavior of a sensor's host are strongly tied together. For visual organs, this interrelationship is heavily influenced by the spatial topology of the sensor and how it is moved with respect to an organism's environment. Here we present a computational approach to the organization of spatial layouts of visual sensors according to given sensor-environment interaction patterns. We propose that prediction and spatiotemporal correlation are key principles for the development of visual sensors well-adapted to an agent's interaction with its environment. This proposition is first motivated by studying the interdependency of morphology and behavior of a number of visual systems in nature. Subsequently, we encode the characteristics observed in living organisms by formulating an optimization problem which maximizes the average spatiotemporal correlation between actual and predicted stimuli. We demonstrate that the proposed formulation leads to spatial self-organization of visual receptive fields, and leads to different sensor topologies according to different sensor displacement patterns. The obtained results demonstrate the explanatory power of our approach with respect to i) the development of spatially coherent light receptive fields on a visual sensor surface, and ii) the particular topological organization of receptive fields depending on sensorimotor activity.

Keywords: visual sensor topology, self-organization, sensorimotor coupling.

1 Introduction

By simply observing the active behavior and visual organs of different animal species, important hints can be obtained on how an organism constructs visual percepts. Primates use a sophisticated oculomotor system to sequentially move and stabilize their eyes with relation to different target locations [1]. Most airborne insects on the other hand, have their eyes rigidly attached to their body or head; instead of focusing on particular target locations, these animals analyze how the projection of the environment translates on their sensors during flight [2]. In general, three interrelated aspects contribute to how biological vision systems record raw visual stimuli: i) the characteristics of the environment in which an animal is living, ii) the way a sensor is moved with respect to the environment, and iii) the physical and morphological design of a visual organ.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 259–268, 2012.

In this work, we consider i) to be a general environment and we investigate a possible principle how ii) influences iii).

A closer look at the morphology of biological visual sensors reveals profound differences between different organisms. While all visual organs found in nature record visual stimuli through a number of light sensitive receptors – and hence always record a spatially discretized stimulus – the spatial density distribution of visual receptors varies greatly between species. Studies measuring the distribution of retinal ganglion cells in camera-type eyes, or the ommatidia distribution in compound eyes, suggest that receptor distributions are directly tied to an animal's behavior and environment. Most prominently, primates and other mammalians with binocular vision feature a fovea – a small, high-resolution area in the center of their retina – and a radially close to logarithmically decreasing receptor density. In [3], it is pointed out that such a log-polar-like receptor distribution corresponds to a mapping function which transforms image rotations and dilations (zoom) into simple coordinate shifts in the log-polar coordinate system. Thus, if an eye featuring such a receptor distribution is focusing on an object and that object is rotated or scaled, the projected image is merely shifted along the log-polar coordinate axes. It was argued that this property results in an advantage for the human visual cortex, as it could achieve image invariance for these transformations at a low computational cost by simply shifting the image. Similar to ganglion cell distributions found in camera-type eyes, the density of ommatidia in arthropods varies significantly over the spatial extension of their compound eye. Many flying insects for example have about a two times higher spatial resolution in the frontal visual eye field than compared to the lateral part [4]. A possible advantage of such a distribution is discussed in [5]. There, it is demonstrated that high density of light recording receptors in frontal and caudal regions, and decreasing density in lateral regions, leads to a uniform translation of projected stimuli on the eye during straight locomotion and can facilitate visual distance estimation.

Motivated by observations related to the relationship of behavior and morphology in natural visual systems, we explore in this paper the hypothesis that visual organs develop such as to simplify neural circuitry for predicting on average experienced stimulus flow patterns. We first propose a criterion based on spatiotemporal cross-correlation to evaluate such a receptor-to-receptor flow property, and we subsequently use the introduced criterion as a cost function to synthesize visual sensor topologies on a given sensor surface using a given set of stimulus transformations. The obtained results suggest that the introduced criterion is able to capture important properties of the relationship between the spatial layout of a visual sensor and the way the sensor is moved with respect to the environment.

1.1 Related Work

In an inventive work [6], Clippingdale and Wilson present a numerical experiment motivated by the spatial organization of visual sensors in nature. Using an abstract setup where visual receptors are represented as a set of points on a disk, an appealing principle is motivated on how to capture the relationship between form and behavior. In line with our observations for natural visual systems, the basic idea is a rule capable of generating sensor layouts which simplify stimulus transformation patterns under a given behavior: assuming the given points are transformed by a set of sensor displacement actions, the relative position of each point is updated such as to reduce the overall motion-prediction error between points. Interestingly, this update rule leads to foveal point distributions when considering stimulus transformations plausible e.g. for the mammalian visual system. Furthermore, using different action probability distributions for horizontal and vertical translations, elliptic (visual streak-like) point layouts can be obtained. For an illustration see Figure 10 in [6]. Formally, Clippingdale and Wilson proved the following: a set of points randomly distributed on a disk converges to a stable configuration given: i) points are conjointly transformed by rotations, dilations and translations which are applied according to a given probability distribution; and ii) after a transformation action is applied, each point is moved towards transformed points which are lying closest to the point under consideration. It was shown, the final point distribution is the configuration where each point has on average the smallest possible distance to the next closest transformed point under the given action probability distribution. This approach is based on two important assumptions: visual receptors have no spatial extension (i.e. are points), and the error between original and transformed receptors can be measured as an Euclidean distance between spatial locations of receptors. The first assumption is clearly an abstraction of a real visual sensor. The second assumption can be further divided into two requirements: the spatial layout of the visual sensor is known to the algorithm, and the prediction error of visual stimuli is directly related to spatial distance. While it is arguable if an agent can have complete knowledge of the spatial layout of its sensor, the assumption that the prediction error is equivalent to spatial distance is unlikely to hold for spatially extended visual receptors of different sizes.

Related to the question of how the distance measure underlying the optimization proposed by Clippingdale and Wilson could be translated to real visual sensors, the authors of this paper investigated in previous work how the interrelationship between form and behavior could be quantified for sensors with spatially extended receptors and unknown topologies [7]. Based on the complexity of the model required to predict stimulus changes, a measure was introduced which evaluates the coupling between sensor displacements and sensor topologies. It has been shown that a given sensor topology implicitly defines actions for which future sensory stimuli can be predicted with less parameters. In this work we use a similar strategy to optimize the coupling between a sensor's topology and executed motor actions.

1.2 Contribution

We develop a computational method for synthesizing visual sensor topologies according to on average experienced stimulus transformations. To establish a relation between a sensor's spatial layout and experienced stimulus transformations, we adopt the basic principle proposed in [6]. Though, instead of considering point-like sensor elements, we simulate a realistic visual sensor which records stimuli through receptors where each receptor integrates luminance according to a receptive field. Different from [6], we impose that the algorithm has no access to information about the topological layout of the sensor being organized. This means, the organization of the sensor layout has to be achieved solely by observing the activation of an orderless array of visual receptors. Hence, the implementation of a rule similar to the one proposed in [6] becomes considerably more challenging. In particular, the Euclidean distance measure between transformed and original points has to be replaced with a measure related to how activation is transported between visual receptors when the recorded stimulus changes. We will address this issue by introducing a criterion based on spatiotemporal cross-correlation of receptor activation. This criterion allows us then to implement an optimization which organizes the layout of visual receptors depending on sensorimotor activity. At the same time, we also required the algorithm to find a suitable shape for the receptive fields (RFs) of the spatially extended receptors. We show that spatially coherent RFs can evolve driven only by the low spatial frequency of natural images. By rewarding spatial correlation within RFs, smoothly overlapping clusters organize on the sensor surface without any further constraint on the spatial shape of a receptor's integration area. In practice, receptors can be initialized with a randomly chosen luminance integration function and eventually develop into compact receptive fields.

The following steps summarize the approach followed in this paper:

1. A system with a given sensor surface, a given motor space and a predefined number of visual receptive fields is considered.
2. Each visual receptive field is described as a discretized, randomly initialized function according to which visual input is integrated from the sensor surface.
3. By maximizing spatial correlation of visual stimuli recorded through receptive fields, the development of spatially coherent visual receptors is achieved.
4. By extending spatial correlation to spatiotemporal correlation between visual stimuli of transformed and original receptors, sensor topologies dependent on the agent's motor activity are developed.

2 Approach

An artificial agent with a given sensor surface $\mathcal{I} \subset \mathbb{R}^2$ and a given number of motion degrees of freedom is considered. The sensor surface records a projection of the environment given as a function $i_s : \mathcal{I} \rightarrow \mathbb{R}$ defining a luminance value for each point on the surface when the agent is in state s. For numerical purposes, i is sampled at N spatial locations $\mathbf{x}_n$ as a discrete grayscale image $\mathbf{i} = [i(\mathbf{x}_1)\ i(\mathbf{x}_2)\ \ldots\ i(\mathbf{x}_N)]^\top$. The topology of the visual sensor is composed of M visual receptors, where M is a parameter of the proposed method and is much smaller than N. Each visual receptor m integrates a visual stimulus through a receptive field (RF). The RF is described as a vector of weights $\mathbf{r}_m$ defining how much each entry in $\mathbf{i}$ contributes to receptor m. Note that $\mathbf{r}_m$ is allowed to encode any receptive field function and no spatial coherence is assumed. By assembling weight vectors $\mathbf{r}_m$ for all M visual receptors as the rows of a matrix $\mathbf{R}$, a stimulus recorded by the agent in state s can be written as $\mathbf{R}\mathbf{i}_s$.

After observing state s, the agent can choose to take an action a from a discrete set of actions $\mathcal{A}$ representative of the agent's behavior. This action induces a change in the observed grayscale image from $\mathbf{i}_s^-$ to $\mathbf{i}_s^+$; here we assume that this change is predictable.[1] As the agent explores its environment, we collect before and after images for each particular action a in the matrices $(\mathbf{I}_a^-, \mathbf{I}_a^+)$, where samples are arranged in columns. For the whole set of actions $\mathcal{A}$, these matrices are collected in a dataset $\mathcal{D} = \{(\mathbf{I}_a^-, \mathbf{I}_a^+), a \in \mathcal{A}\}$.

With the introduced terminology, we now proceed to develop an optimization problem which evolves the sensor topology $\mathbf{R}$ such that the previously described properties are induced: i) spatially coherent receptive fields are formed, and ii) the topological layout of the sensor reflects stimulus translations induced by the behavior of the host. We propose to find an optimal $\mathbf{R}$ as the solution to an optimization problem:

$$\mathbf{R}^* = \underset{\mathbf{R}\in\mathcal{R}}{\operatorname{argmax}}[F(\mathcal{D}, \mathbf{R}) - G(\mathbf{R})], \tag{1}$$

where F denotes a function evaluating the spatiotemporal cross-correlation of a set of samples $(\mathbf{I}_a^-, \mathbf{I}_a^+)$, and G represents a cost for growing receptive fields. The constraint set $\mathcal{R}$ is chosen as $\mathcal{R} = \{\mathbf{R} : \mathbf{R} \geq \mathbf{0},\ \mathbf{R}^\top \mathbf{1} = \mathbf{1}\}$, such as to guarantee that the visual receptive fields occupy the whole sensor surface and luminance cannot be subtracted. In the remainder of this section, we unroll the complete definition of this optimization problem by developing F and G.

Consider first an immobile agent with a single null action leading to a reduced data set $\bar{\mathcal{D}} = \{\mathbf{I}^-\}$ of stimuli recorded in different states s. In this case, we consider a reasonable sensor topology $\mathbf{R}$ to be one which leads to high correlation within a batch of recorded stimuli $\mathbf{R}\mathbf{i}_s$. The rational behind this is that bigger differences between simultaneous receptive field activations indicate that the agent is able to pick-up more information from the images $\mathbf{i}_s$, in an information theoretic sense. Furthermore, correlation must be normalized with respect to the size of a receptive field such that different sized receptive fields are comparable. Implementing these two requests, we propose a first version of F for an immobile agent to be a size normalized correlation between stimuli $\mathbf{i}_s$ like:

$$\bar{F}(\bar{\mathcal{D}}, \mathbf{R}) = \sum_{s=1}^{S} \left(\hat{\mathbf{R}}\mathbf{i}_s^-\right)^\top \left(\hat{\mathbf{R}}\mathbf{i}_s^-\right), \qquad \hat{\mathbf{R}} = \frac{\mathbf{R}}{\sqrt{\mathbf{R}\mathbf{1}\mathbf{1}^\top}}, \tag{2}$$

where in $\hat{\mathbf{R}}$ the division and square root operators are applied element wise.

In a second step, an active agent and a full data set $\mathcal{D} = \{(\mathbf{I}_a^-, \mathbf{I}_a^+)\}$ is considered. To establish a temporal relationship between receptive fields, we now

[1] See also [8], Appendix A for the constraints posed on such actions and how this situation relates to a physical agent acting in a 3-dimensional world.

adapt $\bar{F}$ to compute correlation between pre- and post-action stimuli. We remind the reader that it is a priori unknown how to temporally relate receptive fields and how stimuli change under an action a. This is naturally solved by considering a prediction operator which describes a mapping of receptors for a given action, allowing for comparison of stimuli at different points in time. In [9] Crapse and Sommer provide an excellent review of the ubiquity of stimulus prediction in living organisms and [8] gives an argument for the use of linear prediction. Thus, assuming that for an action a we can predict a visual stimulus as $\mathbf{R}\mathbf{i}_a^+ = \mathbf{P}^a_{(\mathbf{R})}\mathbf{R}\mathbf{i}_a^-$ we revise $\bar{F}$ like

$$F\left(\mathcal{D}, \mathbf{R}\right) = \sum_{a \in \mathcal{A}} \sum_{s=1}^{S} \left(\hat{\mathbf{R}}\mathbf{i}^+_{s,a}\right)^\top \left(\mathbf{P}^a_{(\mathbf{R})}\hat{\mathbf{R}}\mathbf{i}^-_{s,a}\right), \qquad \hat{\mathbf{R}} = \frac{\mathbf{R}}{\sqrt{\mathbf{R}\mathbf{1}\mathbf{1}^\top}}, \tag{3}$$

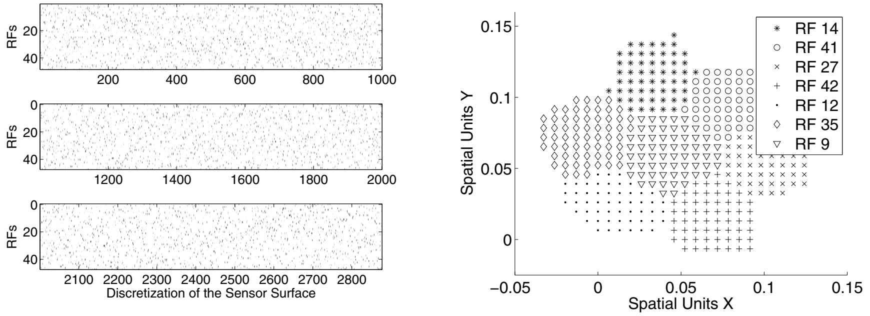
where a prediction operator $\mathbf{P}^a_{(\mathbf{R})}$ is learnt from a batch of samples $(\mathbf{I}_a^-, \mathbf{I}_a^+)$. We request $\mathbf{P}^a_{(\mathbf{R})} \geq \mathbf{0}$ and propose $\mathbf{P}^a_{(\mathbf{R})}$ to be the solution to a positive least squares problem. As demonstrated in [7] this yields a predictor reflecting the complexity of stimulus flow patterns under actions.

Finally $G\left(\mathbf{R}\right)$ is chosen in such a way as to impose a cost on the growth of receptive fields. Choosing $G(\mathbf{R}) = \omega\|\mathbf{R}\|_2^2$ provides control over the smoothness of the receptive field boundaries. For $\omega = 0$ solutions with hard receptive field boundaries are obtained.

3 Method

We consider the sensor surface to be a disk, discretized at $N = 2877$ locations in a grid-like layout, and being organized into $M = 48$ receptive fields. The environment is given as a plane textured by a very high resolution image depicting a real world scene. A state s consists of a position of the sensor surface with respect to this plane. In this paper we assume the sensor surface to be parallel to the plane and each location records luminance over the covered area into discrete grayscale images $\mathbf{i}$. This sensor interacts with the environment through four types of actions, translations in x- and y-directions, rotations and changes in distance to the plane (zoom). An action set $\mathcal{A}$ is obtained by sampling a particular action probability distribution representative of the agent's behavior. For the results presented in this paper each behavior is represented with 60 samples as shown in Fig. 2. For each action a a pair of samples is obtained by positioning the agent in a random state on the environment and taking the chosen action a. This process is repeated 68 ($> M$) times for each a, acquiring the dataset $\mathcal{D} = \{(\mathbf{I}_a^-, \mathbf{I}_a^+)\}$.

To find $\mathbf{R}^*$ we iteratively improve the optimization problem given in Eq. (1) using a projected gradient descent method [10]. At each iteration we learn predictors $\mathbf{P}^a_{(\mathbf{R})}$ that best satisfy $\mathbf{R}\mathbf{i}_a^+ = \mathbf{P}^a_{(\mathbf{R})}\mathbf{R}\mathbf{i}_a^-$ in a positive least squares sense using the optimization method known from [11]. Note that, even though $\mathbf{P}^a_{(\mathbf{R})}$ cannot be obtained as a closed form solution, the gradient needed to iterate Eq. (1)



(a) Converged Matrix $\mathbf{R}^*$

(b) Topology of $\mathbf{R}^*$ on the Sensor Surface

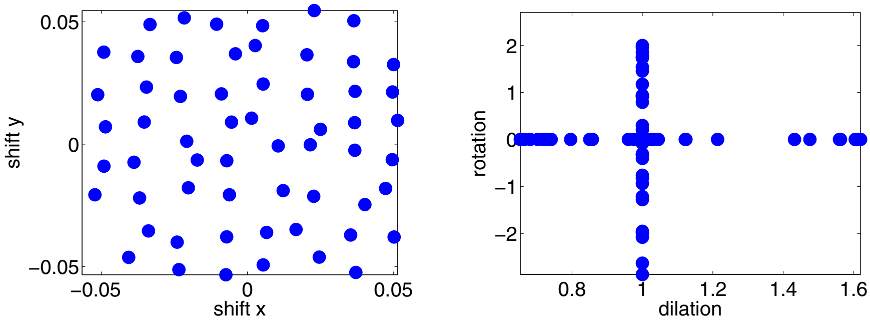
Fig. 1. Emergent clustering of receptive fields (RFs). Left: A converged but topologically orderless matrix $\mathbf{R}$ as seen by the algorithm; each entry specifies the contribution of a location on the sensor surface to a receptive field (RF); the sensor surface is discretized into 2877 pixels (x-axis), and the matrix $\mathbf{R}$ codes for 48 RFs. Right: The sensor surface and the coverage of 7 selected RFs at spatial locations where their contribution is predominant; this view reveals the implicitly present topological clustering in $\mathbf{R}$.

can still be found in closed form by applying the implicit function theorem to the Karush-Kuhn-Tucker optimality conditions of the positive least squares optimization problem [12]. While it is no problem to find a solution for $\mathbf{R}$ with an online method, convergence is much slower, we therefore choose here the batch approach for practical reasons. However, we note that under different circumstances an online implementation might be preferable, e.g. for a purely biologically inspired implementation in a robot with stronger memory constraints and a longer exploration phase.

The experiments presented in Sect. 4 were initialized as follows: the topology of the sensor $\mathbf{R}$ was randomly initialized according to a uniform distribution between zero and one, and then projected to obey the constraints $\mathcal{R}$. The cost for growing receptive fields was kept at a constant level $\omega = 0.3$. It is important to note that with a randomized initialization, nothing prevents the adaptation process from converging to a locally optimal solution. From a biological point of view, we accept these solutions as possible branches of evolutionary development.

4 Results

To demonstrate the correlation principle introduced in Eq.(2) we start by showing the results for an immobile agent. This example, although discarding any meaningful behavior, shows a crucial capability of the proposed method namely the requested property i) the development of spatially coherent light receptive fields on a visual sensor surface. Figure 1 highlights the discovery of topological order from the orderless sampling of the underlying image.



(a) Shift

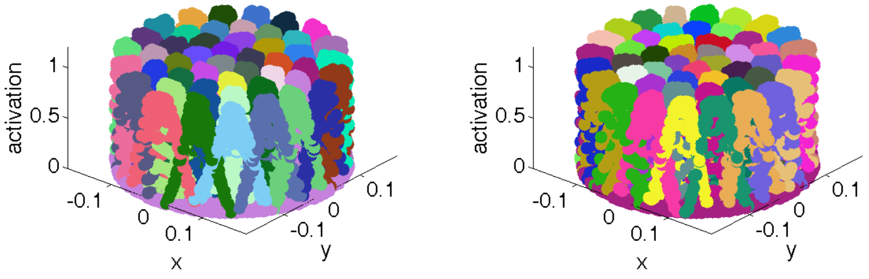
(b) Rotation and Zoom

Fig. 2. Two different behaviors represented as action distributions. Left: uniform 2-dimensional translations in a given range covering 10 times the distance between discrete sampling locations on the sensor surface in each direction. Shift units are normalized with respect to the environment. Right: independent zoom and rotation actions distributed uniformly on each axis. Rotations are given in radians and dilations are given as a scale factor. Both operate with respect to the center of the sensor surface. Zoom actions range from 0.6 to 1.66 and rotations cover $-\pi$ to π.

As external observers we have the privilege of knowing the spatial locations where the sensor surface was sampled and as such we are able to plot the topological ordering of receptive fields on the sensor surface as shown in Fig. 1(b). In the two dimensional visualization we choose to show at each discrete sensor surface location the predominant receptor. The clustering property of the receptive field elements is clearly demonstrated. Since in this case no action is taken, this clustering is a sole consequence of the interaction between the correlation based cost function and the low frequency characteristic of the observed environment. Note that the agent does not have access to the sampling locations of the sensor surface and is thus unaware of the final topological ordering. The proposed algorithm operates solely on matrix **R** which is absent of any topological meaning even in the final converged state, as shown in Fig. 1(a).

For active agents we will now consider two different behaviors as shown in Fig. 2(a) and Fig. 2(b). The first consists of a uniform action probability distribution of 2-dimensional translations over the sensor surface in a given range. This scenario relates to translational unbiased oculomotor control causing random stimulus displacements. The second behavior is composed of independent zoom and rotation actions distributed uniformly on each axis. This mimics the behavior of an object manipulating agent where the oculomotor system stabilizes the sensor on target, mechanically compensating for image translations but not image rotations or scaling. These setups demonstrate that the agent's behavior induces different topologies of receptive fields on the sensor surface.

In Fig. 2 the converged layouts for the two considered action distributions are shown. The nature of the two converged topologies exhibits macroscopic



(a) Shift

(b) Rotation and Zoom

Fig. 3. Sensor topologies obtained under behaviors visualized in Fig. 2(a) and Fig. 2(b)

differences: in the translation only case we can identify a tendency for hexagonal tiling structures over the entire sensor surface (apart from boundary effects), whereas in the rotation and zoom case the receptors organize radially in clear circular rings. Unlike in Fig. 1, the 3-dimensional perspective shows the smooth overlapping between receptive field elements.

To better comprehend the resulting sensor layouts, we refer back to the work of Clippingdale and Wilson [6], where the fitness of a layout relates directly to the distance between predicted and original point locations. In our case, just as in [6] a perfect sensor layout is one where receptors exactly map one onto another for every considered action resulting in $\mathbf{P}^{a}_{(R)}$ matrices where each row contains exactly one non-zero entry. Any deviation from this case leads to an increase in prediction error and lowers correlation. This fact allows us to replace the Euclidean distance as used by Clippingdale and Wilson by one based solely on correlation between sensory readings disregarding any knowledge about the sensor topology.

5 Conclusion and Outlook

This paper explored how the behavior of an artificial agent can shape the topology of a visual sensor. We proposed that a well suited sensor is one which simplifies stimulus flow patterns – and hence stimulus prediction – under a given set of actions. We showed that this quality is captured by spatiotemporal cross-correlation and can be used to self-organize visual sensor topologies on a given surface. The method proposed in this work simultaneously develops spatially coherent receptive fields and organizes their layout according to an executed behavior.

Recognizing the mutual coupling of morphology and active behavior in organisms evolved in nature, we believe that in artificial agents physical structure and actuation should eventually emerge through a co-developmental process. Working in this direction, we will investigate in future contributions the reciprocal influence of physical form on behavior in order to deduce suitable actions from a given sensor topology.

Acknowledgments. This work was supported by the European Commission proj. FP7-ICT-248366 RoboSoM, by the Portuguese Government – Fundação para a Ciência e Tecnologia (FCT) proj. PEst-OE/EEI/LA0009/2011, proj. DC-CAL PTDC/EEA-CRO/105413/2008, and FCT grant SFRH/BD/44649/2008.

References

1. Hayhoe, M., Ballard, D.: Eye movements in natural behavior. Trends in Cognitive Sciences 9, 188–194 (2005)
2. Egelhaaf, M., Kern, R., Krapp, H.G., Kretzberg, J., Kurtz, R., Warzecha, A.K.: Neural enconding of behaviourally relevant visual-motion information in the fly. Trends in Neurosciences 25, 96–102 (2002)
3. Schwartz, E.L.: Computational anatomy and functional architecture of striate cortex: A spatial mapping approach to perceptual coding. Vision Research 20(1), 645–669 (1980)
4. Petrowitz, R., Dahmen, H., Egelhaaf, M., Krapp, H.G.: Arrangement of optical axes and spatial resolution in the compound eye of the female blowfly *Calliphora*. J. Comp. Physiology A 186, 737–746 (2000)
5. Lichtensteiger, L., Eggenberger, P.: Evolving the morphology of a compound eye on a robot. In: Proc. 3rd Europ. Worksh. on Adv. Mobile Robots, pp. 127–134 (1999)
6. Clippingdale, S.M., Wilson, R.: Self-similar neural networks based on a kohonen learning rule. Neural Networks 9(5), 747–763 (1996)
7. Ruesch, J., Ferreira, R., Bernardino, A.: A measure of good motor actions for active visual perception. In: Proc. Int. Conf. Dev. and Learning, ICDL (2011)
8. Ruesch, J., Ferreira, R., Bernardino, A.: Predicting visual stimuli from self-induced actions: an adaptive model of a corollary discharge circuit. IEEE Transactions on Autonomous Mental Development (submitted)
9. Crapse, T.B., Sommer, M.A.: Corollary discharge across the animal kingdom. Nat. Rev. Neuroscience 9, 587–600 (2008)
10. Absil, P.A., Mahoney, R., Sepulchre, R.: Optimization Algorithms on Matrix Manifolds. Princeton University Press (2008)
11. Barzilai, J., Borwein, J.: Two-point step size gradient methods. IMA Journal of Numerical Analysis 8, 141–148 (1988)
12. Bertsekas, D.P.: Constrained Optimization and Lagrange Multiplier Methods. Athena Scientific (1996)

Self-organization of Spinal Reflexes Involving Homonymous, Antagonist and Synergistic Interactions

Hugo Gravato Marques[1,2], Kristin Völk[1], Stefan König[1], and Fumiya Iida[1]

[1] ETH,
Dep. of Mechanical and Process Engineering, BIRL,
Zurich 8092, Switzerland
hgmarques@gmail.com
[2] University of Zurich,
Institute for Informatics, AI Lab.,
Zurich 8050, Switzerland

Abstract. Recent results in spinal research are challenging the historical view that the spinal reflexes are mostly hardwired and fixed behaviours. In previous work we have shown that three of the simplest spinal reflexes could be self-organised in an agonist-antagonist pair of muscles. The simplicity of these reflexes is given from the fact that they entail at most one interneuron mediating the connectivity between afferent inputs and efferent outputs. These reflexes are: the Myotatic, the Reciprocal Inibition and the Reverse Myotatic reflexes. In this paper we apply our framework to a simulated 2D leg model actuated by six muscles (mono- and bi-articular). Our results show that the framework is successful in learning most of the spinal reflex circuitry as well as the corresponding behaviour in the more complicated muscle arrangement.

1 Introduction

Historically the spinal cord has been viewed as a rather inflexible system, composed of hardwired reflexes that have very limited degrees of adaptability and plasticity [9][1]. However, recent studies have shown that at least some of these circuits can be adapted by changing the contingent (i.e. temporally correlated) sensor and motor information. In Petersson et al. [7] showed that the delivery of false tactile feedback in response to spontaneous muscle twitches (SMTs) can systematically modify the response of the spinal withdrawal reflex. One of the major contributions of this work was the identification of SMTs as a major driving force in the adaptive process. SMTs consist of involuntary contractions that can activate muscles independently [1][2]. This kind of motor activity can be

[1] This research was funded by the Swiss SNF, Grant No. PP00P2123387/1, the Swiss NCCR Robotics, and the EU's FP7, Cognitive Systems, Interaction, Robotics no. 207212 – eSMCs.

[2] Note that SMTs are not the only mechanism of spontaneous motor activity (see [1]).

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 269–278, 2012.

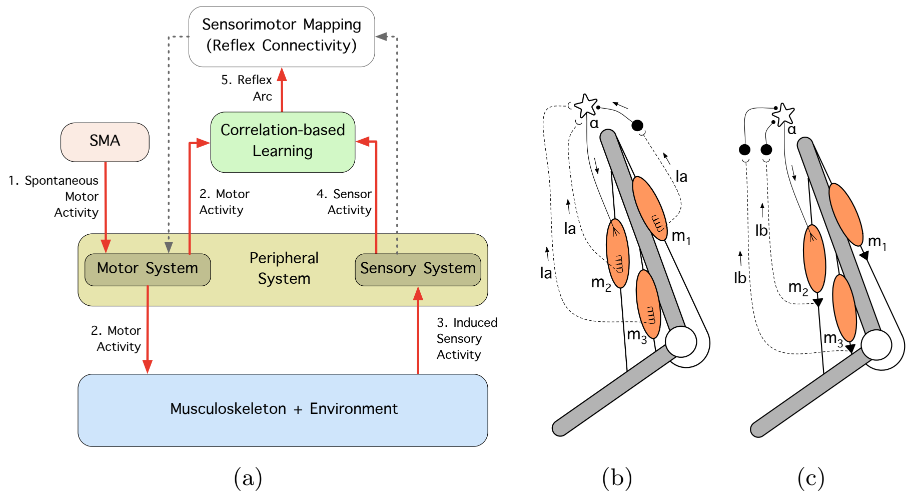


Fig. 1. (a) The framweork proposed; it works as follows: (1) spontaneous motor activity (SMA) produces contractions of individual muscles (SMTs), (2) the muscle contractions produce forces which are propagated through the musculoskeletal system (as well as throughout the environment where it is embedded), (3) the changes produced in the musculoskeletal system are captured by the different sensors, which (4) convert them into sensor activity, (5) the correlation between the sensor and motor activity is used to learn the reflex circuitry between each sensor and motor pair. (b-c) The spinal circuitry of the three reflexes investigated: b) the Myotatic reflex and the Reciprocal Inhibition reflex (which are supposed to counteract the effects of external loads), and c) the Reverse Myotatic reflex (which is supposed to prevent muscles from producing too high forces). The stars represent α-motoneurons, the large solid circles represent inhibitory interneurons, semi-circled arcs represent excitatory connections, small solid circles represent inhibitory connections, the dashed lines represent afferent inputs from a) Ia fibers and b) Ib fibers, m_i indicates an abstract muscle i, and the arrows represent the flow of information. The spirals in (b) represent the muscle spindles, the filled triangles in (c) represent the Golgi-tendon organs. The reflex circuitry has been abstracted from [8, pp.79-86,209-15, 256-60]

observed during active sleep from fetuses to adults, and it has been argued to contribute strongly to the development of the human motor system.

In previous work we proposed a framework to self-organize spinal reflexes from the contingent sensor and motor information induced during SMTs (see Fig.1a). These reflexes were: the Myotatic, the Reciprocal Inhibition and the Revere Myotatic reflexes (see Fig. 1b-c). These reflexes consist of very small local circuits (they include at most one interneuron in their reflex arcs) that coordinate muscle information. They are carried out through two types of afferent inputs: the Ia and the Ib fibers. The Ia fibers estimate changes in the muscle length as well as (positional) muscle length [4], and the Ib fibers respond to small variations in muscle force [5].

In the human spine the circuitry of these reflexes involve mainly three types of sensorimotor (muscle) interactions: homonymous (i.e. interactions between the same muscle), antagonistic (i.e. interaction between muscles that move a joint in opposing directions), and synergistic (i.e. interaction between muscles that move a joint in the same direction). In our previous work we have successfully self-organized the three reflexes in a minimalistic setup involving homonymous and antagonist interactions using a pair of simulated mono-articular muscles [6]. In this paper, our goal is to test the framework in a more complicated muscle arrangement which comprises six muscles and includes synergistic interactions provided by bi-articular muscles.

The remainder of this paper is organized as follows. The second section describes the implementation of our framework (here we will only provide the details relevant to understand this paper; more information can be found in [6]). The third section provides the results obtained. The fourth section discusses the results. And the fifth section concludes the paper and gives some directions for future work.

2 Methods: Implementation of Framework Models

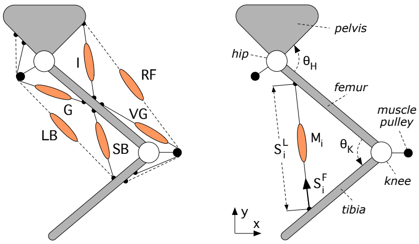
Our framework consists of five interacting models: a musculoskeletal model (and its environment), a peripheral model, a model of spontaneous motor activity (SMA), a learning model based on the correlations between sensor and motor activity, and a model of the reflex sensorimotor mapping (see Fig.1). The first four models are involved in the learning of the reflexes (learning stage) and the last model is responsible for triggering the reflex activity (testing stage). The interaction between these models is described in Fig. 1. In this section we will describe the implementation of each of these models.

reflex learning from

2.1 Implementation of the Musculoskeletal System and the Environment

Our musculoskeletal system consists of a 2D virtual model of a leg actuated by six muscles (see Fig. 2a). We have called these muscles: Glutei (G), Iliacus (I), Rectus Femuris (RF), Long Biceps (LB), Short Biceps (SB), and Vast Group (VG), which in the human leg produce approximately the same type of (flexion and extension) motions. The muscles LB and RF are bi-articular muscles (represented as dashed lines in Fig. 2a). The simulation of the leg dynamics is carried out in SimMechanics/Simulink. The skeleton model consists of three rigid bodies: the pelvis, the femur and the tibia, which are connected by hinge joints (hip and knee). In our simulation the pelvis is fixed in space, and only the femur ($mass = 1kg$, $length = 0.5m$) and tibia ($mass = 0.5kg$, $length = 0.5m$) are allowed to move (e.g. due to gravity).

Each muscle is simulated as a straight line between two rigid bodies (see Fig. 2b). The muscle model used in our investigation is based on a 2-element nonlinear Hill model [3] [10]. The two elements in the model are an active contractile



(a) (b)

Fig. 2. Diagram of the leg model implemented. a) the geometric arrangement of the six muscles in the leg: the Glutei (G), the Iliacus (I), the Rectus Femuris (RF), the Long Biceps (LB), the Short Biceps (SB), and the Vast Group (VG); bi-articular muscles are shown in dashed lines; b) an abstraction of the muscle model used, M_i corresponds to the motor activity of α-motoneurons of muscle i. The filled circles represent pulleys which simulate the way muscles wrap around the joints. Each muscle has two sensors, S_i^L and S_i^F, which are used to estimate the length and force (respectively) of muscle i.

element in parallel with a passive elastic element. The contractile element models the active force generated by the muscle fibers. This element includes a damping mechanism that simulates the force-velocity relation of biological muscles. The passive elastic element models the muscle fiber's resistance to deflection.

The force produced by muscle i at its attachment points is given by:

$$F_{M_i} = F_{CH_i} + F_{SH_i}, \qquad F_{CH_i} = \frac{M_i}{1 + C \cdot \dot{l}_i}, \qquad F_{SH_i} = K \cdot \Delta l_i \tag{1}$$

where F_{CH_i} is the force produced by the Hill contractile element of muscle i and F_{SH_i} is the force produced by the passive spring element of muscle i, C is a constant damping factor, K is a constant spring factor, M_i is the motor activation of motor i, $\dot{l}_i$ is the rate of change of the muscle length relative to muscle i, Δl_i is the passive deformation of muscle i. In biology the force generated by the passive spring element of the muscle, F_{SH_i}, is significantly smaller than the force generated by the contractile element, F_{CH_i}. To achieve these properties we set $K = 10$ and $C = 0.5$. At the beginning of both the learning and testing stages, the model starts with all muscles relaxed ($M = 0$). In this condition the leg falls straight down due to the effect of gravity.

2.2 Implementation of Peripheral System

The peripheral system includes motor and sensing elements. The sensor elements consist of analogues to α-motoneurons the activity of which determines the muscle contraction, M_i. On the sensing side we simulate two types of sensors (see Fig. 2b): one that measures the length of the muscle, S_i^L (i.e. the distance between the two attachment points), and one that measures the force at the attachment points, S_i^F; when referring to sensors indiscriminately we will simply use the symbol S. In our simulation the sensor activity consists of the derivatives of these sensor inputs ($\dot{S}_i^L$ and $\dot{S}_i^F$), which provide qualitative analogues of the Ia and Ib afferent fibers.

2.3 Implementation of Correlation-Based Learning

This process identifies the reflex circuitry based on the correlation between sensor and motor activity. All possible pairs of sensor and motor elements are considered. We use the method of motor-directed somatosensory imprinting (MDSI), which has been used to explain the self-organization of the withdrawal reflex [7]. This method uses the anti-Hebbian rule [2] which is given by the additive inverse of the temporal correlation between the sensor and motor activity. It is important to mention here that despite the fact that the anti-Hebbian rule has been used to justify plastic phenomena in the spinal cord [7], such a rule (to our knowledge) has not yet been explicitly identified. The reflex connectivity, Q, is then given by:

$$Q_{i,j} = -\eta_{ij} \sum_{t=1}^{T} M_{i,t} \cdot \dot{S}_{j,t}, \qquad \eta_{ij} = \left[max(\dot{S}_j) \sum_{t=1}^{T} M_{i,t} \right]^{-1} \tag{2}$$

where η_{ij} is a normalization factor, $M_{i,t}$ is the motor activity of motor i at timestep t, $\dot{S}_{i,t}$ is the sensor activity of sensor j at timestep t, and T is the number of timesteps taken by the learning process. Excitatory connections are characterized by positive values and inhibitory connections by negative values. The strength of each connection is given the magnitude, $|Q|$.

2.4 Implementation of Spontaneous Motor Activity

During the learning stage the generation of single muscle twitches is done by sequentially contracting one muscle after the other, generating a total of six SMTs. Each twitch consists of a short rectangular pulse of amplitude $1mu$ (motor units) and duration of $1s$. The time between twitches is set to a value large enough to allow the system to stop oscillating.

2.5 Implementation of Sensorimotor Mapping

The testing of the reflex behaviours is carried out exclusively by the sensorimotor mapping model. The testing consists of applying an external force, D at the bottom of the tibia ($D = 1N$ applied in the x direction). This causes both the femur and the tibia to rotate counterclockwise (see Fig. 2b), which causes the length of the muscles G, LB and SB to increase and the length of the muscles I, RF and VG to decrease. The reflex behavior, M_i, is then triggered by the measured sensor stimulation at each sensor S_j weighted by the respective connection $Q_{i,j}$:

$$M_{i,t} = G \cdot \sum_{j=1}^{m} A_{i,j,t}, \qquad A_{i,j,t} = Q_{i,j} \frac{\dot{S}_{j,t}}{max(\dot{S}_{j,t})} \tag{3}$$

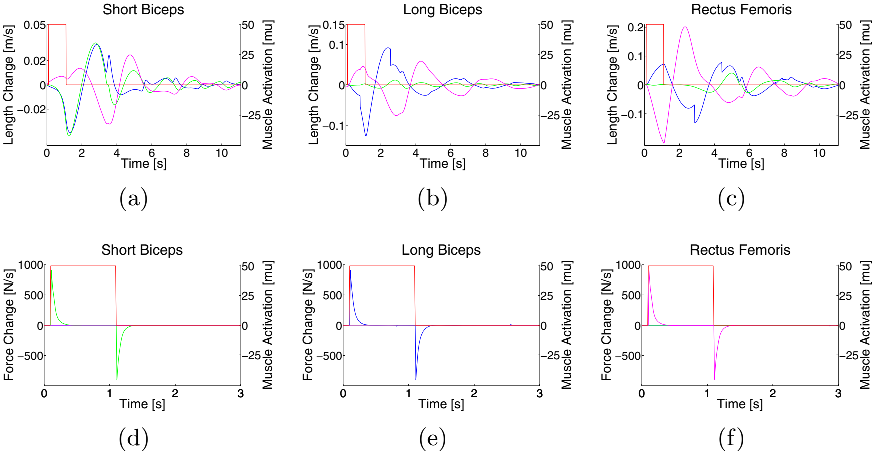
where G is a dimensionless value with the reflex gain, m is the number of motors in the system, t is time, and $A_{i,j}$ is the output of the inhibitory or excitatory connection. In biology the positive or negative effect of a connection on a given neuron is defined by the nature of the connection (excitatory and inhibitory respectively); negative connections can only decrease the activity of afferent neurons, and excitatory connections can only increase the activity of afferent neurons. To ensure these effects in our simulation, we add a condition where we define that inhibitory connections (i,e. those where $Q_{i,j} < 0$) can only have a negative impact on the motor activity; and that excitatory connections (i,e. those where $Q_{i,j} > 0$) can only have a positive impact on the motor activity. This is achieved by setting $A_{i,j} = 0$ whenever $\dot{S}_{j,t} < 0$.

3 Results

Figure 3 shows the sensory activity induced in response to SMTs in different muscles. The muscles shown include the three types of muscle interactions investigated here (i.e. homonymous, antagonistic and synergistic). Figure 3a-c illustrates the reaction of the length sensors to the contraction of one muscle induced by a SMT in this muscle. Figure 3d-e shows the reaction of the force sensors to the same SMTs. As can be seen, all the SMTs induce causal information in all the length sensors of each muscle, whereas the force sensors are affected only by contractions of the homonymous muscle. This is because when a muscle is relaxed the only force in the muscle is due to the passive spring element which has a negligible magnitude when compared with the active force that can be produced by the contractile element of the muscle.

The data shown in Fig. 3 suggest causal relations between force sensors and homonymous muscles, and between length sensors and homonymous, antagonistic and synergistic muscles. The learned connectivity matrix, Q, is shown in Fig. 4a. In relation to the force sensors we obtain inhibitory connections with the homonymous muscles only (e.g. between $\dot{S}^F_G$ and M_G); all the other connections obtained have negligible magnitudes (≤ 0.001). This connectivity corresponds to that of the Reverse Myotatic reflex for homonymous muscles. In comparison to the analogue biological circuitry (see Fig. 1c) we fail to obtain inhibitory connections between force sensors and synergistic muscles (see Sect. 4 for discussion).

In relation to the length sensors we obtain excitatory connections with homonymous muscles (e.g. between $\dot{S}^L_G$ and M_G) as well as with synergistic muscles (e.g. between $\dot{S}^L_{LB}$ and M_G). This connectivity pattern is in line with the connectivity of the Myotatic as shown in Fig. 1b. The only exception to this rule is observed between $\dot{S}^L_{VG}$ and M_{RF} where we obtain an inhibitory connection. This connection is caused by the fact that the twitches are performed with the leg hanging down in a straight line due to gravity. In this position the contraction of the RF muscle leads to a flexion of the hip, which causes the Tibia to swing slightly backwards due to gravity; this in turn causes a knee flexion instead of the knee extension that would typically be observed when the RF contracts.



(a) (b) (c)

(d) (e) (f)

Fig. 3. (a)-(c): The changes in muscle length of the Short Biceps (green), the Long Biceps (blue) and the Rectus Femoris (magenta) in response to a SMT (red) carried out by a) the Short Biceps, b) the Long Biceps and c) the Rectus Femoris. (d)-(f): The changes in muscle force of the Short Biceps (green), the Long Biceps (blue) and the Rectus Femoris (magenta) in response to a SMT (red) carried out by d) the Short Biceps, e) the Long Biceps and f) the Rectus Femoris.

In addition we obtain inhibitory connections between the length sensors and antagonist muscles (e.g. between $\dot{S}^L_{RF}$ and M_G), which is consistent with the connectivity observed in relation to the Reciprocal Inhibition reflex (see Fig. 1b). Figure 4b provides a more clear representation of the connectivity of the length sensors with the homonymous, antagonist and synergistic muscles.

Connections can also be found between muscles in distal joints, (e.g. the excitatory connection between $\dot{S}^L_I$ and M_{SB}). This connectivity is present because a movement produced at a given joint can induce movement in other joints (e.g. due to gravity or inertia). Connections between muscles at distal joints are also present in the human spinal cord but they will not be analysed here. Connections between length sensors and motor elements not shown in the Q-matrix of Fig. 4a have very small magnitudes (≤ 0.006) and do not have any behavioral relevance.

The resulting reflex behavior is shown in Fig. 5. The figure shows the response of the six muscles to the external perturbation (see Sect. 2.5), in three different conditions: using no reflexes ($G = 0$), using the non-modulated Q matrix ($G = 1$) and using a modulation gain of $G = 3$. By comparing the angles of the knee and hip joint at reflex application to those without reflexes, it can be seen that the action of the reflexes has the effect of dampening the system, similar to the function of a proportional controller. Higher gains cause a higher dampening of the system and result in an increase of the overall motor activity. Additionally the muscle activations for agonist-antagonist pairs tend to be opposite (e.g. between RF and LB, or between G and I). Synergistic muscles on the other hand tend to be recruited at the same time (e.g. LB and SB, or RF and I).

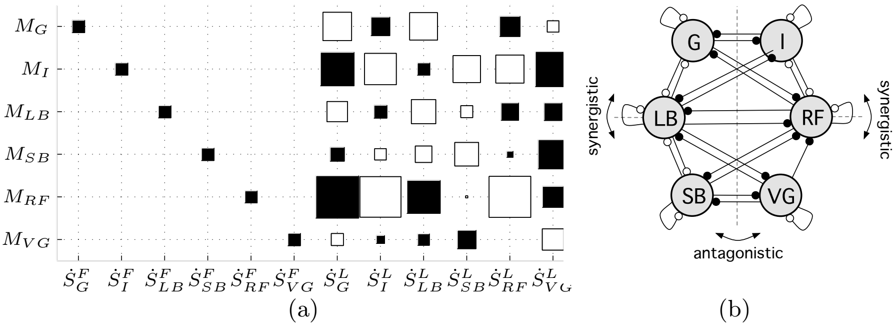

M_G
M_I
M_{LB}
M_{SB}
M_{RF}
M_{VG}
$\dot{S}^F_G$ $\dot{S}^F_I$ $\dot{S}^F_{LB}$ $\dot{S}^F_{SB}$ $\dot{S}^F_{RF}$ $\dot{S}^F_{VG}$ $\dot{S}^L_G$ $\dot{S}^L_I$ $\dot{S}^L_{LB}$ $\dot{S}^L_{SB}$ $\dot{S}^L_{RF}$ $\dot{S}^L_{VG}$
(a)
G
I
LB
RF
SB
VG
synergistic
synergistic
antagonistic
(b)

Fig. 4. The reflex circuitry obtained using our framework a) in a matrix and b) in a graph representations. Filled squares and circles represent inhibitory connections and empty squares and circles represent excitatory connections. In a) the strength of each connection is given by the square size; all the sizes have been normalized by the weight of the highest connection. In b) are shown only the interactions of length sensors with homonymous, antagonist and synergistic muscles.

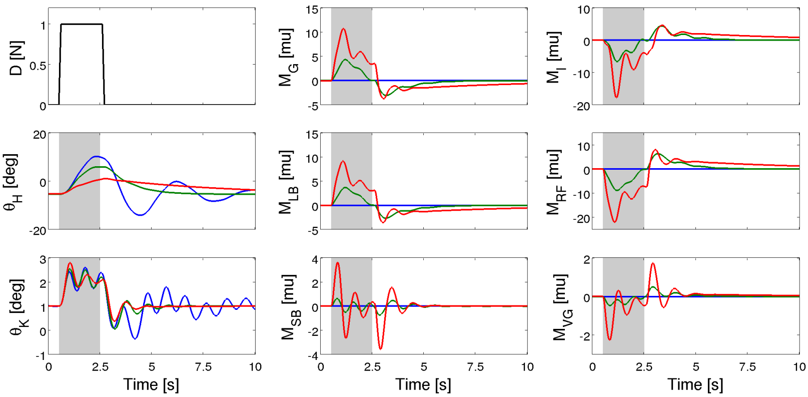


Fig. 5. Muscle-related activity generated by the reflex circuitry in reaction to an external disturbance of $1N$ in the x-direction imposed on the endpoint of the Tibia. Left side from top to bottom: 1) the external disturbance, 2) angle of the hip joint, and 3) angle of the knee joint. In the middle from top to bottom, activations of: 1) Glutei , 2) Long Biceps, and 3) Short Biceps (mu stands for motor units). Right side from top to bottom, activations of: 1) 1) Iliacus, 2) Rectus Femoris, 3) Vast Group. Each line in the figure shows three experimental conditions: blue, no reflex activity, $G = 0$; green, non-modulated reflex activity $G = 1$, and red, modulated reflex activity $G = 5$.

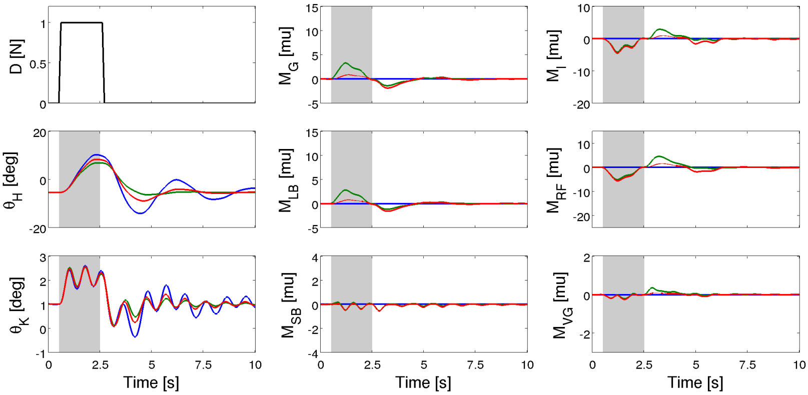


Fig. 6. The Reverse Myotatic reflex. Same elements as Fig. 5, but the three conditions shown are: blue, no reflex activity, $G = 0$; green, non-modulated reflex activity $G = 1$, and red, *dual-gain* condition (see text).

The muscle activity produced by the Myotatic and the Reciprocal Inhibition reflexes is shown in Fig. 5. In relation to the Myotatic reflex it can be observed that the increase in the length of the muscles G, LB and SB (imposed by the external perturbation) results in an increase of motor activity in these muscles[3]. In relation to the Reciprocal Inhibition reflex it can be observed that the decrease in the length of the muscles I, RF and VG (imposed by the external perturbation) results in a decrease of motor activity in these muscles (see also footnote).

The activity of the Reciprocal Inhibition reflex is slightly more complicated to visualize since in the range of forces we use the force component contributes very little to the overall muscle activity. To give expression to the force component we set a higher gain ($G_F = 50$) to the force elements in the matrix Q, while keeping the gains of the length elements ($G_L = 1$); we call this condition *dual-gain*. In this condition the effects of the Myotatic reflex alternate constantly with those of the Reverse Myotatic reflex, leading to peaks that alternate between positive and negative muscle activity. To show the overall muscle activity in this condition we filter all the muscle responses (in the three conditions) with a Savitzky-Golay filter of order 3 and window size of 51 (*sgolayfilt* function in Matlab). Fig. 6 shows the muscle activity in the *dual-gain* condition in comparison with that in $G = 0$ (no reflex activity) and $G = 1$ (non-modulated reflex activity). As can be seen in the *dual-gain* condition the Myotatic reflex leads to a generalized decrease of muscle activity in all the active muscles ($M_i > 0$). This is due to the inhibitory nature of the Reverse Myotatic reflex circuitry. In addition, it can be

[3] The length increase of the SB muscle, as well as the length decrease of the VG muscle, can only be observed immediately after the disturbance is applied; the small oscillations that are observed afterwards are due to the impact of gravity in the vertical alignment of the leg mentioned above.

seen from the joint angles that the system gets less damped than in the $G = 1$ condition, which is due to the overall decrease of muscle activity.

4 Discussion and Conclusions

Our results show that most of the circuitry and the behavior obtained are consistent with human data. Relative to the Myotatic reflex we obtain, excitatory connections between length sensors and motor elements of homonymous and synergistic muscles. Relative to the Reciprocal Inhibition reflex we obtain inhibitory connections between length sensors and motor elements of antagonist muscles. And relative to the Reverse Myotatic reflex we obtain inhibitory connections between the force sensors and motor elements of the homonymous muscles. The only circuits that are systematically absent in our results (and are present in humans) are those involving connections between force sensors of a muscle and motor elements of the synergistic muscles (see Fig. 1b-c).

In reality the contraction of one muscle should not induce force information on any other muscle. Our hypothesis is that the connectivity between force sensors and motor elements of synergistic muscles is accidental; it is formed because synergistic muscles are often recruited at the same time, which prevents casual sensorimotor relations from being fully disambiguated. We are currently working ways of exploiting synchronous muscle activations during SMA.

References

1. Blumberg, M.S., Lucas, D.E.: Dual mechanisms of twitching during sleep in neonatal rats. Behav. Neurosci. 108, 1196–1202 (1994)
2. Földiák, P.: Forming sparse representations by local anti-hebbian learning. Biol. Cybern. 64, 165–170 (1990)
3. Hill, A.V.: The heat of shortening and dynamics constants of muscles. Proc. R. Soc. Lond. B 126(843), 136–195 (1938)
4. Hulliger, M.: The mammalian muscle spindle and its central control. Rev. Physiol. Bioch. P. 101, 1–110 (1984)
5. Jami, L.: Golgi tendon organs in mammalian skeletal muscle: Functional properties and central actions. Physiol. Rev. 72(3), 632–666 (1992)
6. Marques, H.G., Imtiaz, F., Iida, F., Pfeifer, R.: Self-organisation of reflexive behaviour from spontaneous motor activity (2012) (under review)
7. Petersson, P., Waldenström, A., Fåhraeus, C., Schouenborg, J.: Spontaneous muscle twitches during sleep guide spinal self-organization. Nature 424, 72–75 (2003)
8. Pierrot-Deseilligny, E., Burke, D.: The Circuitry of the Human Spinal Cord. Cambridge University Press (2005)
9. Schouenborg, J.: Somatosensory imprinting in spinal reflex modules. Journal of Rehabilitation Medicine (41 suppl.), 73–80 (2003)
10. Zajac, F.E.: Muscle and tendon: properties, models, scaling and application to biomechanics and motor control. Crit. Rev. Biomed. Eng. 17(4), 359–410 (1989)

A Computational Model of the Role of Serotonin in Reversal Learning

Graeme Hattan and Bernd Porr

Biomedical Engineering, School of Engineering, University of Glasgow,
G12 8LT, United Kingdom
g.hattan.1@research.gla.ac.uk,
bernd.porr@glasgow.ac.uk

Abstract. It has been shown that the action of serotonin on the orbitofrontal cortex (OFC) is crucial for the inhibition phase of reversal learning. Serotonin has also been shown to facilitate the induction of LTD throughout the prefrontal cortex. We present a biologically realistic, systems level model which proposes a mechanism for the release of serotonin in response to the omission of an expected reward. Serotonin release, as a result of the combination of excitation of the dorsal raphé nucleus (DRN) pathway and the lack of inhibition of the DRN from the lateral habenula, leads to LTD in the OFC and suppression of excitation of the nucleus accumbens shell due to reward predicting sensory stimuli. Behavioural inhibition is controlled via the shell-ventral pallido-mediodorsal pathway, which serves as a feed forward switching mechanism and enables the behavioural inhibition required to achieve reversal learning.

1 Introduction

There is much evidence to support the involvement of serotonin in a wide range of affective processes, however despite this little is known about the specific computational role of serotonin in any of them [7]. Of particular interest from the perspective of adaptive behaviour is the process of reversal learning. During the process of reversal learning it is necessary for an agent to inhibit it's approach of a previously rewarded stimulus. Multiple studies have shown that serotonin depletion disrupts this process both in marmosets [3,4,5] and rats [9]. Not only does a more general depletion of serotonin produce these results, but so does depletion specific to the orbito-frontal cortex [5].

Here we present a biologically inspired, systems level model which combines the action of both serotonin and dopamine to achieve learning and reversal learning in a simulated food seeking task. Stimulus-reward associations made during the acquisition phase are realised in the same manner as that of the Thompson model [20]. The release of phasic dopamine from the ventral tegmental area (VTA) in response to a reward results in long term potentiation (LTP) of active pathways in the nucleus accumbens and the OFC. Reversal learning is achieved through the action of serotonin which causes long term depression (LTD) in the OFC [22]. Depletion of serotonin within the model leads to substantially more preservative errors being made during reversal learning.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 279–288, 2012.

2 Task and Simulated Agent

The model is tested on a food seeking task based on the experiment conducted by Lapiz-Bluhm [9]. A simple reversal learning paradigm is used where a rat has to locate a reward buried in one of two pots. The rat learns to discriminate between the rewarded pots using either a different scent which has been applied to each pot or a different digging material used within them. Once the rat has made the correct stimulus-reward association and completes six consecutive correct trails, the stimuli are reversed (ie the odour or the digging materials are switched) and the total number of trials for the rat to complete another six consecutive correct trials is measured. Similarly, in our simulated food seeking task (shown in Fig. 1A), the simulated agent must learn to associate one of two markers with the reward. The agent must again complete six consecutive trials for both the acquisition and reversal stages. For simplicity the markers are represented as being of two colours, red and blue.

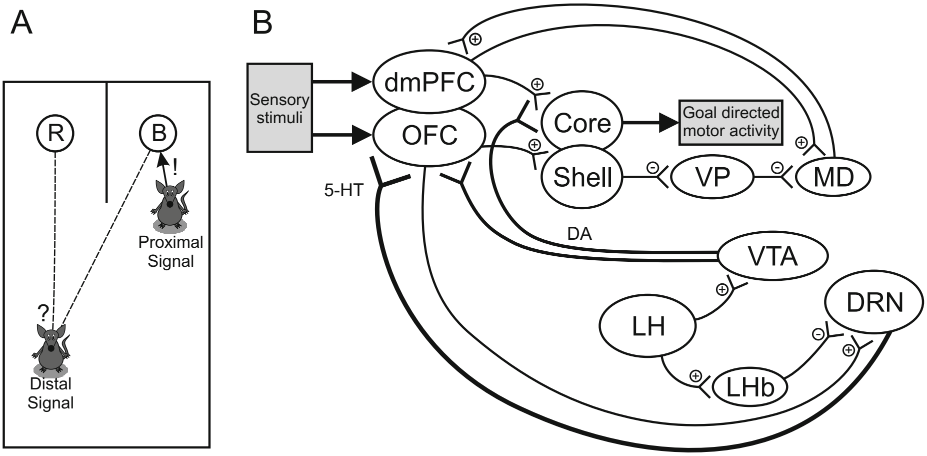


Fig. 1. A. Overview of the simulation environment. The arena has two markers, labelled R and B (red and blue), within one of which lies a reward which the agent cannot sense directly. The agent receives a distal signal from each marker, which indicates how far away the marker is. The agent is required to learn to associate the distal signal with any reward it predicts. Additionally, the agent receives a proximal signal when it nears the marker. This causes it to approach the marker unconditionally and represents a natural curiosity in the marker. B. Overview of the connections in the model, the bottom half of the figure shows the connections necessary to create a serotonin signal coding omission. Also shown is the shell-VP-MD-dmPFC pathway necessary for behavioural inhibition. A detailed view of the NAc core circuitry is shown in Fig. 2. Abbreviations: dmPFC - dorsal medial prefrontal cortex, OFC - orbito-frontal cortex, VP - ventral palladium, MD - mediodorsal nucleus of the thalamus, VTA - ventral tegmental area, LH - lateral hypothalamus, LHb - lateral habenula, DRN - dorsal raphé nucleus, 5-HT - serotonin, DA - dopamine.

The simulated agent moves continuously within the environment and is programmed with two basic behaviours. The first is simply to explore the environment in a random fashion, which is the default. The second is to approach one of the markers if the learning model determines that it is appropriate for it to do so. Approach behaviour is modelled on the action of a Braitenberg vehicle, where the differential between two detectors controls the steering.

3 Model Description

3.1 Acquisition

Neuroplasticity, modulated by the action of phasic dopamine, in the NAc core is primarily responsible for acquisition, where the simulated agent learns to associate a marker with a reward. This is achieved by having separate core units for the approach of each marker (Fig. 2). Inputs to the NAc core are represented as coming from the dorsal medial prefrontal cortex (dmPFC) (Fig. 1B). They are modelled as the lowpass filtered average of each of the agent's sensor outputs which provide distal signals indicating the distance to each marker (Fig. 1A). This data provides a separate conditional stimulus (CS) input to each core unit for each available environmental stimulus. Also feeding into each core unit is an unconditional stimulus (US) for the activity of the unit (Fig. 2). A US signal is activated when the agent enters the close proximity of a marker and simulates a natural curiosity for the object (Fig. 1A). This reduces the initial time spent exploring the environment when reward associations have yet to be made. In contrast to the NAc core, the NAc shell receives sensory input from the OFC. Phasic dopamine is also modelled as causing LTP in the OFC which leads to activation of the NAc core inputs via the shell-VP-MD-dmPFC pathway (Fig. 1B) in response to the appropriate distal signal.

Primary reward information is modelled as originating from the lateral hypothalamus (LH) [2]. Simulated rewards lead to sudden bursts of VTA activity leading to phasic dopamine activity (the LH and the VTA are shown near the bottom of Fig. 1B). These are thought to cause to LTP in the NAc, strengthening stimulus/action associations [17]. The dopamine is modelled as providing the third factor in ISO-3 learning [14] where synaptic weights associated with each CS input active at the time of the burst are increased.

The output of each core unit is the sum of each weighted CS and the US. Additionally, each core unit is allowed to laterally inhibit the other in order to achieve a 'winner takes all' mechanism (Fig. 2). To select the action which the agent engages, the outputs of each core unit are compared and the action associated with the unit with the largest output is taken provided the output is above a threshold. If all outputs are below this threshold, the agent simply engages in exploratory behaviour.

Initially the learning weights will be zero meaning that the distal signals from the environmental stimuli will be effectively ignored by the agent and it will simply explore the environment in a random manner. When the agent nears one

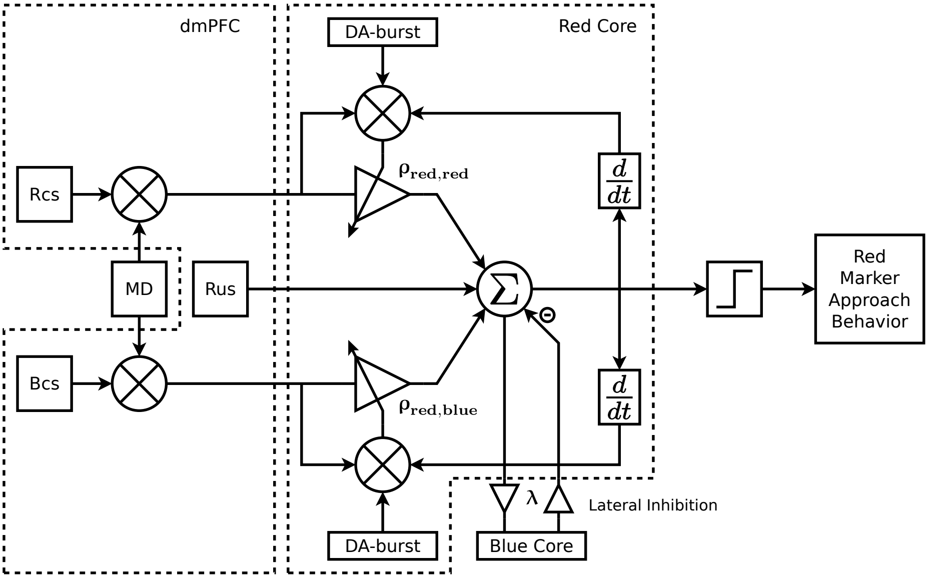


Fig. 2. Circuit diagram of the core unit which switches the approach behaviour for the red marker. When the output is above the threshold, the agent approaches the marker. The left side of the diagram shows the two conditional stimuli (R_{CS} and B_{CS}, related to the red and blue proximal inputs respectively) and single unconditional stimulus (R_{US}) inputs along with the enabling input from the mediodorsal nucleus of the thalamus (MD). The core uses the ISO-3 learning rule [14], where phasic dopamine (DA) bursts are used as a third factor and the ρ variables represent the learning weights. The bottom right shows the input from and output to the core unit used to switch the approach behaviour for the blue marker. These connections enable the winner takes all lateral inhibition mechanism, where λ is a fixed parameter controlling the gain. Similar to the 'red core,' the 'blue core' has the same circuit, except that it receives the blue US (R_{US}).

of the markers, the appropriate US input to the model is activated which in turn pushes the output of the appropriate core unit above the threshold value, causing the agent to make a decision to approach the marker. If the marker is rewarded, contact with it will cause the LH to be activated which will cause a burst of phasic dopamine to be released from the VTA. The synaptic weights associated with the distal inputs to the model which are active at the time of the dopamine burst are then strengthened. The agent is then returned to its starting position.

Over the course of multiple contacts, the weights will increase until the activity of the distal signal alone is enough to push the output of the core above the threshold value. From then, the approach behaviour will be activated by sight of the marker from progressively further distances until the approach behaviour may be instigated from any point in the arena (from the first moment that the marker activates the agent's sensor).

3.2 Extinction

Extinction, the inhibition of previously learned behaviour, is regulated by the action of the neurotransmitter serotonin. Release of serotonin is modelled as leading to LTD in the OFC. The subsequent lack of input to the NAc shell influences the NAc core through the ventral pallido-mediodorsal pathway, causing the behavioural inhibition.

A signal which codes omission is created by comparing the activity in the LH with the activity of neurons in the OFC which fire both in anticipation of and during reward delivery [16]. The OFC exerts an excitatory influence on the DRN while the LH exerts an inhibitory influence via the Lateral Habenula (LHb) [1,18]. In response to an expected reward, the excitatory effect of the OFC is cancelled by the inhibitory effect of the LH (Fig. 3). In the case of an omitted reward, an increase in neuronal activity is seen due to the lack of LH activity (Fig. 4). The connections are shown in the bottom half of Fig. 1B. Equation (13) shows how DRN activity is calculated, this signal is then lowpass filtered to give the serotonin release.

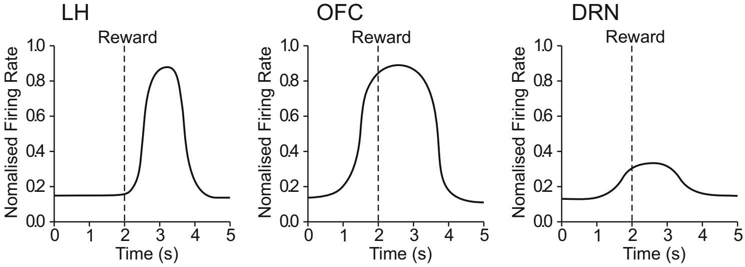


Fig. 3. Stylised plots showing the response of the lateral hypothalamus (LH), the orbito-frontal cortex (OFC) and the resulting response of the dorsal raphé nucleus (DRN) to an expected reward

Behavioural inhibition results from a shutdown of NAc core inputs via the shell-VP-MD-dmPFC pathway (Fig. 1B, equations (8), (9) and (10)). A key part of this mechanism is the ventral palladium (VP) which receives a GABAergic, inhibitory projection from the NAc shell [8]. Without input, the VP inhibits the mediodorsal nucleus of the thalamus (MD) [21]. This circuitry is shown in Fig. 1B. Without MD activity, dmPFC inputs to the NAc core are modelled to be inactivated. Prior to the reversal, sensory stimulation from the rewarded marker excites the OFC and, in turn, the NAc shell leading to activation of the dmPFC-core pathway. As the reversal learning proceeds and LTD results in the OFC, the dmPFC-core pathway becomes progressively less active in response to sensory stimulation from the (now unrewarded) marker. This will continue until the agent no longer approaches the marker and returns to exploring the environment for new rewards, allowing new acquisition to begin when the rewarded marker is discovered.

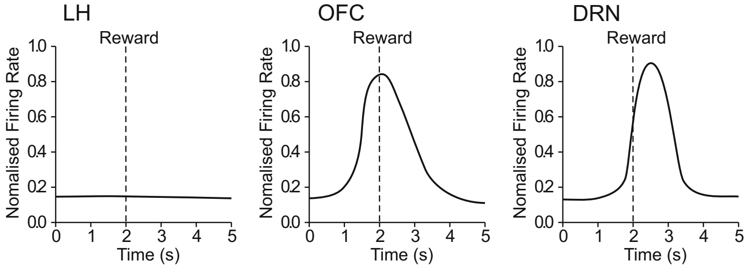


Fig. 4. Stylised plots showing the response of the lateral hypothalamus (LH), the orbito-frontal cortex (OFC) and the resulting response of the dorsal raphé nucleus (DRN) to a reward omission

4 Results

Simulations were run comparing a 60% serotonin depleted version of the model with the non-depleted model as a control. A total of fifty acquisition-reversal runs were made for each. Figure 5 shows the total number of trials required to reach the criterion of six consecutive correct trials for each stage of each run. Figure 6 shows the mean number of trials to the criterion for each stage on each version of the model. The depleted model takes an average of 125.8 trials to reach the criterion, compared to 28.42 trial for the control. A Hotelling t^2 test was run on the two sets of reversal data, giving a $p < 0.001$. Thus, we have demonstrated a model that, while acquisition is still achieved through the action of dopamine, behavioural inhibition can be successfully modelled by introducing LTD in the OFC in response to serotonin release after a reward omission.

5 Discussion

Standard models of learning and extinction are inspired by reinforcement learning theory where the neuromodulator dopamine causes weight growth and decay during acquisition and extinction respectively [12,6]. However, this implies that a similar rate to relearn the association is required when the rewarded stimulus is reintroduced. While such a model is elegant in terms of the underlying theory [19], experimental results show that reacquisition is much quicker than the original acquisition [13,11]. In contrast to previous work we present a model based on serotonergic action which inhibits actions rather than removing unnecessary learned behaviour so that, when contingencies change and the previously irrelevant behaviour becomes useful again, it is no longer suppressed and can very quickly be reinstated.

Central to our model is the interplay of the dopaminergic system with the sertonergic system. This interplay is not completely symmetrical but differs in terms of target areas. Dopamine activity is essential for mediating plasticity in the NAc and has been implemented as an error signal in numerous computational models. In previous models [12,6], the error is calculated in the VTA and

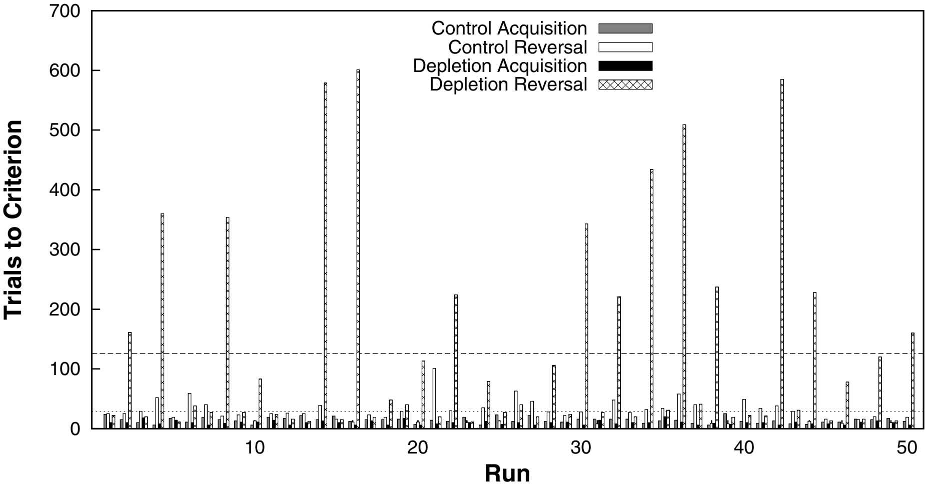


Fig. 5. Responses required to reach the criterion on each acquisition and reversal stage, both for the serotonin depleted model and the non-depleted control. The x-axis shows the run number for each trial, while the horizontal lines show the mean of the control reversal runs (lower line) and the mean of the depleted reversal runs (upper line).

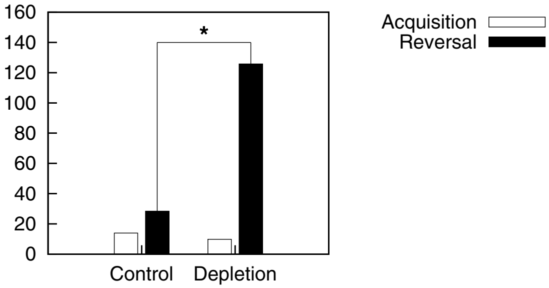


Fig. 6. Mean number of trials taken to reach the criterion of each stage on each model. *The result of a Hotelling t^2 test on the two sets of reversal data gave $p < 0.001$.

delivered globally so that weights increase or decrease in an identical manner depending on its value. As mentioned before this would imply that initial learning and re-learning had the same learning speed. To avoid this problem we use only the positive prediction error, namely dopamine elevation to drive LTP (with the option to also to code motivation) in all target areas.

In contrast to dopamine, serotonin is responsible for LTD however not in all target areas so that learned behaviour can be preserved but at the same time suppressed. Serotonin causes LTD only in the OFC but not in the dmPFC or NAc core.

The mechanism for suppression of behaviour in response to reward omission is based on our 'value control' model [20]. Classical reinforcement learning consists of a critic and an actor. The critic usually causes changes in the actor by either activating or removing actions. In our value control model, the critic does not remove actions but rather suppresses them by a gating mechanism. Central to the critic is the generation of the value of an action which we identify with the NAc shell and OFC whereas the action system is represented by the dorsal parts of the striatum starting from the NAc core to the basal ganglia. Here simply use the NAc core for generating actions but the concept could be extended to the basal ganglia, since this is also modulated by the same shell-VP-MD-dmPFC-core pathway [21].

In our model we consider plasticity in the OFC, where neural activity is correlated with choice activity [15]. An extended model would also consider the role of the amygdala where activity also reflects contingencies [15] and which has been modelled as interacting with the OFC [10]. Table 1 summarises the functional regions of our system. The first column shows the regions in the limbic system namely NAc core/dmPFC and NAc shell/OFC. The second column show the corresponding computational roles namely action and value. The value is then used to generate the prediction error by taking the positive part of its derivative and the actor the enables actions. The last column then shows the actions of dopamine and serotonin on the target structures. While dopamine can cause LTP in all target areas, serotonin causes only LTD in the OFC.

Table 1. The action of dopamine and serotonin in different brain regions

Brain Region	System	DA	5HT
OFC/NAc shell	value	LTP	LTD
dmPFC/NAc core	action	LTP	–

6 Model Equations

The following equations show how the US and CS signals are defined from the proximal and distal signals from the environment and how these are used to define basic OFC functionality. $f_{stimulus}$ is lowpass filter tuned to remove sharp transitions in the raw signals.

$$X_{US} = f_{stimulus}(X\text{-}proximal) \tag{1}$$

$$X_{CS} = MD \cdot f_{stimulus}(X\text{-}distal) \tag{2}$$

$$LH = f_{stimulus}(reward) \tag{3}$$

$$OFC\text{-}shell_x = \alpha_x \cdot f_{stimulus}(X\text{-}distal) \tag{4}$$

$$OFC\text{-}DRN = \alpha_{red} \cdot R^3_{US} + \alpha_{blue} B^3_{US} \tag{5}$$

The following equations define the activity of the other brain regions within the model. The ρ and ω variables represent synaptic weights and are initially set to zero. Set parameters are: $\zeta = 2, \kappa = 6.325, \nu = 2.45, \eta = 0.1, \psi = 3, \theta_{MD} = 2, \lambda = 0.5, o = 3$.

$$core_{red} = R_{US} + (R_{CS} \cdot \rho_{red,red}) + (B_{CS} \cdot \rho_{red,blue}) - \lambda \cdot core_{blue} \tag{6}$$

$$core_{blue} = B_{US} + (R_{CS} \cdot \rho_{blue,red}) + (B_{CS} \cdot \rho_{blue,blue}) - \lambda \cdot core_{red} \tag{7}$$

$$shell = LH + OFC\text{-}shell_{red} + OFC\text{-}shell_{blue} \tag{8}$$

$$VP = \frac{1}{1 + \zeta \cdot shell} \tag{9}$$

$$MD = \theta_{MD}(1 - VP) \tag{10}$$

$$LHb = \psi \cdot LH \tag{11}$$

$$VTA = \frac{\kappa \cdot LH}{1 + \nu \cdot VP + \eta \cdot shell} - \frac{\kappa}{1 + \nu} \tag{11}$$
[1]

$$DRN = \frac{o \cdot OFC\text{-}DRN}{1 + LHb} \tag{12}$$

The next set of equations define the dopamine burst and serotonin signals. Set parameters: $\chi_{burst} = 1, \gamma = -50$. The *depletion* factor was 0.6 for the depleted model and 1 otherwise. Finally, at the end of each time step within the simulation the synaptic weights are updated according to the following equations. Set parameters: $\mu_{core} = 3, \mu_{OFC} = 6,$[2] $\epsilon = 1.3 \cdot 10^{-2}, limit_{core} = 0.85, limit_{OFC} = 1$.

$$DA\text{-}burst = \chi_{burst} \cdot (1 - e^{\gamma \cdot VTA''}) \tag{13}$$

$$serotonin = depletion \cdot f_{serotonin}(DRN) \tag{14}$$

$$\rho_{x,x} := \rho_{x,x} + \mu_{core} \cdot X_{CS} \cdot core_x{}' \cdot DA\text{-}burst \cdot (limit_{core} - \rho_{x,x}) \tag{15}$$

$$\begin{aligned} \alpha_x := \alpha_x &+ \mu_{OFC} \cdot X_{US} \cdot X_{US}{}' \cdot DA\text{-}burst \cdot (limit_{OFC} - \alpha_x) \\ &- \epsilon \cdot X_{US} \cdot serotonin \end{aligned} \tag{16}$$

References

1. Bianco, I.H., Wilson, S.W.: The habenular nuclei: a conserved asymmetric relay station in the vertebrate brain. Philos. Trans. R. Soc. Lond., B, Biol. Sci. 364(1519), 1005–1020 (2009)
2. Brown, J., Bullock, D., Grossberg, S.: How the basal ganglia use parallel excitatory and inhibitory learning pathways to selectively respond to unexpected rewarding cues. J. Neurosci. 19(23), 10502–10511 (1999)
3. Clarke, H.F., Dalley, J.W., Crofts, H.S., Robbins, T.W., Roberts, A.C.: Cognitive inflexibility after prefrontal serotonin depletion. Science 304(5672), 878–881 (2004)
4. Clarke, H.F., Walker, S.C., Crofts, H.S., Dalley, J.W., Robbins, T.W., Roberts, A.C.: Prefrontal serotonin depletion affects reversal learning but not attentional set shifting. J. Neurosci. 25(2), 532–538 (2005)

[1] Additional connections to the VTA from the shell and VP serve to inhibit phasic dopamine in response to an expected reward and create spikes at the onset of a learned CS. These are not essential for the operation of this model but are included for completeness. Please refer to the Thompson model [20] for further information.

[2] Learning rates are very high due to a very short burst duration which was used for simplicity.

5. Clarke, H.F., Walker, S.C., Dalley, J.W., Robbins, T.W., Roberts, A.C.: Cognitive inflexibility after prefrontal serotonin depletion is behaviorally and neurochemically specific. Cereb. Cortex 17(1), 18–27 (2007)
6. Dayan, P.: Motivated reinforcement learning. In: Advances in Neural Information Processing Systems 13 (2001)
7. Dayan, P., Huys, Q.J.M.: Serotonin in affective control. Annual Review of Neuroscience 32, 95–126 (2009)
8. Grace, A.A., Floresco, S.B., Goto, Y., Lodge, D.J.: Regulation of firing of dopaminergic neurons and control of goal-directed behaviors. Trends Neurosci. 30(5), 220–227 (2007)
9. Lapiz-Bluhm, M.D.S., Soto-Piña, A.E., Hensler, J.G., Morilak, D.A.: Chronic intermittent cold stress and serotonin depletion induce deficits of reversal learning in an attentional set-shifting test in rats. Psychopharmacology (Berl.) 202(1-3), 329–341 (2009)
10. Moren, J., Balkenius, C.: A computational model of emotional learning in the amygdala. In: From Animals to Animats 6, pp. 383–391 (2000)
11. Napier, R.M., Macrae, M., Kehoe, E.J.: Rapid reaquisition in conditioning of the rabbit's nictitating membrane response. J. Exp. Psychol. Anim. Behav. Process. 18(2), 182–192 (1992)
12. O'Reilly, R.C., Frank, M.J., Hazy, T.E., Watz, B.: Pvlv: the primary value and learned value pavlovian learning algorithm. Behav. Neurosci. 121(1), 31–49 (2007)
13. Pavlov, I.P.: Conditioned reflexes. Oxford University Press, Oxford (1927)
14. Porr, B., Wörgötter, F.: Learning with "relevance": using a third factor to stabilize hebbian learning. Neural Computation 19(10), 2694–2719 (2007)
15. Schoenbaum, G., Chiba, A.A., Gallagher, M.: Neural encoding in orbitofrontal cortex and basolateral amygdala during olfactory discrimination learning. J. Neurosci. 19(5), 1876–1884 (1999)
16. Schoenbaum, G., Roesch, M.R., Stalnaker, T.A., Takahashi, Y.K.: A new perspective on the role of the orbitofrontal cortex in adaptive behaviour. Nat. Rev. Neurosci. 10(12), 885–892 (2009)
17. Schotanus, S.M., Chergui, K.: Dopamine d1 receptors and group i metabotropic glutamate receptors contribute to the induction of long-term potentiation in the nucleus accumbens. Neuropharmacology 54(5), 837–844 (2008)
18. Stern, W.C., Johnson, A., Bronzino, J.D., Morgane, P.J.: Effects of electrical stimulation of the lateral habenula on single-unit activity of raphe neurons. Exp. Neurol. 65(2), 326–342 (1979)
19. Sutton, R.S., Barto, A.G.: Reinforcement learning: an introduction. MIT, Cambridge (1998)
20. Thompson, A.M., Porr, B., Wörgötter, F.: Learning and reversal learning in the subcortical limbic system: A computational model. Adaptive Behavior 18(3-4), 211 (2010)
21. Zahm, D.S.: An integrative neuroanatomical perspective on some subcortical substrates of adaptive responding with emphasis on the nucleus accumbens. Neurosci. Biobehav. Rev. 24(1), 85–105 (2000)
22. Zhong, P., Liu, W., Gu, Z., Yan, Z.: Serotonin facilitates long-term depression induction in prefrontal cortex via p38 mapk/rab5-mediated enhancement of ampa receptor internalization. J. Physiol (Lond.) 586(pt. 18), 4465–4479 (2008)

Which Temporal Difference Learning Algorithm Best Reproduces Dopamine Activity in a Multi-choice Task?

Jean Bellot[1], Olivier Sigaud[1], and Mehdi Khamassi[1,2]

[1] Institut des Systemes Intelligents et de Robotique (ISIR), Universite Pierre et Marie Curie (UPMC), 4 place Jussieu, 75005 Paris, France
[2] UMR 7222, Centre National de la Recherche Scientifique (CNRS), France
{jean.bellot,olivier.sigaud,mehdi.khamassi}@isir.upmc.fr

Abstract. The activity of dopaminergic (DA) neurons has been hypothesized to encode a reward prediction error (RPE) which corresponds to the error signal in *Temporal Difference (TD) learning* algorithms. This hypothesis has been reinforced by numerous studies showing the relevance of *TD learning* algorithms to describe the role of basal ganglia in classical conditioning. However, recent recordings of DA neurons during multi-choice tasks raised contradictory interpretations on whether DA's RPE signal is action dependent or not. Thus the precise TD algorithm (i.e. Actor-Critic, Q-learning or SARSA) that best describes DA signals remains unknown. Here we simulate and precisely analyze these TD algorithms on a multi-choice task performed by rats. We find that DA activity previously reported in this task is best fitted by a *TD error* which has not fully converged, and which converged faster than observed behavioral adaptation.

Keywords: dopamine, reinforcement learning, reward prediction error, behavioral adaptation, instrumental conditioning.

1 Introduction

The work of Wolfram Schultz and colleagues during the 90s has highlighted the link between the information carried by the activity of dopaminergic (DA) neurons and the error signal computed by *Temporal Difference (TD) learning* algorithms [1,2,3]. However, most experiments involved *Pavlovian conditioning*, where the animal remains passive during the 2 seconds delay between the stimulus and the reward. In contrast, several different *TD learning* algorithms have been proposed with a different way of encoding the choice of actions (i.e. Actor-Critic, Q-learning, SARSA) and which cannot be discriminated based on these data [4].

More recent studies have focused on DA activity during multi-choice tasks, where animals learn to perform the right actions in order to obtain reward [5,6]. This enables to investigate whether the RPE signal in DA neurons is action-dependent or only depends on the conditioned stimuli. With these recent

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 289–298, 2012.

protocols, one can compare the ability of different *TD learning* algorithms to reproduce the activity patterns of DA neurons. However, these studies convey contradictory conclusions.

In [5], monkeys had to choose among two conditioned stimuli presented on a screen, each stimulus being associated to reward with a different probability. This time, the RPE carried by recorded DA neurons appeared to depend on the action the animal would subsequently perform. This RPE signal appeared to be consistent with the SARSA algorithm.

In [6], rats had to choose between two wells delivering two different rewards (large versus small reward; or delayed versus immediate reward). In each trial, an odor (conditioned stimulus) was presented, carrying the information enabling to identify which reward was available in each well. The progressive learning of stimulus-reward associations and changes in these associations enabled to analyze the type of RPE that was encoded by DA neurons during the task. The authors found that DA neurons are encoding an error depending on the value of the current best option, not matter whether the rat would subsequently perform that option or select the wrong option. Such type of RPE is compatible with the Q-learning algorithm.

Thus, the conclusions of both studies are inconsistent. But none of them did attempt to compare DA activity with empirical simulations of the algorithms. Here, we simulate diverse basic *TD learning* algorithms in order to determine which of them best reproduces the results obtained by [6]. We first describe the model used to simulate the multi-choice task studied in [6] and the *TD learning* algorithms. Then we focus on reproducing the behavioral data and the dopamine activity depending on the meta-parameters of the algorithms.

2 Material and Methods: Computational Model

In this section, we first describe the computational model used to simulate the task of [6]. Then we present the three *TD learning* algorithms introduced in [4].

2.1 Modelling the Blocks

We have modelled the experiments of [6] with a Markov Decision Process (MDP) (see Fig.1): in a given block of trials one well (right or left) delivers the best reward, the other contains the worst one (big vs small in the size condition; immediate versus delayed in the delay condition). After nosepoking, the animal receives one of three odors: odor 1 informs it that only the left well delivers reward; with odor 2 only the right well is rewarding; odor 3 indicates a free-choice trial where the animal is rewarded everywhere but needs to find the best reward.

At each block change, there is a shift in the well that delivers the best reward and the animal learns the new place-reward association. In this work, we only present simulations of free-choice trials (where odor 3 is presented) which are crucial to discriminate between the competing algorithms.

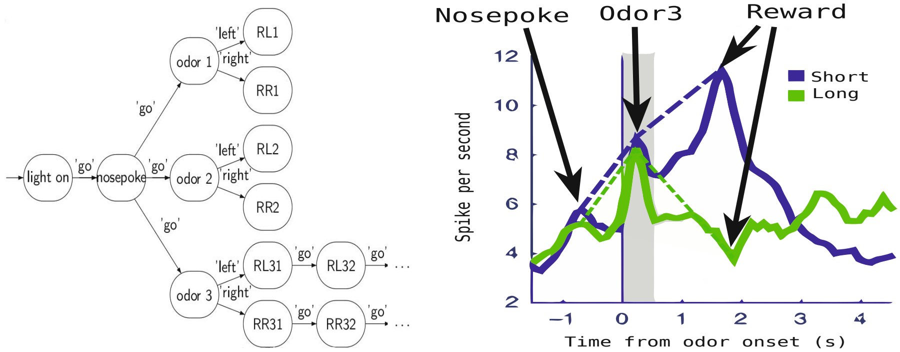


Fig. 1. Modelling the state of the task used in [6]. Left: Markov Decision Process used to model the task; RL31, Reward Left following odor 3; RR31, Reward Right following odor 3. The other states represent the delay. Right: State decomposition illustrated on DA activity reported by Roesch. We use the RPE calculated by the different algorithms at the three different states : 'nosepoke', 'odor3' and 'RL31' or 'RR31' (depending on the choice of the algorithm) to fit the DA activity extracted from the right graph.

The DA activity in Fig. 1 Right was obtained in this task by averaging the DA activity over all trials once the performance of rats went above 50%. The high response to the odor cue, independent from the future choice, is interpreted by the authors as consistent with a *TD error* calculated by Q-LEARNING [6]. However, since the behavioral performance converge quickly, the RPE should also have converged towards 0 at the time of the reward, as observed in Schultz's work [1]. This is not the case of this DA activity. It rather looks like an error that has not converged yet. Thus we simulate here the three concurrent TD-learning algorithms to empirically evaluate the nature of the RPE signal encoded by this DA activity.

2.2 Studied Algorithms

Our study of the performance of all algorithms is focused on the match between the evolution of the *TD error* and the DA activity. We compare three algorithms: Q-LEARNING, SARSA and ACTOR-CRITIC. Q-LEARNING and SARSA are based on the same principles. They update for each (s, a) pair a Q-table that stores the utility expectation for performing action a in state s. The ACTOR-CRITIC architecture contains a critic, i.e. a model of the value function V that stores the utility expectation from each state s, and an actor, the policy P which associates to any (s, a) pair the probability of choosing action a in state s (i.e. $P(a|s)$).

These value functions are updated from the *TD error* δ using $\forall f \epsilon \{Q, V, P\}$ $f_{t+1} = f_t + \alpha\delta_t$. But the computation of the *TD error* differs depending on the algorithm. The update rule is:

- Q-learning: $\delta_t = r_{t+1} + \gamma \max_a (Q(s_{t+1}, a)) - Q(s_t, a_t)$
- Sarsa: $\delta_t = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$
- Actor-Critic: $\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$

For action selection, we use a *softMax* policy which chooses an action with a probability proportional to the value of this action: $P(a|s_t) = \frac{\exp(\beta Q(s_t,a))}{\sum_b \exp(\beta Q(s_t,b))}$.

3 Reproduction of Behavioral Results

With the model described above we first fit the behavioral data from [6] to determine which set of parameters can better reproduce the learning dynamics of the rats.

3.1 Methods

In order to reproduce the behavioral results of [6], the simulated agent learns during 30 trials of a block where the best reward is on the left (block 1 or 4) using the above algorithms. From these trials, the agent learns the block (it goes more often to the best reward). After this initial learning stage, a block change occurs where the side of the best reward is reversed (right instead of left). The behavior of the agent is compared to that of the animals from 15 trials before the block change to 30 trials after. The performance is computed as the number of left choices. It is averaged over 100 simulated agents.We test different meta-parameter sets of the algorithms:

- *max_iter*: 100
- α: from 0.1 to 0.9 with 0.1 steps,
- β: from 0.1 to 0.9 with 0.1 steps,
- γ: $\epsilon[0.6, 0.7, 0.8, 0.9]$.
- After empirical tuning, we set the reward value to 5.

The obtained results are compared to those of [6] (see Fig. 2) by minimizing the squared error between a set of points over the curves. The points are extracted from the curves of [6] in the *size* case and the *delay* case. Then we look for the set of meta-parameters that optimize the match in both cases (see Fig. 2 for the *delay* case).

3.2 Results

Figure 2 Left shows the reproduction of the behavior with Q-learning, Sarsa and Actor-Critic. We obtain the following optimal meta-parameter sets for the different algorithms:

- Q-learning: $\alpha = 0.3, \beta = 0.3, \gamma = 0.7$
- Sarsa: $\alpha = 0.3, \beta = 0.3, \gamma = 0.7$
- Actor-Critic: $\alpha = 0.7, \beta = 0.8, \gamma = 0.7$

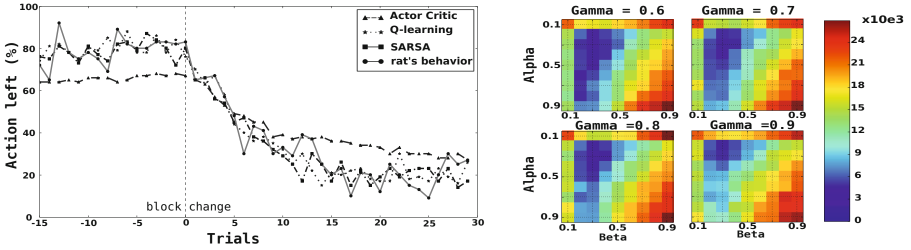


Fig. 2. Reproducing the behavior of the rat for the *delay* case with Q-LEARNING, SARSA and ACTOR-CRITIC. Left: reproduction of the behavior using the parameters obtained from the best fit with the behavioral data of Roesch et al. Right: squared error as a function of α, β and γ illustrated for the case of Q-LEARNING.

Figure 2 Right shows the squared error as a function of α, β and γ for the *delay* case (*size* case not shown). One can see that *Qlearning* and SARSA show a similar sensitivity to the parameters. The error is lower for α and β close to 0.3. ACTOR-CRITIC requires a larger α and β. As can be seen on Fig 2 , the smallest error for ACTOR-CRITIC is obtained for a β value near the tested limit (0.9). Thus we additionally test higher β values for ACTOR-CRITIC (1, 1.5 and 2). This does not change the results: the same parameter set enables the ACTOR-CRITIC model to give the best compromise between fitting the behavior during the delay condition and fitting the behavior during the size condition ($\alpha = 0.7$, $\beta = 0.8$ and $\gamma = 0.7$).

4 Comparing DA Activity with *TD Error*

In this section, we describe how we compare the DA activity to the *TD error* depending on the algorithms' meta-parameters.

4.1 Methods

Based on the parameters obtained from the behavioral results, we then investigate whether the *TD error* computed in simulation can match the DA activity observed in rats. The idea is that, if DA activity reflects the RPE signal of the learning algorithm by which rats learn the task, algorithms tuned to fit the behavior should display the same pattern of activity as responses of DA neurons.

Thus we compare the reported DA activity with the *TD error* computed by the different algorithms. We fit three states of our MDP with three points of the experimental curve (see Fig. 1 Right), corresponding to moments where the rat: (1) touches the port, *nosepokes*; (2) receives an odor, *odor*; (3) receives the reward, *reward*.

The DA activity and the *TD error* do not share a common scale. Thus, we minimize with least square the difference $||(a\delta_s + b) - R_s||^2$ where R_s is the experimental DA activity in state s and δ_s is the average *TD error* computed in s over the different trials. Thus we have: $\delta_s = \frac{1}{n}\Sigma_{e=0}^{n}\delta_s(e)$, where n is the number of considered trials and $\delta_s(e)$ is the *TD error* computed from the e^{th} trial in s. The (a, b) pair is determined with the least square method.

4.2 Results

The curves in [6] are obtained by averaging the DA activity over all trials once the performance of rats is over 50%. In the *delay* case, this happens after the fifth trial after the block changed (see Fig. 2 Left). The *TD error* should have converged towards 0 over learning. This is not the case of the DA activity in [6]. It rather looks like an error that has not converged yet because the reported response of DA neurons to rewards does not vanish with learning. Thus we look for a temporal window where the *TD error* may behave like the DA activity recorded in [6]. More precisely, we vary the number of trials considered in our average on δ_s so as to match the DA activity as well as we can. Fig. 3 Left shows the squared error as a function of this number of trials with Q-LEARNING.

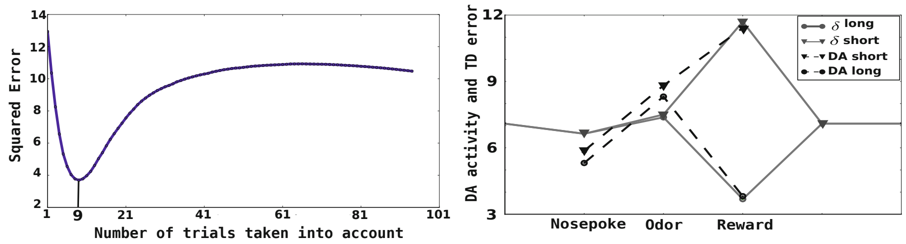


Fig. 3. Left: Evolution of the error obtained when fitting DA activity with the *TD error* in function of the number of trials taken into account in the calculation of the averaged RPE. Right: Best fit of the DA activity recorded in [6] during the delay case, from the *TD error* computed with the Q-LEARNING algorithm and averaged over the first 9 trials (minimizing the error as shown in Left).

The results are consistent with the conclusions of [6], since they show that, in the *delay* case, Q-LEARNING is the algorithm that best matches the DA activity. However, this is the case only if we just consider the 9 first trials after the performance got over 50%. When we take all the 20 *free-choice* trials used in [6], the error is much larger. The same applies to SARSA. But SARSA has different errors for the two odors, which is not observed in DA activity.

In the ACTOR-CRITIC case, the high fitting error is mainly due to the mismatch at the nosepoke state. Like other algorithms, the ACTOR-CRITIC's RPE signal which best fits DA activity is produced by a learning process that has already converged (even more strongly converged due to the high α, see Fig. 2). Thus the RPE signal is almost flat and requires a high amplification factor a to be compared to DA activity. This high a amplifies the noise at the nosepoke state.

In Table 1, we report the error of each algorithm as well as the optimal number of trials n to be considered when computing the average *TD error*. In the *size* case, none of the algorithms obtains a low error. Thus this case is reproduced worse than the *delay* case.

Table 1. Squared error (e^2) and percentage of error (e%) obtained with the different algorithms with respect to the value in [6]. This latter value is computed as $e\% = \frac{1}{n}\Sigma_i \frac{|d_i - s_i|}{d_i}$ where d_i is the value on [6]'s curve at instant i, s_i is the value obtained in simulation and n is the number of points.

	size			delay		
Algorithm	n	e^2	$e\%$	n	e^2	$e\%$
Qlearning	4	8.2	14.6%	9	3.8	10.7%
Sarsa	3	8.2	14.5%	3	9	16.9%
Actor-Critic	1	10.1	15.5%	41	8.3	16%

In summary, the results show that although Q-learning obtains better results, as expected by [6], the three algorithms, when fitted on the rat's behavior, have too much converged to reproduce the observed pattern of responses of DA neurons. This suggests that the rate with which behavior is adapted may be different from the rate with which the RPE signal encoded by DA neurons is learned, as if their responses were not tightly linked to the behavior. To assess this simple interpretation, we next test the algorithms after releasing the constraint on the fit with the behavior.

4.3 Optimization of Parameters over DA Activity Only

So far, we have used the same parameters for reproducing behavioral results and DA activity, considering that the behavior of the rat was directly driven by the *TD error*. This assumption was consistent with other studies in the literature [7,8]. Nevertheless, from Fig. 2, it is clear that the DA activity cannot be fitted with a *TD error* that has converged, whereas the behavior itself has converged.

In order to test the assumption that DA activity may reflect a learning dynamics slower than the one reflected in behavior, we now fit this activity with the model without constraining the meta-parameters on the behavioral data. However, we restrict the matching process to the trials where the behavior of the rat is above 50% of correct choice. Thus we cannot avoid at least a minor influence of the behavior in this fitting process.

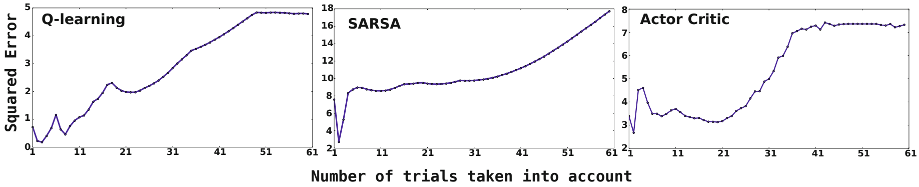


Fig. 4. Squared error as a function of the number of trials, between the DA activity from [6] and the *TD error* computed from different algorithms with free parameters. Left: Q-LEARNING, Middle: SARSA, Right: ACTOR-CRITIC.

Under this new condition, we obtain a better fit than previously (see Fig.4). These results show a large difference between the algorithms, in terms of their capability to reproduce the DA activity as a function of the parameters. As previously, Q-LEARNING can fit DA data. SARSA cannot do so as well as Q-LEARNING. Finally, ACTOR-CRITIC obtains a better performance than for previous results, performing comparably with Q-LEARNING. Indeed, if we only consider the 20 first trials (corresponding to the number of *free-choice* used in [6]), then the error and the corresponding meta-parameters are given in Table 2.

Table 2. Meta-parameters when fitting only with DA activity

	Qlearning				SARSA				ACTOR-CRITIC			
	α	β	γ	error	α	β	γ	error	α	β	γ	error
size	0.8	0.9	0.6	7.2	0.1	0.1	0.9	8.1	0.2	0.9	0.7	9.6
delay	0.1	0.6	0.9	2.0	0.1	0.5	0.9	9.5	0.1	0.1	0.9	3.4

Globally, one can see that to get a minimal error with respect to DA activity with the three algorithms in this task, the meta-parameters have to differ from the ones used to match behavioral results. In particular, the learning rate must be lower so that the value does not converge too quickly. One can conclude that the DA activity is compatible with a *TD error* computed by Q-LEARNING or ACTOR-CRITIC in the *delay* case. SARSA is a much less likely candidate algorithm with these data. In the *size* case, the three algorithms still cannot reproduce DA activity satisfyingly (see Table 2) which are discussed below.

5 Discussion

In this work, we have tried to fit DA activity observed during a multi-choice task with various RL algorithms. The starting hypothesis, initially resulting from DA recordings in passive monkeys, was that the response of these neurons would encode an RPE similar to the error signal used in RL [1].

Here we studied the link between the information carried by DA neurons recorded by [6] and the error computed by Q-learning, Sarsa and Actor-Critic algorithms in a task where animals are to perform active action selection. We found that none of these algorithms could satisfyingly reproduce the observed patterns of responses when keeping the behavioral parameters. However, when the learning rate of behavioral adaptation and the learning rate of the adaptation of the expected value were dissociated, we found that Q-learning and Actor-Critic could both reproduce DA activity during the *delay* condition while Sarsa could not.

The *size* condition remains problematic because after a block change, DA activity reported by [6] does not reflect the reversal of the contingencies: instead of displaying a negative RPE when the reward is worst than previously expected, the response of DA neurons to the small reward remains high. This pattern prevents the standard RL algorithms from reproducing neural activity.

Another important issue is the global increase of DA activity along the trial, getting higher when time gets close to reward delivery (see Fig. 6 in [6]). At first glance, this could look like a learned value function instead of an RPE. This possible confusion between value and RPE is reflected by the frequent usage of the term ”value” instead of ”RPE” in the original article [6]. It could be interesting to see whether the simulated value function of the tested algorithm can contribute to the reproduction of DA activity in this task. However, this would be inconsistent with the now well established theory that the phasic responses of DA neurons encode an RPE [1,9,10,11,12,13].

The alternative interpretation that we propose and whose plausibility is confirmed by our empirical simulations is that the DA signal recorded by [6] may correspond to an RPE that has not yet fully converged while the animals behavior has already converged. In our simulations, Q-learning and Actor-Critic algorithms could fit DA activity during the delay condition when the simulated error signal was averaged only during early learning trials. The fact that we cannot fit DA activity when considering all post-learning trials seems to reveal that the observed choice behavior of the animal has a different dynamics of adaptation than the learning process encoded by DA neurons. First, this suggests that instead of only reporting the averaged post learning DA activity, showing the trial-by-trial evolution of DA's response accross learning may be more informative, and may lead to different conclusions on which algorithm among Actor-Critic, Q-learning and SARSA best describes the activity. Second, this suggests that the observed behavior is not the direct consequence of a unique learning system that we suppose relies on the recorded DA activity. This could indicate the presence of a second parallel decision system which speeds up the behavioral adaptation: the behavior would result from the influence of a cortically-driven fast learning process while the slower habitual learning subserved via DA neurons in the striatum would take more time to converge. This idea would be consistent with the proposal of dual decision-making systems subserving parallel learning processes for goal-directed behaviors and habits in mammals [8,14]:

a fast model-based RL system combined with a slower model-free system such as TD-learning algorithms studied here.

In future work, it would be interesting to test the ability of such dual system model to explain both behavioral adaptation and DA activity reported in [6]. A simpler alternative explanation that we could compare would be that the Actor and the Critic underlying behavioral adaptation in this task may have different learning rates.

Acknowledgements. We are very grateful to Matthew R. Roesch and Geoffrey Schoenbaum for useful discussions. This work was supported by: L'Agence Nationale de la Recherche ANR-11-BSV4-0006 "LU2" (Learning Under Uncertainty) project.

References

1. Schultz, W., Dayan, P., Montague, P.R.: A neural substrate of prediction and reward. Science 275(5306), 1593–1599 (1997)
2. Hollerman, J.R., Schultz, W.: Dopamine neurons report an error in the temporal prediction of reward during learning. Nat. Neurosci. 1(4), 304–309 (1998)
3. Schultz, W.: Predictive reward signal of dopamine neurons. Journal of Neurophysiology 80(1), 1–27 (1998)
4. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press (March 1998)
5. Morris, G., Nevet, A., Arkadir, D., Vaadia, E., Bergman, H.: Midbrain dopamine neurons encode decisions for future action. Nat. Neurosci. 9(8), 1057–1063 (2006)
6. Roesch, M.R., Calu, D.J., Schoenbaum, G.: Dopamine neurons encode the better option in rats deciding between differently delayed or sized rewards. Nat. Neurosci. 10(12), 1615–1624 (2007)
7. Tanaka, S.C., Doya, K., Okada, G., Ueda, K., Okamoto, Y., Yamawaki, S.: Prediction of immediate and future rewards differentially recruits cortico-basal ganglia loops. Nature Neuroscience 7(8), 887–893 (2004)
8. Daw, N.D., Niv, Y., Dayan, P.: Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. Nat. Neurosci. 8(12), 1704–1711 (2005)
9. Bayer, H.M., Glimcher, P.W.: Midbrain dopamine neurons encode a quantitative reward prediction error signal. Neuron 47(1), 129–141 (2005)
10. Niv, Y., Daw, N.D., Dayan, P.: Choice values. Nature Neuroscience 9(8), 987–988 (2006)
11. Daw, N.D.: Dopamine: at the intersection of reward and action. Nat. Neurosci. 10(12), 1505–1507 (2007)
12. Niv, Y., Schoenbaum, G.: Dialogues on prediction errors. Trends in Cognitive Sciences 12(7), 265–272 (2008)
13. Matsumoto, M., Hikosaka, O.: Two types of dopamine neuron distinctly convey positive and negative motivational signals. Nature 459(7248), 837–841 (2009)
14. Keramati, M., Dezfouli, A., Piray, P.: Speed/Accuracy Trade-Off between the habitual and the Goal-Directed processes. PLoS Comput. Biol. 7(5), e1002055 (2011)

Multi-timescale Nexting in a Reinforcement Learning Robot

Joseph Modayil, Adam White, and Richard S. Sutton

Reinforcement Learning and Artificial Intelligence Laboratory
University of Alberta, Edmonton, Alberta, Canada

Abstract. The term "nexting" has been used by psychologists to refer to the propensity of people and many other animals to continually predict what will happen next in an immediate, local, and personal sense. The ability to "next" constitutes a basic kind of awareness and knowledge of one's environment. In this paper we present results with a robot that learns to next in real time, predicting thousands of features of the world's state, including all sensory inputs, at timescales from 0.1 to 8 seconds. This was achieved by treating each state feature as a reward-like target and applying temporal-difference methods to learn a corresponding value function with a discount rate corresponding to the timescale. We show that two thousand predictions, each dependent on six thousand state features, can be learned and updated online at better than 10Hz on a laptop computer, using the standard TD(λ) algorithm with linear function approximation. We show that this approach is efficient enough to be practical, with most of the learning complete within 30 minutes. We also show that a single tile-coded feature representation suffices to accurately predict many different signals at a significant range of timescales. Finally, we show that the accuracy of our learned predictions compares favorably with the optimal off-line solution.

1 Multi-timescale Nexting

Psychologists have noted that people and other animals seem to continually make large numbers of short-term predictions about their sensory input (e.g., see Gilbert 2006, Brogden 1939, Pezzulo 2008, Carlsson et al. 2000). When we hear a melody we predict what the next note will be or when the next downbeat will occur, and are surprised and interested (or annoyed) when our predictions are disconfirmed (Huron 2006, Levitin 2006). When we see a bird in flight, hear our own footsteps, or handle an object, we continually make and confirm multiple predictions about our sensory input. When we ride a bike, ski, or rollerblade, we have finely tuned moment-by-moment predictions of whether we will fall, and of how our trajectory will change in a turn. In all these examples, we continually predict what will happen to us *next*. Making predictions of this simple, personal, short-term kind has been called *nexting* (Gilbert, 2006).

Nexting predictions are specific to one individual and to their personal, immediate sensory signals or state variables. A special name for these predictions

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 299–309, 2012.

seems appropriate because they are unlike predictions of the stock market, of political events, or of fashion trends. Predictions of such public events seem to involve more cognition and deliberation, and are fewer in number. In nexting we envision that one individual may be continually making massive numbers of small predictions in parallel. Moreover, nexting predictions seem to be made simultaneously at multiple time scales. When we read, for example, it seems likely that we next at the letter, word, and sentence levels, each involving substantially different time scales.

The ability to predict and anticipate has often been proposed as a key part of intelligence (e.g., see Tolman 1951, Hawkins & Blakeslee 2004, Butz et al. 2003, Wolpert et al. 1995, Clark in press). Nexting can be seen as the most basic kind of prediction, preceding and possibly underlying all the others. That people and a wide variety of animals learn and make simple predictions at a range of short time scales in conditioning experiments was established so long ago that it is known as *classical conditioning* (Pavlov 1927). Predictions of upcoming shock to a paw may reveal themselves in limb-retraction attempts a fraction of a second before the shock, and as increases in heart rate 30 seconds prior. In other experiments, for example those known as *sensory preconditioning* (Brogden 1939, Rescorla 1980), it has been clearly shown that animals learn predictive relationships between stimuli even when none of them are inherently good or bad (like food and shock) or connected to an innate response. In this case the predictions are made, but not expressed in behaviour until some later experimental manipulation connects them to a response. Animals seem to just be wired to learn the many predictive relationships in their world.

To be able to next is to have a basic kind of knowledge about how the world works in interaction with one's body. It is to have a limited form of forward model of the world's dynamics. To be able to learn to next—to notice any disconfirmed predictions and continually adjust your nexting—is to be aware of one's world in a significant way. Thus, to build a robot that can do both of these things is a natural goal for artificial intelligence. Prior attempts to achieve artificial nexting can be grouped in two approaches.

The first approach is to build a *myopic* forward model of the world's dynamics, either in terms of differential equations or state-transition probabilities (e.g., Wolpert et al. 1995, Grush 2004, Sutton 1990). In this approach a small number of carefully chosen predictions are made of selected state variables with a public meaning. The model is myopic in that the predictions are only short term, either infinitesimally short in the case of differential equations, or maximally short in the case of the one-step predictions of Markov models. In these ways, this approach has ended up in practice being very different from nexting.

The second approach, which we follow here, is to use temporal-difference (TD) methods to learn long-term predictions directly. The prior work pursuing this approach has almost all been in simulation, and has used table-lookup representations and a small number of predictions (e.g., Sutton 1995, Kaelbling 1993, Singh 1992, Sutton, Precup & Singh 1999, Dayan and Hinton 1993). Sutton et al. (2011) showed real-time learning of TD predictions on a robot, but did not demonstrate the ability to learn many predictions in real time or with a single feature representation.

2 Nexting as Multiple Value Functions

We take a reinforcement-learning approach to achieving nexting. In reinforcement learning it is commonplace to learn long-term predictions of reward, called *value functions*, and to learn these using temporal-difference (TD) methods such as TD(λ) (Sutton 1988). However, TD(λ) has also been used as a model of classical conditioning, where the predictions are shorter term and where more than one signal might be viewed as a reward (Sutton & Barto, 1990). Our approach to nexting can be seen as taking this latter approach to the extreme of predicting massive numbers of target signals of all kinds at multiple time scales.

We use a notation for our multiple predictions that mirrors—or rather multiplies—that used for conventional value functions. Time is taken to be discrete, $t = 1, 2, 3, \ldots$, with each time step corresponding to approximately 0.1 seconds of real time. Our ith prediction at time t, denoted v_t^i, is meant to anticipate the future values of the ith prediction's target signal, r_t^i, over a designated time scale given by the discount-rate parameter γ^i. In our experiments, the target signal r_t^i was either a raw sensory signal or else a component of a state-feature vector (that we will introduce shortly), and the discount-rate parameter was one of four fixed values. The goal of learning is for each prediction to approximately equal the correspondingly discounted sum of the future values of the corresponding target signal:

$$v_t^i \quad \approx \quad \sum_{k=0}^{\infty} (\gamma^i)^k r_{t+k+1}^i \quad \overset{\text{def}}{=} \quad G_t^i. \tag{1}$$

The random quantity G_t^i is known as the *return*.

We use linear function approximation to form each prediction. That is, we assume that the state of the world at time t is characterized by the feature vector $\phi_t \in \mathbb{R}^n$, and that all the predictions v_t^i are formed as inner products of ϕ_t with the corresponding weight vectors θ_t^i:

$$v_t^i \quad = \quad \phi_t^\top \theta_t^i \quad \overset{\text{def}}{=} \quad \sum_j \phi_t(j) \theta_t^i(j), \tag{2}$$

where $\phi_t^\top$ denotes the transpose of ϕ_t (all vectors are column vectors unless transposed) and $\phi_t(j)$ denotes its jth component. The predictions at each time are thus determined by the weight vectors θ_t^i. One natural algorithm for learning the weight vectors is linear TD(λ):

$$\theta_{t+1}^i = \theta_t^i + \alpha \left(r_{t+1}^i + \gamma^i \phi_{t+1}^\top \theta_t^i - \phi_t^\top \theta_t^i \right) \mathbf{e}_t^i \tag{3}$$

where $\alpha > 0$ is a step-size parameter and $\mathbf{e}_t^i \in \mathbb{R}^n$ is an *eligibility trace* vector, initially set to zero and then updated on each step by

$$\mathbf{e}_t^i = \gamma^i \lambda \mathbf{e}_{t-1}^i + \phi_t, \tag{4}$$

where $\lambda \in [0, 1]$ is a trace-decay parameter.

Under common assumptions and a decreasing step-size parameter, TD(λ) with $\lambda = 1$ converges asymptotically to the weight vector that minimizes the mean squared error between the prediction and its return. In practice, smaller values of $\lambda \in [0, 1)$ are almost always used because they can result in significantly faster learning (e.g., see Sutton & Barto 1998), but the $\lambda = 1$ case still provides an important theoretical touchstone. In this case we can define an optimal weight value θ_*^i that minimizes the squared error from the return over the first N predictions:

$$\theta_*^i = \arg\min_\theta \sum_{t=1}^{N} \left(\phi_t^\top \theta - G_t^i\right)^2 . \tag{5}$$

This value can be computed offline by standard algorithms for solving large least-squares regression problems, and the performance of this offline-optimal value can be compared with that of the weight vectors found online by TD(λ). The offline algorithm is $O(n^3)$ in computation and $O(n^2)$ in memory, and thus is just barely tractable for the cases we consider here, in which $n = 6065$. Nevertheless, θ_*^i provides an important performance standard in that it provides an upper limit on one measure of the quality of the predictions found by learning. This upper limit is determined not by any learning algorithm, but by the feature representation. As we will see, even the predictions due to θ_*^i will have residual error. Thus, this analysis provides a method for determining when performance can be improved with more experience and when performance improvements require a better representation. Note that this technique is applicable even when experience is gathered from the physical world, where no formal notion of state is available.

3 Experimental Setup

We investigated the practicality of nexting on the Critterbot, a custom-designed robust and sensor-rich mobile robot platform (Figure 1, left). The robot has a diverse set of sensors and has holonomic motion provided by three omni-wheels. Sensors attached to the motors report the electrical current, the input motor voltage, motor temperature, wheel rotational velocities, and an overheating flag, providing substantial observability of the internal physical state of the robot. Other sensors collect information from the external environment. Passive sensors detect ambient light in several directions from the top of the robot in the visible and infrared spectrum. Active sensors emit infrared light and measure the reflectance, providing information about the distance to nearby obstacles. Other sensors report acceleration, rotation, and the magnetic field. In total, we consider 53 different sensor readings, all normalized to values between 0 and 1 based on sensor limits.

For our experiments, the agent's state representation was a binary vector, $\phi_t \in \{0, 1\}^n$, with a constant number of 1 features, constructed by tile coding (see Sutton & Barto 1998). The features provided no history and performed no

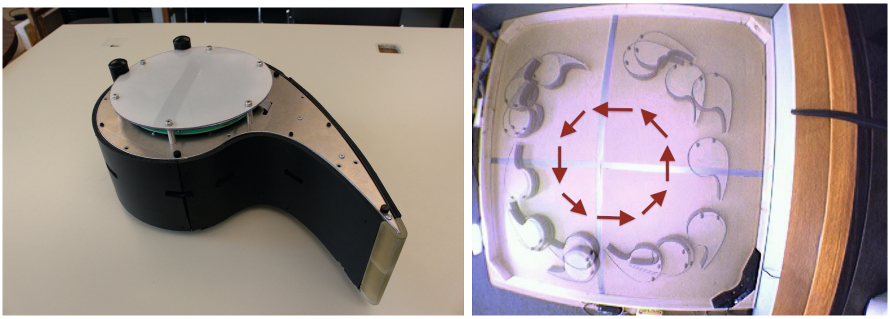


Fig. 1. Left: The Critterbot, a custom mobile robot with multiple sensors. Right: The Critterbot gathering experience while wall-following in its pen. This experience contains observations of both stochastic events (such as ambient light variations from the sun) and regular events (such as passing a lamp on the lower-left side of the pen).

averaging of sensor values. The sensory signals were partitioned based on sensor modalities. Within each sensor modality, each individual sensor (e.g., Light0) has multiple overlapping tilings at random offsets (up to 8 tilings), where each tiling splits the sensor range into disjoint intervals of fixed width (up to 8 intervals). Additionally, pairs of sensors within a sensor modality were tiled together using multiple two-dimensional overlapping grids. Pairs of sensors were jointly tiled if they were spatially adjacent on the robot (e.g., IRLight0 with IRLight1) or if there was a single sensor in between them (e.g., IRDistance1 with IRDistance3, IRDistance2 with IRDistance4, etc.). All in all, this tiling scheme produced a feature vector with $n = 6065$ components, most of which were 0s, but exactly 457 of which were 1s, including one bias feature that was always 1.

The robot experiment was conducted in a square wooden pen, approximately two meters on a side, with a lamp on one edge (see Figure 1). The robot's actions were selected according to a fixed stochastic wall-following policy. This policy moved forward by default, slid left or right to keep a side IRDistance sensor within a bounded range (50-200), and drove backward while turning when the front IRDistance sensor reported a nearby obstacle. The robot completed a loop of the pen approximately once every 40 seconds. Due to overheating protection, the motors would stop to cool down at approximately 14 minute intervals. To increase the diversity of the data, the policy selected an action at random with a probability $p = 0.05$. At every time step (approximately 100ms), sensory data was gathered and an action performed. This simple policy was sufficient for the robot to reliably follow the wall for hours, even with overheating interruptions.

The wall-following policy, tile-coding, and the TD(λ) learning algorithm were all implemented in Java and run on a laptop connected to the robot by a dedicated wireless link. The laptop used an Intel Core 2 Duo processor with a 2.4GHz clock cycle, 3MB of shared L3 cache, and 4GB DDR3 RAM. The system garbage collector was called on every time step to reduce variability. Four threads were used for the learning code. For offline analysis, data was also logged to disk for 120000 time steps (3 hours and 20 minutes).

4 Results

We applied TD(λ) to learn 2160 predictions in parallel. The first 212 predictions had the target signal, r_t^i, set to the sensor reading of one of the 53 sensors and the discount rate, γ^i, set to one of four timescales; the remaining 1948 predictions had the target signal set to one of 487 randomly selected components of the feature vector and the discount rate set to one of four timescales. The discount rates were one of the four values in $\{0, 0.8, 0.95, 0.9875\}$, corresponding to time scales of approximately 0.1, 0.5, 2, and 8 seconds respectively. The learning parameters were $\lambda = 0.9$ and $\alpha = 0.1/457(= \#$ of active features). The initial weight vector was set to zero.

Our initial performance question was scalability, in particular, whether so many predictions could be made and learned in real time. We found that the total computation time for a cycle under our conditions was 55ms, well within the 100ms duty cycle of the robot. The total memory consumption was 400MB. Note that with faster computers the number of predictions or the size of the weight and feature vectors could be increased at least proportionally. This strategy for nexting should be easily scalable to millions of predictions with foreseeable increases in parallel computing power over the next decade.

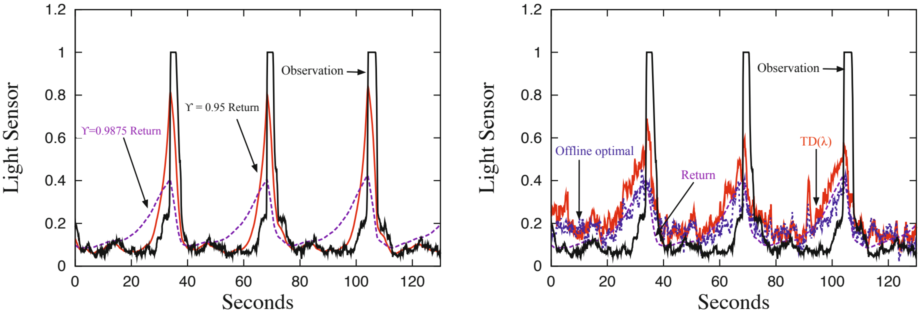


Fig. 2. Nexting is demonstrated in these graphs with predictions that rise and fall prior to the increase and decrease of a sensory signal. Comparison of ideal (left) and learned (right) predictions of one of the light sensors for three trips around the pen after 2.5 hours of experience. On each trip, the sensor value saturates at 1.0. The returns for the 2 and 8-second predictions, shown on the left, rise in anticipation of the high value, and then fall in anticipation of the low value. The 8-second predictions in the second panel of the offline-optimal weights (dotted blue line) and the TD(λ)-learned weights (solid red line) behave similarly both to each other and to the returns (albeit with more noise).

For an initial assesment of accuracy, let us take a close look at one of the predictions, in particular, at the prediction for one of the light sensors. Notice that there is a bright lamp in the lower left corner of the pen in Figure 1 (right). On each trip around the pen, the light sensor increases to its maximal level and then falls back to a low level, as shown by the black line in Figure 2. If the state features are sufficiently informative, then the robot may be able to anticipate the rising and falling of this sensor value. The ideal prediction is the return G_t^i, shown on the left in the colored lines in Figure 2 for two time scales (two seconds and eight seconds). Of course, to determine these lines, we had to use the future values of the light sensor; the idea here is to approximate these ideal predictions (as in Equation 5) using only the sensory information available to the robot in its feature vector. The second panel of the figure shows the predictions due to the weight vector adapted online by TD(λ) and due to the optimal weight vector, θ_*^i, computed offline (both for the 8-second time scale). The key result is that the robot has learned to anticipate both the rise and fall of the light. Both the learned prediction and the optimal offline prediction match the return closely, though with substantial noisy perturbations.

Figure 3 is a still closer look at this same prediction, obtained by averaging over 100 circuits around the pen, aligning each circuit's data so that the time of initial saturation of the light sensor is the same. We can now see very clearly how the predictions and returns anticipate both the rise and fall of the sensor value, and that both the TD(λ) prediction and the optimal prediction, when averaged, closely match the return.

Having demonstrated that accurate prediction is possible, we now consider the rate of learning in Figure 4. The graphs shows that learning is fast in terms of data (despite the large number of features), converging to solutions with low error in the familiar exponential way. This result is important as is demonstrates that learning online in real time is possible on robots with a few hours of experience, even with a large distributed representation. For contrast, we also show the learning curve for a trivial representation consisting only of a bias unit (the single feature that is always 1). The comparison serves to highlight that large informative feature sets are beneficial. The comparison to the predictive performance of the offline-optimal solution shows a vanishing performance gap by the end of the experiment. The second panel of the figure shows a similar pattern of decreasing errors for a sample of the 2160 TD(λ) predictions, showing that learning many predictions in parallel yields similar results.

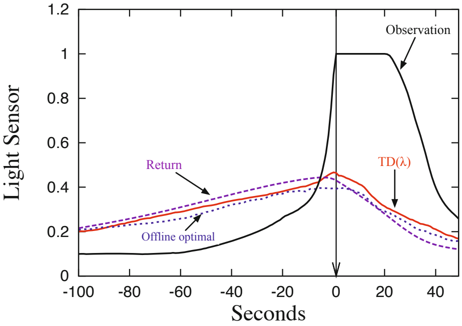


Fig. 3. An average of 100 cycles like the three shown in Figure 2 (right panel), aligned on the onset of sensor saturation. Error bars are slightly wider than the lines themselves and overlap substantially, so are omitted for clarity

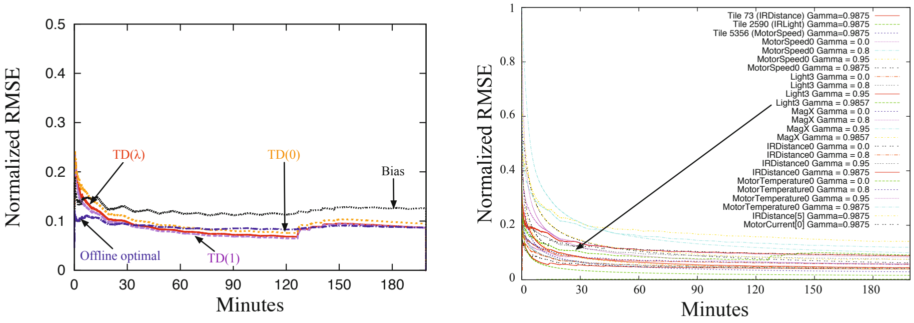


Fig. 4. Nexting learning curves for the 8-second light sensor predictions (left) and for a representative sample of the TD(λ) predictions (right). Predictions at different time scales have had their root mean squared error (RMSE) normalized by $\frac{1}{1-\gamma^i}$. The graph on the left is a comparison of different learning algorithms. The jog in the middle of the first graph occurs when the robot stops by the light to cool off its motors, causing the online learners to start making poor predictions. In spite of the unusual event, the TD(λ) solution still approaches the offline-optimal solution. TD(λ) performs similarly to a supervised learner TD(1), and they both slightly outperform TD(0). The curve for the bias unit shows the poor performance of a learner with a trivial representation. The graph on the right shows that seemingly all the TD(λ) predictions are learning well with a single feature representation and a single set of learning parameters.

A noteworthy result is that the same learning parameters and representation suffice for learning answers to a wide variety of nexting predictions without any convergence problems. Although the answers continue to improve over time, the most dramatic gains were achieved after 30 minutes of real time.

5 Discussion

These results provide evidence that online learning of thousands of nexting predictions on a robot in parallel is possible, practical, and accurate. Moreover, the predictive accuracy is reasonable with just a few hours of robot experience, no tuning of algorithm parameters, and using a single feature representation for all predictions. The parallel scalability of knowledge-acquisition in this approach is substantially novel when compared with the predominately sequential existing approaches common for robot learning. These results also show that online methods can be competitive in accuracy with an offline optimization of mean squared error.

The ease with which a simple reinforcement learning algorithm enables nexting on a robot is somewhat surprising. Although the formal theories of reinforcement learning sometimes give mathematical guarantees of convergence, there is little guidance for the choice of features for a task, for selecting learning parameters across a range of tasks, or for how much experience is required before a reinforcement learning system will approach convergence. The experiments show that we can use the same features across a range of tasks, anticipate events before they occur, and achieve predictive accuracy approaching that of an offline-optimal solution with a limited amount of robot experience.

6 More General Nexting

The exponentially discounted predictions that we have focused on in this paper constitute the simplest kind of nexting. They are a natural first kind of predictive knowledge to be learned. Online TD-style algorithms can be extended to handle a much broader set of predictions, including time-varying choices of γ, time-varying λ, and even off-policy prediction (Maei & Sutton 2010). It has even been proposed that all world knowledge can be represented by a sufficiently large and diverse set of predictions (Sutton 2009).

As one example of such an extension, consider allowing the discount rate γ^i to vary as a function of the agent's state. The algorithmic modifications required are straightforward. In the definition of the return in Equation 1, $(\gamma^i)^k$ is replaced with $\Pi_{j=0}^{k}\gamma^i_{t+j}$. In Equation 3, γ^i is replaced with γ^i_{t+1} and finally, in Equation 4, γ^i is replaced with γ^i_t. Using the modified definitions, the robot can predict how much motor power it will consume until either the light sensor is saturated or approximately two seconds elapse. This prediction can be formalized by setting the prediction's target signal to be the sum of instantaneous power consumption of each wheel, ($r = \sum_{i=1}^{3}$ MotorVoltage$i$ × MotorCurrent$i$) and throttling gamma when the light sensor is saturated ($\gamma^i_t =$ 0.1 when the light sensor is saturated and 0.95 otherwise). The plots in Figure 5 shows that the robot has learned to anticipate how much power will be expended prior to reach the light or spontaneously terminating.

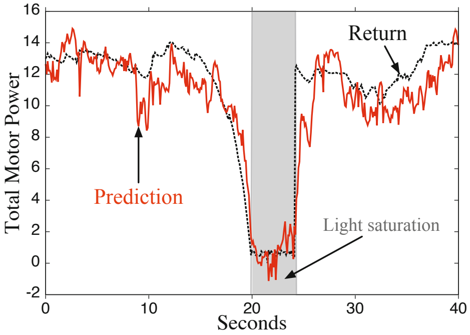


Fig. 5. Nexting can be extended, for example to consider time-varying gamma to predict of the amount of power that the robot will expend before a probabilistic pseudo-termination with a 2-second time horizon or a saturation event on the light sensor

7 Conclusions

We have demonstrated multi-timescale nexting on a physical robot; thousands of anticipatory predictions at various time-scales can be learned in parallel on a physical robot in real-time using a reinforcement learning methodology. This approach uses a large feature representation with an online learning algorithm to provide an efficient means for making parallel predictions. The algorithms are capable of making real-time predictions about the future of the robot's sensors at multiple time-scales using the computational horsepower of a laptop. Finally, and key to the practical application of our approach, we have shown that a single feature representation and a single set of learning parameters are sufficient for learning many diverse predictions. A natural direction for future work would be to extend these results to more general predictions and to control.

Acknowledgements. The authors thank Mike Sokolsky for creating the Critterbot and Patrick Pilarski and Thomas Degris for preparation of Figure 1 and for essential assistance with the experiment briefly reported in Section 6. This work was supported by grants from Alberta Innovates – Technology Futures, the National Science and Engineering Reseach Council of Canada, and the Alberta Innovates Centre for Machine Learning.

References

Brogden, W.: Sensory pre-conditioning. Journal of Experimental Psychology 25(4), 323–332 (1939)

Butz, M.V., Sigaud, O., Gérard, P. (eds.): Anticipatory Behavior in Adaptive Learning Systems. LNCS (LNAI), vol. 2684. Springer, Heidelberg (2003)

Carlsson, K., Petrovic, P., Skare, S., Petersson, K., Ingvar, M.: Tickling expectations: neural processing in anticipation of a sensory stimulus. Journal of Cognitive Neuroscience 12(4), 691–703 (2000)

Clark, A.: Whatever Next? Predictive Brains, Situated Agents, and the Future of Cognitive Science. Behavioral and Brain Sciences (in press)

Dayan, P., Hinton, G.: Feudal reinforcement learning. In: Advances in Neural Information Processing Systems 5, pp. 271–278 (1993)

Gilbert, D.: Stumbling on Happiness. Knopf Press (2006)

Grush, R.: The emulation theory of representation: motor control, imagery, and perception. Behavioural and Brain Sciences 27, 377–442 (2004)

Hawkins, J., Blakeslee, S.: On Intelligence. Times Books (2004)

Huron, D.: Sweet anticipation: Music and the Psychology of Expectation. MIT Press (2006)

Kaelbling, L.: Learning to achieve goals. In: Proceedings of International Joint Conference on Artificial Intelligence (1993)

Levitin, D.: This is Your Brain on Music. Dutton Books (2006)

Pavlov, I.: Conditioned Reflexes: An Investigations of the Physiological Activity of the Cerebral Cortex, translated and edited by Anrep, G.V. Oxford University Press (1927)

Pezzulo, G.: Coordinating with the future: The anticipatory nature of representation. Minds and Machines 18(2), 179–225 (2008)

Rescorla, R.: Simultaneous and successive associations in sensory preconditioning. Journal of Experimental Psychology: Animal Behavior Processes 6(3), 207–216 (1980)

Singh, S.: Reinforcement learning with a hierarchy of abstract models. In: Proceedings of the Tenth National Conference on Artificial Intelligence, pp. 202–207 (1992)

Sutton, R.S.: Learning to predict by the method of temporal differences. Machine Learning 3, 9–44 (1988)

Sutton, R.S.: Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In: Proceedings of the Seventh International Conference on Machine Learning, pp. 216–224 (1990)

Sutton, R.S.: TD models: Modeling the world at a mixture of time scales. In: Proceedings of the International Conference on Machine Learning, pp. 531–539 (1995)

Sutton, R.S.: The grand challenge of predictive empirical abstract knowledge. In: Working Notes of the IJCAI 2009 Workshop on Grand Challenges for Reasoning from Experiences (2009)

Sutton, R.S., Barto, A.G.: Time-derivative models of Pavlovian reinforcement. In: Learning and Computational Neuroscience: Foundations of Adaptive Networks, pp. 497–537. MIT Press (1990)

Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (1998)

Sutton, R.S., Modayil, J., Delp, M., Degris, T., Pilarski, P.M., White, A., Precup, D.: Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In: Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems, pp. 761–768 (2011)

Sutton, R.S., Precup, D., Singh, S.: Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence 112, 181–211 (1999)

Tolman, E.C.: Purposive Behavior in Animals and Men. University of California Press (1951)

Wolpert, D., Ghahramani, Z., Jordan, M.: An internal model for sensorimotor integration. Science 269(5232), 1880–1882 (1995)

Self-organizing Developmental Reinforcement Learning

Alain Dutech

LORIA - INRIA Campus Scientifique - BP 239
54506 Vandoeuvre les Nancy, France
alain.dutech@loria.fr

Abstract. This paper presents a developmental reinforcement learning framework aimed at exploring rich, complex and large sensorimotor spaces. The core of this architecture is made of a function approximator based on a Dynamic Self-Organizing Map (DSOM). The life-long online learning property of the DSOM allows us to take a developmental approach to learning a robotic task: the perception and motor skills of the robot can grow in richness and complexity during learning. This architecture is tested on a robotic task that looks simple but is still challenging for reinforcement learning.

1 Overview

The framework of reinforcement learning [1] is particularly attractive because it enables an artificial agent to learn an action plan to solve a problem that may be unfamiliar or uncertain while only using a scalar signal to distinguish between good and bad situations. Learning is not supervised and does not require an expert or an oracle. Moreover, Markov Decision Processes (MDP) [2] provide a formal background for analyzing theoretical properties of many algorithms, ensuring, for example, the existence of an optimal solution to the learning problem and convergence to this optimal solution.

Although reinforcement learning led to some successful applications, the practical use of reinforcement learning for realistic problems is not easy. Difficulties usually arise from the size of the problems to be solved that can only be described through a very large number of states, making them computationally intractable for reinforcement learning algorithms searching an exact solution. Algorithms providing approximate solutions do exist (approximate value iteration, LSPI, direct policy search through gradient ascent, *etc.*, see Chap. 3,4 of [3],[4]), but they are far from providing answers to all problems encountered in practice.

We are particularly interested in the use of reinforcement learning in the context of autonomous robotics. Several difficulties reduce the applicability of the learning algorithms. The sensor data are continuous in value, often noisy, expensive to obtain (in time and energy) and very (too) rich in information (*e.g.* in the case of a video stream). Our main concern is for the agent to be able to explore

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 310–319, 2012.

efficiently these rich and complex environments. We propose a *developmental* approach where the agent starts with crude actions and perceptions that gradually get more complex. Thus, starting from a smaller and simpler sensorimotor space, the agent should be able to learn simple tasks and, building over acquired behaviors, learn more complex tasks as its sensorimotor world gets richer. This developmental learning is made possible by using a self-organizing adaptive map architecture for function approximation in a reinforcement learning framework.

This paper first quickly presents the reinforcement learning framework used and original architecture for a developmental approach (Sections 2 and 3). Then, the experimental setting is detailed and results are given (Sections 4 and 5). Section 6 discusses the our work, pointing out some limitations and offering perspectives for future work.

2 Principles and Motivations

Our approach of developmental reinforcement learning is largely based on the *Q-learning* algorithm [5] and therefore relies on estimating the optimal value function Q^* in the sensorimotor space $\mathcal{S} \times \mathcal{A}$. The principle of the algorithm is to use each training sample (s_t, a_t, s_{t+1}, r_t) provided at time t through an interaction between the agent and its environment (the agent perceives s_t, chooses action a_t that changes the environment now perceived as s_{t+1}, and gets a reward r_t) to update the estimate of the value function Q^* for the pair (s_t, a_t) as follows:

$$Q^*_{t+1}(s_t, a_t) \longleftarrow Q^*_t(s_t, a_t) + \overbrace{\alpha\left(r + \gamma \max_{a' \in \mathcal{A}} Q^*_t(s_{t+1}, a') - Q^*_t(s_t, a_t)\right)}^{\Delta}, \qquad (1)$$

where α is a learning coefficient, which may depend on s, a and t.

A major limitation to the practical use of reinforcement learning methods is related to the complexity of algorithms. Theoretically, to guarantee the convergence of reinforcement learning algorithms to an optimal solution one requires that each $(s,\ a)$ pair is visited an infinite number of times. Even with only a very large number of visits, this constraint is one of the main limitations to the practical use of reinforcement learning techniques, and particularly in the context of robotics, and the only way to learn good behavior would be to explore intelligently and efficiently the environment.

The sensorimotor space of a robot is rich, continuous and complicated, furthermore it can be time and energy consuming to get a new training sample, and also risky (like bumping in a wall). As a result, an efficient and complete exploration of the unknown environment is very unlikely. The main idea of our developmental reinforcement learning framework is to ease the exploration by starting with a very small and limited sensorimotor space and gradually increasing it when the robot learned performances improve. Here, this support to the agent mainly takes the form of a gradual increase of perceptions richness and potential actions of the agent as its performance in the learning task is improving.

From this perspective, several problems arise. First, the learning architecture chosen must allow to progressively increase the sensorimotor space of the agent and the difficulty of the task. Second, we want the agent to rely on what he has already learned to learn in a richer environment. This last issue is particularly delicate, it is similar to the problem of knowledge transfer, a research field still wide open [6]. Within the framework of reinforcement learning, a central element of learning is the estimation of the value function of the sensorimotor space. The problems we have mentioned are related to estimating a function: how to take advantage of the current estimate of a function to extend it to a modified input space (when sensorimotor space is enriched).

3 Architecture for Developmental Learning

The core of our architecture is an adaptive function approximation to learn the *Q-value* function in a continuous sensorimotor space. The principle of our architecture is depicted in Fig. 1. Perceptions of the robot (we give more details on these perceptions in Section 4), denoted s, feed a Dynamic Self-Organizing Map (DSOM) [7]. The activity of this self-organizing map is the input of a perceptron with one layer of neurons that has as many outputs as possible actions. Thus, the output neuron associated with the action a learns the value function $Q(s, a)$ in a supervised way using the *Q-learning* update equation (1) a linear regression pattern, by modifying its weights ω using: $\omega \leftarrow \omega - \epsilon_L \varDelta$. The DSOM map uses unsupervised learning to adapt to the perceptual inputs received.

Although we have not used a more conventional architecture like LSTD or LSQ [8,4], our system is actually quite close. One can consider that the use of a DSOM map projects the perceptions on basis function $\varPhi(s)$ that evolve over time, and that the linear regression estimates the value function as a linear combination $w^T \varPhi(s)$ of these basis functions. The search for optimal coefficients of this linear combination is made by a stochastic gradient descent and the basis functions being adaptable, our algorithm does not guarantee convergence.

The neuronal learning architecture is well adapted to the developmental approach that we want to experience. The action space is discrete, its richness is closely tied to its cardinality. Increasing the richness of the action space can be dealt with by adding neurons to the output layer. These new neurons take advantage of the already adapted DSOM map. The learning of the coefficients of the linear combination of a new output neuron can be accelerated by initialising these coefficients using the coefficient of neurons coding "semantically" close actions. In the example that we present below (see Section 4), a new neuron connected with action `turn-right-slowly` will be initialized with the coefficients related to the neuron for the action `turn-right`.

For the continuous state space, the richness comes in part from to the dimension of this space. Increasing the size of the input space is not as simple as for the output space. One solution is to increase the size of all vectors of input weight without changing their projection on the previous perceptual space. Another solution that we would like to test is to provide the architecture with input weight

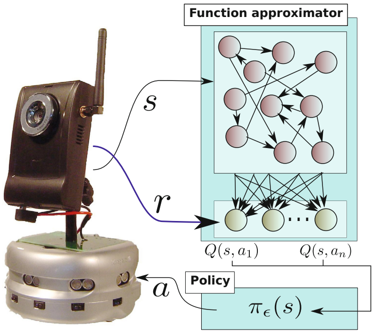


Fig. 1. Learning architecture. Robot perceptions s are the input of a dynamic self-organizing map. A linear output layer with one neuron for each action$a$ learns $Q(s, a)$.

of the maximum dimension of the input space and, while the current perceptual vector has dimension less than this maximum, to artificially increase its size by cloning a few of these values to the extra dimensions. The latter approach seems better adapted to transfer what is learned in small size to larger tasks.

4 Experimental Platform

Our experiments are performed on a KheperaIII robot of K-Team company[1]. Although this robot embarks a processor, we chose to deport processing and computation on an external computer, communication between this computer and the robot is done by wifi.

The robot moves using two drive wheels which allow it to turn on the spot. The robot can use discrete actions, such as moving forward or backward a certain distance (on average, as the actions of the robots are noisy) or turn right or left for a given angle. We use discrete actions because we want to explore a framework based on Q-Learning and estimating a value function for continuous actions is still largely an open problem. The robot's action space is a discrete space and its richness is characterized by the number of available actions.

The main sensor we use is a wireless camera placed above the robot which captures a 320x200 bitmap color image, the image is then processed to provide sensory information in a lower dimensional space. Through a logical sensor we call "*retina*", the robot is provided with perceptions made of a vector of dimension dim^{per} where each coefficient of the vector, between 0 and 1, is the ratio of a particular hue computed on a vertical strip of the image. The number and positions of the vertical stripes are customizable and can evolve during the experiments. On the example of the right side of Fig. 2, the robot is facing a blue diamond on a red background, the retina is configured to show the ratio of blue vertical stripes - named `B1`, `B2` and `B3` - positioned at the abscissa 80, 160 and

[1] See `http://www.k-team.com`

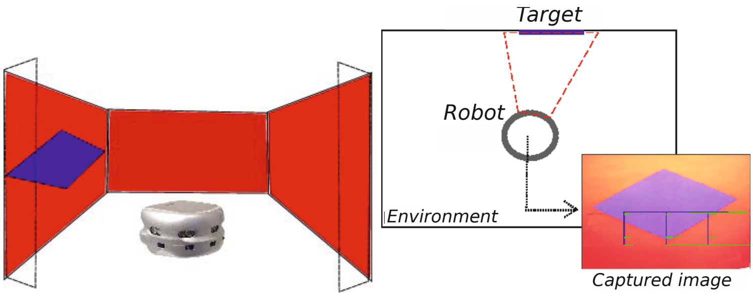


Fig. 2. Left: the robot in its environment. Right: the perception given by the `retina` of the robot. For every vertical stripe, positioned at $(80, 160, 240)$, the ratio of blue is computed, here $[0,42; 0,67; 0,33]$.

240 of the image. Perceptions of the robot in this case are $[0.42, 0.67, 0.33]$. The state space of the robot is a continuous space with a richness measured by the dimension of this space (that is to say the number of vertical stripes).

Given our experimental platform, the tasks we want to teach our robot are navigation tasks in mazes with visual cues. Our idea is to set visual cues in the environment to guide the robot, *e.g.* arrows that tell in which direction to head in order to reach the exit. If the robot needs a fine perception to distinguish and recognize the arrows, it does not need this degree of perception in order to be "attracted" by the highly contrasted symbols. In addition, the robot can learn "satisfactory" behaviors with only crude actions. Then, these behaviors can be refined and precised as actions and perceptions become richer. We believe that this kind of scenario, which can be broken down into sub-tasks of varying complexity, is well suited to the experimentation of the concepts of developmental reinforcement learning.

5 Experimental Results

The learning architecture proposed is tested in a very simple testbed. This experiment, which only involves an increase in the richness of the action space of the robot, is only a preliminary to other experiments. Increase in perceptions space and in the complexity of the task are scheduled.

In this experiment, the robot is positioned in a closed environment. The surrounding walls are red and a blue diamond (the target) is set on one of the walls. The robot must choose between 5 discrete actions: `turn-right` and `turn-left` (about 34 degree turn), `stop`, `turn-right-slowly` and `turn-left-slowly` (about 20 degree). These action are noisy by nature because of the imprecision in the robot command. The perceptual space is made of 3 vertical stripes that return the ratio of blue. These stripes, called $B1$, $B2$ and $B3$ are respectively set to the extreme left, at the middle and at the extreme right of the image given by the camera of the robot, their relative angle to the front of the robot are -16, 0 and 16 degree. The robot has to learn to face the target and it receives a reward if it stops while the target is in front of it. If $0.7B1 + B2 + 0.7B3 > 1.4$ and the robot stops, then it gets a reward of +1, otherwise the reward is zero. After getting a non-zero reward, of after 12 moves without success, the robot is repositioned randomly.

Learning samples come from real movements and perceptions on the robotic platform but our learning algorithm needs too many iterations and the sample set is not enough for that. As explained later, 10,000 to 15,000 iterations are needed for the algorithm to learn. Thus, we use, and more importantly reuse, the same set of 2,000 samples generated on the robot. Each iteration of the algorithm is done with one sample randomly chosen from the sample set, with respect to the learning scenario (*e.g.*, a sample must use one of the actions currently allowed to the robot).

Three different learning scenarii are tested. In the first one, the robot can only use 3 actions `turn-left`, `turn-left` and `stop`. In the second scenario, the robot can choose among the 5 different actions. In the "*developmental*" scenario, the robot starts learning with 3 actions during N_B iterations then it can also use the 2 other actions, with some optional adaptation of learning parameters.

Results presented were obtained using the following parameters: 64 neurons for the DSOM map, with a "small world" neighborhood and at least on neighbor for every neuron ($\text{nb}_{\text{neur}} = 64$, $\text{nb}_{\text{link}} = 1$) ; $\epsilon_D = 0.5$ and $\eta = 1$ are the learning parameters of the DSOM map; $\alpha = 0.1$ and $\gamma = 0.9$ are parameters for the reinforcement learning framework; the exploration policy used is an epsilon-greedy

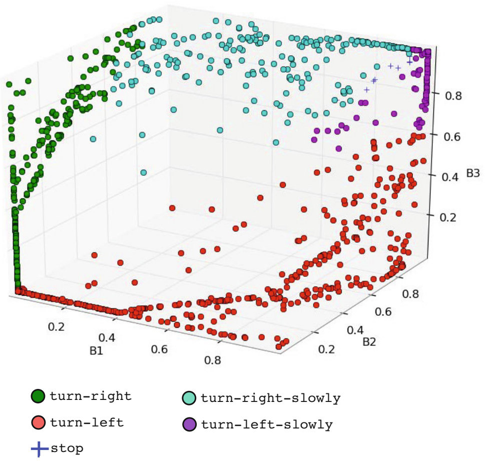


Fig. 3. Representation of a policy learned with 5 actions in the sensorimotor space (the coordinate B1, B2 and B3 are the dimension of the perceptual space). The agent stops when facing the target (toward the (1,1,1) vertex) and uses big turns when the target is far from the center (weak values for B1 or B3).

policy, with $\epsilon_\pi = 0.25$. The learning parameter ϵ_L of the linear regressor varies with the experiments (usually between 0.001 and 0.5). This parameter seems to have a more important influence on the quality of the results and on the speed with which performances are reached. These parameters have been set by hand, they give good results, as suggested by the learned policy depicted on Fig. 3.

For every experiment, the quality of learning is evaluated every 1,000 iterations by testing the greedy policy on a fixed set of starting positions. The set of positions is defined so as to discriminate as much as possible the different policies. These positions are given by the following angles: 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 25, 30, 35, 40, 45, 50, 90, 180 and their symmetric. The mean of the reward received from these positions is a measure of the quality of a policy, and thus of the learning. The best hand-made policy using 3 actions has a performance of 0.138 and, with 5 actions, one can reach a performance of 0.39.

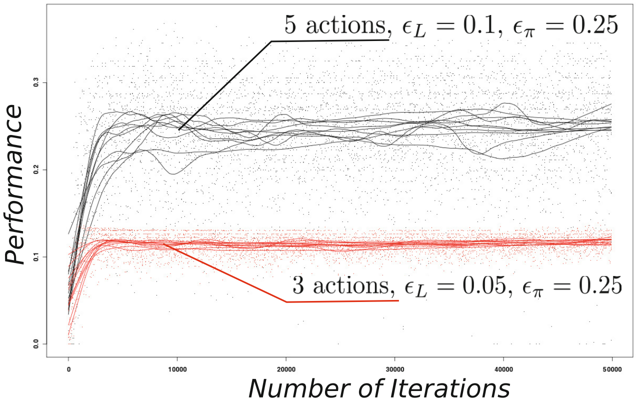


Fig. 4. Learning with 3 and 5 actions, points depict the performances every 100 iterations, for 10 different trajectories. Each trajectory is smoothed and represented by a line.

Fig. 4 shows the performance reached by the algorithm as a function of the number of iterations when using 3 or 5 actions, with $\epsilon_L = 0.05$ and $\epsilon_L = 0.1$. Each graph is made of 10 learning trajectories. The points are the performances measured and the line are sliding mean of these trajectories. For this set of parameters, as for most of the other settings tested, one can notice a large variability in the performance of the algorithm along a trajectory. The variability can be limited by using very low ϵ_L value.

We want to point out that the learning speed is largely increased with the DSOM based architecture. In a previous work using a multi-layer perceptron and eligibility traces [9], 100,000 iterations were needed before reaching good performances, whereas here only 5,000 to 15,000 are required.

The main focus of the experiments is the exploration of the developmental scenario. Fig. 5 compares the performances of the algorithm when using directly 5 actions with performances obtained with the developmental scenario (3 then 5 actions). The direct learning is quicker but the developmental learning can rather quickly (around 10, 000 iterations) reach better performances. To reach these performances, the learning parameter ϵ_L of the linear regressor must be adapted during the developmental scenario: starting at a value of 0.1, it is reduced to 0.01 when adding the two new actions. If this parameter is kept at 0.1, the performance are reduced (red curve in figure 5, right).

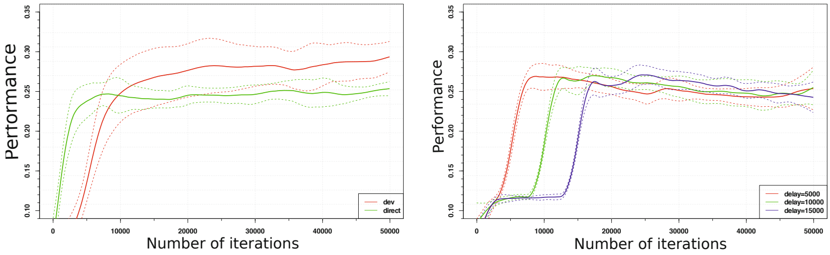


Fig. 5. Developmental learning. Left: The direct (5 actions) and developmental learning (3 actions, then 2 more after $N_B = 5000$ iterations, ϵ_L goes from 0.1 to 0.01) are compared. The curve show the mean and variance for 10 trajectories. Right: Developmental learning with various delay before using the full action set ($\epsilon_L = 0.1$, $N_B = \{5000, 10000, 15000\}$.

Fig. 5:right shows that, in a developmental scenario, the number if iterations before adding the last 2 actions does not influence the reached performance. For this set of parameters ($\epsilon_L = 0.1$ and $\epsilon_\pi = 0.25$), as for many others, performances tend to decrease after 20, 000 iterations. This fact, combined with the large variability within a learning trajectory, may suggest that the task to learn is not that simple. This may be related to the difficulties of using non-linear approximators in a reinforcement learning framework [10,11].

6 Discussion

Results presented here are still preliminary and are part of a work in progress. Many parameters influence the learning algorithm and these parameters should be more systematically explored. Some of these parameters seem to balance each other, like, for example, the number of neurons in the DSOM map and its learning parameters: with a large number of neurons, the DSOM map does not need to be very adaptive and ϵ_D can be low.

The intrinsic properties of DSOM map, like the capacity to adapt continuously to the input while not being oversensible to the distribution of the input, seem particularly interesting when the perceptual space of the agent will change with time. We want to test this, by increasing the perceptions space (add more stripes to the retina) but also by moving these stripes or making them more noisy.

Learning performances are very unstable. In only 100 iterations, the performance of the learned policy can change a lot. Even when the DSOM map is stabilized, which happens very quickly in our experiments as the perceptual space is stable and "small", performances are still unstable, even for low learning parameters of the linear regressor. We would like to study more systematically how the various parameters could be tuned to decrease this variability while preserving both the level of performance and the learning speed, but these criteria are qualitatively assessed right now. We lack an automated procedure for comparing the overall performance (speed, stability, level) of different parameter settings. This kind of automated overall performance evaluation could also be used to guide and control the increase of the richness of the sensorimotor space of the agent. Thus, using original developmental path, it could be possible to link the increase in performance and the "maturation" process of the agent, as done in [12] for example.

7 Conclusion

In this paper, we have presented the principles of a developmental reinforcement learning architecture in order to ease the exploration of large and complicated sensorimotor spaces. At the heart of this architecture lies a dynamic self-organizing map that participates in learning an approximation of the value function of the sensorimotor space. Thus, our learning algorithm is a kind of developmental neuro *Q-Learning* algorithm.

Experiments conducted so far on a simple robotic task are quite promising but far from complete, a lot of testing is still needed. When increasing the action space, the learning architecture learns more quickly than previous architectures [9,13]. Furthermore, even on this simple task, the developmental scenario tested brings better results than a direct approach. This work opens a lot of perspective and future work, in particular a more systematic study of the parameters, the testing of evolving perceptual spaces and more complex tasks.

References

1. Sutton, R., Barto, A.: Reinforcement Learning. Bradford Book, MIT Press, Cambridge, MA (1998)
2. Puterman, M.: Markov Decision Processes: discrete stochastic dynamic programming. John Wiley & Sons, Inc., New York (1994)
3. Sigaud, O., Buffet, O. (eds.): Markov Decision Processes in Artificial Intelligence. ISTE - Jonh Wiley & Sons (2010)

4. Lagoudakis, M.G., Parr, R., Littman, M.L.: Least-Squares Methods in Reinforcement Learning for Control. In: Vlahavas, I.P., Spyropoulos, C.D. (eds.) SETN 2002. LNCS (LNAI), vol. 2308, pp. 249–260. Springer, Heidelberg (2002)
5. Watkins, C.: Learning from delayed rewards. PhD thesis, King's College of Cambridge, UK (1989)
6. Caruana, R.: Multitask Learning. In: Learning to Learn. Kluwer Academic Publishers (1997)
7. Rougier, N.P., Boniface, Y.: Dynamic Self-Organising Map. Neurocomputing 74(11), 1840–1847 (2011)
8. Bradtke, S., Barto, A.: Linear least-squares algorithms for temporal difference learning. Machine Learning 22, 33–57 (1996)
9. Sarzyniec, L., Buffet, O., Dutech, A.: Apprentissage par renforcement développemental en robotique autonome. In: Conférence Francophone d'Apprentissage (CAp 2011), Chambéry (2011)
10. Bertsekas, D., Tsitsiklis, J.: Neuro-dynamic programming. Athena Scientific, Belmont (1996)
11. Coulom, R.: Reinforcement learning using Neural Networks, with Applications to Motor Control. PhD thesis, Institut National Polytechnique de Grenoble (2002)
12. Baranes, A., Oudeyer, P.Y.: Maturationally-constrained competence-based intrinsically motivated learning. In: Proc. of IEEE Int. Conference on Development and Learning (ICDL 2010), Ann Arbor, Michigan, USA (2010)
13. Dutech, A.: Dynamic reservoir for developmental reinforcement learning. In: Workhop on Development and Learning in Artificial Neural Networks (DevLeaNN), Paris (2011)

Adaptive Learning in Continuous Environment Using Actor-Critic Design and Echo-State Networks

Mohamed Oubbati, Johannes Uhlemann, and Günther Palm

Institute of Neural Information Processing, University of Ulm
89069 Ulm, Germany
{mohamed.oubbati,johannes.uhlemann,guenther.palm}@uni-ulm.de
http://www.uni-ulm.de/in/neuroinformatik.html

Abstract. Approximating adaptive dynamic programming has been studied extensively in recent years for its potential scalability to solve problems involving continuous state and action spaces. The framework of adaptive critic design (ACD) addresses this issue and has been demonstrated in several case studies. The present paper proposes an implementation of ACD using an echo state network as the critic. The ESN is trained online to estimate the utility function and adapt the control policy of an embodied agent. In addition to its simple training algorithm, the ESN structure facilitates backpropagation of derivatives needed for adapting the controller. Experimental results using a mobile robot are provided to validate the proposed learning architecture.

1 Introduction

Dynamic programming [1] is one of the most general approaches to obtain optimal control policies for autonomous systems. Unfortunately, it has often been dismissed because of the commonly known "curse of dimensionality" problem, i.e. computational complexity increases exponentially with dimensionality of the application [2]. The framework of adaptive critic design (ACD) adresses this problem by approximating the cost-to-go function J in dynamic programming [3]. Three basic approaches are proposed for implementing ACDs: *Heuristic Dynamic Programming* (HDP) which trains the critic to approximate J [4], the *Dual Heuristic Programming* (DHP) which adapts the critic to estimate the derivatives of J [5] and *Globalized Dual Heuristic Programming* (GDHP) which combines the best of HDP and DHP, i.e. minimizing error prediction of both J and its derivatives [6]. A detailed description of these structures and their action-dependent versions can be found in [7]. Although ACDs have been mathematically well formalized from the adaptive control theory point of view [8][9][10], the challenge to convert existing theory to real practical applications exists, mainly due to its increased computational demands. As any adaptive approach, the success of ACD implementation depends on the efficiency of the online training algorithm used. Implementing ACDs using recurrent neural networks (RNNs) emerges as one possible and very promising approach. RNNs are universal approximators of dynamical systems [11] and can, indeed, be trained to estimate the value function in a continuous space. Although training algorithms for RNNs exist and have shown some success, such as the backpropagation through time [12], some problems like high computational costs and

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 320–329, 2012.

slow convergence remain unresolved preventing RNNs to be applied for a wider class of problems [13].

Reservoir computing networks have recently gained wide attention, due to their new architectures and efficient training methods. Maybe the most prominent network is the echo state network (ESN) [14] which has been successfully implemented in several real world application and performed exceptionally well. Its general idea consists of using a large RNN as a “pool” of excitable complex neural dynamics, from which readout neurons can learn to extract the current state of the network. This reduces the complexity of training to simple linear regression while preserving the recurrent property of the network. In a recent work [15] [16], we implemented ACD using ESN as the critic. The preliminary results show that we can benefit significantly from the proposed ESN-ACD architecture to solve a variety of problems. In this paper, we go further in our investigation by testing how well the ESN critic can adapt in a changing environment. The ESN is trained online to learn a control policy that acquires reward and avoids punishment regions in a real environment. At unpredictable time the experimenter changes the environment by interchanging reward and punishment regions, and the ESN must adapt the previously learned policy to the new situation.

The remaining of this paper is divided as follows. Section 2 explains the learning design. Section 3 presents the implementation of the approach and shows the obtained results. Finally, this work is discussed and concluded in Section 4.

2 Learning Framework

We assume that the world (agent-environment) is described by a dynamical system

$$s(k+1) = F[s(k), a(k)] \tag{1}$$

where $s \in \Re^n$ represents the state vector, k discrete time, $a \in \Re^m$ denotes the control action, and F is the system function. The environment includes contextual stimuli and the agent body (robot). Suppose that one associates with this system the performance index

$$J[s(i)] = \sum_{k=i}^{\infty} \gamma^{k-i} U[s(i), a(i)], \tag{2}$$

where U is the utility function (reward), and γ is a discount factor with $0 < \gamma < 1$. The goal is to obtain optimal action sequences $a(k)$ that maximizes J. The key idea is that any action sequences $a(k)$ that maximizes J in the short term will also maximize the sum of U over all future times [5].

2.1 The Critic

The mission of the ESN is to deliver $\hat{J}$, an estimation of J, according to the heuristic dynamic programming [4]. An ESN is formed by a so-called ”dynamic reservoir”, which contains a large number of sparsely interconnected neurons with non-trainable weights. The activation of internal neurons is updated according to

$$X(k+1) = f(W_{in}U(k+1) + WX(k) + W_{back}Y(k+1)) \tag{3}$$

and the outputs are calculated as

$$Y(k+1) = f_{out}(W_{out}(U(k+1), X(k+1), Y(k))) \tag{4}$$

An essential condition for successful using of ESN is the "echo state" property. It is a property of the network prior to training, related to the weight matrices (W^{in}, W, W^{back}). A network (W_{in}, W, W_{back}) has echo states with respect to the compact intervals U and D, if the network state $X(n)$ is uniquely determined by the history of the input-output data. In practice it was found that when the spectral radius of the internal matrix $|\lambda_{max}| < 1$, we do have an echo state network. The following procedure seems to give a practical solution to guaranty echo state property [14]:

1. The order of input and output neurons should be stated according to the task at hand.
2. Generate randomly the input weights W_{in} and output backpropagated weights W_{back}.
3. Generate randomly an internal weight matrix W_0.
4. Normalize W_0 with its spectral radius λ_{max} and put it in $W_1 : W_1 = \frac{1}{|\lambda_{max}|}W_0$.
5. Scale W_1 with a factor $0 < \alpha < 1$ and put the new internal matrix $W = \alpha W_1$ (in the remaining of this paper α is called the spectral radius).

If this condition is met, only weights connections from the internal neurons to the output (W_{out}) are to be adjusted using any linear training method, such as least squares method. In this work, the output weights W_{out} are adjusted recursively using the temporal difference (TD) learning algorithm [17], by minimizing the following error

$$\|E\| = \sum_k E_k = \sum_k [\hat{J}(k) - U(k) - \gamma J(k+1)]^2 \tag{5}$$

where $\hat{J}(k) = \hat{J}[s(k), a(k), k, \theta_c]$ and θ_c represents the parameters of the ESN (Fig. 1). After each adjustment of W_{out} the ESN's internal state is updated according to

$$X(k) = f(W_{in}S(k) + WX(k-1)) \tag{6}$$

and its output $\hat{J}$ is computed as

$$\hat{J}(k) = f_{out}(W_{out}(k)X(k)) \tag{7}$$

where $f = tanh()$, and f_{out} is chosen as a linear activation function (identity).

2.2 The Actor

The role of the actor is to generate control actions that maximize J in the short term. Since in this approach we don't train an action network, we use a control law that decreases the gradient of the predicted $\hat{J}$. At each time step the control action is calculated as

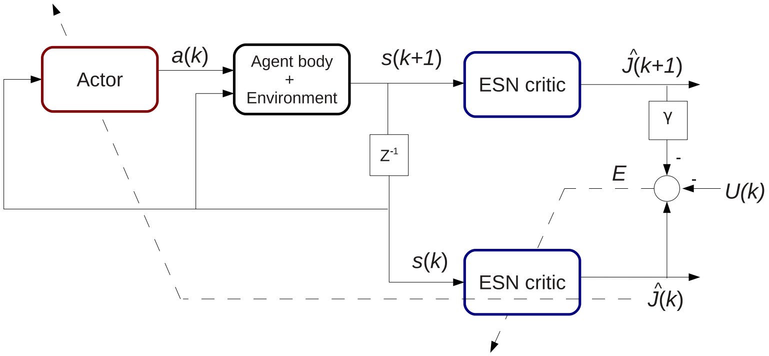


Fig. 1. The ESN critic in two consecutive time steps. Dashed lines represent the information flow of parameter updating for the critic and the actor.

$$a(k+1) = a(k) - \delta \frac{\partial \hat{J}(k)}{\partial a(k)} \tag{8}$$

where δ is the learning rate.

The gradient of $\hat{J}$ with respect to $a(k)$ can be computed using the chaine rule

$$\frac{\partial \hat{J}(k)}{\partial a(k)} = \frac{\partial \hat{J}(k)}{\partial S(k)} \frac{\partial S(k)}{\partial a(k)} \tag{9}$$

The term $\frac{\partial S(k)}{\partial a(k)}$ represents the agent-environment interaction model (1), and $\frac{\partial \hat{J}(k)}{\partial S(k)}$ is computed as follows

$$\frac{\partial \hat{J}(k)}{\partial S(k)} = \frac{\partial \hat{J}(k)}{\partial X(k)} \frac{\partial X(k)}{\partial S(k)} \tag{10}$$

where

$$\frac{\partial \hat{J}(k)}{\partial X(k)} = W_{out} \tag{11}$$

To compute the term $\frac{\partial X(k)}{\partial S(k)}$ we proceed as follows. We put

$$\xi = W_{in} S(k) + W X(k-1) \tag{12}$$

and using the chaine rule again we obtain

$$\frac{\partial X(k)}{\partial S(k)} = \frac{\partial f(\xi)}{\partial \xi} \frac{\partial \xi}{\partial S(k)} = \left(1 - X^2(n)\right) W_{in}^T \tag{13}$$

where I denotes the column vector of 1.

From (11) and (13) we obtain

$$\frac{\partial \hat{J}(k)}{\partial S(k)} = W_{out}(I - X^2(k)) W_{in}^T \tag{14}$$

Equation (14) shows that the partial derivative of $\hat{J}$ with respect to S depends only on the input and output weights and on the current reservoir state. In contrast to a typical layered neural networks the backpropagation of derivatives is then simplified because of the ESN structure that separates the dynamic reservoir from the input and output units.

3 Implementation and Results

Fig. 2 illustrates a scenario of a robot that explores an environment containing positive and negative reward regions represented by the green point and the red point, respectively. The objective is to learn a control policy that acquires rewards and avoid punishments, i.e. navigating from a given starting state to the green region, and avoid the red region. At unpredictable time, the experimenter changes the environmental state, by interchanging reward and punishment regions, and the previously learned policy must be adapted to this new situation. In this task, the ESN critic is asked to consider continuous state space, deal with uncertainties in sensory data, adapt to changes in the environment and be computationally cheap.

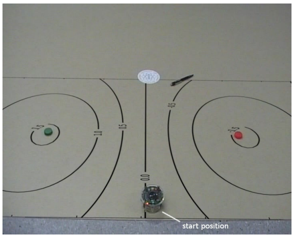
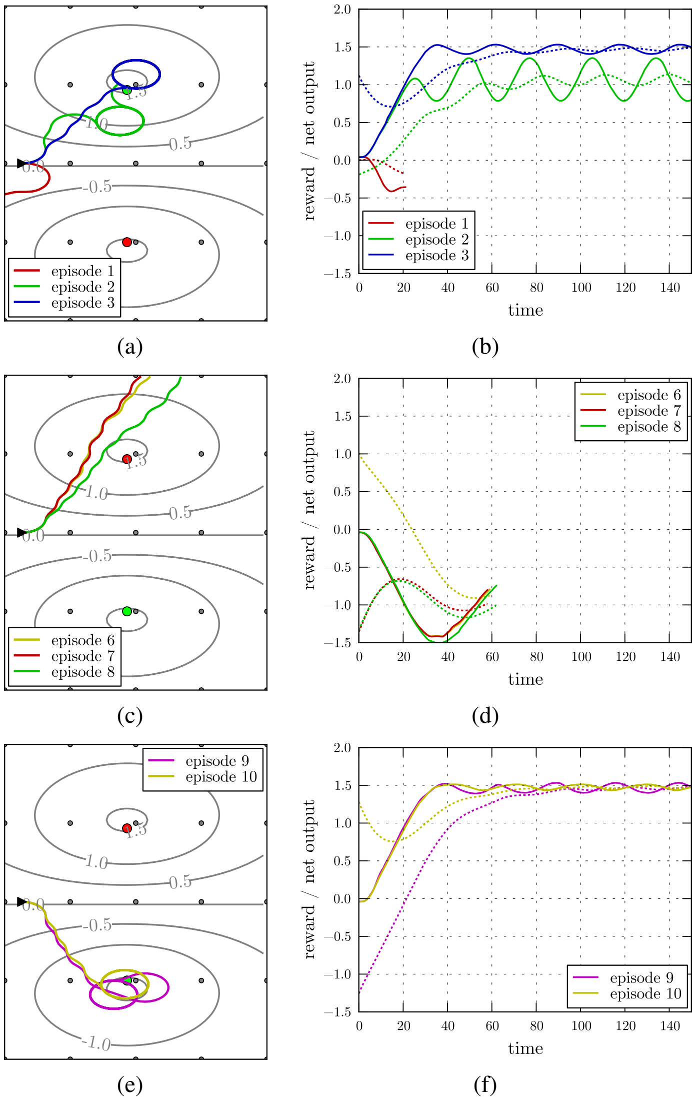


Fig. 2. Scenario for adaptive behavior in ESN-ACD. The robot has to learn a control policy to acquire rewards (green point) and avoid punishments (red point), and revise the learned policy whenever the reward and punishment regions are relocated. Virtual lines (in black) show the distribution of the utility value on the environment.

The robot has a differential-drive, and its geometric configuration is described by $q = [x, y, \phi]^T$ where (x, y) are its coordinates, and ϕ its heading angle relative to the local coordinate system. We consider a continuous state space $S(k) = \{x(k), y(k)\}$, and continuous action space in a form of desired heading $a(k) = \phi_d(k)$. At each time a new state $S(k)$ is measured, the utility function $U(k)$ is provided as

$$U(k) = \frac{2}{(1 + {d_r}^2)} - \frac{2}{(1 + {d_p}^2)} \tag{15}$$

where d_r and d_p define the distance between $S(k)$ and the reward and punishment regions, respectively. The characteristics of the ESN are described by a set of 5 parameters: α (spectral radius), N (number of neurons in the reservoir), c_{dr} (connectivity of the internal weight matrice W), c_i (connectivity of the input weight matrice W_{in}),



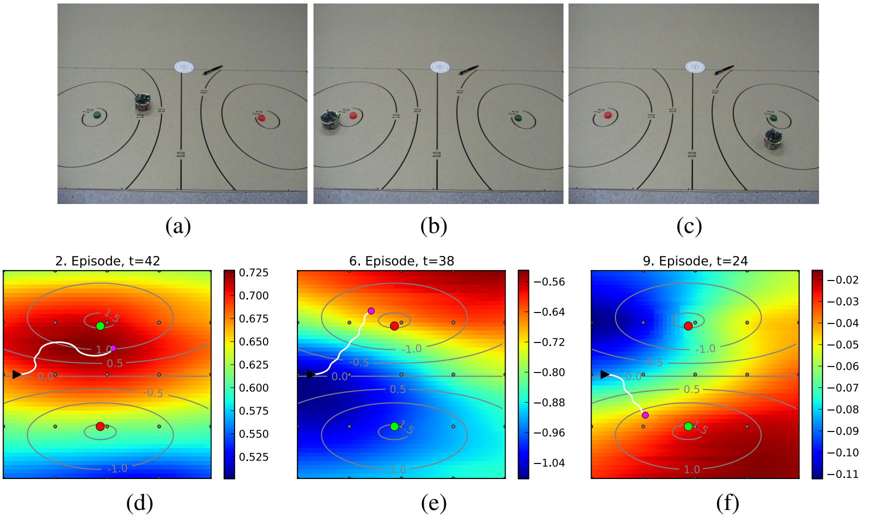
(a) (b)

(c) (d)

(e) (f)

Fig. 3. The behavior of the learning robot in a sequence of training episodes. Plots on the right side show the received and predicted Utility from ESN-Critic (dashed lines), and those on the left side show the 2d path of the robot. a) and b) show that after 3 trial episodes the ESN-Critic approximately predicts the true utility value, and the robot successfully applies a correct control action leading to the reward area. In (c) and (d) the reward/punishment areas are relocated, but the robot keeps the same control policy and moves into the punishment region. During the next three episodes the robot leaves the borders of the environment, and the episodes were stopped. In episode 9 ((e) and (f)) the robot has recognised that its current policy must be changed. After one more episode, it learns a new policy that avoids the punishment area and acquires the new reward region.

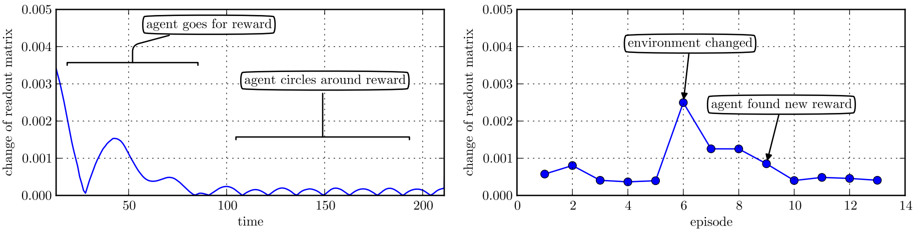
and r (amplitude of the interval distribution where the internal synaptic connections weights are randomly initialized). No back-connection W_{back} from the output to the reservoir, and no synaptic weight connections from the input directly to the output are used. The training is performed in a sequence of episodes. After completion of one episode, the robot is automatically transferred to the start position for the next one. We set $\alpha = 0.8$, $N = 400$, $r = 4$, $c_{dr} = 30\%$ and $c_i = 20\%$, and we start the experiment with an untrained ESN. After the first 3 trial episodes the agent has successfully learned a correct policy leading to the reward region (Fig. 3. a.). This is also confirmed by the convergence of ESN-Critic to a maximum of the utility value (Fig. 3. b.). The robot now knows a path to the goal, and with continuing learning time (episodes 4 and 5), no significant refinment were noticed. After 5 episodes we changed the environmental state, and started the 6th episode. The robot keeps the same control policy and moves into the punishment region (Fig. 3. c.). At that moment and after receiving punishment signals, it recognised that the current policy is no longer correct (Fig. 3. d.). After two more episodes, the ESN critic provides new reinforcement signals to adjust the learned policy, and the robot generates a new behavior that avoids the punishment area, maximize again the expected rewards (Fig. 3. f.), and moves to the new reward region (Fig. 3. e.). Fig. 4 illustrates a global view of the utility distribution seen by the ESN critic throughout the episodes. This could also be considered as an internal representation of the environment. Finally, Fig.5 shows how output weights of the ESN critic are adjusted over all episodes.



(a) (b) (c)

(d) (e) (f)

Fig. 4. (a), (b) and (c) are photos from video sequences of the robot behavior at some time steps in episodes 2, 6 and 9 respectively. The graphs (d), (e) and (f) show the utility distribution estimated by the ESN critic at the same time steps. They were obtained by simulating how would be the utility prediction if the robot visits all possible positions on the environment. The lowest utility estimation is represented by the color red, and the highest one by the color blue.



(a) adaptation during one episode (3rd). (b) the average changes over all episodes

Fig. 5. Adaptation of W_{out}. (a) shows that the learning process starts with higher adaptation of reward expectations, and it decreases when the robot starts converging toward to a local rewarded region. As expected, we can see in (b) that the adaptation curve decreases with increasing the episodes (from 1st to 4th episode). Immediately after relocating the reward/punishment areas, at episode 5, the changes of weights go higher, which reflects an exploratory behavior of the agent in order to update the control policy. Updating the control policy to the novel environment has the effect to reduce again the changes of learning weights.

4 Conclusion

The main contribution of the paper is the implementation of an ESN within the heuristic dynamic programming. The ESN is trained online to estimate the utility function in a continuous space of states and actions, and to adapt the agents' control policy accordingly. The high speed of convergence, which is a consequence of the ESN's training algorithm and of the simple calculation of derivatives that are needed to update the actor, provides the proposed ESN-HDP approach the capability to tackle adaptive control tasks in real time. The experimental results are very promising, and pave the way for further tests. Our next step is to increase the complexity of the environment by adding several regions with different reward levels, and to extend the learning algorithm to non-episodic continuous tasks.

At the design level there are many issues that need further investigation. We repeated the previous experiment with different ESN parameter settings (the results will be presented in a extended version), but we admit that their influence on the performance is still not well understood. The constraint imposed on the spectral radius and the random initialisation of the reservoir prevented us to find out what ESN properties are strongly involved in this task. There are many extensions of the standard ESN, e.g. intrinsic plasticity [18], transfer entropy [19] and others [20][21][22], which could provide better understanding of phenomena governing the reservoir dynamics. One possibility for future work is to implement the topology templates proposed in [22] that may simplify the reservoir analysis, and to optimise the information transfer at each internal unit with respect to the learning task [19]. These modifications could provide a clear insight into the reservoir dynamics organization.

At the application level there is an important issue which concerns the characterisation of the agent-environment interaction. In this work, no explicit model of the environment was built, instead, the ESN estimates future rewards directly from the actual state of the agent. The utility distribution over the whole environment illustrated in

Fig. 4 could also be seen as a world-model whose input represents the current state/reward situation, and the output correspond to the expected reward associated with the other possible states of the agent. These estimates can then be used for example in multiple-choice tasks. A multiple-choice task (also called multi-objective problem [23]) could have several possible solutions, and the difficulty is that most often no one can be considered to be better than any others with respect to all objectives. A future work could be, first, the utilization of the reservoir as a source of these solutions. Then, the choice of one solutions among the others will depend on the previously experienced situations. It is maybe premature to go further now in this issue, but our efforts along this line of research are in progress.

References

1. Bellman, R.E.: Dynamic Programming. Princeton Univ. Press, NJ (1957)
2. Dreyfus, S.E., Law, A.M.: Art and Theory of Dynamic Programming. Academic Press, Inc., Orlando (1977)
3. Werbos, P.: Approximate dynamic programming for realtime control and neural modeling. In: Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches. Van Nostrand Reinhold, New York (1992)
4. Werbos, P.J.: Consistency of HDP applied to a simple reinforcement learning problem. Neural Networks 2, 179–189 (1990)
5. White, D.A., Sofge, D.A. (eds.): Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches. Van Nostrand Reinhold, New York (1992)
6. Werbos, P.J.: A menu of designs for reinforcement learning over time. In: Neural Networks for Control, pp. 67–95. MIT Press, Cambridge (1990)
7. Prokhorov, D., Wunsch, D.: Adaptive critic designs. IEEE Transactions on Neural Networks 8, 997–1007 (1997)
8. Al-Tamimi, A., Lewis, F.L., Abu-Khalaf, M.: Discrete-time nonlinear hjb solution using approximate dynamic programming: Convergence proof. IEEE Transactions on Systems, Man, and Cybernetics, Part B 38(4), 943–949 (2008)
9. Vrabie, D., Pastravanu, O., Abu-Khalaf, M., Lewis, F.L.: Brief paper: Adaptive optimal control for continuous-time linear systems based on policy iteration. Automatica 45(2), 477–484 (2009)
10. Vrabie, D., Lewis, F.L.: Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. Neural Networks 22(3), 237–246 (2009)
11. Funahashi, K.-I., Nakamura, Y.: Approximation of dynamical systems by continuous time recurrent neural networks. Neural Network 6(6), 801–806 (1993)
12. Werbos, P.J.: Backpropagation through time: What it does and how to do it. Proceedings of the IEEE 78(10), 1550–1560 (1990)
13. Lukoševičius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. Computer Science Review 3(3), 127–149 (2009)
14. Jaeger, H.: The 'echo state' approach to analysing and training recurrent neural networks. Technical Report 148, AIS Fraunhofer, St. Augustin, Germany (2001)
15. Koprinkova, H.P., Oubbati, M., Palm, G.: Adaptive critic design with echo state network. In: IEEE Int. Conference on Systems, Man, and Cybernetics, pp. 1010–1015 (2010)
16. Oubbati, M., Kächele, M., Koprinkova, P., Palm, G.: Anticipating rewards in continuous time and space with echo state networks and actor-critic design. In: 19th European Symposium on Artificial Neural Networks (ESANN 2011), pp. 117–122 (2011)

17. Sutton, R.S.: Learning to predict by the methods of temporal differences. Machine Learning 3, 9–44 (1988)
18. Schrauwen, B., Wardermann, M., Verstraeten, D., Steil, J.J., Stroobandt, D.: Improving reservoirs using intrinsic plasticity. Neurocomputing 71, 1159–1171 (2008)
19. Obst, O., Boedecker, J., Asada, M.: Improving Recurrent Neural Network Performance Using Transfer Entropy. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) ICONIP 2010, Part II. LNCS, vol. 6444, pp. 193–200. Springer, Heidelberg (2010)
20. Xue, Y., Yang, L., Haykin, S.: Decoupled echo state networks with lateral inhibition. Neural Networks 20, 365–376 (2007)
21. Zhidong, D., Yi, Z.: Collective behavior of a small-world recurrent neural system with scale-free distribution. IEEE Transactions on Neural Networks 18(5), 1364–1375 (2007)
22. Rodan, A., Tino, P.: Minimum complexity echo state network. IEEE Transactions on Neural Networks 22(1), 131–144 (2011)
23. Coello Coello, C.A., Lamont, G.B.: Applications of multi-objective evolutionary algorithms. Advances in Natural Computation, vol. 1 (2004)

Learning Adjectives and Nouns from Affordances on the iCub Humanoid Robot

Onur Yürüten[1], Kadir Fırat Uyanık[1,3], Yiğit Çalışkan[1,2],
Asil Kaan Bozcuoğlu[1], Erol Şahin[1], and Sinan Kalkan[1]

[1] Kovan Res. Lab, Dept. of Computer Eng., Middle East Technical University, Turkey
{oyuruten,kadir,asil,erol,skalkan}@ceng.metu.edu.tr,
[2] Dept. of Computer Eng., Bilkent University, Turkey
caliskan@cs.bilkent.edu.tr
[3] Dept. of Electrical and Electronics Eng., Middle East Technical University, Turkey

Abstract. This article studies how a robot can learn *nouns* and *adjectives* in language. Towards this end, we extended a framework that enabled robots to learn affordances from its sensorimotor interactions, to learn nouns and adjectives using labeling from humans. Specifically, an iCub humanoid robot interacted with a set of objects (each labeled with a set of adjectives and a noun) and learned to predict the effects (as labeled with a set of verbs) it can generate on them with its behaviors. Different from appearance-based studies that directly link the appearances of objects to nouns and adjectives, we first predict the affordances of an object through a set of Support Vector Machine classifiers which provided a functional view of the object. Then, we learned the mapping between these predicted affordance values and nouns and adjectives. We evaluated and compared a number of different approaches towards the learning of nouns and adjectives on a small set of novel objects.

The results show that the proposed method provides better generalization than the appearance-based approaches towards learning adjectives whereas, for nouns, the reverse is the case. We conclude that affordances of objects can be more informative for (a subset of) adjectives describing objects in language.

Keywords: affordances, nouns, adjectives.

1 Introduction

Humanoid robots are expected to be part of our daily life and to communicate with humans using natural language. In order to accomplish this long-term goal, such agents should have the capability to perceive, to generalize and also to communicate about what they perceive and cognize. To have the human-like perceptual and cognitive abilities, an agent should be able (i) to relate its symbols or symbolic representations to its internal and external sensorimotor data/experiences, which is mostly called *the symbol grounding problem* [1] and (ii) to conceptualize over raw sensorimotor experiences towards abstract, compact and general representations. Problems (i) and (ii) are two challenges an embodied agent faces and in this article, we focus on problem (i).

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 330–340, 2012.

The term concept is defined by psychologists [2] as the information associated with its referent and what the referrer knows about it. For example, the concept of an apple is all the information that we know about apples. This concept includes not only how an apple looks like but also how it tastes, how it feels etc. The appearance related aspects of objects correspond to a subset of noun concepts whereas the ones related to their affordances (e.g., edible, small, round) correspond to a subset of adjective concepts.

Affordances, a concept introduced by J. J. Gibson [3], offers a promising solution towards symbol grounding since it ties perception, action and language naturally. J. J. Gibson defined affordances as the action possibilities offered by objects to an agent: Firstly, he argued that organisms infer possible actions that can be applied on a certain object directly and without any mental calculation. In addition, he stated that, while organisms process such possible actions, they only take into account relevant perceptual data, which is called as perceptual economy. Finally, Gibson indicated that affordances are relative, and it is neither defined by the habitat nor by the organism alone but through their interactions with the habitat.

In our previous studies [4,5], we proposed methods for linking affordances to object concepts and verb concepts. In this article, we extend these to learn nouns and adjectives from the affordances of objects.

Using a set of Support Vector Machines, our humanoid robot, iCub, learns the affordances of objects in the environment by interacting with them. After these interactions, iCub learns nouns and adjectives either (i) by directly linking appearance to noun and adjective labels, or (ii) by linking the affordances of objects to noun and adjective labels. In other words, we have two different approaches (appearance-based and affordance-based models) for learning nouns and adjectives, which we compare and evaluate. Later, when shown a novel object, iCub can recognize the noun and adjectives describing the object.

2 Related Studies

The symbol grounding problem in the scope of noun learning has been studied by many. For example, Yu and Ballard [6] proposed a system that collects sequences of images alongside speech. After speech processing and object detection, objects and nouns inside the given speech are related using a generative correspondence model. Carbonetto et al. [7] presented a system that splits a given image into regions and finds a proper mapping between regions and nouns inside the given dictionary using a probabilistic translation mode similar to a machine translation problem. On another side, Saunders et al. [8] suggested an interactive approach to learn lexical semantics by demonstrating how an agent can use heuristics to learn simple shapes which are presented by a tutor with unrestricted speech. Their method matches perceptual changes in robot's sensors with the spoken words and trains k-nearest neighbor algorithm in order to learn the names of shapes. In similar studies, Cangelosi et al. [9,10] use neural networks to link words with behaviours of robots and the extracted visual features.

Based on Gibson's ideas and observations, Şahin et al. [11] formalized affordances as a triplet (see, e.g., [12,13,14] for similar formalizations):

$$(o, b, f), \tag{1}$$

where f is the effect of applying behaviour b on object o. As an example, a behaviour b_{lift} that produces an effect f_{lifted} on an object o_{cup} forms an affordance relation $(o_{\text{cup}}, b_{\text{lift}}, f_{\text{lifted}})$. Note that an agent would require more of such relations on different objects and behaviours to learn more general affordance relations and to conceptualize over its sensorimotor experiences.

During the last decade, similar formalizations of affordances proved to be very practical with successful applications to domains such as navigation [15], manipulation [16,17,18,19,20], conceptualization and language [5,4], planning [18], imitation and emulation [12,18,4], tool use [21,22,13] and vision [4]. A notable one with a notion of affordances similar to ours is presented by Montesano et al. [23,24]. Using the data obtained from the interactions with the environment, they construct a Bayesian network where the correlations between actions, entities and effects are probabilistically mapped. Such an architecture allows action, entity and effect information to be separately queried (given the other two information) and used in various tasks, such as goal emulation.

In this article, our focus is linking affordances with nouns and adjectives. In addition to directly linking the appearance of objects with nouns and adjectives, we learn them from the affordances of objects and compare the two approaches.

3 Methodology

3.1 Setup and Perception

We use the humanoid robot iCub to demonstrate and assess the performance of the models we develop.

iCub perceives the environment with a Kinect sensor and a motion capture system (VisualEyez VZ2). In order to simplify perceptual processing, we assumed that iCub's interaction workspace is dominated by an interaction table. We use PCL[25] to process raw sensory data. The table is assumed to be planar and is segmented out as background. After segmentation, the point cloud is clustered into objects and the following features extracted from the point cloud represent an object o (Eq. 1):

- *Surface features:* surface normals (azimuth and zenith angles), principal curvatures (min and max), and shape index. They are represented as a 20-bin histogram in addition to the minimum, maximum, mean, standard deviation and variance information.
- *Spatial features:* bounding box pose (x, y, z, theta), bounding box dimensions (x, y, z), and object presence.

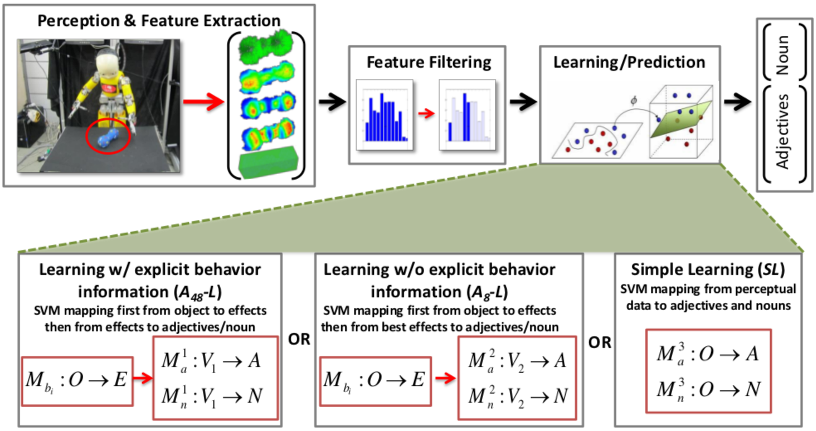

Perception & Feature Extraction
Feature Filtering
Learning/Prediction
ϕ
Noun
Adjectives
Learning w/ explicit behavior information (A_{48}-L)
SVM mapping first from object to effects then from effects to adjectives/noun
$M_{b_i}: O \rightarrow E$
$M_a^1: V_1 \rightarrow A$
$M_n^1: V_1 \rightarrow N$
OR
Learning w/o explicit behavior information (A_8-L)
SVM mapping first from object to effects then from best effects to adjectives/noun
$M_{b_i}: O \rightarrow E$
$M_a^2: V_2 \rightarrow A$
$M_n^2: V_2 \rightarrow N$
OR
Simple Learning (SL)
SVM mapping from perceptual data to adjectives and nouns
$M_a^3: O \rightarrow A$
$M_n^3: O \rightarrow N$

Fig. 1. Overview of the system. iCub perceives the environment and learnes the affordances. From either the perceptual data or the affordances, it learns different models for learning nouns and affordances.

3.2 Data Collection

The robot interacted with a set of 35 objects of variable shapes and sizes, which are assigned the nouns "cylinder", "ball", "cup", "box" (Fig. 2).

The robot's behaviour repertoire $\mathcal{B}$ contains six behaviors ($b_1, ..., b_6$ - Eq. 1): *push-left*, *push-right*, *push-forward*, *pull*, *top-grasp*, *side-grasp*. iCub applies each behaviour b_j on each object o_i and observes an effect $f_{o_i}^{b_j} = o_i' - o_i$, where o_i' is the set of features extracted from the object after behaviour b_j is applied. After each interaction epoch, we give an appropriate effect label $E_k \in \mathcal{E}$ to the observed effect $f_{o_i}^{b_j}$, where E can take values *moved-left*, *moved-right*, *moved-forward*, *moved-backward*, *grasped*, *knocked*, *disappeared*, *no-change*[1]. Thus, we have a collection of $\{o_i, b_j, E_{o_i}^{b_j}\}$, including an effect label $E_{o_i}^{b_j}$ for the effect of applying each behaviour b_j to each object o_i.

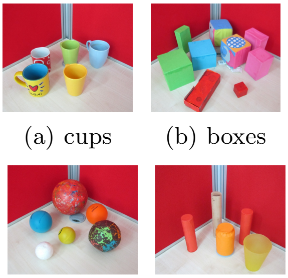

(a) cups (b) boxes
(c) balls (d) cylinders

Fig. 2. The objects in our dataset

3.3 Learning Affordances

Using the effect labels $E \in \mathcal{E}$, we train a Support Vector Machine (SVM) classifier for each behavior b_i to learn a mapping $\mathcal{M}_{b_i} : \mathcal{O} \rightarrow \mathcal{E}$ from the initial

[1] The *no-change* label means that the applied behavior could not generate any notable change on the object. For example, iCub cannot properly grasp objects larger than its hand, hence, the *grasp* behaviour on large objects do not generate any change.

representation of the objects (i.e., $\mathcal{O}$) to the effect labels ($\mathcal{E}$). The trained SVMs can be then used to predict the effect (label) $E_{o_l}^{b_k}$ of a behavior b_k on a novel object o_l using the trained mapping $\mathcal{M}_{b_k}$. Before training SVMs, we use ReliefF feature selection algorithm [26] and only use the features with important contribution (weight > 0) to training.

3.4 Adjectives

We train SVMs for learning the adjectives of objects from their affordances (see Fig. 1). We have six adjectives, i.e., $\mathcal{A} = \{$'*edgy*'-'*round*', '*short*'-'*tall*', '*thin*'-'*thick*'$\}$, for which we require three SVMs (one for each pair). We have the following three adjective learning models:

- *Adjective learning with explicit behavior information (**$\boldsymbol{A}_{48}$-AL**)*:
  In the first adjective learning model, for learning adjectives $a \in \mathcal{A}$, we use the trained SVMs for affordances (i.e., $\mathcal{M}_b$ in Sect. 3.3) to acquire a **48-dimensional** space, $\mathcal{V}_1 = (\hat{E}_1^{b_1}, ..., \hat{E}_8^{b_1}, ..., \hat{E}_1^{b_6}, ..., \hat{E}_8^{b_6})$, where $\hat{E}_i^{b_j}$ is the confidence of behaviour b_j producing effect E_i on the object o. We train an SVM for learning the mapping $\mathcal{M}_a^1 : \mathcal{V}_1 \rightarrow \mathcal{A}$.
- *Adjective learning without explicit behavior information (**$\boldsymbol{A}_{8}$-AL**)*:
  In the second adjective learning model, for learning adjectives $a \in \mathcal{A}$, we use the trained SVMs for affordances to acquire an **8-dimensional** affordance vector, $\mathcal{V}_2 = (p(E_1), ..., p(E_8))$, where $p(E_i)$ is the maximum SVM confidence of a behaviour b_j leading to the effect E_i on object o. From $\mathcal{V}_2$, we train an SVM for learning the mapping $\mathcal{M}_a^2 : \mathcal{V}_2 \rightarrow \mathcal{A}$.
- *Simple adjective learning (**SAL**)*:
  In the third adjective learning model, we learn $\mathcal{M}_a^3 : \mathcal{O} \rightarrow \mathcal{A}$ directly from the appearance of the objects.

After learning, iCub can predict the noun and adjective labels for a novel object (Fig. 3).

3.5 Nouns

We train **one SVM** for nouns $\mathcal{N} = \{$'ball', 'cylinder', 'box', 'cup'$\}$, for which we have 413 instances.

Similar to adjectives, we have three models:

- *Noun learning with explicit behavior information (**$\boldsymbol{A}_{48}$-NL**)*:
  Similar to A$_{48}$-AL, we train an SVM for learning the mapping $\mathcal{M}_n^1 : \mathcal{V}_1 \rightarrow \mathcal{N}$.
- *Noun learning without explicit behavior information (**$\boldsymbol{A}_{8}$-NL**)*:
  Similar to A$_8$-AL, we train an SVM for learning the mapping $\mathcal{M}_n^2 : \mathcal{V}_2 \rightarrow \mathcal{N}$.
- *Simple noun learning (**SNL**)*:
  Similar to SAL, we train an SVM for learning the mapping $\mathcal{M}_n^3 : \mathcal{O} \rightarrow \mathcal{N}$ directly from the appearance of the objects.

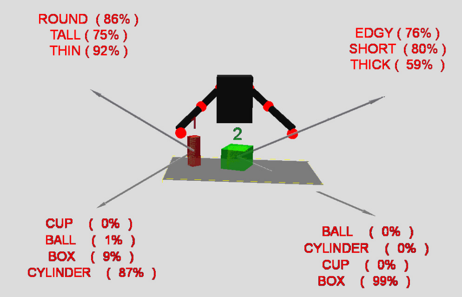


Fig. 3. After learning nouns and adjectives, iCub can refer to an object with its higher level representations or understand what is meant if such representations are used by a human

4 Results

The prediction accuracy of the trained SVMs that map each behaviour b_i on an object to an effect label (i.e., $\mathcal{M}_{b_i} : \mathcal{O} \rightarrow \mathcal{E}$)

is as follows: 90% for *top-grasp*, 100% for *side-grasp*, 96% for *pull*, 100% for *push-forward*, 92% for *push-left* and 96% for *push-right*.

4.1 Results on Adjectives

Using Robust Growing Neural Gas [27], we clustered the types of dependence between each adjective and the effects of the behaviours into *Consistently Small* (-), *Consistently Large* (+) and *Highly Variant* (*). These dependencies allow iCub to relate adjectives with what it can and cannot do with them. Table 1 shows these dependencies for the model A$_{48}$-AL ($\mathcal{M}_a^1$) introduced in Sect. 3.4.

We see from the table what behaviours can consistently generate which effects on which types of objects (specified with their adjectives). For example, with a

Table 1. The dependence between adjectives and affordances for the model A$_{48}$-AL ($\mathcal{M}_a^1$). TG: Top Grasp, SG: Side Grasp, PR: Push Right, PL: Push Left, PF: Push Forward, PB: Pull. For each behavior, there are eight effect categories: a: Moved Right, b: Moved Left, c: Moved Forward, d: Pulled, e: Knocked, f: No Change g: Grasped, h: Disappeared.

Adjective	TG	SG	PR	PL	PF	PB
	$abcdefgh$	$abcdefgh$	$abcdefgh$	$abcdefgh$	$abcdefgh$	$abcdefgh$
Edgy	-----+--	-----**-	*---**-+	-*--**-+	---***-+	---*++-+
Round	-----**-	-----+--	*---+*-+	-*--+*-+	---**+-*	---**+-*
Short	-----**-	-----+--	+---**-+	-+--**-+	---+**-+	---+*+-+
Tall	-----**-	-----**-	*---+*-+	-*--+*-+	---*++-*	---*++-*
Thin	-----**-	-----**-	*---+*-+	-*--+*-+	---*+*-+	----++-+
Thick	-----+--	-----**-	*---**-*	-*--**-*	---**+-*	---+*+-*

consistently large probability, the robot would generate *no change* effect on *edgy* or *thick* objects when *top grasp* behavior was applied. Furthermore, the *short* and *tall* objects show a clear distinction in response to pushing behaviors (*tall* objects have a high probability to be *knocked* while *short* objects simply get pushed).

The dependencies for the no-explicit-behavior model A$_8$-AL ($\mathcal{M}_a^2$) is in Table 2. We see from the table that *round* objects have a consistently high probability to generate *disappeared* effect, whereas *edgy* objects do not have such consistency. Furthermore, *tall* objects have consistently low probabilities in obtaining *moved-left, -right, -forward* or *pulled* effects. Almost all effects can be generated on *thin* objects with consistently high probability.

Table 2. The dependence between adjectives and affordances for the model A$_8$-AL ($\mathcal{M}_a^2$). MR: Moved Right, ML: Moved Left, MF: Moved Forward, P: Pulled, K: Knocked, NC: No Change, G: Grasped, D: Disappeared.

Adjective	MR	ML	MF	P	K	NC	G	D
Edgy	*	*	*	*	+	+	*	*
Round	*	*	*	*	*	+	+	+
Short	*	*	*	*	*	+	+	+
Tall	−	−	−	−	+	+	+	+
Thin	*	+	+	+	+	+	+	+
Thick	*	*	*	*	*	*	+	+

The comparison between the different adjective learning methods is displayed in Table 3, which displays the average 5-fold cross-validation accuracies. We see that the explicit-behavior model (A$_{48}$-AL) performs better than A$_8$-AL and SAL models. The reason that A$_8$-AL is worse than the other methods is eminent in Table 2, where we see that different adjective categories end up with similar descriptor vectors, losing distinctiveness. On the other hand, the A$_{48}$-AL model that has learned adjectives from the affordances of objects performs better than directly learning SAL model.

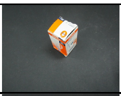
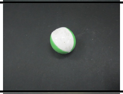
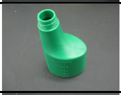
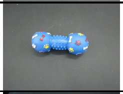
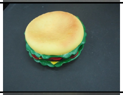
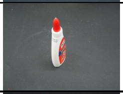
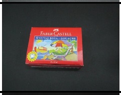
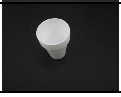
An important point is whether adjectives should include explicit behaviour information (i.e., A$_{48}$-AL vs. A$_8$-AL). Theoretically, the performance of these models should converge while one-to-one, unique behavior-to-effect relations dominate the set of known affordances. In such cases, the behavior information would be redundant. On the other hand, with a behavior repertoire that may pose many-to-one-effect mappings, behavior information must be taken into account to obtain more distinguishable adjectives.

Table 3. Avg. prediction results for the three adjective models in Sect. 3.4

	A$_{48}$-AL	A$_8$-AL	SAL
	$\mathcal{M}_a^1$	$\mathcal{M}_a^2$	$\mathcal{M}_a^3$
Edgy-Round	87%	72%	89%
Short-Tall	93%	95%	89%
Thin-Thick	95%	72%	91%

Results on Adjectives of Novel Objects. Table 4 shows the predicted adjectives from the different models on novel objects. We see that, for adjectives, $\mathcal{M}_a^1$ is better in naming adjectives than $\mathcal{M}_a^2$. For example, $\mathcal{M}_a^2$ mis-classifies object-5 as *edgy*, object-7 as *thin* and object-1 as *thick* whereas $\mathcal{M}_a^1$ correctly

Table 4. Predicted adjectives for novel objects using 3 different models (bold labels denote correct classifications)

ID	Object	A_{48}-AL	A_8-AL	SAL
		$\mathcal{M}_a^1$	$\mathcal{M}_a^2$	$\mathcal{M}_a^3$
1		**edgy (54 %)** **short (97 %)** **thin (59 %)**	**edgy (89 %)** **short (91 %)** **thick (52 %)**	**edgy (89 %)** **short (55 %)** **thin (52 %)**
2		**round (77 %)** **short (77 %)** **thin (89 %)**	**round (90 %)** **short (91 %)** **thin (67 %)**	edgy (79 %) **short (58 %)** **thin 67 %**
3		**edgy (63 %)** short (94 %) **thin (96 %)**	round (72 %) short (92 %) **thin (72 %)**	**edgy (64 %)** **tall (67 %)** **thin 84 %**
4		**round (84 %)** **short (98 %)** **thick (91 %)**	edgy (%94) **short (% 87)** thin (% 68)	**round (77 %)** **short (68%)** thin (62 %)
5		**round (84 %)** **short (97 %)** **thick (95 %)**	edgy (% 81) **short (% 93)** **thick (% 59)**	**round (89 %)** **short (67 %)** **thick (58 %)**
6		**edgy (84 %)** **short (98 %)** **thin (92 %)**	**edgy (79 %)** **short (80 %)** **thin (79 %)**	**edgy (79 %)** tall (55 %) thick (62 %)
7		**edgy (62 %)** **short (98 %)** **thick (78 %)**	**edgy (52 %)** **short (93 %)** thin (53 %)	round (84 %) **short (54 %)** **thick (68 %)**
8		**round (72 %)** **short (98 %)** **thick (79 %)**	**round (69 %)** **short (95 %)** **thick (64 %)**	edgy (89 %) **short (67 %)** **thick (52 %)**

names them. On some objects (e.g., object-3), where there are disagreements between the models, correctness cannot be evaluated due to the complexity of the object. If we look at the direct mapping from objects' appearance to adjectives ($\mathcal{M}_a^3$), we see that it misclassifies object-7 as *round*, object-6 as *tall* and objects 2 and 8 as *edgy*.

4.2 Results on Nouns

For the three models trained on nouns (Sect. 3.5), we get the following 5-fold cross-validation accuracies: A_{48}-NL: 87.5%, A_8-NL: 78.1% and SNL: 94%. We see that, unlike the case in adjectives, directly learning the mapping from appearance to nouns performs better than using the affordances of objects. This suggests that the affordances of the objects (used in our experiments) are less descriptive for the noun labels we have used. The dependency results for nouns (similar to the ones in adjectives shown in Tables 1 and 2) are not provided for the sake of space.

Results on Nouns of Novel Objects. Table 5 shows the results obtained on novel objects. Unlike the case in adjectives, the simple learner (SNL) significantly outperforms the A_{48}-NL and A_8-NL models. Hence, we conclude that the set of nouns (cup, cylinder, box, ball) we have are more of appearance-based.

5 Conclusion

We proposed linking affordances with nouns and adjectives. Using its interactions with the objects, iCub learned the affordances of the objects and from these, built different types of SVM models for predicting the nouns and the adjectives for the objects. We compared the results of learning nouns and adjectives with classifiers that directly try to link nouns and adjectives with the appearances of objects.

Table 5. Noun prediction for novel objects using 3 different models (see Table 4 for pictures of the objects)

ID	A_{48}-NL	A_8-NL	SNL
1	box (74 %)	cylinder (42 %)	box (97 %)
2	ball (83 %)	ball (44 %)	ball (97 %)
3	cylinder (87 %)	cylinder (39 %)	cylinder (95 %)
4	box (94 %)	cylinder (38 %)	cylinder (86 %)
5	box (89 %)	cylinder (35 %)	box (94 %)
6	cup (89 %)	cylinder (44 %)	box (46 %)
7	box (89 %)	box (32 %)	box (93 %)
8	cup (89 %)	cylinder (44 %)	cup (98 %)

We showed that, by using learned affordances, iCub can predict adjectives with more accuracy than the direct mode. However, for the nouns, direct methods are better. This suggests that a subset of adjectives describing objects in a language can be learned from the affordances of objects. We also demonstrated that explicit behavior information in learning adjectives can provide better representations. It is important to note that these findings are subject to the sensorimotor limitations of the robot, which are maintained by the number and the quality of the behaviors and the properties of the perceptual system. For example, had we included a behavior to try to fill objects with some liquid, the *cups* concept would be much easier to be formed and predicted. A sample video footage can be viewed at http://youtu.be/DxLFZseasYA

Acknowledgements. This work is partially funded by the EU project ROSSI (FP7-ICT-216125) and by TÜBİTAK through projects 109E033 and 111E287. The authors Onur Yuruten, Kadir Uyanik and Asil Bozcuoglu acknowledge the support of TÜBİTAK 2210 scholarship program.

References

1. Harnad, S.: The symbol grounding problem. Physica D: Nonlinear Phenomena 42(1-3), 335–346 (1990)
2. Borghi, A.M.: Object concepts and embodiment: Why sensorimotor and cognitive processes cannot be separated. La Nuova Critica 49(50), 90–107 (2007)
3. Gibson, J.J.: The ecological approach to visual perception. Lawrence Erlbaum (1986)

4. Dag, N., Atil, I., Kalkan, S., Sahin, E.: Learning affordances for categorizing objects and their properties. In: IEEE ICPR. IEEE (2010)
5. Atil, I., Dag, N., Kalkan, S., Şahin, E.: Affordances and emergence of concepts. In: Epigenetic Robotics (2010)
6. Yu, C., Ballard, D.H.: On the integration of grounding language and learning objects. In: Proc. 19th Int. Conf. on Artifical Intelligence, AAAI 2004, pp. 488–493. AAAI Press (2004)
7. Carbonetto, P., de Freitas, N.: Why can't jose read? the problem of learning semantic associations in a robot environment. In: Proc. the HLT-NAACL 2003 Workshop on Learning Word Meaning from Non-Linguistic Data, pp. 54–61 (2003)
8. Nehaniv, C.L., Saunders, J., Lyon, C.: Robot learning of lexical semantics from sensorimotor interaction and the unrestricted speech of human tutors. In: Proc. 2nd Int. Symp. New Frontiers Human-Robot Interact. ASIB Convent. (2010)
9. Cangelosi, A.: Evolution of communication and language using signals, symbols, and words. IEEE Tran. on Evolutionary Computation 5(2), 93–101 (2001)
10. Cangelosi, A.: Grounding language in action and perception: From cognitive agents to humanoid robots. Physics of Life Reviews 7(2), 139–151 (2010)
11. Şahin, E., Çakmak, M., Doğar, M.R., Uğur, E., Üçoluk, G.: To afford or not to afford: A new formalization of affordances toward affordance-based robot control. Adaptive Behavior 15(4), 447–472 (2007)
12. Montesano, L., Lopes, M., Bernardino, A., Santos-Victor, J.: Learning object affordances: From sensory–motor coordination to imitation. IEEE Tran. on Robotics 24(1), 15–26 (2008)
13. Stoytchev, A.: Learning the Affordances of Tools Using a Behavior-Grounded Approach. In: Rome, E., Hertzberg, J., Dorffner, G. (eds.) Towards Affordance-Based Robot Control. LNCS (LNAI), vol. 4760, pp. 140–158. Springer, Heidelberg (2008)
14. Kraft, D., Pugeault, N., Baseski, E., Popovic, M., Kragic, D., Kalkan, S., Wörgötter, F., Krüger, N.: Birth of the object: Detection of objectness and extraction of object shape through object action complexes. International Journal of Humanoid Robotics 5(2), 247–265 (2008)
15. Ugur, E., Şahin, E.: Traversability: A case study for learning and perceiving affordances in robots. Adaptive Behavior 18(3-4), 258–284 (2010)
16. Fitzpatrick, P., Metta, G., Natale, L., Rao, S., Sandini, G.: Learning about objects through action - initial steps towards artificial cognition. In: IEEE ICRA (2003)
17. Detry, R., Kraft, D., Buch, A.G., Kruger, N., Piater, J.: Refining grasp affordance models by experience. In: IEEE ICRA, pp. 2287–2293 (2010)
18. Ugur, E., Oztop, E., Şahin, E.: Goal emulation and planning in perceptual space using learned affordances. Robotics and Autonomous Systems 59(7-8) (2011)
19. Ugur, E., Şahin, E., Oztop, E.: Affordance learning from range data for multi-step planning. In: Int. Conf. on Epigenetic Robotics (2009)
20. Montesano, L., Lopes, M., Bernardino, A., Santos-Victor, J.: A computational model of object affordances. In: Advances in Cognitive Systems. IET (2009)
21. Sinapov, J., Stoytchev, A.: Learning and generalization of behavior-grounded tool affordances. In: IEEE 6th International Conference on Development and Learning, ICDL 2007, pp. 19–24 (July 2007)
22. Sinapov, J., Stoytchev, A.: Detecting the functional similarities between tools using a hierarchical representation of outcomes. In: 7th IEEE International Conference on Development and Learning, ICDL 2008, pp. 91–96 (August 2008)
23. Montesano, L., Lopes, M., Bernardino, A., Santos-Victor, J.: Learning object affordances: From sensory–motor coordination to imitation. IEEE Tran. on Robotics 24(1), 15–26 (2008)

24. Montesano, L., Lopes, M., Melo, F., Bernardino, A., Santos-Victor, J.: A computational model of object affordances. Advances in Cognitive Systems 54, 258 (2009)
25. Rusu, R.B., Cousins, S.: 3d is here: Point cloud library (pcl). In: IEEE ICRA, pp. 1–4. IEEE (2011)
26. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: Proc. 9th Int. Workshop on Machine Learning, pp. 249–256 (1992)
27. Qin, A.K., Suganthan, P.N.: Robust growing neural gas algorithm with application in cluster analysis. Neural Networks 17(8-9), 1135–1148 (2004)

Learning and Adaptation of Sensorimotor Contingencies: Prism-Adaptation, a Case Study

Gert Kootstra[1], Niklas Wilming[2], Nico M. Schmidt[3], Mikael Djurfeldt[4],
Danica Kragic[1], and Peter König[5]

[1] CAS-CVAP, CSC, Royal Institute of Technology (KTH)
{kootstra,dani}@kth.se
[2] Institute of Cognitive Science, University of Osnabrück
nwilming@uos.de
[3] AI Lab, University of Zürich
nico.schmidt@uzh.ch
[4] PDC, Royal Institute of Technology (KTH)
mdj@kth.se
[5] Institute of Cognitive Science, University of Osnabrück; Department of Neurophysiology and Pathophysiology, University Medical Center Hamburg-Eppendorf
pkoenig@uos.de

Abstract. This paper focuses on learning and adaptation of sensorimotor contingencies. As a specific case, we investigate the application of prism glasses, which change visual-motor contingencies. After an initial disruption of sensorimotor coordination, humans quickly adapt. However, scope and generalization of that adaptation is highly dependent on the type of feedback and exhibits markedly different degrees of generalization. We apply a model with a specific interaction of forward and inverse models to a robotic setup and subject it to the identical experiments that have been used on previous human psychophysical studies. Our model demonstrates both locally specific adaptation and global generalization in accordance with the psychophysical experiments. These results emphasize the role of the motor system for sensory processes and open an avenue to improve on sensorimotor processing.

Keywords: Sensorimotor contingencies, prism-adaptation, motor learning/ adaptation, body maps, inverse kinematics.

1 Introduction

Humans adapt easily to changes in sensorimotor coordination during development and adulthood. A remarkable demonstration is adaptation to prism glasses that displace the visual field horizontally by a constant angle. Despite such drastic changes of a sensorimotor relationship, eye-hand coordination quickly adapts [6].

This is, however, not a passive process but requires active exploration [4, 5]. Prism adaptation is specific to the involved body parts [7, 15] and actions [1, 7, 16]. The adaptation to changed sensorimotor dependencies therefore appears to crucially depend on the ability to relate one's own motor actions to their observed sensory consequences. This bears a striking resemblance to the concept of sensorimotor contingencies (SMC) [11], which also stresses the importance of action for perception [3]. The case

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 341–350, 2012.

of prism adaptation is well investigated and therefore may serve as a prime example for investigating how the brain achieves “mastery of a sensorimotor contingency” [11].

When prisms are donned, subject’s pointing movements are offset due to the visual displacement. However, with repeated movements this offset diminishes and original performance restores. When subsequently the prisms are removed, pointing movements are offset in the opposite direction. This *aftereffect*, i.e. the difference in pointing between pre- and post-exposure, is a convenient measure of the degree of adaptation.

The adaptation is thought to combine two separate processes: Recalibration and realignment [15]. Recalibration is believed to utilize a cognitive learning strategy that quickly reduces pointing errors. The effects of recalibration are local to specific actions. Recalibration is quantified by the aftereffect measured with the identical movement that was carried out during the prism exposure phase [15]. Realignment, in contrast, is thought to be an automatic process that aligns, for example, visual and proprioceptive maps. It reveals a global generalization of adaptation to new pointing targets and actions. Realignment is measured by actions not practiced during prism exposure.

Redding and Wallace observe that the realignment effect is modulated by different kinds of feedback during exposure. When participants could see their own hand-movement (concurrent feedback) during exposure, Redding and Wallace [15] observed a small shift of “visual straight ahead” and a large shift of “proprioceptive straight ahead”. This pattern was reversed when participants could only see the end-position of their hand (terminal feedback) during exposure. Related to this, Redding and Wallace found that the aftereffect generalized differently to targets not shown during exposure. Specifically, concurrent feedback produced aftereffects that increased for targets in the direction of the prismatic shift, whereas they decreased for terminal feedback. Both results are explained by two different references: During concurrent feedback the visual system acts as a reference and the proprioceptive system is aligned to it and during terminal feedback the situation is reversed.

In this paper, we investigate properties of recalibration and realignment and their dependence on different types of feedback. For this purpose, a computational model of prism adaptation is developed for the control of a simulated robotic arm and subjected to the identical experiments focused on eye-hand coordination of the previous psychophysical study [15]. Specifically, we test the hypothesis that the different forms of adaptation aftereffects can be described in a unified approach based on the concept of sensorimotor contingencies. We take as our starting point that adaptive agents have to relate changes in perception to their own actions. At the same time we strive for a computational description of recalibration and realignment to foster our understanding of sensorimotor adaption and for facilitating the development of versatile robotic agents. The ability to quickly learn eye-hand coordination from own experience makes manual calibrations redundant, which is interesting, for instance, for robotic grasping [12].

2 Computational Model of Prism Adaptation

The contingency between sensory and motor signals involved in pointing movements is captured by forward and inverse-kinematic models. The forward model defines the position of the effector, e.g. the hand, based on the joint angles of the arm. The inverse-kinematic model provides the joint configuration of the arm necessary to reach a

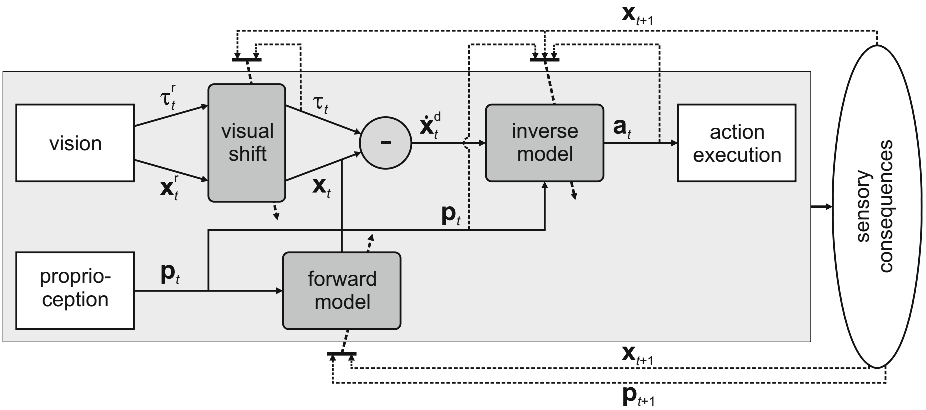


Fig. 1. Theoretical model, including three sub-modules: the visual shift, the forward model, and the inverse model. Based on visual and proprioceptive observations, the model estimates the action to get the end effector to the target pose. All three sub-modules learn on-line using the sensory consequences of the executed action as observed by the system itself. Information flow for control is indicated by solid arrows, and the flow for learning by dashed arrows. The variables are explained in the text.

specific target, and thereby allows direct control of a robot. Motivated by [2, 10] and in accordance with the SMC theory [11] we use a differential inverse-kinematic model that generates the action necessary to reach a desired *change* in visual position of the end-effector. Given a current joint configuration, $\mathbf{q}$, and a desired *change* in pose of the end effector, $\dot{\mathbf{x}}$, the necessary *change* in joint angles, $\dot{\mathbf{q}}$, is determined by the model, expressed by the mapping $(\mathbf{q}, \dot{\mathbf{x}}) \mapsto \dot{\mathbf{q}}$. For small enough changes, the optimal gradient $\dot{\mathbf{q}}$ for any given state $\mathbf{q}$ is well defined and resolves ambiguities inherent in inverse-kinematics.

When the robotic system is subject to changes, for instance, due to mechanical wear or damage or adaptation to prisms, the forward and inverse mappings are not fixed, but changes over time. This calls for adaptive systems that can learn and adapt the inverse-kinematics model from own experience [9]. Others have used, for instance, locally-weighted projection regression [2, 17], Gaussian-process regression (GPR) [13], and local Gaussian-process regression [10]. Here we use Gaussian-process regression (GPR) to learn the inverse kinematics, since it has the best reported performance [10, 13] at the cost of higher computational load. But note, that the GPR can be easily substituted by other regression methods, if real-time performance is more of an issue.

In total, our model consists of three sub-modules: the forward model, the inverse model, and the visual-shift model (see Figure 1). In the following, we give the model details of all components in turn.

2.1 Forward Model and Inverse Model

Based on the proprioceptive information $\mathbf{p}_t$, the *forward model* gives an estimation of the pose of the end effector in visual coordinates $\mathbf{x}_t$ at time t (see Figure 1). This estimation is used when the end effector cannot be observed visually. The *inverse model* provides an estimate of the necessary action, $\mathbf{a}_t$, for instance a change in joint angles, based on $\mathbf{p}_t$ and the desired change in pose, $\dot{\mathbf{x}}_t^{\mathbf{d}}$, which is determined based on the difference between the target and the end effector, $\dot{\mathbf{x}}_t^{\mathbf{d}} = \tau_t - \mathbf{x}_t$ (see Figure 1).

Using GPR, an estimation is made of the functions $\mathbf{a}_t = f(\mathbf{p}_t, \dot{\mathbf{x}}_t^{\mathbf{d}})$ for the inverse and $\mathbf{x}_t = g(\mathbf{p}_t)$ for the forward model, based on a set of training examples, which the system acquires from own experience. GPR has a few hyperparameters, including $\{\lambda_1^2, \ldots, \lambda_D^2\}$, which are the characteristic length-scales of the different dimensions of the squared-exponential kernel, where D is the dimensionality of the input to the GPR. The hyperparameters are learned from data by maximizing the marginal likelihood. For details on GPR, we refer to [13].

Recency Effect. A problem with a standard GPR implementation of the forward and inverse models is that adaptation to changed SMCs is slow. When prism glasses are donned, the old training samples contribute as strong to the estimation as the new samples, which results in a rather slow adaptation. To increase speed of adaptation and to intensify the aftereffect, we include a forgetting mechanism using a recency effect, such that more recent training data make a stronger contribution to the estimation. Specifically, we add a time dimension to the input, resulting in input $\mathbf{z}^{\mathrm{i}} = \{\mathbf{p}, \dot{\mathbf{x}}, t\}$ for the inverse, and $\mathbf{z}^{\mathrm{f}} = \{\mathbf{p}, t\}$ for the forward model, both with a time constant λ_T. This characteristic length-scale parameter is not included in the optimizing of the hyperparameters, but instead used as a free parameter to control the recency effect, which is strong for low values of λ_T, whereas for $\lambda_T \to \infty$, the effect is absent.

Execution and On-Line Learning. The Gaussian process regressors that implement the inverse and forward models are continuously updated while the robot performs its task, thus on-line learning the eye-hand coordination. In execution, the GPRs are used to predict the output of the models based on the SMC experience, resulting in an action. When the robot executes this action, it results in a movement of the arm. The system then observes the visual and proprioceptive consequences of that action and uses the observation of these new SMCs as training samples. In case of the forward model, the training data is $\{\mathbf{p}_{t+1}, \mathbf{x}_{t+1}, t\}$. The training data for the inverse model is $\{\mathbf{p}_t, \dot{\mathbf{x}}_t^{\mathbf{o}}, \mathbf{a}_t, t\}$, where $\dot{\mathbf{x}}_t^{\mathbf{o}} = \mathbf{x}_{t+1} - \mathbf{x}_t$ is the observed change in end effector.

2.2 Visual-Shift Model

The visual system provides information about the pose of the end effector and the target in visual coordinates. The visual-shift model applies a transformation, T, to these visual observations, so that the retinotopic (or camera) coordinate system is transformed into an internal visual coordinate system: $\mathbf{x} = T(\mathbf{x}^{\mathrm{r}})$ and $\tau = T(\tau^{\mathrm{r}})$, where $\mathbf{x}_t^{\mathrm{r}}$ is the pose of the end effector and τ_t^{r} the pose of the target, both in the retinotopic coordinate system.

The transformation T is updated based on the visually observed error in pointing, which is caused by the error of the model in predicting the effects of the applied action. T needs to counterbalance the visual transformation caused by the prism glasses. This can be done in different ways, but since in our experimental setup (see Section 3.1), the prism effect causes a rotation of visual observations, we chose to implement T as a rotation of the visual coordinates as well: $\mathbf{x} = T(\mathbf{x}^{\mathrm{r}}) = R^{\theta} \cdot \mathbf{x}^{\mathrm{r}}$, where the rotation matrix R^{θ} applies a rotation over θ. The rotation angle θ is updated by the system at the end of each movement by $\theta_{t+1} = \theta_t + \eta \cdot \varepsilon$, where ϵ is the visually-observed terminal error. This error is based on the angular difference between the desired pose of the end effector at the end of the trajectory, τ_M, and the actual pose observed by the system after movement, $\mathbf{x}_{M+1}$: $\varepsilon = \angle\tau_M - \angle\mathbf{x}_{M+1}$, where M is the last time step in the action sequence. $\eta \in [0, 1]$ is the transformation learning rate, which determines the influence of the visual-shift model in the complete adaptation system.

2.3 Concurrent and Terminal Feedback

As in [15], we distinguish two different types of feedback: concurrent and terminal. In the first case, the end effector is continuously observed visually to obtain the internal pose $\mathbf{x}_t$, which is used by the inverse model to determine the action. This forms a visual closed-loop control system, where the effect of the action is visually observed and used in the next iteration. In the terminal-feedback condition, there is no visual closed-loop control, since the internal pose of the end effector is estimated based on proprioceptive information using the forward model. As a results, there will be a difference in terminal pointing error in the two conditions when SMCs are altered through prism exposure.

Another consequence of the feedback condition is the available data to train the inverse and forward model. In case of concurrent feedback, the new SMCs can reliably be observed over the complete trajectory. However, in the terminal condition, the SMCs related to the individual actions need to be estimated from the terminal feedback and will be incorrect due to the non-linear relations involved. We generate training data in the terminal-feedback condition by interpolating the internal visual pose of the end effector based on the visually observed end pose and the start pose estimated by the forward model.

3 Experiments

3.1 Simulation and Model Setup

We use a 2D simulated robotic setup to test our computational model of prism adaptation, see Figure 2a-c. The setup consists of a two degrees-of-freedom arm, a vision sensor observing the 2D position of the end effector, $\mathbf{x}_t = \{x_t^{\mathrm{x}}, x_t^{\mathrm{y}}\}$, and the target, $\tau_t = \{\tau_t^{\mathrm{x}}, \tau_t^{\mathrm{y}}\}$, and proprioceptive sensors in each of the joints, $\mathbf{p_t} = \{p_t^1, p_t^2\}$ giving information about the joint angles. In this setup, an action is a change in joint angles, $\mathbf{a}_t = \dot{\mathbf{q}}_t$. The visual observations are made from the bird's-eye perspective. In the experiment, pointing is done at two different heights, high and low, causing different arm poses during pointing. To experiment with the similarity of different poses, we add an arm-pose dimension to the input of the GPRs, with an associated length-scale parameter λ_{P}, which is by default set to 2.0 unless stated otherwise.

In a pointing trial, a target is positioned at a specific angle with respect to the robot. The system observes the target's position, $\mathbf{g}$, and then plans a target trajectory, $\{\tau_1, \ldots, \tau_M\}$, where $M = 5$ is the number of actions involved in the pointing trajectory (see Figure 2d). The prism glasses are implemented as a rotation of the visual coordinates, so that $\mathbf{y}^{\mathrm{r}} = R^{\gamma} \cdot \mathbf{y}^{\mathrm{t}}$, where $\mathbf{y}^{\mathrm{t}}$ is the true position, $\mathbf{y}^{\mathrm{r}}$ is the position in retinotopic

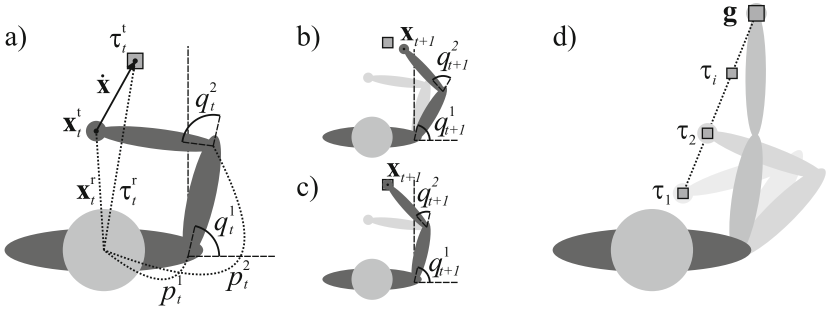


Fig. 2. Experimental setup. a) The robot has joint angles $\mathbf{q}_t$, the end effector is at true pose $\mathbf{x}_t^{\mathrm{t}}$, and the target at true pose τ_t^{t}. However, this information is not known to the system. Instead, the system observes (dotted lines) the proprioceptive signals $\mathbf{p_t}$, and the retinotopic coordinates of the end effector $\mathbf{x}_t^{\mathrm{r}}$ and the target τ_t^{r}. The model will suggest an action, which will change the joint angles. b) At first, the SMCs will not be correctly learned, and the action will result in an incorrect movement of the arm. c) After some time, the model will correctly estimate the action, so that $\mathbf{x}_{t+1} \equiv \tau_t$. d) Based on the visual target at position $\mathbf{g}$, the system plans a target trajectory giving a set of intermediate targets $\{\tau_1, \ldots, \tau_M\}$.

coordinates, and R^γ is a rotation matrix applying deviation angle γ. The application of the virtual prism glasses effects all visual observations, that is the observation of the end effector and the target.

We set the recency effect used in the forward and inverse models to $\lambda_{\mathrm{T}} = 1000$, and the transformation learning rate to $\eta = 0.35$. These parameters change the slopes of the learning curves and the influence of the forward and inverse models versus the visual-shift model. The general results are robust to small changes in these values.

3.2 Experimental Setup

To compare the performance of our model to psychophysical data, we adopt the experiments performed by Redding and Wallace [15]. The task of the system is to point to visual targets. In the experiment, two different starting positions are used. In the proximal starting position, the end effector starts at the origin, and with a low arm pose. The distal starting position is halfway to the target, and with a high arm pose.

When the system has been initialized, as described below, we perform two sets of pre-tests, one for realignment and one for recalibration, to measure the performance before prism exposure. The prisms are then turned on in the simulation by rotating the visual observations over the origin with an angle of 11.4°. Under prism exposure, the system performs 12 consecutive pointing movements from the distal starting position with either concurrent or terminal feedback. After that, the prisms are switched off, and the recalibration and realignment tests are performed again as post-tests. No adaptation learning is going on during the pre- and post-tests.

The realignment tests consist of a visual-shift test, a proprioceptive-shift test and a total-shift test, and are all done from the proximal starting position, i.e., different from the starting position during prism exposure. In the visual-shift test we measure subjective *straight-ahead*, by reading out $-\theta$, the negative value of the applied transformation

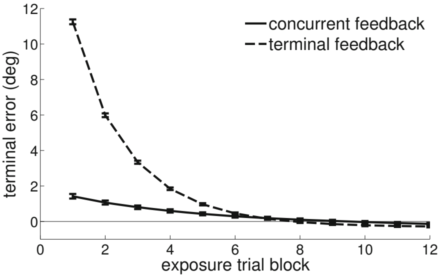


Fig. 3. The terminal pointing error as a function of pointing trial during prism exposure for concurrent and terminal feedback. The values are the means over 20 trials, and the error bars give the 95% confidence intervals.

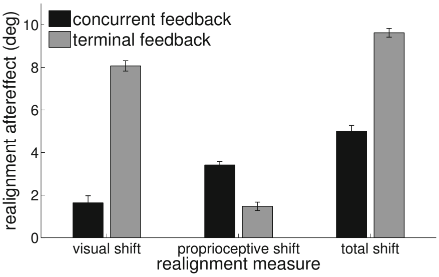


Fig. 4. The terminal error or aftereffect for the different realignment measures. With concurrent feedback, the visual shift remains small, while the proprioceptive shift is large, the reverse is true in case of terminal feedback.

in the visual-shift model. The proprioceptive-shift test measures proprioceptive straight ahead by having the system point without a visual target. To do so, the system sets a virtual target at 0° in internal coordinates. Finally, during the total-shift test, a visual target is presented at different angles $\{-20^\circ, -15^\circ, \dots, 20^\circ\}$, and the system points to the targets. Pointing is always done without visual feedback.

The recalibration test is similar to the total-shift test, but the system is tested with the distal starting position, i.e., similar to the starting position during prism exposure. Moreover, we perform the test using different values for λ_P, to change the level of similarity in arm pose between distal and proximal starting conditions.

An experimental trial starts by initializing the system. The robot initiates 300 random movements in different parts of the work space and using the two different arm poses. Next, the system specializes on the pointing task by performing 24 pointing movements of $M = 5$ actions, using three different target positions (at -15°, 0° and 15°) and two different arm poses. Since this initialization is a noisy process and influences the performance of the system during the experiment, we repeat the experiment 20 times and report the mean values and 95% confidence intervals.

4 Results

Adaptation of pointing during prism exposure for the concurrent and terminal feedback condition is shown in Figure 3. In both conditions, the system quickly adapts to the prism effect, demonstrating accurate pointing behavior after 8 pointing trials. The error in the terminal condition is considerably larger than in the concurrent condition, which is expected, since the trajectory can be adjusted during pointing in the concurrent case. Negative error values can be observed in later pointing trials, showing overcompensation by the model, which is due to the collaboration of the forward and inverse models with the visual-shift model. These results correspond well with the psychophysical results observed by Redding and Wallace [15].

The size of the realignment aftereffect is shown in Figure 4. There is a clear difference in visual shift and proprioceptive shift between the concurrent and terminal

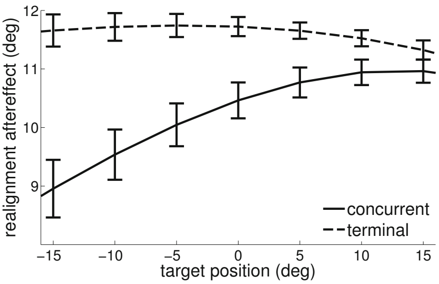


Fig. 5. The realignment aftereffect as a function of target position for concurrent and terminal feedback. The curves show different slopes depending on the feedback condition.

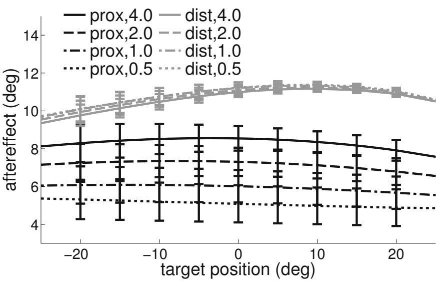


Fig. 6. The aftereffect for target positions tested at the proximal (realignment) and distal (recalibration) starting positions. λ_P indicates levels of pose similarity.

condition, matching the psychophysical results [14, 15]. The visual shift is small in the concurrent condition and large in the terminal condition. In the proprioceptive test, the aftereffects are reversed, with a large effect in the concurrent and a small effect in the terminal condition. The total aftereffects are approximately the sum of the visual- and proprioceptive aftereffects.

To investigate the generalization over space, we next investigate the total-shift (realignment) aftereffects as a function of target position for concurrent and terminal feedback (see Figure 5). With concurrent feedback, the aftereffect shows a positive slope towards the prismatic shift (11.4°), whereas with terminal feedback, the curve shows a slight negative slope. This matches the psychophysical results [15].

Generalization over different actions is investigated by the aftereffects for the total shift and recalibration tests with proximal and distal starting position as shown in Figure 6. We use four different levels of similarity between the arm pose at those two starting positions, $\lambda_P = \{4.0, 2.0, 1.0, 0.5\}$, where a higher value is a higher level of similarity. The results for the concurrent and terminal feedback condition are combined, as was done in [15]. The aftereffect is largest when the starting position is identical to the exposure phase, that is, the distal position. The curves for the distal position show a local generalization effect, with lower values for target positions that have not been used during prism exposure. This can be explained by the local learning in GPR used in forward and inverse model. The curves for the proximal position are more flat, indicating a more global generalization effect caused by the visual-shift model. Reducing the similarity in arm pose between exposure condition (distal) and the proximal condition during recalibration testing, i.e., lower values of λ_P, results in lower and more flat curves, indicating that aftereffects are weaker and mainly dominated by the global effect. These results correspond with psychophysical observations [1, 15].

5 Discussion

We presented a computational model capable of learning the sensorimotor contingencies involved in pointing, and of adapting to changes in these contingencies. Moreover, it accounts for the observations made in human prism-adaptation studies [1, 14, 15], i.e. adaptation is achieved jointly through a specific local and a general global effect. Whereas local adaptation, or recalibration, is specific for the sensorimotor contingencies involved in the trained action, realignment, shows a global effect and generalizes to different actions

Using two feedback conditions, the realignment shows fundamentally different aftereffects for visual and proprioceptive tests. Where the visual shift is small and the proprioceptive shift is large for concurrent feedback, the reverse is true for terminal feedback. In our model, the visual shift is larger during terminal feedback because the observed pointing error at the end of the terminal feedback trials is larger compared to concurrent feedback trials. During concurrent feedback the end effector is continuously observed and the pointing error can be reduced while the movement is executed. In contrast, the results for the proprioceptive test are explained by the forward and inverse models. Since the sensory consequences of actions are directly observable in the concurrent condition, low-error training data is available which leads to quick adaptation of the inverse model and thus a large proprioceptive after effect. In the terminal condition, these training data need to be estimated based on the terminal observation, causing them to be less correct.

Our model gives a different explanation of prism adaptation compared to Redding and Wallace [15]. Where they consider recalibration to be a cognitive learning strategy and realignment to be an automatic process aligning different spatial maps, our model solely consists of automatic and low-level processes. Although cognitive strategies are an equally valid explanation, our model shows that similar effects can be reached with a simpler mechanism. Furthermore, in our case, the model does not include an explicit proprioceptive-shift model. Instead the results for the proprioceptive-shift test can be explained by adaptation of the forward and inverse models to changed SMCs. The presented model offers a potential system implementation for learning sensorimotor contingencies and emphasize the importance of the motor system for perception [8].

Currently, we assume a setup with a fixed eye-head system. However, in humans, both eye-head and head-hand systems are involved. The presented work is a first step towards modeling both systems. The current model is not able to learn multiple sensorimotor mappings. To account for dual adaptation effects observed in alternating prism-exposure experiments [18], future additions to the model will be necessary.

The model presented is directed at fostering our understanding of human adaptation as well as to improve on the current state of robotic systems. Inverse-kinematics learning with regression models has already been addressed by others, see e.g., [2, 9], but the addition of a recency effect and the synergy with the visual-shift model results in a fast adaptation to changes in sensorimotor contingencies.

Acknowledgments. This work was supported by the EU through the project eSMCs (FP7-IST-270212), and by the Swedish Foundation for Strategic Research.

Reference

[1] Baraduc, P., Wolpert, D.: Adaptation to a visuomotor shift depends on the starting posture. Journal of Neurophysiology 88(2), 973–981 (2002)

[2] D'Souza, A., Vijayakumar, S., Schaal, S.: Learning inverse kinematics. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2001)

[3] Einhäuser, W., Martin, K., König, P.: Are switches in perception of the necker cube related to eye position? European Journal of Neuroscience 20(10), 2811–2818 (2004)

[4] Held, R., Hein, A.V.: Adaptation of disarranged hand-eye coordination contingent upon re-afferent stimulation. Perceptual and Motor Skills 8(3), 87–90 (1958)

[5] Held, R., Schlank, M.: Adaptation to disarranged eye-hand coördination in the distance-dimension. The American Journal of Psychology 72(4), 603–605 (1959)

[6] Kornheiser, A.: Adaptation to laterally displaced vision: A review. Psychological Bulletin 83(5), 783–816 (1976)

[7] Martin, T., Keating, J., Goodkin, H., Bastian, A., Thach, W.: Throwing while looking through prisms. ii. Specificity and storage of multiple gaze-throw calibrations. Brain 119(4), 1199–1212 (1996)

[8] Nagel, S., Carl, C., Kringe, T., Märtin, R., König, P.: Beyond sensory substitution – learning the sixth sense. Journal of Neural Engineering 2, R13 (2005)

[9] Nguyen-Tuong, D., Peters, J.: Model learning for robot control: A survey. Cognitive Processing 12(4), 319–340 (2011)

[10] Nguyen-Tuong, D., Seeger, M., Peters, J.: Model learning with local gaussian process regression. Advanced Robotics 23(15), 2015–2034 (2009)

[11] O'Regan, J., Noe, A.: A sensorimotor account of vision and visual consciousness. Behavioral and Brain Sciences 24(5), 939–1031 (2001)

[12] Popović, M., Kootstra, G., Jørgensen, J.A., Kragic, D., Krüger, N.: Grasping unknown objects using an early cognitive vision system for general scene understanding. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 987–994. IEEE, San Francisco (2011)

[13] Rasmussen, C.E., Williams, C.: Gaussian Processes for Machine Learning. The MIT Press (2006)

[14] Redding, G.M., Wallace, B.: Components of prism adaptation in terminal and concurrent exposure: Organization of the eye-hand coordination loop. Perception and Psychophysics 44(1), 59–68 (1988)

[15] Redding, G.M., Wallace, B.: Generalization of prism adaptation. Journal of Experimental Psychology: Human Perception and Performance 32(4), 1006–1022 (2006)

[16] Redding, G.M., Wallace, B.: Intermanual transfer of prism adaptation. Journal of Motor Behavior 40(3), 246–262 (2008)

[17] Schaal, S., Atkeson, C.G., Vijayakumar, S.: Scalable techniques from nonparametric statistics for real-time robot learning. Applied Intelligence 17(1), 49–60 (2002)

[18] Welch, R., Bridgeman, B., Anand, S., Browman, K.: Alternating prism exposure causes dual adaptation and generalization to a novel displacement. Perceptual Psychophysics 54(2), 195–205 (1993)

Unsupervised Learning of a Reduced Dimensional Controller for a Tendon Driven Robot Platform

Hugo Gravato Marques[1,2], Philip Schaffner[1], and Naveen Kuppuswamy[1]

[1] University of Zurich,
Institute for Informatics, AI Lab.
Zurich 8050, Switzerland
[2] ETH,
Dep. of Mechanical and Process Engineering, BIRL,
Zurich 8092, Switzerland
hgmarques@gmail.com

Abstract. In this paper we present a developmental framework to carry out goal-oriented learning in a low-dimensional space. The framework uses two stages of learning: one to synthesise a set of motor synergies and reduce the dimensionality of the control space in an unsupervised manner, and another to carry out supervised learning in the reduced control space. We test our framework in a reaching task carried out on a (real) tendon-driven robot actuated by four artificial muscles. Our results show that the robot is capable of learning to reach using a reduced control space using no prior information about its body apart from that inherent to the unsupervised and supervised learning rules.

1 Introduction

Current theories of biological motor control in mammals feature almost invariably some kind of hierarchical and modular architecture, where low-level motor primitives (also called stored motor programs, motor units, or muscle synergies), are combined by higher-level mechanisms to produce coordinated behaviour [4][1]. In this context, one of the most prominent areas of research is the identification of motor primitives. This process typically entails some form of factor analysis (e.g. PCA) applied on sensory and motor data collected during behaviour, the outcome of which results on a small number of abstract control units which can explain the behavior of the animal in statistical terms. However, the presence of this statistical units in the animal is difficult to justify or validate, and many open questions remain of whether motor primitives are innate or can emerge from experience [10], [7].

[1] The research leading to these results has received funding from the European Community's 7th Framework Programme FP7 Cognitive Systems, Interaction, Robotics - under grant agreement no. 207212 – eSMCs – and 235065 – RobotDoC, Marie Curie ITN, and the Swiss NCCR Robotics. We thank Cristiano Alessandro for constructing the robot platform used in our experiments.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 351–360, 2012.

This question is paramount in artificial systems which make use of hierarchical motor architectures to solve issues of redundancy [11]. In artificial systems motor primitives are typically enforced based on the designer's intuition, and established independently of mechanical considerations, to fulfill a certain functional role (as in variants of the subsumption architecture [2]). But this solution is neither convenient (as primitives are synthesised without following any general principle), nor adaptive (as small unpredictable changes in the system dynamics, or in its environment, might render the primitives useless). In principle, a motor primitive should somehow reflect the regularities and the contingencies of the system to be controlled, rather then being designed independently of the body dynamics [8],[11], which is an argument often used in the field of developmental robotics [1],[5].

In this work we will favour the term "motor synergies" over that of "motor primitives" as the former highlights the co-activation of muscles required to achieve a given task. We present a developmental framework that synthesises a set of motor synergies guided by a self-exploration process. The number of synergies obtained is directly given by the task dimensionality which is lower than the number of actuated muscles. A supervised learning strategy is then used to find the appropriate combination of these synergies and to achieve a desired goal. Our framework has been tested successfully on a (real) tendon-driven pendulum robot actuated by four artificial muscles, where the goal of the robot is to reach to different targets in 2D space.

The reminder of this paper is organised as follows. The second section describes our developmental framework. The third section provides the implementation details of each mechanism in the framework. The fourth section describes the experimental results. The fifth section concludes the paper and provides the outlook of our research.

2 Framework Description

The schematic diagram of the framework proposed in this paper is shown in Figure 1. The framework entails two stages of learning: one carried out by an unsupervised learning process (ULP), and the other by a supervised learning process (SLP). The former synthesises the motor synergies based on Hebbian learning, while the latter uses the synthesised synergies to carry out goal-oriented learning in the resulting reduced control space.

The ULP consists of four interacting mechanisms: a musculoskeletal system (and its environment), a peripheral system, a mechanism capable of triggering spontaneous muscle twitches (SMTs), and a Hebbian-learning mechanism. It works as follows. First, SMTs produce spontaneous and independent contractions in individual muscles. Second, each of these contractions produces forces which are propagated through the musculoskeletal system (as well as through the environment where it is embedded). Third, the changes produced in the musculoskeletal system are captured by the various sensor modalities, which (fourth)

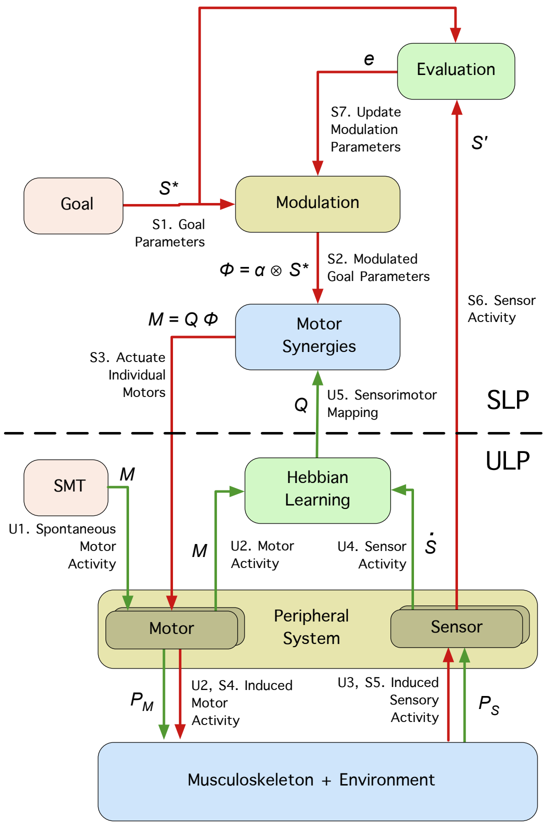
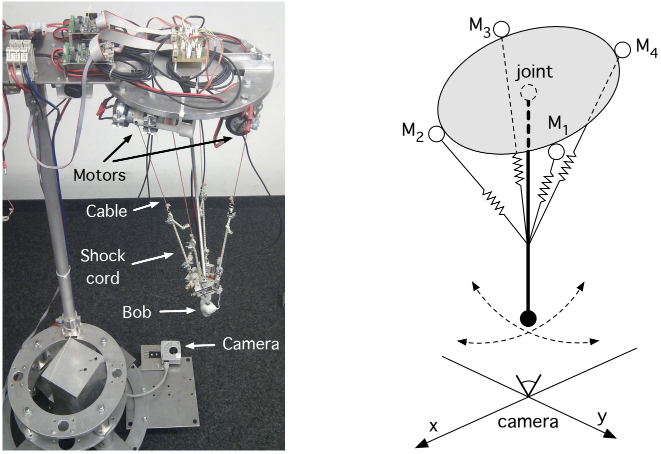


Fig. 1. The proposed developmental framework. The framework entails two processes (divided by a dashed line): a unsupervised learning process (ULP) and a supervised learning process (SLP); the former is shown in gray and the latter is shown in color and it is enclosed by the dashed rectangle. The sequence of events in each process is given by Ui for the ULP and Si for the SLP (see text).

convert them into sensor activity. Fifth, the correlation between the sensor and motor activity is used to synthesise the motor synergies. We have used elsewhere a variant of this scheme to self-organise spinal reflexes [7].

Once the motor synergies are synthesised they are used in the SLP. The SLP consists of an iterative process which aims to identify the appropriate modulation gains required to achieve a given goal. This process entails six mechanisms: a peripheral and a musculoskeletal mechanisms (which are shared with the ULP), a goal mechanism, a mechanism that processes the motor synergies, a modulation mechanism, and an evaluation mechanism. It works as follows. First, a goal is set in the system. Second, the parameters of the goal signals are modulated (i.e. scaled). Third, the modulated signals activate the motor synergies, which



(a)

(b)

Fig. 2. The experimental pendulum platform (a) and its schematic diagram (b)

combine the control of the individual muscles. Fourth the actuation of the different motors cause the musculoskeleton to move, which (fifth) induces sensory stimulation (as in the ULP). Sixth, the novel sensor activity is evaluated with respect to the goal. And seventh, the modulation gains are changed and the process is repeated iteratively.

3 Implementation

3.1 Musculoskeletal System and Environment

Our musculoskeletal system consists of a single-joint pendulum actuated by four artificial muscles (see Fig. 2). A camera placed at the bottom tracks the position and velocity of the pendulum. Each muscle consists of a DC Motor a cable and an elastic shock cord arranged in series (see [3],[6]). When the motor is actuated in one direction it reels the cable and creates an analogue to a muscle contraction; when it is actuated in the opposite direction it allows for the muscle to extend. Like their biological counterparts, the artificial muscles have asymmetrical conditioning, i.e. they can only produce force when contracting but not when extending. But unlike biological muscles, our muscles offer a strong resistance to passive extension (i.e. when they are OFF). To overcome this difference we implemented a force controller for each muscle which keeps it (actively) at a minimum tension value when the muscle is supposed to be relaxed.

In both the ULP and the SLP the system starts with all the muscles relaxed, i.e. in minimum tension mode. In this condition the pendulum moves to its resting position due to the effects of gravity. It is noteworthy that the usage of non-standard motors (the motors used are taken from cheap screw-drivers) combined with the minimum tension procedure introduces variances in the resting position of the pendulum. Nonetheless our architecture is robust to these issues as demonstrated by the results described in Sec. 4.

3.2 Peripheral System

The peripheral system provides the interface between the physical system and the neural (or computational) apparatus; it includes the sensory as well as the motor elements. In our platform, the sensory signal is a 2-element row vector which contains the velocity of the pendulum (obtained from the camera) in the two axes, $\dot{S} = \{\dot{x}, \dot{y}\}$. The derivatives have been filtered using the Savitzky-Golay-filter of 3rd order using a window size of 51. The motor system is 4-element row vector which defines the activity, M_i, of each muscle.

3.3 Unsupervised Learning Process

The single muscle twitches (SMTs) consist of short and spontaneous contractions of single muscles. In mammals this type of motor activity has been observed before birth as well as after birth, during sleep [9]. In our experiments the generation of SMTs is done by sequentially twitching one muscle after the other (each muscle is twitched 20 times). Each twitch consists of a square signal of amplitude $M_i = 4V$ and duration $0.5s$. We have tested different combinations of amplitude and duration and no qualitative impact has been observed on the results presented here.

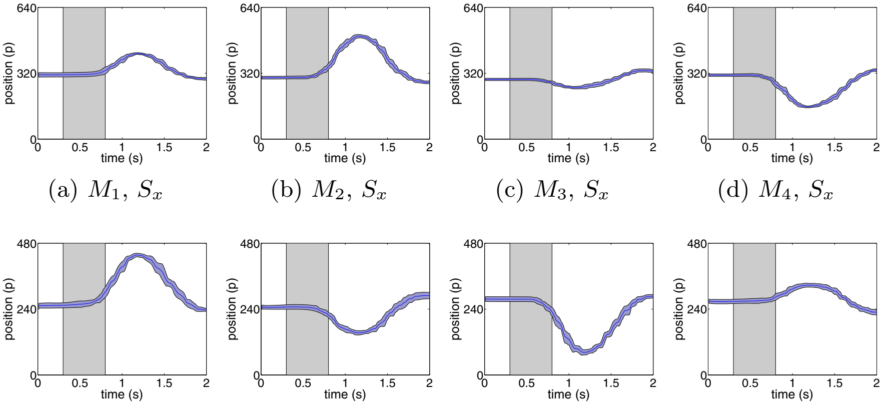
The Hebbian learning mechanism allows to identify the motor synergies based on the correlation between sensor and motor signals during the SMTs . All possible combinations between sensor and motor elements are considered. The motor synergies are described by a single matrix P:

$$P_{i,j} = \eta_{ij} \sum_{t=1}^{T} M_{i,t} \cdot \dot{S}_{j,t+L}, \qquad \eta_{i,j} = \bar{M}_i \left[max(\dot{S}) \sum_{t=1}^{T} M_{i,t} \right]^{-1} \tag{1}$$

where, P is a 4×2 matrix containing information about which motors can be recruited to produce a given sensor activation (positive or negative), η_{ij} is a scaling factor, T is the number of time samples, $M_{i,t}$ is the value of the motor signal M at time sample t, $\bar{M}_i$ is the maximum twitch amplitude of muscle i, $\dot{S}_{j,t}$ is the value of the sensor signal j at time sample t, and L is the lag between sensor and motor activity (obtained *a priori* through direct measurements).

3.4 Supervised Learning Process

In our platform a controller without motor synergies would require a four control signals (one for each motor); the use of muscle synergies reduces the number of controlled variables to only two. This paper investigates a reaching task, where the goal consists of a 2-element row vector, S^* which is defined by $S^* = S_d - S_0$,



(a) M_1, S_x (b) M_2, S_x (c) M_3, S_x (d) M_4, S_x

(e) M_1, S_y (f) M_2, S_y (g) M_3, S_y (h) M_4, S_y

Fig. 3. The raw sensor data collected during the ULP. The data is shown for the x (a-d) and y (e-h) sensor values for SMTs carried out in M_1 (a,e), M_2 (b,f), M_3 (c, g), and M_4 (d,h). Each plot shows the mean and standard deviation of the sensor data for 20 SMTs triggered by the respective muscle. The filled rectangle shows the duration of the twitch.

where S_d is the desired target position of the pendulum, and S_0 is the position of the pendulum at the beginning of each iteration.

The role of the modulation mechanism is to scale the goal signal. Intuitively, a simple multiplication between the goal signal, S^*, and the motor synergies P will move the pendulum in the direction of the goal (see next section). The modulation mechanism allows the amplitude of the movement in the target direction. The modulation mechanism is then given by:

$$\phi = \alpha \otimes S^* \tag{2}$$

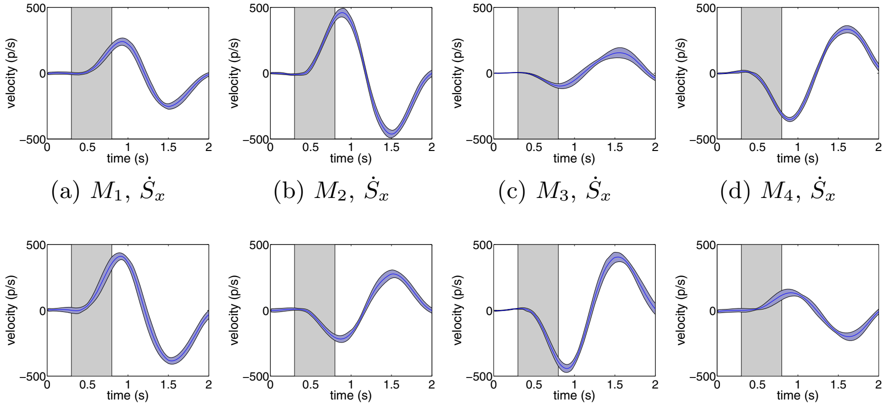
where ϕ is 2-element row vector containing the the modulated goal signals, α is a 2-element row vector containing the modulation gains, and $\otimes$ is the operator for element-wise multiplication.

The motor synergies are responsible to transform the modulated goal signal into individual motor activity. This is done by:

$$M = P\phi \tag{3}$$

The motor signals obtained are activated for a fixed period of time. Here, we used the same time as that used for the twitch duration, i.e. $0.5s$. At the end of each iteration the evaluation mechanism is used to measure the error, e, obtained between the goal vector, S^*, and the movement vector, S'_k, achieved by the pendulum at the end of each iteration. This is given by $S'_k = S_k - S_0$, where S_k is the final position at the end of iteration k. The error in each sensor signal, S, is used to modify the gain parameters according to a gradient descent scheme:

$$\alpha_{k+1} = \alpha_k - \tau \cdot e, \qquad e = (S^* - S') \otimes S^* \oslash |S^*| \tag{4}$$



(a) M_1, $\dot{S}_x$ (b) M_2, $\dot{S}_x$ (c) M_3, $\dot{S}_x$ (d) M_4, $\dot{S}_x$

(e) M_1, $\dot{S}_y$ (f) M_2, $\dot{S}_y$ (g) M_3, $\dot{S}_y$ (h) M_4, $\dot{S}_y$

Fig. 4. The filtered and derived sensor data collected during the ULP. The data is shown for the $\dot{S}_x$ (a-d) and $\dot{S}_y$ (e-h) sensor values for SMTs carried out in M_1 (a,e), M_2 (b,f), M_3 (c, g), and M_4 (d,h). Each plot shows the mean and standard deviation of the filtered and derived sensor data for 10 SMTs triggered by the respective muscle. The filled rectangle shows the duration of the twitch.

where α_k is the gain vector at iteration k, τ is the learning rate (set to 0.15 in our experiments), e is the estimated error in the two dimensions (x and y), $\oslash$ is the operator for element-wise division, and $|S^*|$ is the absolute value of the elements in vector S^*. The error is calculated as the difference between the desired movement vector, S^*, and the vector achieved S'_k; the division between S^* and $|S^*|$ enforces the pendulum to move in the same direction as the target. Note that the only parameters that are being modified during the SLP are the scaling factors, α_k, the dimensionality of which is the same as that of the goal S^* and it is independent of the number of muscles needed to achieve the task.

4 Results

The raw sensor signals S, collected during the ULP are shown in Fig. 3. Each plot shows the mean and the standard deviation of x and y sensor values for a total of 20 SMTs carried out in each muscle. As can be seen SMTs carried out by M_1 increase both the x and y positions of pendulum with respect to the camera. In contrast, SMTs carried out by M_2 increase the x position of the pendulum and decrease the y value. The filtered and derived sensor signals $\dot{S}$ are shown in Fig. 4. As can be observed the profiles match those in Fig. 3; the activation of M_1 produces a positive velocity in both the x and y axes, while the activation of M_2 produces a positive velocity in x and a negative velocity in y. It can also be observed that the relative sensor change is different for each muscle;

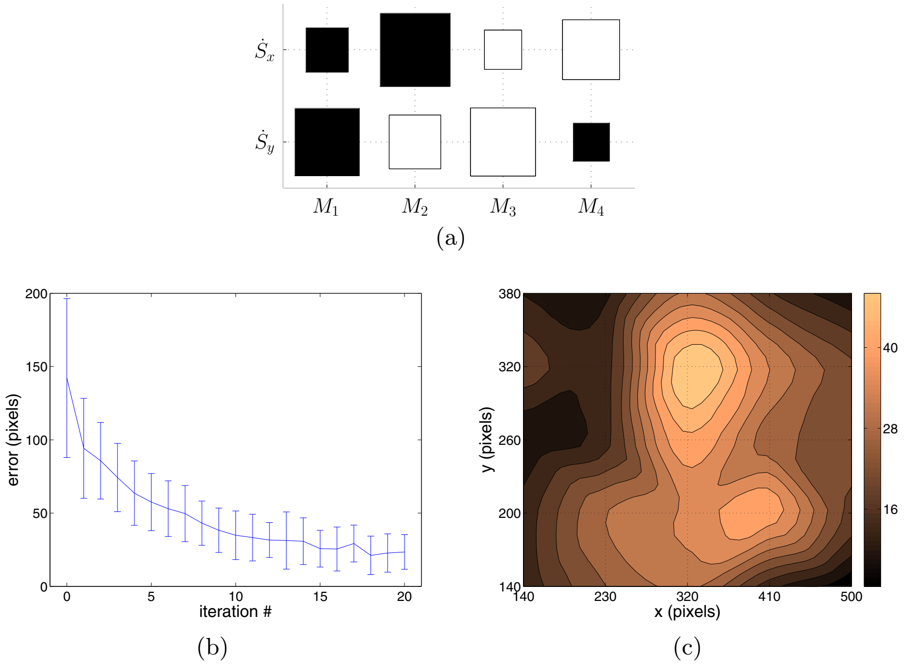
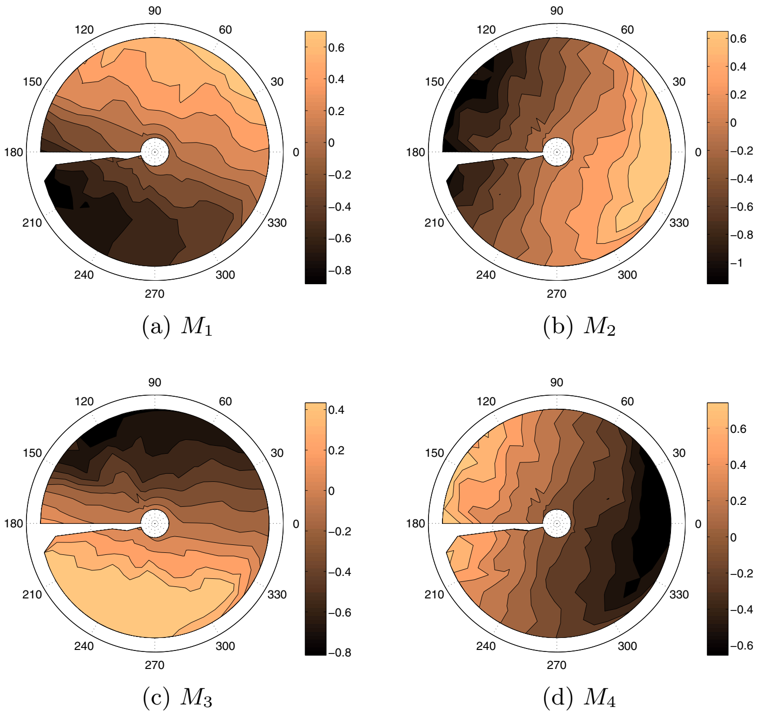


Fig. 5. The motor synergies synthesised during the ULP (a) and the reaching error obtained for the SLP (b-c). a) filled squares represent positive connections and empty squares represent negative connections. The strength of each connection is represented by the size of the respective square, b) the error (mean and standard deviation) obtained for each iteration of the SLP, c) the interpolated error as function of the target position.

for example, SMTs produced by M_1 reach x velocities with smaller amplitudes than those produced by M_2 in the same axis. This is relevant because these amplitudes ultimately affect the weights, $P_{i.j}$ which define the motor synergies.

Matrix P is shown in Fig. 5a (depicted transposed). The resulting connectivity obtained is consistent with the Hebbian learning rule applied to the data in Fig. 4, both qualitatively and quantitatively. From the qualitative point of view, we obtain positive connections between M_1 and both $\dot{S}_x$ and $\dot{S}_y$ signals; this is consistent with the positive velocities reached in both x and y during SMTs produced by M_1. Negative connections can be seen for example between M_2 and $\dot{S}_y$; this is consistent with the negative y velocity achieved during SMTs produced by M_2.

From the quantitative point of view, the connectivity strength is consistent with the signal amplitudes shown in Fig. 4; for example, the connection between M_2 and $\dot{S}_x$ is stronger than that between M_1 and $\dot{S}_x$, which is consistent with the observation that SMTs carried out by M_2 reach higher amplitudes in $\dot{S}_x$ than those carried out by M_1.



(a) M_1 (b) M_2

(c) M_3 (d) M_4

Fig. 6. The motor recruitment as a function of the target position in polar coordinates

Once identified the sensor and motor connectivity using the ULP, we test the SLP by setting a grid of 25 target points and allow the system to make 20 attempts towards each target (each attempt is an iteration in the SLP). The targets are disposed in a grid which tries to maximize the real workspace of the pendulum. At the end of each iteration all the muscles are reset to the minimum tension value to allow the pendulum to move back to its resting position.

Figure 5b shows the mean and the standard deviation of the error achieved when reaching the 25 targets as a function of the iteration step, k. The results show that average of the error decreases rather steadily, although a few error increases can be observed (e.g at iteration 17). These can be explained from 1) the uncertainties of our DC motors, and 2) the fact that the starting position of the pendulum is different at each iteration step (see Section 3.1). The distribution of final error achieved for each target over the entire workspace is shown in Fig. 5c. The smaller errors at the extremities of the workspace can be attributed to the higher precision of the motors when larger voltages are applied.

Figure 6 shows the recruitment of each artificial muscle as a function of the target angle and the target distance. As can be seen the recruitment of each muscle is consistent with its direction of applied force. For example, motor M_1, which pulls in the positive x and y directions (see Figs. 4a,d) is mostly active for targets located at 45°. In addition, all the motors are mostly active for targets which are farther away, which is in fact expected. Conversely, all the motors are mostly inhibited for targets aligned with the direction of force but located contralaterally to the muscle (e.g. the inhibition of M_1 at 210°).

5 Conclusion

In this paper we have presented a developmental framework to carry out goal-oriented learning in a reduced dimensional space. Our framework was tested in a 2D reaching task carried out by a tendon-driven pendulum robot actuated by four artificial muscles. Our results show that our framework is capable of synthesising a set of motor synergies in an unsupervised manner and combined them effectively to accomplish the proposed reaching task. At the moment we are investigating the use of interpolation methods to generalise the target positions in the task space.

References

1. Asada, M., Hosoda, K., Kuniyoshi, Y., Ishiguro, H., Inui, T., Yoshikawa, Y., Origino, M., Yoshida, C.: Cognititive developmental robotics: A survey. IEEE Transactions on Autonomous Mental Development 1(1), 12–34 (2009)
2. Brooks, R.: A robust layered control system for a mobile robot. Technical Report 864, MIT AI Lab (1985)
3. Holland, O., Knight, R.: The anthropomimetic principle. In: Burn, J., Wilson, M. (eds.) Proceedings of the AISB 2006 Symposium on Biologically Inspired Robotics (2006)
4. Latash, M.L.: Evolution of motor control: From reflexes and motor programs to the equilibrium-point hypothesis. Journal of Human Kinetics 19, 3–24 (2008)
5. Lungarella, M., Metta, G., Pfeifer, R., Sandini, G.: Developmental robotics: a survey. Connection Science 15(4), 151–190 (2003)
6. Marques, H., Jäntsch, M., Wittmeier, S., Cristiano, A., Lungarella, M., Knight, R., Holland, O.: Ecce1: the first of a series of anthropomimetic musculoskelal upper torsos. In: Humanoids 2010 (2010) (in press)
7. Marques, H.G., Imtiaz, F., Iida, F., Pfeifer, R.: Self-organisation of reflexive behaviour from spontaneous motor activity (2012) (under review)
8. Pfeifer, R., Lungarella, M., Iida, F.: Self-organization, embodiment, and biologically inspired robotics. Science 318, 1088–1093 (2007)
9. Robinson, S.R., Blumberg, M.S., Lane, M.S., Kreber, L.A.: Spontaneous motor activity in fetal and infant rats is organized into discrete multilimb bouts. Behav. Neurosci. 114(2), 328–336 (2000)
10. Smotherman, W., Robinson, S.: Prenatal ontogeny of sensory responsiveness and learning. In: Greenberg, G., Haraway, M. (eds.) Comparative Psychology: A Handbook, pp. 586–601. Garland Publishing, Inc., New York (1998)
11. Todorov, E., Ghahrammani, Z.: Unsupervised learning of sensory-motor primitives. In: Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (2003)

Adaptive Quadruped Locomotion: Learning to Detect and Avoid an Obstacle

Pedro Silva, Vitor Matos, and Cristina P. Santos

Industrial Electronics Department
University of Minho, Portugal
{psilva,vmatos,cristina}@dei.uminho.pt
http://asbg.dei.uminho.pt

Abstract. Autonomy and adaptability are key features in the design and construction of a robotic system capable of carrying out tasks in an unstructured and not predefined environment. Such adaptability is generally observed in animals, biological systems that often serve as inspiration models to the design of robots. The autonomy and adaptability of these systems partially arises from their ability to learn.

This work proposes a mechanism to enable a quadruped robot to detect and avoid an obstacle in its path. The detection is based on a Forward Internal Model trained in real-time to create estimations about the robot's perceptive information. In order to avoid tripping on an obstacle, the detections are used to create a map of responses that will change the locomotion according to previous experience.

Both learning tasks occur in real time and are combined together, defining a Sensorimotor Map that enables the robot to learn to avoid an obstacle.

Keywords: Adaptive robot controller, Autonomy in robotics, Quadruped locomotion, Learning, Forward Internal Model, Biological inspiration, Sensorimotor Map.

1 Introduction

Autonomy in a robotic system requires the adaptation to unexpected situations without the need of recalibration or specific configuration of the surrounding environment. In biological systems autonomy partially arises from their ability to learn and adapt to the environment. Likewise, the construction of robotic systems may take advantage from these characteristics in order to produce adaptive behaviors, allowing the robot to fulfill its task in a dynamical environment. In fact, the design of robotic systems is often inspired in biology, both in terms of mechanical features (quadruped, biped and hexapod robots), as in terms of control concepts and structures (Central Pattern Generators, reflexes and neural networks).

Robotic legged locomotion is also an intensive focus of investigation, namely the generation of adaptive locomotion. The autonomy and adaptability associated with the advantages of legged robotic locomotion, as the ability to move in

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 361–370, 2012.

human environments, uneven terrain and overcome obstacles; could create ideal robotic systems able to move and act in a dynamic environment.

Reflexes are a common way to seek adaptability when conceiving a legged locomotion controller. Such type of approach usually does not rely in environmental models, but in adapting to the necessary conditions of the environment.

Fukuoka and colleagues in [1] proposed a quadruped robot controller that uses a neural controller composed of a Central Pattern Generator (CPG) and reflexes. The system seeks to grant stability and energy consumption optimization. Later, this work was extended to enable locomotion in natural ground by defining additional reflexes to maintain the same level of stability. A similar approach to the control that relies in CPG and reflexes is presented in [2], in which locomotion is generated by the interaction of three different layers that operate at joint, intra-leg and inter-leg coordination. The controller uses reinforcement learning to trigger the reflexes in the quadruped robot BISAM.

Other works that address the generation of quadruped adaptive locomotion, seek the coupling between the locomotion generator and the mechanical system through sensory feedback. In [3] the cyclic sensory input defines a master-oscillator-base CPG that drives the different joints, and in [4] an adaptive controller tracks the resonant frequency of a quadruped robot with compliant knees.

In [5] the little dog is able to detect and select the best possible foot positions of a rough terrain, using a classifier based on Support Vector Machines (SVM) and Ada Boost, trained off-line. A different approach applied to the same robot is presented in [6], addressing the problem of crossing a terrain with a high level of roughness by breaking the problem down into simpler ones and solving each at a time. Although efficient and with good results, both methods rely on pre-processed information about the environment.

A Forward Internal Model (FIM) ([7]) translates the robot's motor commands into the consequent sensory changes. This enables the robot to evaluate its own movements and state, as well as to perceive external disturbances in the environment.

A FIM is also used in [8] to detect changes in the ground's slope and enable a tethered biped robot to stabilize itself by shifting its COM through an upper body component. In [9] authors propose a control model based on Unit CPGs, that enables a quadruped robot to quickly learn a stable locomotion. These are simple reflexs and adaptive models that together control the robot's movements; a FIM is used to estimate the sensory inputs according to the motor commands, thus enabling the robot to evaluate itself.

Obstacle avoidance is achieved in [10] by employing a FIM. A tethered biped robot estimates disparity information using its own state during the locomotion cycle and is able to discern the presence of the obstacle. A related work, [11], learns the alterations to the locomotion that are necessary to enable a similar robot to step over an obstacle.

Similarly to these works, we propose an adaptive controller for a legged robot, that enables a robot to move and act in the environment with some level of autonomy. However, we seek a specific goal: the robot has to learn to detect and

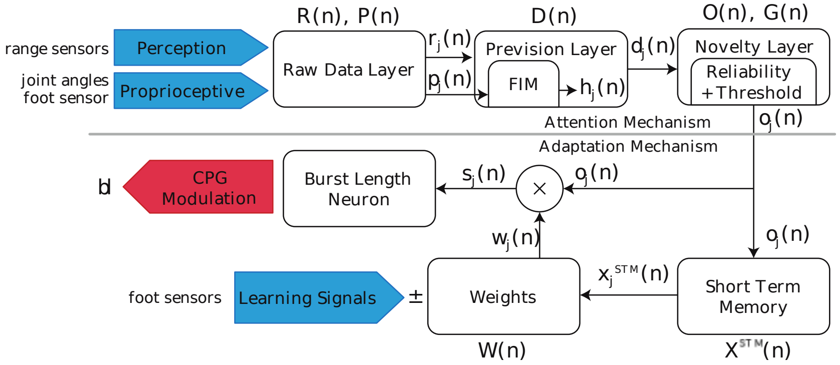


Fig. 1. Mechanism architecture. The attention mechanism yields obstacle detections and the adaptation mechanism learns to change the locomotion during the approach to the obstacle.

to avoid an obstacle in its path. Furthermore, unlike most of the previous works where the adaptation relies in off-line methods or predetermined reactions to specific situations, one of our main concern was to achieve run-time learning of the perception and actuation tasks.

The proposed system is partly inspired in [10,11], and extends the authors current work [12] in generating quadruped locomotion with a CPG controller. It enables a quadruped robot to autonomously learn in real-time to detect obstacles in its path through an array of range sensors and proprioceptive information, and avoid stumbling on the obstacle by adapting its stride length when approaching it. The detection is based on estimations about the range sensors, which are computed by a FIM. In order to adapt the locomotion, the mechanism learns the necessary alterations to the stride length. The learning relies on the obstacle detections and the failed attempts to step over the obstacle. The output continuously modulates the CPG's amplitude, which results in changes in the stride length during the approach to the obstacle.

2 Architecture Overview

The proposed architecture (fig. 1) is divided into two main mechanisms: i) the attention and ii) the adaptation mechanism.

The attention mechanism is responsible for detecting the obstacles in the robot's path, and is composed by three layers as follows:

Raw. The Raw Layer receives the continuous inputs from the n_s range sensors and the n_f proprioceptive information features (joint angles and foot sensors), and divides both inputs, first according to each stride (n) and then according to the n_p stride phases. This process yields two matrices: $R_{n_s \times n_p}$, which correlates the distances scanned by each sensor i and moments of the stride j (r_{ij}); and $P_{n_f \times n_p}$, which columns are defined by the vectors p_j (length n_f), referrent to the stride phases j. We want to stress the concept of a cell as an explicit relation between a distance sensed by one of the robot's sensors (i), and a moment of the stride (j). Here we define $r_{ij} \in R$ as a cell, and the same concept is present throughout the whole Mechanism, always as a part of a matrix of dimensions $n_s \times n_p$. At each stride phase j this layer outputs the range sensors values r_j, and the proprioceptive information p_j.

Prevision. In the Prevision Layer an estimation $h_{ij} \in Hn_s \times n_p$ is created for each $r_{ij} \in Rn_s \times n_p$, generating a difference value, $d_{ij} \in Dn_s \times n_p$. These matrices follow the structure defined before: each of its cells relates a distance sensor (i) to a moment of the stride (j). If the difference (d_{ij}) between an observed distance (r_{ij}) and the estimated value (h_{ij}) is high, then, something changed in the environment (or eventually in the robot) that caused such difference. There is a Least Mean Square (LMS) rule to compute each h_{ij}, which works as a Forward Internal Model (FIM), exploiting the movements generated by the locomotion at every stride to estimate the changes in the robot's perception. The Prevision Layer outputs the difference for all sensors, d_j, during each stride phase j.

Novelty. The Novelty Layer evaluates each difference value $d_{ij} \in D_{n_f \times n_p}$ according to the reliability of the estimations (h_{ij}), and generates the obstacle detection signals $o_{ij} \in O_{n_f \times n_p}$ accordingly. A threshold and a reliability variable $g_{ij} \in G_{n_f \times n_p}$ are used for that purpose. g_{ij} enhances the difference value if the correspondent cell usually produces good estimations, or reduces it otherwise. Thus, an obstacle is detected at by a sensor i durring the stride phase j if $o_{ij} \neq 0$. The reliability variables are updated according to the difference value of its correspondent cell by a feedback mechanism, which evaluates the performance and maintains g_{ij} within $]0, 1.2]$. The resulting obstacle detections o_j are produced every stride phase j.

The adaptation mechanism acquires experience during the attempts to step over the obstacle, addressing such situations in the future. This experience is built based on the obstacle detection output by the attention mechanism, and the triggering of learning signals, as follows:

Short Term Memory. The Short Term Memory cells ($X^{STM}_{n_f \times n_p}$) are activated taking into account the obstacle detections $O_{n_f \times n_p}$, that is, if o_{ij} is activated, so will x^{STM}_{ij}. The activated STM cells hold their values through time as a register of an obstacle detected at a certain distance i and moment of the stride j. This enables the relation of two different moments in time: the detection and the encounter with the obstacle. The value of x^{STM}_{ij} decreases exponentially with time, thus, the sooner an encounter occurs, the stronger the signal in x^{STM}_{ij} will be.

Learning Signals. When the robot stumbles on the obstacle, a learning signal δ is triggered, which indicates how the locomotion should be changed in the future. Therefore, enabling such situation to be avoided. The type of response depends on the moment of the stride in which the signal is triggered: if it is during the paw extension phase (fig. 2:1), $\delta = -1$, the stride length must be reduced. If otherwise it happens during the paw placement phase (fig. 2:2), $\delta = 1$, the stride length should be increased.

Weights. The weights $w_{ij} \in W_{n_f \times n_p}$ hold the alterations to the stride length and are computed when a learning signal δ is triggered. When the robot detects the obstacle with the sensor i during the moment of the stride j, the alteration to the locomotion, w_{ij}, is defined according to the magnitude of the remains in x_{ij}^{STM} and the signal of δ. These weights belong to $[-1, 1]$, and their values are normalized so that their sum over all the n_s range sensors is $||w_j|| = 1$.

Burst Length Neuron. The CPG's amplitude is continuously modulated by bl. This value is updated every stride phase j according to the experience acquired so far in W for that stride phase w_j, and the obstacle detection signals o_j. When an obstacle is detected by a distance sensor i and moment of stride j, a synapse s_{ij} occurs between o_{ij} and w_{ij}, which indicates the necessary alteration to the locomotion. At each stride phase j the synapses of all sensors, s_j, are used to update bl. After enough iterations, these changes grant the robot the ability to avoid stumbling on the obstacle.

The equations that implement each of the described components are presented in the appendix.

3 Results

The proposed architecture was evaluated in simulation using a quadruped robot BioloidQuad with 3 DOFs in each leg. The BioloidQuad is equipped with a set of 10 range sensors arranged in order to detect the obstacle ahead, as well as touch sensors in the paws, in the front and back sides, and in the bottom (fig. 2, respectively). The robot's knee and hip-swing joints are driven by the CPG presented in [12]. The hip-swing joints and the bottom touch sensors on the paws define the proprioceptive information, which is used to indicate the state of the robot's locomotion cycle.

3.1 Simulation Setup

Results were obtained through a series of simulations that consist of several trials. At each trial the robot starts walking at a determined distance from the obstacle, located ahead to the right, obstructing the right foreleg. As the robot walks, the obstacle is detected by the infra-red range sensors, up until the point that the robot reaches the obstacle 0.85 cm tall and 0.3 cm wide. The goal is to show that both the detection and the adjustments to the locomotion happen as expected, as well as the ability for the whole system to achieve the desired goal, that is to step over the obstacle without stumbling.

3.2 Adaptive Locomotion Evaluation

A typical simulation is depicted in fig. 2, with the robot executing several trials walking towards an obstacle. In fig. 2:1 the robot faces the obstacle for the first time and stumbles with the front of the paw. In the second trial (fig. 2:2) despite

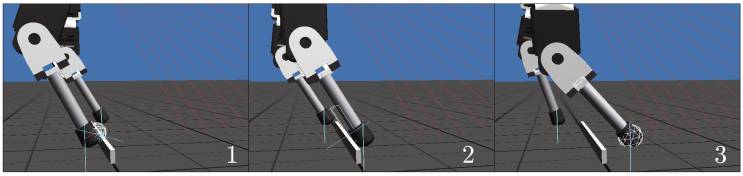


Fig. 2. In the first trial the robot tries to step over too close to the obstacle, so it stumbles on it. On the second trial with the prior experience, it starts too far away and collides with the obstacle after stepping over the obstacle. After the third trial, with enough experience, the robot succeeds in stepping over the obstacle without any collision.

the adjustment influenced by the previous trial, the robot still touches the obstacle but now with the back of the paw. At the third trial the robot steps over the obstacle without touching it. From the previous two trials, it has learnt how to adjust the stride length in order to correctly approach and step over the obstacle.

Fig. 3 shows the values of a single cell throughout the different layers of the proposed system. In this simulation the robot learns to avoid the obstacle after three trials. Shaded areas separate the trials.

The raw data and the estimated values for each stride of that specific cell are depicted in fig. 3:a (dashed blue and solid red, respectively). The visible difference defines a candidate obstacle detection in the novelty layer (fig. 3:b, solid blue) and will be evaluated according to the cell's reliability value and filter threshold. As cell previsions improve, these differences are accepted as obstacle detections and output from the novelty layer (fig. 3:b, red dots). At each trial the robot detects the obstacle at sensibly the same distance, observable by the activations in fig. 3:b periodically in the shaded areas . These obstacles detections are temporarily stored in the STM cells (fig. 3:c, blue dashed).

At the end of each trial the robot reaches the obstacle and a previously defined mechanism for overcoming it is elicited. If the approach to the obstacle is appropriate, the robot can overcome the obstacle without any collision. In this simulation the robot collides with the obstacle three times, triggering three paw extension learning signals (fig. 3:c, blue columns). Each triggered learning signal combined with the recent obstacle detection adjusts the corresponding cell's weight (fig. 3:d).

The modulation signal output by the mechanism in fig. 3:e is adjusted continuously throughout the trial, resulting in different stride lengths (fig. 3:f). Positive spikes at the end of each trial correspond to elicited step over. It is observable that after a weight adjustment, the output modulation signal of the subsequent trial changes. After the first weight adjustment, the mechanism produces an increased negative modulation signal, which is fed to the CPG and reduces the stride length during the approach to the obstacle. However that change is not enough, and the robot still stumbled on the obstacle, triggering another learning signal and altering the weights even further. The output signal becomes stronger and the changes in the stride length become more pronounced.

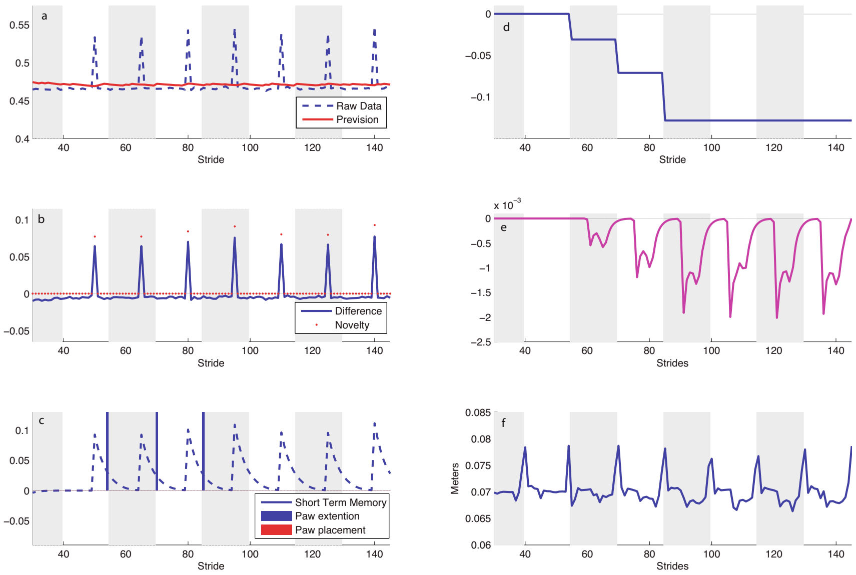


Fig. 3. Value of a single cell throughout the different components. Shaded areas separate the several trials. Raw data and prevision layer (a). Difference values and the novelty signal (b). Short Term Memory and learning signals (c). Weight (d). Output of the burst length neuron (e). Measured stride length (f).

3.3 Reacting to Activation Patterns by Experience

During the approach to the obstacle a specific set of cells is activated, as each sensor detects the obstacle at a particular moment of the stride. If we consider the aforementioned matrix structure, the activated cells form a specific pattern, which we call an Activation Pattern (AP). The AP depends on the trial situation, which is defined by the starting distance to the obstacle of each trial.

Fig. 4:left shows the triggered learning signals when the robot is faced with two distinct APs, starting the approach to the obstacle at two distinct distances. The mechanism learns to respond to the first AP (strides 0-200) and the second as well (strides 200-325) - we can see the different responses in the output. After this, the robot is able to avoid the obstacle in both situations regardless of the AP the mechanism is being faced with (strides 320-600).

If in two trial situations, the obstacle distance differ in a multiple of the nominal stride length, the resulting APs will be similar because the robot's locomotion is periodic. Therefore the approach conditions are the same, only one stride length closer or further away. We demonstrate this feature in fig. 4, right, where learning signals are only triggered in the first five trials, and the learned adaptations are valid for both trial situations, switched alternatively at stride 200, 320 and 450. As shown, the output signal is similar for the two situations.

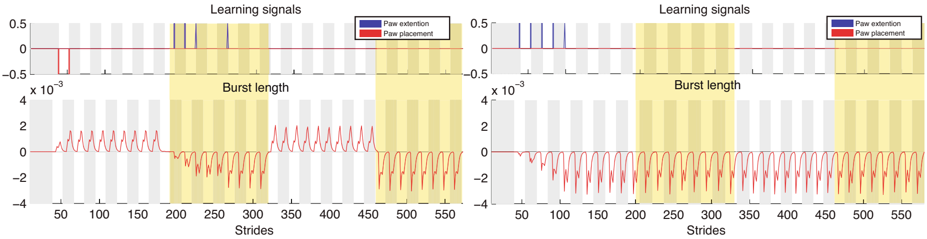


Fig. 4. Two different obtained APs (left) and two equivalent APs (right). The upper plots display the triggered learning signals throughout each trial and bellow the output signals to the CPG.

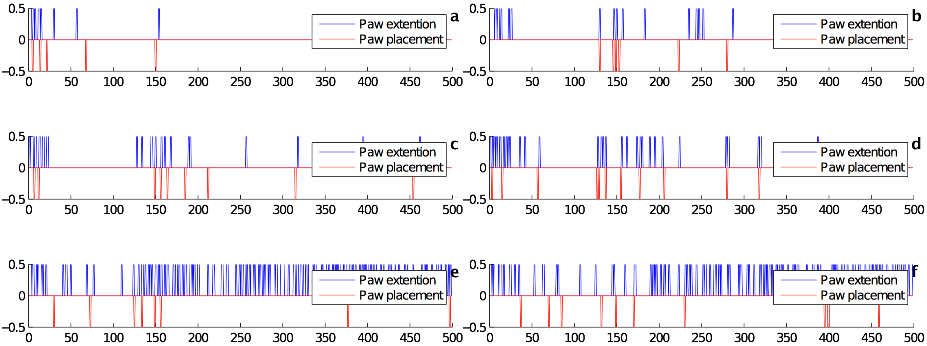


Fig. 5. Learning signals concerning several possible APs. With 6, 7, 8 and 9, respectively (a), (b), (c) and (d), and with 14 and 16 APs, respectively (e) and (f).

Further tests were made in order to evaluate the mechanism's ability to respond to several APs. As an AP is repeated according to the locomotion cycle, the starting distances are randomly varied from within a stride's length, plus a base distance of 1 meter.

Fig. 5 presents the results using different possible starting positions. The decreasing frequency of the activated learning signals demonstrates the learning convergence. As visible in fig. 5, less APs means a quicker convergence and a more robust learning process. When using 6 and 7 APs (respectively, 5:a and b), the convergence is faster and the performance is maintained afterwards, since the learning signals stopped. In the worse case, when using 14 and 16 APs (respectively, fig. 5:e and f), despite an initial reduction in the learning signals, the system appears not to be able to deal with this many possible conditions. When faced with 8 or 9 (respectively, fig. 5:c and d) the mechanism is able to achieve a good performance. However, it seems that the mechanism requires sparse learning signals to keep avoiding the obstacle, nevertheless, it maintains an acceptable level of performance.

4 Conclusion and Discussion

The proposed architecture is able to successfully achieve real-time learning of obstacle detection and obstacle approach to prevent stumbling. The robot is able to discern the obstacles ahead resorting to the Forward Internal Model (FIM) in the attention mechanism. Further, using these obstacle detections, the mechanism is able to gain experience from failed attempts to step over the obstacle, and then use this experience to avoid those same situations. Both tasks were carried out in real-time and kept active, enabling continuous adaptation. The robot learns to detect and avoid obstacles in its path without explicit specification of the obstacles, simply reacting to undesirable stimulus.

However, further tests indicated that there is a limit to the number of APs the mechanism can deal with. Further and deeper knowledge about the relation that should exist between the scanned distances and the stride division, as well as the one that exists between the mechanism's output and the actual alterations to the stride length would give us better understanding to overcome this limitation. Further work will also include the application of the proposed mechanism to a real quadruped robot.

Acknowledgments. This work is funded by FEDER Funding supported by the Operational Program Competitive Factors - COMPETE and National Funding supported by the FCT - Portuguese Science Foundation through project PTDC/EEACRO/100655/2008 and FCOMP/01/0124/FEDER/022674. Pedro Silva is supported by grant UMINHO/BI/155/2011 and Vitor Matos by SFRH/BD/62047/2009.

References

1. Fukuoka, Y., Kimura, H., Cohen, A.H.: Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts. The International Journal of Robotics Research 22(3-4), 187 (2003)
2. Ilg, W., Albiez, J., Jedele, H., Berns, K., Dillmann, R.: Adaptive periodic movement control for the four legged walking machine bisam. In: Proceedings of the 1999 IEEE International Conference on Robotics and Automation, vol. 3, pp. 2354–2359. IEEE (1999)
3. Heliot, R., Espiau, B.: Multisensor input for cpg-based sensory—motor coordination. IEEE Transactions on Robotics 24(1), 191–195 (2008)
4. Buchli, J., Ijspeert, A.J.: Self-organized adaptive legged locomotion in a compliant quadruped robot. Auton. Robots 25, 331–347 (2008)
5. Doshi, F., Brunskill, E., Shkolnik, A., Kollar, T., Rohanimanesh, K., Tedrake, R., Roy, N.: Collision detection in legged locomotion using supervised learning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007, pp. 317–322 (October 2007)
6. Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M., Schaal, S.: Fast, robust quadruped locomotion over challenging terrain. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 2665–2670. IEEE (2010)
7. Wolpert, D.M., Kawato, M.: Multiple paired forward and inverse models for motor control. Neural Networks 11(7-8), 1317–1329 (1998)

8. Schröder-Schetelig, J., Manoonpong, P., Wörgötter, F.: Using efference copy and a forward internal model for adaptive biped walking. Autonomous Robots, 1–10 (2010)
9. Lewis, M.A., Bekey, G.A.: Gait adaptation in a quadruped robot. Autonomous Robots 12(3), 301–312 (2002)
10. Lewis, M.A., Simó, L.S.: Certain principles of biomorphic robots. Auton. Robots 11(3), 221–226 (2001)
11. Lewis, M.A., Simó, L.S.: Elegant stepping: A model of visually triggered gait adaptation. Connection Science 11(3), 331–344 (1999)
12. Matos, V., Santos, C.P.: Omnidirectional locomotion in a quadruped robot: A cpg-based approach. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3392–3397. IEEE (2010)

Appendix: Mechanism Summary

Raw Layer	Prevision Layer
Proprioceptive information divided into n_p phases ($P(n)$). Perception raw data divided into n_p and composed of n_s elements ($R(n)$). n is stride number.	$D(n) = R(n) - L\,(P(n))$. $L(P(n))$ computes $H(n)$ using a Least Mean Square (LMS) rule.
Novelty Layer	Short Term Memory
$O(n) = \left.(G(n)D(n))\right\rvert_{th}$ $G(n) = G(n-1) + F(D(n))G(n-1)\alpha^{gf}$ Feedback mechanism F updates G through a gaussian function. α^{gf} is rate of change. th is threshold.	$\tau^{STM} \Delta X^{STM} = -X^{STM}(n) + O(n)$ $\tau^{STM} = 1 + \frac{2}{1+e^{l}}$ $l = X^{STM}(n) - O(n)$
Weights Update	Burst Length Neuron
$\Delta W = \delta X^{STM}(n)\left(\lvert W(n)\rvert + c\right)\alpha^{STM}$ $\delta = \begin{cases} 1 \text{ if paw placement,} \\ -1 \text{ if paw extension.} \end{cases}$ c is a small constant that starts up the learning task and α^{STM} is the rate of learning.	$\tau^{bl}\frac{dbl}{dt}bl_j = -bl + \sum_{i=1}^{n_s} s_{ij}$ $s_{ij} = w_{ij}(n)\lvert x_{ij}^{STM}(n)\rvert$ Synapse s_j is computed at evert stride phase j. bl is the output to the CPG.

Attentional Action Selection Using Reinforcement Learning

Dario Di Nocera[1], Alberto Finzi[1], Silvia Rossi[1], and Mariacarla Staffa[2]

[1] Dipartimento di Scienze Fisiche
[2] Dipartimento di Informatica e Sistemistica,
University of Naples "Federico II" – Naples, Italy
d.dinocera@studenti.unina.it, {finzi,srossi,mariacarla.staffa}@unina.it

Abstract. Reinforcement learning is typically used to model and optimize action selection strategies, in this work we deploy it to optimize attentional allocation strategies while action selection is obtained as a side effect. We present a reinforcement learning approach to attentional allocation and action selection in a behavior-based robotic systems. We detail our attentional allocation mechanisms describing the reinforcement learning problem and analysing its performance in a survival domain.

Keywords: attention allocation, reinforcement learning, action selection.

1 Introduction

Beyond their role in perception orientation and filtering, attentional mechanisms are considered as key mechanisms in sensorimotor coordination and action control. Indeed, in biological systems, executive attention and attention allocation strategies are strictly connected with action selection and execution [5,7,10]. In this work we explore this connection in a robotic setting deploying a reinforcement learning framework. More specifically, we propose a reinforcement learning approach to attention allocation and action selection in behavior-based robotic system. Reinforcement learning (RL) is typically used to model and optimize action selection strategies both in artificial [12] and biological systems [6,4,8]. In contrast, in this work we deploy RL to learn attention allocation strategies, while action selection is obtained as a side effect of the resulting attentional behavior. RL models for attention allocation have been mainly proposed for visual attentions and gaze control [1,9], here we apply an analogous approach to executive attention considering the problem of a supervisory attentional system [7] suitable for monitoring and coordinating multiple parallel behaviors. Our attentional system is obtained as a reactive, behavior-based system, endowed with simple, bottom-up, attentional mechanisms capable of monitoring multiple concurrent tasks. We assume a frequency-based model of attention allocation [11]. Specifically, we introduce simple attentional mechanisms regulating sensors sampling rates and action activations [2,3]: the higher the attention the higher the

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 371–380, 2012.

resolution at which a process is monitored and controlled. In this framework, reinforcement learning is used to select the best regulations for these mechanisms. We detail the approach describing the reinforcement learning problem and analyzing its performance in a simulated survival domain. The collected results show that the approach is feasible and effective in different settings. That is, reinforcement learning applied to attentional allocation allows not only to reduce and focus sensor processing, but also to significantly improve sensorimotor coordination and action selection.

2 Background and Model

2.1 Attentional System

Our attentional system is obtained as a reactive behavior-based system where each behavior is endowed with an attentional mechanism represented by an internal adaptive clock [2].

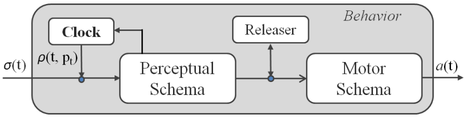


Fig. 1. Schema theory representation of an attentional behavior

In Figure 1 we show a schema theory representation of an attentional behavior. This is characterized by a *Perceptual Schema* (PS), which elaborates sensor data, a *Motor Schema* (MS), producing the pattern of motor actions, a *Releaser* working as a trigger for the MS activation and an attention control mechanism, called Adaptive Innate Releasing Mechanism (AIRM), based on a *Clock* regulating sensors' sampling rate and behaviors' activations (if enabled). The clock regulation mechanism represents our frequency-based attentional mechanism: it regulates the resolution/frequency at which a behavior is monitored and controlled, moreover, it provides a simple prioritization criterion. This attentional mechanism is characterized by:

- An activation period p^b ranging in an interval $[p^b_{min}, p^b_{max}]$, where b is the behavior's identifier. It represents the sensors sampling rate (i.e. how long the sensors are not processed before the next reading).
- An *monitoring function* $f(\sigma^b(t), p^b_{t-1}) : \mathbb{R}^n \to \mathbb{R}$ that adjusts the current clock period p^b_t, according to the internal state of the behavior and to the environmental changes.
- A trigger function $\rho(t, p^b_t)$, assuming a 0/1 value, which enables/disables the data flow $\sigma^b(t)$ from sensors to PS at each p^b_t time unit.
- Finally, a normalization function $\phi(f(\sigma^b(t), p^b_{t-1})) : \mathbb{R} \to \mathbb{N}$ that maps the values returned by f into the allowed range $[p^b_{min}, p^b_{max}]$.

The clock period at time t is regulated as follows:

$$p_t^b = \rho(t, p_{t-1}^b) \times \phi(f(\sigma^b(t), p_{t-1}^b) + (1 - \rho(t, p_{t-1}^b)) \times p_{t-1}^b \tag{1}$$

That is, if the behavior is disabled, the clock period remains unchanged, i.e. p_{t-1}^b. Otherwise, when the trigger function is 1, the behavior is activated and the clock period changes according to the $\phi(f)$.

2.2 Reinforcement Learning for Attentional Action Selection

Given the attention mechanisms introduced above, our aim is to exploit a Reinforcement Learning (RL) [12] technique to regulate the monitoring functions.

Q-learning for attentional regulation. Our approach exploits Q-learning [14] (QL) to produce a policy for attentional allocation. In this context the learning problem can be cast as follows. For each behavior, we introduce a suitable space state S while the action space A represents a set of possible regulations for its clock. In this paper, we assume that this set spans a discretized set of possible allowed periods $P = \{p_1, \ldots, p_n\}$, i.e. A coincides with P. Since the current state $s \in S$ should track both the attentional state (clock period) and the perceptive state (i.e. internal and external perceived status), this will be represented by a pair $s = (p, x)$, where $p \in P$ is the current clock period and $x \in X$ is for the current perceived status. Then, an attentional allocation policy $\pi : S \to P$ defines a mapping between the current state s and the next attentional period p. Given a reward function R for each behavior, the QL task is to find the optimal attention allocation policy π: for each state $s \in S$ we have to find the activation period $p \in P$ that maximizes the behavior's expected reward. Notice that each behavior concurrently runs its own QL algorithm as an independent agent (independent versus cooperative RL is discussed in [13]). We can rely on this model because here the attentional mechanisms are not mutually dependent (only stigmergic interactions). Therefore, for each behavior, for each clock activation p_t leading from the state s_t to the state s_{t+1}, the agent receives a reward r_{t+1}, and the Q-value is updated as follows:

$$Q(s_t, p_t) \leftarrow (1 - \alpha_t) \cdot Q(s_t, p_t) + \alpha_t(\mathcal{R}_{t+1} + \gamma \cdot max_{p_{t+1} \in A} Q(s_{t+1}, p_{t+1})),$$

where γ is the discount factor (which determines the importance of future rewards) and α is the learning rate (a factor of 0 will make the agent not learn anything, while a factor of 1 would make the agent consider only the most recent information). QL requires clever exploration mechanisms, we will refer to *softmax* that uses a Boltzmann distribution [12]. We also tested our system using the ϵ-greedy exploration policy, but in our case we obtained better results adopting the *softmax* technique because it allows to balance exploration and exploitation by means of the temperature value: the higher the temperature, the closer to a random policy (exploration), the lower the closer to $Q(s, p)$ maximization (exploitation). Namely, we experimentally set a temperature value that allows us to prefer actions with high reward, in so producing a convergence which is faster than the one obtained with the ϵ-greedy policy.

3 Case Study

In order to test our approach we consider a *Survival Problem*: the robot must survive for a predefined amount of time within an environment (Fig. 2) avoiding obstacles (objects, walls, etc.), escaping from possible sources of danger (red objects) and recharging its batteries when necessary. We consider simulated environments of area $16m^2$. Obstacles, dangerous, and recharge locations are cubes of size $0.5m \times 0.5m \times 0.5m$ respectively of black, red, and green color (Fig.2). An experiment ends in a positive way if the robot is able to survive till the end of the test, while it fails in three cases: the robot collides with an obstacle, the recharge value goes under the minimum value; the robot goes very close to a source of danger. We tested our approach using a simulated *Pioneer3-DX* mobile robot (using the Player/Stage tool), endowed with a blob camera and 16 sonar sensors.

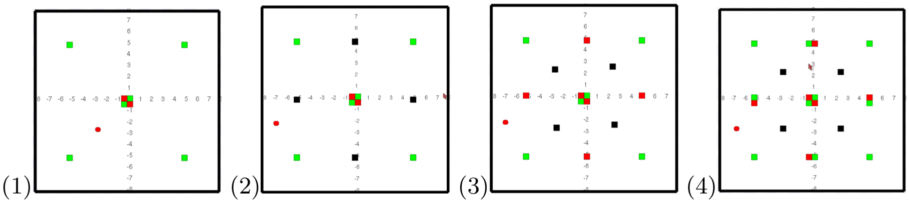


Fig. 2. Testing Environments

3.1 Attentional Architecture

In Fig. 3 we illustrate the attentional control system designed for the survival domain. It combines three behaviors: *Avoid*, *Recharge*, and *Escape*, each endowed with its releaser and adaptive clock. In the following we detail these behaviors.

Avoid manages obstacle avoidance, its input signal $\sigma^a(t)$ is the distance vector generated by the 8 frontal sonar sensors; its motor schema controls the robot velocity and angular velocity $(v(t), \omega(t))$ (for this reason its motor schema always produce a pattern of motor action) eventually generating a movement away from an obstacle, when detected. The obstacle avoidance is obtained as follows: $v(t)$ is proportional to the obstacle proximity, i.e. $v(t) = v_{max} \times \frac{min(\sigma^a(t))}{max\_sonar}$, where v_{max}, $min(\sigma^a(t))$ and max_sonar, are respectively the maximum velocity, the minimum distance from the obstacle and the maximum sonar range; $\omega(t)$ is obtained as weighted sum of the angular velocities generated by the active sonars, i.e. $\omega(t) = \sum_{i \in A(t)} rot_{max} \times w_i$, where $A(t)$ is the set of active sonars detecting an obstacle at time t, rot_{max} is the maximal rotation, w_i is a suitable weight depending on the sonar position (frontal higher, lateral lower).

Recharge monitors an internal function $\sigma^r(t)$ representing the energy status. At each execution cycle the energy decreases of a unit of charge and $\frac{1}{3}$

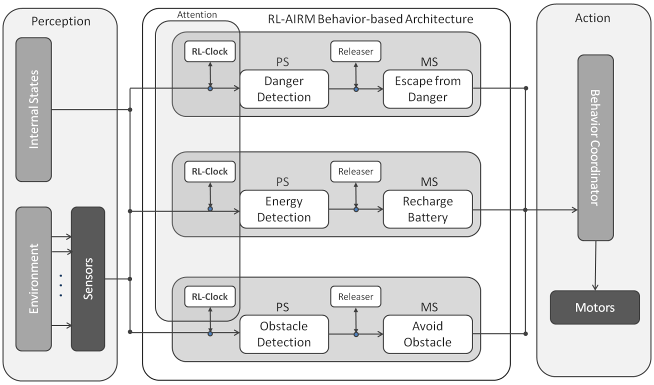


Fig. 3. Attentional Architecture Overview

(where 3 is the number of behaviors) of the unit of charge for each active behavior. Therefore, Recharge is active when $\sigma^r(t)$ goes below a suitable threshold. When enabled, if a green blob (representing the energy source) is detected by the camera, the motor schema generates a movement towards it, otherwise it starts looking around for the green, generating a random direction.

Escape monitors a function $\sigma^e(t)$ that represents fear and considers the height (pixels in FOV) of a detected red object in the environment as an indirect measure of the distance from the object. The motor schema is enabled whenever the $\sigma^e(t)$ is greater then a suitable threshold and generates a movement away from the red object. In this case, the red object is avoided with an angular velocity proportional to the fear, i.e. $\omega(t) = \alpha \times \sigma^e(t)$.

For each behavior, the clock regulation depends on an monitoring function that should be learned at run-time.

3.2 Reinforcement Learning and Attention Allocation

In the following we formulate the RL problem in the case study. We start formalizing the action space and the state space.

Action Space. In the attentional allocation problem, for each behavior, the action space is represented by a set of possible periods $\{p_1, \ldots, p_n\}$ for the adaptive clock. In the case study, assuming the minimum clock period as 1 machine cycle, the possible periods' set for *Avoid*, *Recharge* and *Escape* is: $P^a, P^r, P^e = \{1, 4, 8, 12\}$.

State Space. We recall that for a generic behavior, the state s is determined by a pair (p, x), where p represents the current clock period and x is the current perceptive state. For each behavior, we obtain the perceptive state by partitioning the total perceptive domain in equidimensional intervals (i.e. each perceptive state is a subrange of the input signal). The domain for *Avoid* spans the interval

$[0, max\_sonar]$; the domain of *Recharge* is $[0, max\_charge]$, where max_charge represents the maximum battery charge; the *Escape* domain is in $[0, max\_fear]$, where max_fear is the maximum height (in pixel) of a red object in the FOV. We tested our system discretizing the perceptive state at different granularities.

Reward function. We assume the reward always negative, with a strong penalty (r_{max}) if the system cannot survive. For the other cases the penalty is as follows. Concerning *Avoid*, each activation is penalized proportional to the distance from the obstacle ($\mathcal{R}^a_t = r_{max}$ if $x_t < th\_crash$ and $\frac{-x_t}{max\_sonar}$ otherwise). As for *Recharge*, for each activation the penalty is inversely proportional to the current charge ($\mathcal{R}^r_t = r_{max}$ if $x_t < th\_charge$ and $\frac{(x_t - max\_charge)}{max\_charge}$ otherwise). Finally each activation of *Escape* is penalized proportionally to the current amount of fear ($\mathcal{R}^e_t = r_{max}$ for $x_t < th\_fear$ and $\frac{-x_t}{max\_fear}$ otherwise). For our experiments we adopt the following settings:

- r_max: maximum penalty (-3000 units of penalties), where 3000 is the medium number of cycles for episode multiplied by the number of behaviors;
- max_time: maximum time allowed to accomplish the task (180 seconds);
- max_sonar: maximum sonar range (1 meter);
- th_crash: minimum distance under which the robot stops (0.4 meters);
- max_charge: maximum value for the charge (150 units of charge);
- th_charge: minimum value of the charge under which the robot needs to recharge (140 units of charge);
- max_fear: maximum height of a red blob (dangerous object) perceived by the camera (30 pixels);
- th_fear: minimum height of a red blob beyond which the robot does not work (23 pixels);

Setting the state space. First of all, we carried out some tests evaluating the convergence of the Q-learning process, while changing granularity and dimension of the state space. Each test consists of 5 experiments, each subdivided into 1000 episodes. We evaluated the system performance in 4 different representations of the state space. Namely, for each behavior, we considered 20, 24, 28, and 32 states, obtained by changing the size of the intervals used to partition the perceptive domain, while we use a fixed discretization of clock periods for each test. We set the *discount factor* γ at 0.9 and we tested the system using different values for the *learning rate*. In the following we report the performance obtained with $\alpha = 0.8$ (selected setting). In Fig.4-(a) we illustrate the variation of the fitness value with respect to the state representation. The fitness function evaluates the success percentage, i.e. the number of positive endings. We observe that for all the state representations we get a good percentage of success (up 98% of positive endings) after 200 episodes. However, the one with 24 states converges faster reaching 100% positive endings after 300 episodes. In Fig.4-(b), we show the accumulated rewards for each representation. Also in this case, we obtain the best regulation with the 24 states setting, therefore, we decided to employ this representation for our experiments.

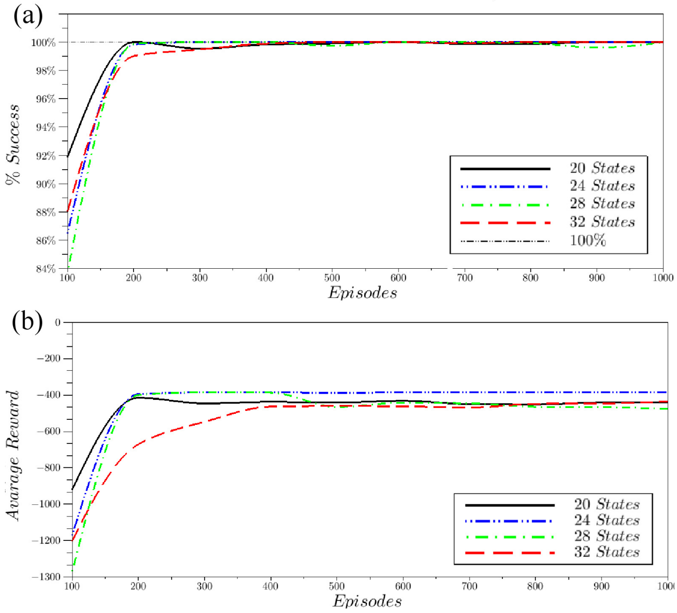


Fig. 4. (a) Fitness convergence and (b) Average Reward varying the states space representation

Setting the learning rate. Learning rate α is a crucial parameter that strongly affects Q-learning velocity and convergence. We tested 4 different settings, namely, 0.2, 0.4, 0.6, 0.8. In Fig. 5-(a) and (b) we compare, respectively, the convergence curves and the reward values in the case of 24 states. Analogous tests have been carried out with other state spaces, the best regulation was obtained with $\alpha = 0.8$.

4 Experiments and Results

We tested the attentional system in 4 environments (see Fig. 2) with different complexity in the number and disposition of the objects (red, green, and black cubes). Each experiment starts with initial values set to 0 in the Q-tables. In Fig. 6, we show the success rate for each environment. Here, the learning curve always converges to 100%. i.e. during the episodes the system is effective in learning the attention allocation strategies used to select the suitable actions for survival. Furthermore, we analized the reliability, efficiency, and effectiveness of the learned attentional strategies (RL-AIRM) comparing them with respect to the results obtained with manually tuned attentional strategies (AIRM). We tested these two architectures in the 4 environments collecting means and standard deviation on 100 tests. In Fig. 7 we can see that in almost all the environments RL-AIRM shows a higher success rate and lesser cost (less cost

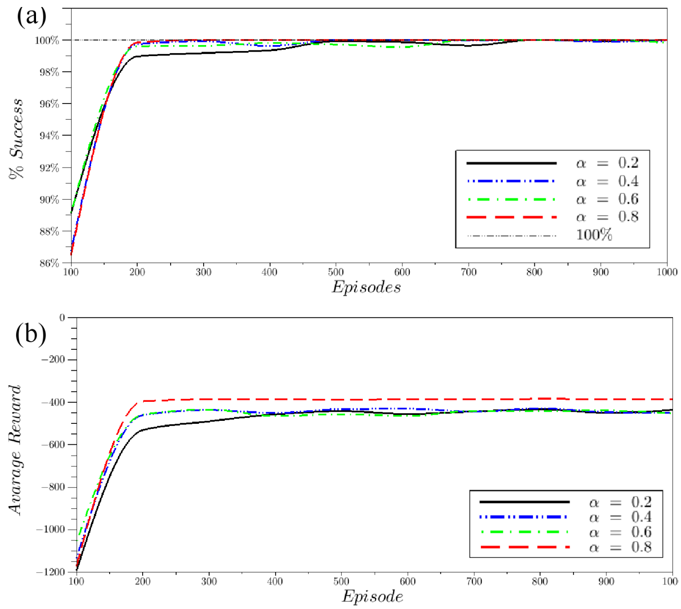


Fig. 5. (a) Fitness convergence and (b) Average Rewards relative to different values of the learning rate

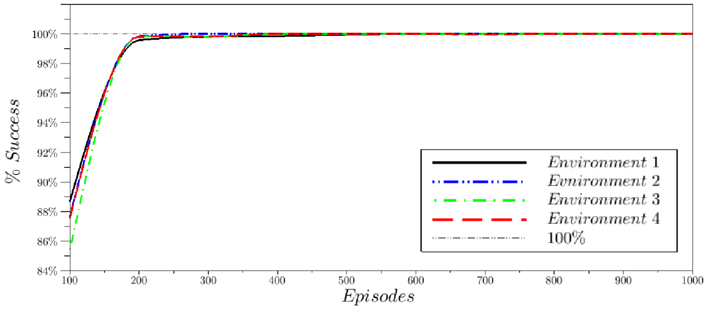


Fig. 6. Success rate in the survival domain

means better performance). The AIRM's large STD deviations are associated with failures which are not produced in the RL-AIRM settings. Note that these results are obtained despite the AIRM works with a more refined state space (we used updating functions spanning continuous values). Concerning efficiency, in Table 1 we can observe that both RL-AIRM and AIRM are able to reduce and focus the behaviors' activations (i.e. the total number of cycles these behaviors are activated). AIRM seems more efficient, but it is also less reliable and effective (as shown in Fig. 7), hence RL-AIRM seems to provide a better balance of efficiency (minimum activations), reliability (maximum success rate), and effectiveness (minimum cost).

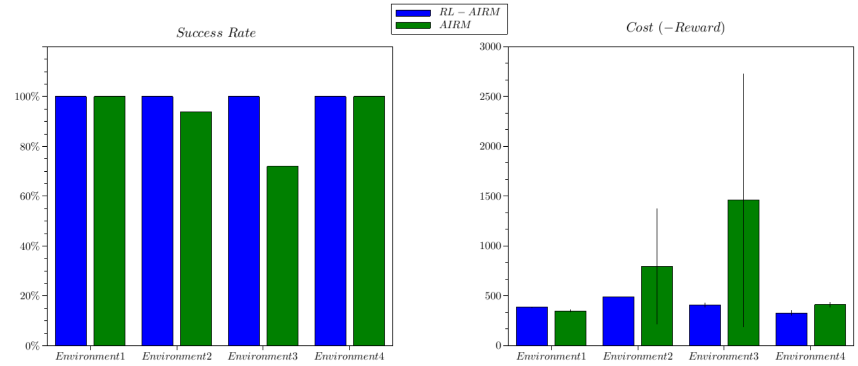


Fig. 7. Evaluation of the RL-AIRM and AIRM architectures in terms of Success rates and Costs

Table 1. Means and variances of performance measures collected on 100 validation tests

	RL-AIRM				AIRM			
Data	Env1	Env2	Env3	Env4	Env1	Env2	Env3	Env4
Rewards	-386±7	-492±6	-408±23	-329±28	-350±12	-797±584	-1462±1271	-412±28
Avoid	404±9	404±9	404±10	406±10	320±10	355±20	312±76	394±23
Recharge	224±27	266±43	339±73	270±42	217±45	235±69	252±97	198±24
Escape	192±14	199±37	272±35	234±29	95±1	98±3	90±17	104±3
Survival	180	180	180	180	180	179±4	160±30	180
Failures	0%	0%	0%	0%	0%	6%	28%	0%
Cycles	1135±1	1135±1	1135±1	1135±1	1135±1	1130±20	1000±200	1135±1

Overall, reinforcement learning seems effective in regulating attention allocation strategies and behaviors' activations. The combined use of attentional mechanisms and learning strategies permits good performance in terms of reliability, adaptivity, effectiveness, and efficiency.

5 Conclusions

We presented a RL approach to attentional action selection in a robotic setting. Differently from classical RL models for action selection, where actions are chosen according to the operative/perceptive contexts, in our case the action selection is mediated by the attentional status of the behavior. In our setting, the learning process adapts and modulates attentional allocation strategies while action selection is obtained as a consequence. We discussed the approach considering learning and executive performance in a survival domain. The collected results show that RL is effective in regulating simple attention allocation mechanisms and the associated behaviors' activations strategies.

Acknowledgments. Work supported by the European Community, within the FP7 ICT-287513 SAPHARI project.

References

1. Bandera, C., Vico, F.J., Bravo, J.M., Harmon, M.E., Iii, L.C.B.: Residual q-learning applied to visual attention. In: ICML 1996, pp. 20–27 (1996)
2. Burattini, E., Rossi, S.: Periodic adaptive activation of behaviors in robotic system. IJPRAI 22(5), 987–999 (2008)
3. Burattini, E., Rossi, S., Finzi, A., Staffa, M.: Attentional Modulation of Mutually Dependent Behaviors. In: Doncieux, S., Girard, B., Guillot, A., Hallam, J., Meyer, J.-A., Mouret, J.-B. (eds.) SAB 2010. LNCS, vol. 6226, pp. 283–292. Springer, Heidelberg (2010)
4. Houk, J.C., Adams, J.L., Barto, A.G.: A model of how the basal ganglia generate and use neural signals that predict reinforcement. In: Houk, J.C., Davis, J.L., Beiser, D.G. (eds.) Models of Information Processing in the Basal Ganglia, pp. 249–270. MIT Press, Cambridge (1995)
5. Kahneman, D.: Attention and Effort. Prentice-Hall, Englewood Cliffs (1973)
6. Montague, P.R., Dayan, P., Sejnowskw, T.J.: A framework for mesencephalic dopamine systems based on predictive hebbian learning. J. Neur. 1936–1947 (1996)
7. Norman, D., Shallice, T.: Attention in action: willed and automatic control of behaviour. Consciousness and Self-Regulation: Advances in Research and Theory 4, 1–18 (1986)
8. O'Reilly, R., Frank, M.: Making working memory work: A computational model of learning in the frontal cortex and basal ganglia. Neural Computation 18, 283–328 (2006)
9. Paletta, L., Fritz, G., Seifert, C.: Q-learning of sequential attention for visual object recognition from informative local descriptors. In: ICML 2005 (2005)
10. Posner, M.I., Presti, D.E.: Selective attention and cognitive control. TINS 10, 13–17 (1987)
11. Senders, J.: The human operator as a monitor and controller of multidegree of freedom systems, pp. 2–6 (1964)
12. Sutton, R., Barto, A.: Reinforcement learning: An introduction, vol. 1. Cambridge Univ. Press (1998)
13. Tan, M.: Multi-agent reinforcement learning: Independent vs. cooperative agents. In: ICML 1993, pp. 330–337. Morgan Kaufmann (1993)
14. Watkins, C., Dayan, P.: Q-learning. Machine Learning 8(3), 279–292 (1992)

Analysing an Evolved Robotic Behaviour Using a Biological Model of Collegial Decision Making

Gianpiero Francesca[1], Manuele Brambilla[1], Vito Trianni[2],
Marco Dorigo[1], and Mauro Birattari[1]

[1] IRIDIA, CoDE, ULB, Brussels, Belgium
{gianpiero.francesca,mbrambil,mdorigo,mbiro}@ulb.ac.be,
[2] ISTC-CNR, Rome, Italy
vito.trianni@istc.cnr.it

Abstract. Evolutionary robotics can be a powerful tool in studies on the evolutionary origins of self-organising behaviours in biological systems. However, these studies are viable only when the behaviour of the evolved artificial system closely corresponds to the one observed in biology, as described by available models. In this paper, we compare the behaviour evolved in a robotic system with the collegial decision making displayed by cockroaches in selecting a resting shelter. We show that artificial evolution can synthesise a simple self-organising behaviour for a swarm of robots, which presents dynamics that are comparable with the cockroaches behaviour.

1 Introduction

In recent studies, evolutionary robotics (ER, see [1]) has been used as an instrument to investigate the evolutionary conditions for the emergence of adaptive behaviour in groups of interacting agents. The main motivation behind these studies is that the evolution of certain adaptive traits and behavioural responses is tightly linked to ecological and social conditions. These conditions are extremely difficult or impossible to be controlled and replicated with empirical studies [2], while they can be completely managed in ER studies. The use of ER to analyse adaptive behaviours has been demonstrated in several occasions. For instance, the effects of genetic relatedness on the evolution of cooperative communication strategies can be investigated by systematically varying the composition of interacting groups [3,4]. Similarly, thanks to a simple ER experiment, it has been shown that the effect of stochastic variations in the evolutionary history could be at the basis of the emergence of diverse signalling strategies [5].

At the same time, ER represents a powerful design tool for the synthesis of collective, self-organising behaviours in swarms of robots [6]. It provides an automatic design methodology to synthesise the individual mechanisms leading to an optimal group response, according to a user-defined performance metric. Additionally, ER can shed light on the evolutionary pressures leading to the emergence of observed collective behaviours. However, it is necessary to understand whether or not the target behaviour can be evolved in the artificial system, and whether it displays dynamics comparable with the natural counterpart.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 381–390, 2012.

In this paper, we perform this first step, that is, the validation of an ER system with respect to collegial decision making by cockroaches in selecting a resting shelter [7]. Cockroaches (*Blattella germanica*) are gregarious insects that manifest cooperative behaviour in selecting a resting site: whenever more than one site is present, the insects collectively choose to aggregate in one single place (provided that it is large enough to host them all). Experimental studies allowed to determine which are the social influences that lead to such a collegial decision-making process, and a dynamical model has been developed (see [7] and Section 3.1 for more details). The identified mechanisms have been successfully exploited for designing collective aggregation and decision-making behaviours in swarms of robots [8,9,10], allowing also mixed insect-robot experiments [11]. However, to the best of our knowledge, there has been no attempt to study the evolution of a similar decision-making behaviour in swarms of robots. In this paper, we demonstrate that similar collegial decisions can be evolved in an artificial system. Our goal is to (i) verify the evolvability of the collegial decision making in the artificial system, and (ii) determine whether the dynamics of the system correspond qualitatively and quantitatively to the ones predicted by the biological model [7]. This will allow us to determine whether or not evolutionary robotics is suitable for formulating hypotheses about the evolutionary pressures that resulted in collective decision-making in cockroaches.

The paper is organised as follows. In Section 2, we describe in detail the experimental setup for the ER experiments. In Section 3, we discuss the results obtained from the evolutionary experiments with respect to the evolvability of the decision-making behaviour in a robotic system. In Section 3.1, we present the dynamical model proposed in [7], and we discuss the methodology that leads us to fit the evolved behaviour to the model. In Section 3.2, we present a comparison of the dynamics of the evolved behaviour with the ones predicted by the model. Section 4 concludes the paper with some final remarks.

2 Experimental Setup

We study the evolution of collegial decision making in a swarm of robots that have to aggregate in one of two areas within the experimental arena. Our experimental setup is based on the one used in Amé et al. [7]. The robots operate in a dodecagonal arena (Figure 1 left) of area 4.91 m^2 surrounded by walls. The floor of the arena is white with two black circular areas having the same radius ($r_a = r_b = 35$ cm) and centred at 67 cm from the walls. In the following, we refer to the two black areas as area a and b, and the remaining white area as c.

The experiments are carried out in simulation using ARGoS [12], a multi-engine simulator of swarm robotics systems. The robots and the environment are modelled using a 2D dynamic physics engine. We use a simulated model of the e-puck robot (Figure 1 right), a small wheeled robot designed for research and education [13]. In our experimental setup, each robot can perceive walls and other robots through eight infrared proximity sensors placed all around its chassis. It can sense the colour of the floor using three ground sensors placed under its front.

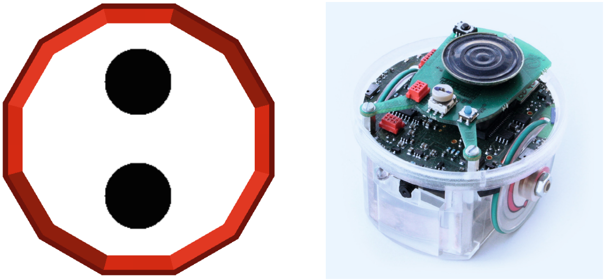


Fig. 1. Left: the simulated experimental arena used for the experiments. Right: the e-puck robot, used for the simulated evolutionary experiments presented in this paper.

Additionally, each robot features another sensor called *range and bearing* [14]. This sensor allows the robot to communicate locally with other robots by sending and receiving messages. In our experiments, the robot uses such a sensor only to perceive the number of other robots within a 70 cm range. To normalise the output of the sensor, we use the preprocessing function $z(n) = 1 - (\frac{2}{1+e^n})$, where n is the number of robots perceived at any given moment. Since the real e-puck can perceive no more than 5 robots at a given time, $z(n)$ saturates to 1 for $n > 5$.

The controller that governs each robot is an artificial neural network. We assume that robots can achieve aggregation using a memoryless behaviour, that is, the behaviour of each robot depends only on the present values of sensors without any kind of internal state. For this reason, we use as controller a fully connected, feed-forward neural network. This neural network has 12 inputs, one for each sensor (8 infrared proximity, 3 ground sensors, 1 from the range and bearing), 2 outputs, one for each wheel, and no hidden units. The input values are linearly scaled in [0,1] when necessary. The activation of the output neurons is computed as the weighted sum of all input units plus a bias term, filtered through a standard logistic function. The two output neurons control the speed of the two wheels, by scaling their activation in the range $[-v_m, v_m]$, with $v_m = 16\,\text{cm/s}$.

We use a simple evolutionary algorithm to set the parameters of the neural network. Each parameter is represented in the genotype by a real number in the range [-5,5]. The evolutionary algorithm works on a population of 100 genotypes, evolved for 200 generations. The population of the first generation is randomly generated. Subsequent generations are created using a selection and reproduction process that involves elitism and mutation. The 20 best genotypes—i.e., the elite—are included unchanged in the next generation. The remaining genotypes of the population are generated by mutation of the genotypes of the elite. The mutation is done by adding a random value to each element of the genotype. The random value is drawn from a normal distribution with mean 0 and variance 1.

The genotype is mapped into a controller that is instantiated in all the robots of the group ($N = 10$). To evaluate the performance, 10 trials of $T = 250$ seconds are run. The evaluation of the performance of the genotype is based on the function $f(t)$:

$$f(t) = \frac{|x_a(t) - x_b(t)|}{N} \in [0, 1] \tag{1}$$

where $x_i(t)$ is the number of robots in area $i \in \{a, b\}$ at time t and N is the total number of robots. The function $f(t)$ is equal to zero when a and b contain the same number of robots. On the contrary, $f(t)$ is equal to 1 when all the robots aggregate on the same area. Fluctuations of $f(t)$ are smoothed through an exponential moving average with time constant $\alpha = 0.9$:

$$G(t) = \alpha G(t-1) + (1-\alpha) f(t) \in [0, 1] \tag{2}$$

where $G(0) = 0$. Finally, the fitness F of the genotype is the average of $G(T)$ over all the 10 trials.

3 Results

We performed 20 evolutionary runs starting from different randomly generated populations. For each run, we selected the best controller within the final population: we evaluated the performance of every controller of the last generation for $K = 200$ trials, and we selected the one with the highest average fitness. All the evolutionary runs were able to produce controllers with high performance (data available as supplementary material in [15]).

A qualitative analysis of the obtained controllers reveals that the evolved behaviours are quite similar one to the other. In general, the robots act differently according to their position in the arena. When a robot is in the white area c, it explores the arena following a wide curved trajectory. If the robot reaches the external wall of the arena, it starts to follow it. The robot motion is influenced by the presence of other robots: curves become sharper when other robots are nearby. Such a perturbation makes the robot leave the border of the arena and eventually enter in one of the two black areas. When the robot is in one black area it follows a circular trajectory. The radius of the trajectory decreases as the number of robots in the area increases. In this way, if the area is empty the robot follows a wide trajectory and eventually leaves. On the contrary if the area is crowded the robot almost rotates on its axis. If the robot goes out of the black area it starts again to explore the arena. Example videos of the obtained controller are available as supplementary material [15].

There are qualitative similarities between the evolved behaviour just described and the self-organizing aggregation behaviour observed in groups of cockroaches. In particular, we observed that the probability that a robot leaves an area is inversely proportional to the number of the robots located in the area itself. To determine whether or not the evolved behaviour presents dynamics quantitatively similar to the biological system, we check the adherence of the evolved robotic behaviour[1] with the model introduced in [7]. In Section 3.1, we introduce the model and the methodology we used to estimate its parameters. In Section 3.2, we compare the dynamics of the evolved behaviour with the predictions of the mathematical model.

[1] To this aim, we select the best obtained controller among all evolutionary runs.

3.1 Model

In Amé et al.'s model [7], the behaviour of each individual insect is characterised by J_i, its probability to join area i, and L_i, its probability to leave area i. Both probabilities depend on x_i, the number of insects located in area i, and on S, the *carrying capacity*, that is, the maximum number of insects that can be hosted in a single area.

The joining probability J_i decreases slightly with the number of insects in area i because of crowding effects. This accounts for the observation that it is less probable to join an area that is already densely populated. Amé et al. define J_i as:

$$J_i = \mu \left(1 - \frac{x_i}{S}\right), \quad i = [a, b]; \tag{3}$$

where μ represents the area quality, that is, the probability that an individual joins the area without social influences, x_i is the number of insects already in area i, and S is the carrying capacity.

Similarly, the leaving probability L_i is inversely proportional to the number of individuals in area i. This accounts for social influences among individuals, which tend to stay close together. L_i is low when the area is densely populated and high when it is sparsely populated. Amé et al. define L_i as:

$$L_i = \frac{\theta}{1 + \rho \left(\frac{x_i}{S}\right)^2}, \quad i = [a, b]; \tag{4}$$

where θ depends on the quality of the area, and ρ is a reference surface ratio related to the area carrying capacity. Using J_i and L_i it is possible to describe the time evolution of the number of individuals in the different areas through a system of differential equations:

$$\frac{dx_i}{dt} = J_i x_c - L_i x_i = \mu x_c \left(1 - \frac{x_i}{S}\right) - \frac{\theta x_i}{1 + \rho \left(\frac{x_i}{S}\right)^2}, \quad i = [a, b] \tag{5}$$

$$N = x_c + x_a + x_b \tag{6}$$

where N is the total number of individuals and x_c is the number of individuals in c, that is, the individuals outside the black areas. This model therefore describes the dynamics of the aggregation behaviour in terms of the number of individuals present in the different areas of the arena (see [7] for details).

To evaluate the correspondence of the evolved behaviour with the model, we estimated the model parameters from the results of simulated experiments.

To estimate the parameters of J_i and L_i, we gathered the empirical probabilities by performing 200 simulated experiments in the same conditions as presented in Section 2 ($N = 10$, $r = 0.35\,\mathrm{cm}$, $S = 29$). We separately fitted the parameters for J_i and L_i to our data using the non-linear least squares method. The obtained parameters are: $\mu = 0.008$, $\theta = 0.008$ and $\rho = 138.574$. We measured the quality of the fitting by computing the coefficient of determination R^2.

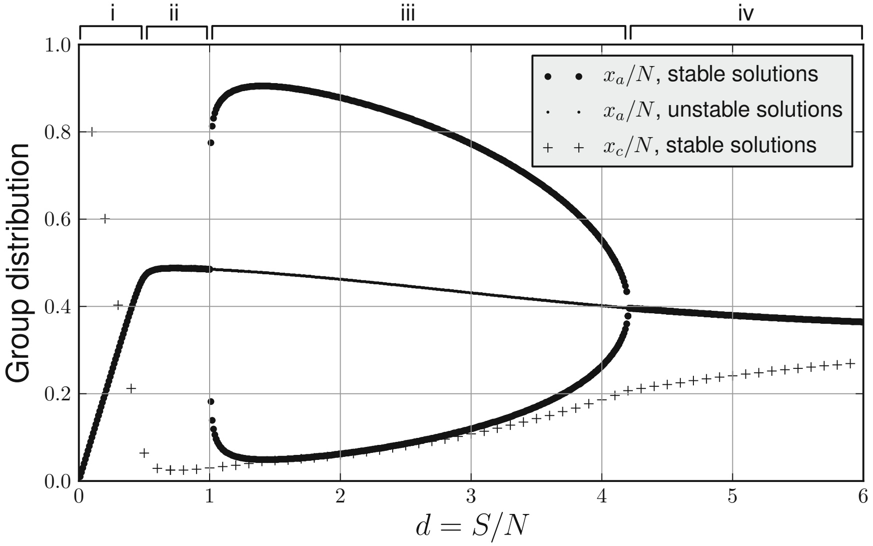


Fig. 2. The bifurcation diagram of our model for different values of $d = \frac{S}{N}$. The percentage of robots in area a and area c.

While the fitting on L_i is excellent ($R^2 = 0.979$, p-$val < 0.001$), the fitting on J_i is not as good ($R^2 = 0.560$, p-$val = 0.148$). This is due to the fact that in our robotic system, J_i appears to be non-linear, differently from Amé et al.'s model. Even though the fitting is not good, we decided to be consistent with Amé et al.'s model and not change J_i. A discussion of the possible effects of this decision is presented in Section 4.

Following the analysis presented in [7], we studied the system behaviour described by eq. (5) and (6) for different values of $d = \frac{S}{N}$. In Fig. 2, it is possible to see the bifurcation diagram of the model. Four different situations can be observed: (i) For d lower than 0.5, the areas are too small to host all the robots; the robots fill completely the areas and some remain in c. (ii) For $0.5 \leq d < 1$, a single area is too small to host all the robots, so the areas are filled equally. However, in this second case, since there is enough space on the areas for all the robots, only few robots are on c. (iii) For $1 < d \leq 4.2$ the areas are big enough for aggregation to happen. Two stable solutions are found, corresponding to area a or area b hosting the majority of the robots. Additionally an unstable solution is found, corresponding to both areas filled equally. (iv) For d greater than 4.2 the areas are too big and the robots are less likely to perceive the presence of other robots in the same area. Thus, the stable solution corresponds to both areas filled equally.

The number of robots present in c described by eq. (6) also varies with d. Two different situations can be observed: For d lower than 0.5, x_c/N decreases sharply: as areas a and b get bigger, more space is available and the areas can host more and more robots; For d greater than 0.5, the population fraction on c increases steadily. This is due to the fact that, as S becomes bigger, the probabilities J_i and L_i increase, resulting in a system less likely to converge on a state in which all robots are in areas a or b.

We consider that a collective decision has occurred when $x_a/N > 0.8$. In the bifurcation diagram in Fig. 2 this happens only for d between 1 and 2.8. For d between 2.8 and 4.2 the model predicts a more variable condition with a still unbalanced distribution of robots among the two areas, and an increasing number of robots that move from one area to the other. In the following, we verify these model predictions with respect to the experimental data, presenting a comparison between the results obtained in simulation and those obtained with the model.

3.2 Dynamics of Robotics and Model Simulations

We compared the results obtained from simulated robotics experiments and Monte Carlo experiments for different values of $d = S/N$. We carried out two different analyses. In the first one, the different values of d are obtained by keeping the number of robots fixed to $N = 10$ and varying the carrying capacity S, by changing r_i. In the second, we keep $r_i = 35\,\text{cm}$ (which corresponds to $S = 29$) and we vary the number of robots. For each value of d, we run 1000 trials of $T = 500$ seconds, both for the robotic and the Monte Carlo simulations. For each trial, we collected the final group distribution x_i over the different areas.

Figures 3 and 4 show the obtained results. For each value of d, one bar for each area of the arena is reported. The colours in the stacked bars show the frequency of individual distributions. We divided the distributions in five classes (0-20%, 20-40%, 40-60%, 60-80%, 80-100%), giving each class a different colour. The size of each class in the figures is proportional to its frequency. If the robots are able to perform a collegial decision and aggregate in one single area most frequently, the bars of the areas a and b are mostly dark blue, corresponding to a bimodal distribution with peaks in 0-20% and 80-100%. On the contrary, if the group splits by aggregating in both areas, the bars of the areas a and b are mostly white, corresponding to a unimodal distribution centred in 40-60%. Area c is depicted in dark red when empty (0-20%) and white when full (80-100%).

Figure 3 shows the comparison between robotics and model simulations when the number of robots is fixed to $N = 10$. Apart from low values of S, there is a good correspondence between the model and the evolved behaviour. Moreover, the evolved behaviour looks more stable for $d > 2.9$, indicating that the robots have a better tendency to perform collegial decision than predicted by the model. For $S = \{5, 11\}$—corresponding to $r_i = \{15, 20\}\,\text{cm}$—the robots find the areas with difficulty due to the small radius and aggregates are less stable.

Figure 4 shows the results when the carrying capacity S is fixed to 29. The evolved behaviour presents a smoother transition from equally occupying the areas at low d to collegial decisions at high d. For $8 < N < 12$ there is a good correspondence, as the robotic system is close to the evolutionary conditions. Differently, for $N \geq 13$ robots split more frequently than aggregating, while the

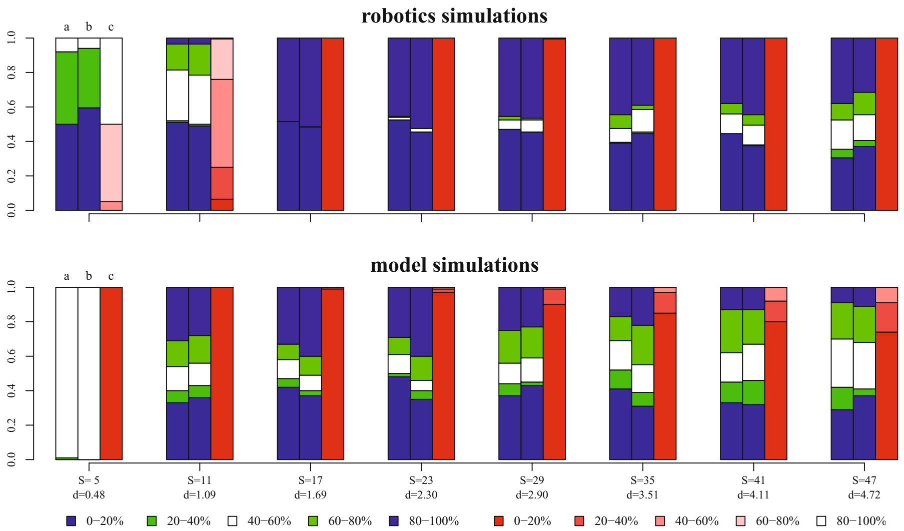


Fig. 3. Comparison between the behaviour of the simulated experiments and the Monte Carlo experiments keeping the number of robots fixed to $N = 10$, and varying the size of the black areas from $r_i = 15\,\text{cm}$ to $r_i = 50\,\text{cm}$

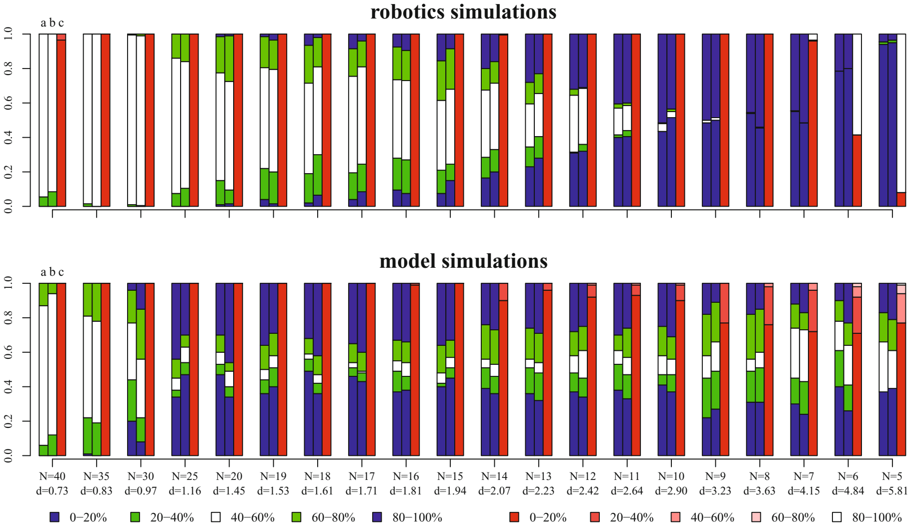


Fig. 4. Comparison between the behaviour of the simulated experiments and the Monte Carlo experiments keeping a constant radius $r_i = 35\,\text{cm}$ ($S = 29$) and varying the number of robots from $N = 5$ to $N = 40$

model predicts splitting only when one area is saturated. Overall, we observe a good qualitative correspondence between robotics simulations and the model, but mostly within the range of parameters used to evolve the robotics behaviour. A more detailed discussion about these discrepancies follows in the next section.

4 Conclusions

In this paper, we demonstrated that evolutionary robotics techniques can be used to synthesise a collegial decision making behaviour similar to the one observed in cockroaches. This is an important result, especially considering that the robotic controllers are simple feed-forward neural networks without internal states. That is, also in a robotic system collegial decisions can emerge solely from simple individual behaviours modulated by social interactions.

We compared the dynamics of the evolved robotic behaviour with the predictions of the model proposed in [7], finding some qualitative correspondence. However, quantitative comparisons revealed similar dynamics mostly for a small parameter range around the evolutionary conditions ($N = 10$, $S = 29$). We identify two reasons for these discrepancies: (i) the evolved system exploits geometric regularities, such as the arena dimension and the positioning of the areas; (ii) the sensing radius for the robots (75 cm) is quite large with respect to the arena dimensions. Both these issues have a bearing on the probability of joining an area, which also explains the non perfect fit of the model parameters observed in Section 3.1. In practice, we observe that the evolved behaviour depends on both S and N, and not only on the their ratio d, as predicted by the model. In future work, by removing geometric regularities from the evolutionary setup, we hope to obtain a better quantitative matching with the model predictions. If successful, we plan to exploit this artificial experimental setup to investigate the optimality of the evolved behaviour with respect to different selective pressures, genetic relatedness among individuals in the group, and variable ecological conditions. We believe this can be useful to better understand the evolutionary path leading to collegial decision making.

Acknowledgements. The research leading to the results presented in this paper has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement n 246939. M. Brambilla, M. Birattari and M. Dorigo acknowledge support from the belgian F.R.S.-FNRS.

References

1. Nolfi, S., Floreano, D.: Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines. MIT Press, Cambridge (2000)
2. Adami, C.: Digital genetics: unravelling the genetic basis of evolution. Nature Reviews Genetics 7(2), 109–118 (2006)
3. Waibel, M., Keller, L., Floreano, D.: Genetic team composition and level of selection in the evolution of cooperation. IEEE Transactions on Evolutionary Computation 13(3), 648–660 (2009)

4. Mitri, S., Floreano, D., Keller, L.: The evolution of information suppression in communicating robots with conflicting interests. Proceedings of the National Academy of Sciences 106(37), 15786–15790 (2009)
5. Wischmann, S., Floreano, D., Keller, L.: Historical contingency affects signaling strategies and competitive abilities in evolving populations of simulated robots. Proceedings of the National Academy of Sciences 109(3), 864–868 (2012)
6. Trianni, V., Nolfi, S.: Engineering the evolution of self-organizing behaviors in swarm robotics: A case study. Artificial Life 17(3), 183–202 (2011)
7. Amé, J., Halloy, J., Rivault, C., Detrain, C., Deneubourg, J.-L.: Collegial decision making based on social amplification leads to optimal group formation. Proceedings of the National Academy of Sciences 103(15), 5835–5840 (2006)
8. Garnier, S., Gautrais, J., Asadpour, M., Jost, C., Theraulaz, G.: Self-organized aggregation triggers collective decision making in a group of cockroach-like robots. Adaptive Behavior 17(2), 109–133 (2009)
9. Campo, A., Garnier, S., Dédriche, O., Zekkri, M., Dorigo, M.: Self-organized discrimination of resources. PLoS ONE 6(5), e19888 (2011)
10. Brambilla, M., Pinciroli, C., Birattari, M., Dorigo, M.: Property-driven design for swarm robotics. In: Proceedings of 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012), Richland, SC, pp. 139–146. IFAAMAS (2012)
11. Halloy, J., Sempo, G., Caprari, G., Rivault, C., Asadpour, M., Tâche, F., Said, I., Durier, V., Canonge, S., Amé, J.M., Detrain, C., Correll, N., Martinoli, A., Mondada, F., Siegwart, R., Deneubourg, J.-L.: Social integration of robots into groups of cockroaches to control self-organized choices. Science 318(5853), 1155–1158 (2007)
12. Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Caro, G.D., Ducatelle, F., Stirling, T., Gutiérrez, A., Gambardella, L.M., Dorigo, M.: ARGoS: a modular, multi-engine simulator for heterogeneous swarm robotics. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011), pp. 5027–5034. IEEE Computer Society Press, Los Alamitos (2011)
13. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering, pp. 59–65. IPCB: Instituto Politécnico de Castelo Branco (2009)
14. Gutiérrez, Á., Campo, A., Dorigo, M., Donate, J., Monasterio-Huelin, F., Magdalena, L.: Open E-puck range & bearing miniaturized board for local communication in swarm robotics. In: ICRA 2009: Proceedings of the 2009 IEEE International Conference on Robotics and Automation, pp. 3111–3116. IEEE Press, Piscataway (2009)
15. Francesca, G., Brambilla, M., Trianni, V., Dorigo, M., Birattari, M.: Analysing an evolved robotic behaviour using a biological model of collegial decision making: Complete data (2012), Supplementary information page at http://iridia.ulb.ac.be/supp/IridiaSupp2012-009/

On the Evolution of Homogeneous Multi-robot Teams: Clonal versus Aclonal Approach

Elio Tuci[1] and Vito Trianni[2]

[1] Aberystwyth University, Computer Science Department
Llandinam Building, Aberystwyth, Ceredigion, SY23 3DB, UK
elt7@aber.ac.uk
http://users.aber.ac.uk/elt7/
[2] Institute of Cognitive Science and Technology
Via San Martino della Battaglia 44, 00185 Rome, Italy
vito.trianni@istc.cnr.it
http://laral.istc.cnr.it/trianni/

Abstract. This study compares two different evolutionary approaches to the design of homogeneous multi-robot teams in a task that requires the agents to specialise in different roles. Our results diverge from what illustrated in a previous similar comparative study, which advocates for the superiority of the aclonal versus the clonal approach. We question this argument in view of new empirical evidence showing that the two approaches perform equally well in generating homogeneous teams.

Keywords: Swarm Robotics, Evolutionary Robotics.

1 Introduction

Homogeneous multi-robot systems are a class of autonomous multi-agent systems in which all robots of a team have identical physical structure and identical decentralised control system. Like social insects, homogeneous robots are generally required to coordinate their actions in order to maximise the efficiency of the team. If there are various tasks that need to be performed than a dynamic task allocation phase should assign agents to different tasks in order to optimise the system level performance [1,10]. The synthesis of controllers for homogeneous multi-robot teams is a complex problem that has been faced with a large number of different techniques [7]. Among the various possibilities, Evolutionary Robotics represents a viable approach for the automatic synthesis of robot controllers requiring little a priori knowledge about the solution of a given problem [6]. In recent years, a number of studies have investigated the dynamics underlying the evolution of multi-robot systems in order to develop a principled understanding of how to guide self-organisation [8,3,11,9]

One of the first papers to focus on this issue is the study described in [8], in which the author compares the clonal and the aclonal approach for the design of homogeneous controllers for teams of two agents required to move in a coordinated way by remaining within sensor range. The clonal approach refers to the

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 391–400, 2012.

use of a single genotype to generate a homogeneous team. The aclonal approach uses different genotypes from the same evolving population to generate teams in which the agents have different controllers (i.e., heterogeneous teams). The study shows that, regardless of the theoretical disadvantages clearly listed and discussed in the paper, the aclonal approach can be a more efficient way than the clonal approach to generate homogeneous systems in which the team members have to autonomously specialise for the benefit of the team. To account for these counter-intuitive results, the author formulates a hypothesis according to which the aclonal approach takes advantages of specific evolutionary dynamics that are precluded to the clonal approach. In particular, in the aclonal approach behavioural roles can be developed and refined in genetically specialised agents, prior to the evolution of generalist solutions, in which roles are dynamically allocated between the agents during life-time. In the clonal approach, this gradual evolution from genetically specialised to generalist solutions is not possible, because, given that the agents are clones, the adoption of complementary roles necessarily requires the existence of some dynamic role allocation mechanisms. Thus, behavioural roles and the mechanisms to allocate them have to (laboriously) evolve simultaneously.

This study provides further comparisons between clonal and aclonal approaches for the evolution of homogeneous multi-robot teams for tasks that require individuals to take specific roles. We are moved by the hypothesis that the results shown in [8], concerning the superiority of the aclonal versus the clonal approach, may have been affected by task-specific features, such as: a) the fact that functional differentiation between the roles may not be a prerequisite to perform the task; and b) the use of an evaluation function composed of team-based metrics (e.g., the position of the team given by the centre-point between the robots). Point (a) calls into question the causal relationship between the supposed nature of the task and the evolutionary dynamics observed. Point (b) calls into question the strong constraints that the evaluation function may have imposed to the evolutionary dynamics of the clonal approach. Contrary to what shown in [8], in our scenario, the roles are clearly different, and the evaluation function is made of robot-based (instead of team-based) factors. We show that in these conditions the clonal and aclonal approaches perform equally well in generating homogeneous teams. Our results indicate that the argument formulated in [8] concerning the superiority of the aclonal versus the clonal approach is based on an interpretation of the data that would require further empirical support to be validated. This is because we reproduced those data showing that they are open to alternative interpretations, not considered in [8].

2 Methods

2.1 The Task and the Simulation Environment

Teams comprising two simulated Khepera mini-robots are evaluated in the context of a dynamic role-allocation task. By taking inspiration from the behaviour of social insects, the roles are nest patrolling and foraging (hereafter, we refer to

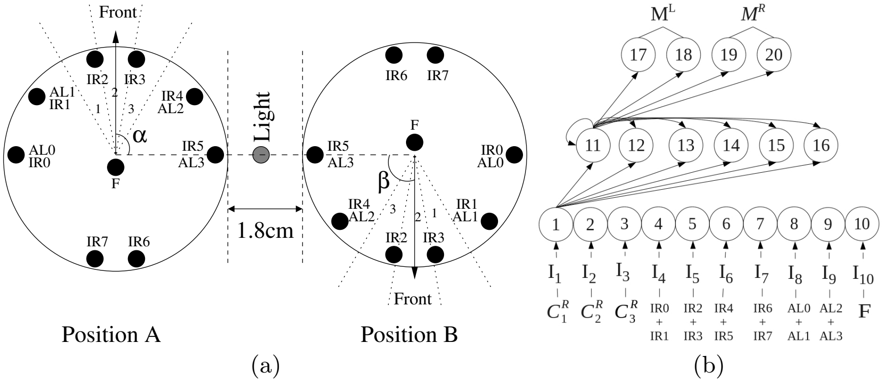


Fig. 1. a) Kheperas' body-plan. The black circles refer to the position of infra-red (IR), ambient-light (AL), and floor sensors (F). The dotted lines indicated view with the camera's sectors. α and β are the parameters defining the set of 15 different initial team positions. The grey circle is the light. b) The neural network.

them as *role P*, and *role F*, respectively). Roughly speaking, *role P* requires a robot to remain within the nest (i.e., an area in which the colour of the floor is in shades of grey). *Role F* requires a robot to move back and forth between the nest and any of the two foraging sites located in the environment. The robots are required to execute both roles simultaneously. Therefore, they should go through a role-allocation phase in which they autonomously decide who is doing what, and then execute their role[1].

The environment is a boundless arena with a light bulb positioned 6cm above the floor, and two red cylindrical objects (2.7cm radius, and 10cm height) positioned at 40cm on the left and on the right of the light, respectively, and referred to as *L-Site*, and *R-Site*. The colour of the arena floor is white except for a circular area (15cm radius), centred around the lamp, within which the floor is in shades of grey. The inner part of the circular area (up to 5cm to the light) is black, the middle part (from 5cm to 10cm from the light) is dark grey, and the outer part (from 10cm to 15cm to the light) is light grey. The area in shades of grey represents the nest. The cylindrical objects represent the foraging sites.

The robots kinematics are simulated using a modified version of the "minimal simulation" technique described by Jakobi in [4]. Our simulation models a Khepera robot, a 2.7cm radius cylindrical robot. It is provided with eight infra-red sensors (IR^i with $i = \{0, .., 7\}$), which give the robot a noisy and non-linear indication of the proximity of an obstacle (in this task, an obstacle can be another robot or a foraging site); four ambient light (AL^i with $i = \{0, .., 3\}$) sensors to detect light; a linear camera; and a floor sensor (F) positioned facing downward on the underside of the robot (see Fig. 1a). The IR and AL sensor values are

[1] See also `http://users.aber.ac.uk/elt7/suppPagn/sab2012/suppMat.html` for further methodological details, pictures, and movies of best teams.

extrapolated from look-up tables provided with the Evorobot simulator (see [5]). The F sensor can be conceived of as a IR sensor capable of detecting the level of grey of the floor. It returns 0 if the robot is on white floor, 0.5 if is on light grey floor, 0.75 if is on dark grey floor, and 1 if is on black floor. The robots camera has a receptive field of 30°, divided in three equal sectors, each of which has three binary sensors (C_i^B for blue, C_i^G for green, and C_i^R for red, with $i = \{1, 2, 3\}$ indicating the sector). Each sensor returns a value in between $[0, 1]$. The camera can detect coloured objects up to a distance of 60cm. The robots can not see each other through the camera. The robot has left and right motors which can be independently driven forward or reverse, allowing it to turn fully in any direction. The robot maximum speed is 8cm/s.

2.2 Robot Controllers and the Evolutionary Algorithm

The robot controller is composed of a continuous time recurrent neural network (CTRNN) of 10 sensor neurons, 6 inter-neurons, and 4 motor neurons (see [2]). The structure of the network is shown in Fig. 1b. The states of the motor neurons are used to control the speed of the left and right wheels as explained later. The sensor neurons are simply relay units. The states of inter and motor neurons are updated using the following equations:

$$\tau_i \dot{y}_i = -y_i + \sum_{j=1}^{10} \omega_{ji}\sigma(gI_j + \beta_j) + \sum_{j=11}^{16} \omega_{ji}\sigma(y_j + \beta_j);\ \text{for } i = \{11, .., 16\}; \quad (1)$$

$$\Delta y_i = -y_i + \sum_{j=11}^{16} \omega_{ji}\sigma(y_j + \beta_j);\ \text{for } i = \{17, .., 20\}; \quad (2)$$

with $\sigma(x) = (1+e^{-x})^{-1}$. In these equations, using terms derived from an analogy with real neurons, y_i represents the cell potential, τ_i the decay constant, g is a gain factor, I_i with $i = \{1, .., 10\}$ is the activation of the i^{th} sensor neuron (see Fig. 1b for the correspondence between robot's sensors and sensor neuron), ω_{ji} the strength of the synaptic connection from neuron j to neuron i, β_j the bias term, $\sigma(y_j + \beta_j)$ the firing rate (hereafter, f_i). All sensory neurons share the same bias (β^I), and the same holds for all motor neurons (β^O). τ_i and β_i with $i = \{11, .., 16\}$, β^I, β^O, all the network connection weights ω_{ij}, and g are genetically specified networks' parameters. At each time step, the output of the left motor is $M^L = f_{17} - f_{18}$, and the right motor is $M^R = f_{19} - f_{20}$, with $M_L, M_R \in [-1, 1]$. Cell potentials are set to 0 when the network is initialised or reset, and equation 1 is integrated using the forward Euler method with an integration time step $\Delta T = 0.1$.

A simple evolutionary algorithm using linear ranking is employed to set the parameters of the networks. The population contains 100 genotypes. Generations following the first one are produced by a combination of selection with elitism, and mutation. For each new generation, the three highest scoring individuals ("the elite") from the previous generation are retained unchanged. The remainder of the new population is generated by fitness-proportional selection from the

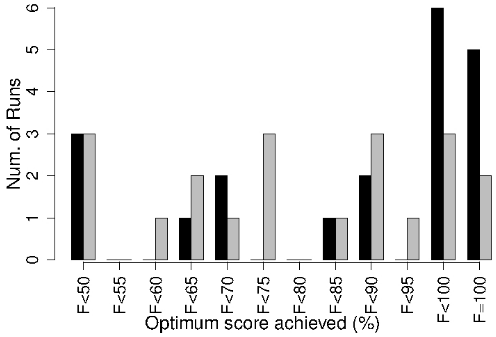
70 best individuals of the old population. Each genotype is a vector comprising 135 real values (120 connections, 6 decay constants, 8 bias terms, and a gain factor). Initially, a random population of vectors is generated by initialising each component of each genotype to values chosen uniformly random from the range [0,1]. New genotypes, except "the elite", are produced by applying mutation, which entails that a random Gaussian offset is applied to each real-valued vector component encoded in the genotype, with a probability of 0.07. The mean of the Gaussian is 0, and its standard deviation is 0.1. During evolution, all vector component values are constrained to remain within the range [0,1].

2.3 Evaluation and Fitness Function

At the beginning of each evaluation trial, the robots are placed in the nest, located symmetrically on the left and on the right of the light, at 1.8cm away from each other. Their controllers are reset. The initial relative orientation of the two robots is sufficiently described by a vector of two variables (α, β, see Fig. 1a). A sample set of starting configuration is chosen such that $\alpha, \beta \in (0, \frac{2\pi}{5}, \frac{4\pi}{5}, \frac{6\pi}{5}, \frac{8\pi}{5})$. From these combinations, 10 have been removed because they are rotational duplicates. This leaves the set of 15 team starting relative orientations that have been used. Each trial differs from the others in the initialisation of the random number generator, which influences the robots' initial distance and orientation, and the noise added to motors and sensors (see [4] for further details on sensors and motor noise). Within a trial, the team life-span is $40s$ (T=400 simulation cycles). Trials are terminated earlier if either one of the robot exceeds the arena limits (i.e., a circle of 120cm radius, centred on the light), or the team exceeds the maximum number of collisions (i.e., 10), or a robot completes two foraging trips (i.e., twice the journey to any of the food sites and back to the nest).

The evaluation procedure for what concerns the clonal and aclonal runs substantially matches the one described in [8]. In clonal runs, the fitness of a genotype is its average team evaluation score after it has been assessed twice for each of the 15 starting configurations, for a total of $E = 30$ trials. The fitness of a genotype in an aclonal run is the average evaluation score of the team in which it participates. In aclonal runs, a genotype is evaluated four times for each starting configuration, twice from each of the robots positions (i.e., position A and position B, see Fig. 1a) comprising each configuration, for a total of $E = 60$ trials. Each one of an aclonal individual 60's trails is undertaken with a different, randomly chosen, partner.

In each trial e, the team is rewarded by an evaluation function F_e which corresponds to the product of the following components: $F_e = \max(C_1^{role\ P} \times C_2^{role\ F}; C_1^{role\ F} \times C_2^{role\ P}) \times P^a \times P^b$. $C_r^{role\ P} \in [0,1]$ rewards a robot $r = \{1,2\}$ for staying in the nest; $C_r^{role\ F} \in [0,4]$ rewards a robot r for travelling twice the distance from the nest and to any of the two food sites; the team collision penalty P^a is inversely proportional to the number of collisions, with $P^a = 1$ with no collisions, and $P^a = 0$ with 10 collisions in a team; P^b is the team penalty for exceeding the arena's limits, with $P^b = 1$ if none of the robots exceeds the limits, $P^b = 0.3$ otherwise. The average team evaluation score is $F = \frac{1}{E}\sum_{e=1}^{E} F_e$.



	mean ranking	median	mean (s.d.)
clonal	22.9	96.21	81.50 (25.73)
aclonal	18.1	78.59	74.66 (24.54)

Fig. 2. The histogram shows the distributions of the highest average re-evaluation scores achieved by each run of the clonal (black bars) and aclonal approach (grey bars). Values represent percentage of the theoretical optimum average evaluation score F. The Table on the right shows, for clonal and aclonal approach, mean ranking, median, mean and standard deviation of the scores mentioned above.

3 Results

20 evolutionary runs, each using a different random initialisation, were carried out for each of the two approaches (i.e., clonal and aclonal). Each run lasted 2500 generations. Recall that our objective is to compare the performances of the clonal and aclonal approach for the evolution of homogeneous team of two robots capable of dynamically allocating and simultaneously executing *role P* (i.e., nest patrolling) and *role F* (i.e., foraging). Following the procedure illustrated in [8], at the end of the evolutionary phase, we run a first set of re-evaluations consisting of 60 trials per team (i.e., 4 times for each of the 15 starting orientation mentioned in Sec. 2.3). In these tests, the 10 fittest genotypes of each generation of both clonal and aclonal runs are re-evaluated in a homogeneous setup. The average re-evaluation score of each genotype is measured using the metrics F illustrated in Sec. 2.3. The highest average re-evaluation score recorded during a given run is assumed to be an adequate measure of the success of that run.

Contrary to what illustrated in [8], the results of our re-evaluations show no significant difference between the performances of homogeneous teams generated clonally and aclonally. 11 runs of the clonal approach produced high-scoring teams exceeding 95% of the optimal score, with 5 of them 100% successful (see Fig. 2, black bars). In contrast, only 5 out of 20 of the aclonal runs generated a homogeneous team that exceeds 95% of the optimal score, with 2 of them capable of completing the 60 re-evaluation trials with the highest score. The Table on the right of Fig. 2 shows a comparisons of mean and median scores and the mean ranking of both approaches. The difference between the two set of results is not significant (Mann-Whitney U test, $p > 0.1$);

From a statistical point of view, there is not enough evidence to prefer one approach over the other for the evolution of homogeneous multi-robot teams engaged in a dynamic task allocation scenario. From the point of view of

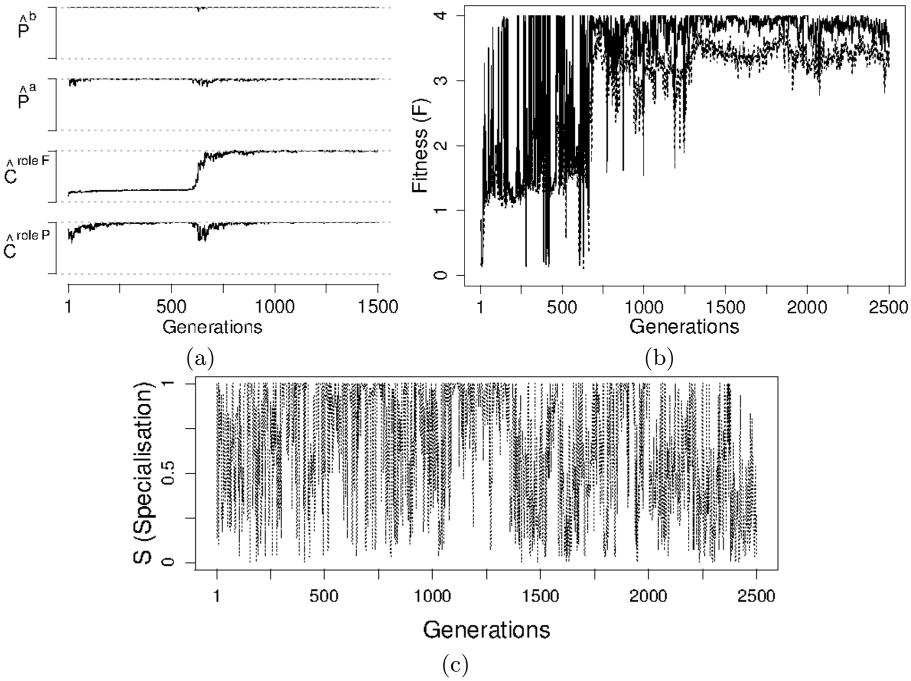


Fig. 3. a) The evolutionary history of the fitness components of the best genotypes of one of the best clonal run. At each generation, $\hat{C}^{role\ P}$, $\hat{C}^{role\ F}$, $\hat{P^a}$, and $\hat{P^b}$ are the mean values of $C_e^{role\ P}$, $C_e^{role\ F}$, P_e^a, and P_e^b, respectively, over 60 trials. b) Average evaluation score (F) of best heterogeneous combination of genotypes (continuous line), and best homogeneous team (dashed line), computed during a set of re-evaluation tests on the 10 fittest genotypes of the best aclonal run. c) Level of specialisation (S) in the best heterogeneous combination of genotypes that contributed to the fitness curve (continuous line) shown in (b).

generating optimal controllers, the clonal approach does better than the aclonal one (i.e., 5 teams with a 100% success rate generated with the clonal approach, see Fig. 2, black bars, versus 2 teams with the aclonal approach, see Fig. 2, grey bars). This evidence not only diverges from what shown in [8], but also questions the argument, put forward in Quinn's paper, concerning the superiority of the aclonal approach. Recall that, according to Quinn, the aclonal approach takes advantage of the fact that roles can be evolved and refined prior to the evolution of any dynamical allocation mechanism. This is assumed not possible in the clonal approach for which the roles and the allocation mechanisms have to evolve simultaneously. Our results do not fit into the explanatory framework proposed by Quinn. To account for this divergence, we question the simultaneity argument. Should we always assume that in the clonal approach, behavioural roles and mechanisms to allocate them have to evolve simultaneously?

To answer this question, we analyse the evolutionary trajectory of the best clonal runs looking for the emergence of roles. In our scenario, the clonal approach generated optimal (in term of fitness) solutions by avoiding the simultaneous appearance of behavioural roles and role-allocation mechanisms. All the best clonal runs follow a very similar evolutionary trajectory. Fig. 3a shows the evolutionary history of the fitness components of the best genotypes of one of the best clonal run. The graph shows that early generation teams are very good in *role P* (see $\hat{C}^{role\ P}$ in Fig 3a), and very bad in *role F* (see $\hat{C}^{role\ F}$ in Fig 3a). After some generations, in which no progress can be observed, the fitness starts to increase owe to the evolution of the behavioural skills required to perform *role F*. By visual inspection of the team behaviour, we noticed that foraging behaviour appears first in a limited number of trials. These trials are those in which the robots, due to their initial relative orientations, experience different perceptual states which break the team symmetry and facilitate the role-allocation process. In following generations, we observe that robots perform foraging behaviour in a larger set of trials, and the role-allocation process becomes less dependent on the robots' initial perceptual states. Eventually, the robots acquire the skills to negotiate their role regardless of their initial perceptual states and become capable of performing the task in all the experienced initial conditions.

In Sec. 4, we will come back to the issue of simultaneity with further speculations. In what remains of this section, we focus on another important aspect of the Quinn's argument, the specialisation in heterogeneous teams generated aclonally. We run a second set of re-evaluation tests identical to the one used in [8] to investigate whether in the aclonal approach specialisation appears before the role allocation ability. In these tests, the 10 fittest genotypes from every generation of the best aclonal run are taken and retested in a heterogeneous setup. That is, each genotype is re-evaluated in combination with every other fittest genotypes of its generation, for 60 trials per combination. As in [8], for each generation, we selected the highest average score F among those produced by each combination, and we compared them with the highest average scores F obtained by these genotypes when re-evaluated in a homogeneous setup (i.e., during the first set of re-evaluation tests). The continuous line in Fig 3b refers to the average re-evaluation score (F) of the best heterogeneous combinations; the dashed line refers to the average score of the best homogeneous team.

As in [8], we also have, for significant period of this aclonal run, a certain disparity between the fitness of heterogeneous and homogeneous teams. This disparity appears to be more prominent right after generation 750, when the fitness of heterogeneous teams oscillates around the optimum, while the fitness of homogeneous teams is lower. In [8], this disparity is accounted to by the existence of genetic specialisation in aclonally generated heterogeneous teams. The logic is rather simple. If robots are genetically predisposed to play either one or the other role, they get high fitness (F) when they are evaluated in combination with a partner that is specialised to play the complementary role, and low or lower fitness when they are evaluated with their clones. In [8], evolutionary dynamics driven by the "specialisation factor" represent the basis of the superiority of the aclonal over the clonal approach. Contrary to clonal runs, aclonal runs can partition the task allocation scenario in two successive phases: a first one, in which evolution generates genetically specialised behavioural roles; a second one, in which evolution finds the way to move from specialised to generalist solutions (i.e., a single genotype that generates agents capable of dynamic allocation and execution of the roles of the task).

In order to deepen our understanding on the causal relationship between specialisation and the fitness disparity observed in Fig 3b, we measured the level of specialisation (S) of the best heterogeneous combinations that contributed to the generation of the fitness curve in Fig 3b, continuous line. In particular, we looked at the number of times, in each combination, the robots play each role during the 60 re-evaluation trials. The role that a robot plays in a trial is determined by how it contributes to the team fitness in that trial. For example, robot 1 plays *role P* and robot 2 plays *role F* if $C_0^{role\ P} \times C_1^{role\ F}$ is bigger than $C_0^{role\ F} \times C_1^{role\ P}$, and vice versa. For each combination, $S = \frac{|N^{role\ P} - N^{role\ F}|}{60}$, with $N^{role\ P}$ and $N^{role\ F}$ being the number of times in 60 trials in which one of the robot plays *role P* and *role F*, respectively. $S = 1$ means that robots are highly specialised; $S = 0$ no specialisation at all. Results are show in Fig 3c.

By comparing the graph in Fig 3b, continuous line, with the graph in Fig 3c, in particular focusing on the performances after generation 750, we clearly see that not all the best combinations are highly specialised. While specialisation rises and falls, the best average score of these heterogeneous combinations only slightly fluctuates around the optimal score. This indicates that both specialisation and a certain level of generalist solutions are simultaneously present in the aclonal population. As far as it concerns the argument formulated in [8], it seems that the disparity between the scores of heterogeneous and homogeneous teams is not necessarily a sign of specialisation among the individuals of an aclonal population, as mentioned in [8]. To conclude, in our dynamic task allocation scenario, the aclonal population dynamics appear to be more articulated than what described in [8]. Specialisation did not turn out to be completely alternative and antecedent to generalist solutions. We believe that this evidence points at interesting evolutionary dynamics, not considered in [8], which suggests that the argument concerning the superiority of the aclonal versus the clonal approach in task-allocation scenario needs to be revisited.

4 Conclusions

This study aimed at testing the hypothesis that the results shown in [8], concerning the superiority of the aclonal versus the clonal approach, have been affected by task-specific features. We designed a similar task-allocation scenario which mainly differs from what shown in [8] for the nature of the behavioural roles. Basically, in our task the roles are clearly distinct, whereas in [8] the roles are rather similar. Contrary to what illustrated in [8], we found no significant difference between the aclonal and clonal approach. The analysis of the evolutionary trajectories of clonal and aclonal runs produced evidence that did not support the reasoning put forward in [8], to account for his results. First, we showed that, in a scenario in which the differences between the roles is captured by an evaluation function that multiply robot-based (instead of team-based) factors, the clonal approach can rely on a broader range of evolutionary trajectories not necessarily limited by the simultaneity argument formulated in [8]. In the clonal approach, the fitness landscape can be successfully explored by capitalising on gradual improvements on the execution of single behavioural roles, and on the appearance of allocation mechanisms (initially) bounded to specific ecological conditions. Second, we showed that the author in [8] drew conclusions on the presence of role specialisation among the individuals generated aclonally from empirical evidence (i.e., the fitness disparity shown in Fig. 3b) that do not seem to be uniquely produced by genetically specialised individuals. We showed that in the aclonal approach, specialised and generalist solutions can live together in the same populations, with role allocation mechanisms evolving together with, and not necessarily after, genetic specialisation. In conclusion, the evolutionary dynamics of both clonal and aclonal approaches seem to be richer then what assumed in [8]. Although further investigation is required, in view of this, we speculate that the argument formulated in [8] applies to a rather restricted set of task-allocation scenarios, in which the selective pressures limit the possible successful evolutionary trajectories in the clonal approach.

Acknowledgments. V. Trianni acknowledges funding from the H^{2}SWARM research project (European Science Foundation).

References

1. Ampatzis, C., Tuci, E., Trianni, V., Christensen, A.L., Dorigo, M.: Evolving self-assembly in autonomous homogeneous robots: Experiments with two physical robots. Artificial Life 15(4), 465–484 (2009)
2. Beer, R.D., Gallagher, J.C.: Evolving dynamic neural networks for adaptive behavior. Adaptive Behavior 1(1), 91–122 (1992)
3. Floreano, D., Mitri, S., Magnenat, S., Keller, L.: Evolutionary conditions for the emergence of communication in robots. Current Biology 17, 514–519 (2007)
4. Jakobi, N.: Evolutionary robotics and the radical envelope of noise hypothesis. Adaptive Behavior 6, 325–368 (1997)
5. Nolfi, S.: EvoRob 1.1 User Manual. Institute of Psychology, National Research Council (CNR) (2000), http://gral.ip.rm.cnr.it/evorobot/simulator.html
6. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-organising Machine. MIT Press, Cambridge (2001)
7. Parker, L.: Springer Handbook of Robotics, ch. 40. Springer, Berlin (2008)
8. Quinn, M.: A comparison of approaches to the evolution of homogeneous multi/robot teams. In: Proceedings of the International Conference on Evolutionary Computation (CEC), vol. 1, pp. 128–135 (2001)
9. Trianni, V., Nolfi, S.: Engineering the evolution of self-organizing behaviors in swarm robotics: A case study. Artifcial Life 17(3), 183–202 (2011)
10. Tuci, E.: An investigation of the evolutionary origin of reciprocal communication using simulated autonomous agents. Biological Cybernetics 101(3), 183–199 (2009)
11. Waibel, M., Keller, L., Floreano, D.: Genetic team composition and level of selection in the evolution of cooperation. IEEE Transaction of Evolutionary Computation 13(3), 648–660 (2009)

Biologically-Inspired Deceptive Behavior for a Robot*

Jaeeun Shim and Ronald C. Arkin

Mobile Robot Laboratory, School of Interactive Computing
Georgia Institute of Technology, Atlanta GA 30308, USA
jaeeun.shim@gatech.edu, arkin@cc.gatech.edu

Abstract. A common behavior in animals or human beings is deception. We focus on deceptive behavior in robotics because the appropriate use of deception is beneficial in several domains ranging from the military to a more everyday context. In this research, novel algorithms are developed for the deceptive behavior of a robot, inspired by the observed deceptive behavior of squirrels for cache protection strategies, evaluating the results via simulation studies.

1 Introduction

A common and essential behavior for survival in a variety of intelligent systems ranging from insects to human beings is deception. Many biologists and psychologists define deception in various ways. According to Vrij [1], deception is "A successful or unsuccessful deliberate attempt, without forewarning, to create in another a belief that the communicator considers to be untrue in order to increase the communicators' payoff at the expense of the other side." Da Waal argued that "Deception can be defined as the projection, to one's own advantage, of an inaccurate or false image of knowledge, intentions, or motivations" in his paper [4]. We can find a simpler definition of deceptive behavior from a paper by Bond and Robinson [2] who defined it as "a false communication that tends to benefit the communicator." We have used this straightforward definition in earlier research in our laboratory on deceptive behavior for robots [19] and we continue to do so here.

In other words, animals act deceptively to gain benefits from others. Biological and psychological findings show that deception plays an important role not only in providing an evolutionary advantage [2]. It appears also in higher-level primates to involve the theory of mind mechanism [3]. We argue that robots can also potentially gain advantage over adversaries by possessing deceptive behaviors. For example, it is obvious that the use of deception is important with respect to the military context [10]. We further posit that to achieve more socially intelligent robots operating in the presence of humans, we must develop robots that interpret, generate, and respond to deceptive behavior. Therefore, we investigate deception in robotics using approaches inspired by biological findings [19,21].

* This work was supported in part by the Office of Naval Research under MURI Grant # N00014-08-1-0696.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 401–411, 2012.

Fig. 1. Black Eastern gray squirrel moving peanuts [29]

In this paper, we present a novel approach for deceptive behavior by a robot, inspired by observations of squirrels (fig 1.) in cache protection strategies [15]. Section 2 reviews relevant animal deceptive behaviors and existing research in robotic deception. In Section 3, deceptive behaviors in food hoarding and protection strategies of squirrels are introduced. In Section 4, a computational model enabling robots to emulate deception behaviors of squirrels is integrated into *MissionLab*, and results discussed in Section 5. Section 6 concludes the paper. We note that we are well aware of the ethical implications associated with robotic deception and our perspective on this subject is discussed elsewhere [22].

2 Related Work

Animal Deception

Animals use various forms of misinformation. These deception mechanisms, achieved by sending false signals either intentionally or unintentionally, are essential for animals' survival. For example, camouflage and mimicry are well known in many species. By resembling other animal species or inanimate objects, animals transmit misinformation to others so that they can avoid detection by both predators and their prey. While camouflage or mimicry are examples of unknowingly deceiving, a deceptive behavior can include seemingly more intentional misinformation.

Many deceptive behaviors are observed from different animals ranging from insects to primates. The spider genus *Portia*, which preys primarily on other spiders, deceives their prey by vibrating the web in ways that resemble a small insect getting ensnared. When the web's resident spider comes to investigate the insects, *Portia* preys on it [19].

According to Ristau's experiment [13], another interesting deceptive behavior appears in piping plovers. These birds exhibit a “broken-wing display" deceptive behavior. By feigning an injured wing and hopping farther and farther from the nest, birds lead the predator away from their young, thus protecting them.

Primates are the species most commonly ascribed with the ability to deceive [3,6]. For example, chimpanzees have multiple deceptive behaviors with several different objectives. When chimpanzees find fruit, they do not move directly so that they do not give any indication to the competitors that they have noticed the location of the foods. This food protective strategy is not that dissimilar to the one we discuss in squirrels later in the paper. Deceptive behavior of chimpanzees is also observed during interactions with humans. According to one observation, a chimpanzee feigned having his arm stuck in the bars of his cage in order to lure a zookeeper nearby. As soon as the human entered to help free his arm, he leaped onto the zookeeper [4].

Another relevant class of deceptive behavior occurs in the food hoarding strategies of animals. Food hoarding (caching) is an important type of animal behavior needed for their survival through periods when nourishment is not readily available. In particular, these caching behaviors are commonly observed in rodents such as hamsters or squirrels [8].

This caching behavior is of particular interest as it can also be useful in the robot context. In this paper we investigate caching and protecting resources for application as a resource protection strategy. In the military domain, robots might face this situation, where it is important to discourage an adversary from discovering a protected site, so the application of these bio-inspired animal food protection behaviors can be particularly beneficial.

In this paper, we focus specifically on the observed deceptive behavior of squirrels while they protect cached food acquired during hoarding [15]. Recent research in the field of biorobotics suggested the robotic squirrel models [28]. According to this study, *robosquirrel* are successfully used for long-term studies on rattlesnake behavior after squirrel encounters. Even though this research showed a good model of squirrel's behavior in robotics, it does not include squirrel's deceptive behaviors, which are potentially useful in several contexts. Different from this work, our research focuses on employing squirrel's deceptive behaviors to robot systems. Section 3 describes this set of behaviors in more detail.

Robot Deception

Endowing robots with the capacity for deception has significant potential utility [18], similar to its use in animals. Clearly, deception behaviors are useful in the military domain [7,10]. Sun Tzu stated in *The Art of War*, "All warfare is based on deception". Military robots capable of deception could mislead opponents in a variety of ways. As both individual and teams of robots become more prevalent in the military's future, [23] robotic deception can provide new advantages apart from the more traditional one of force multiplication. In other areas, such as search and or healthcare, deceptive robots might also add value, for example, for calming victims or patients when required for their own protection. Conceivably even in the field of educational robots, the deceptive behavior of a robot teacher may potentially play a role in improving human learning efficiency.

Despite its ubiquity in nature and its potential benefits, very few studies have been done on deceptive behaviors in robotics to date. Floreano's research group [20] demonstrated robots evolving deceptive strategies in an evolutionary manner, learning to protect energy sources. Their work illustrates the ties between biology, evolution, and signal communication and does so on a robotic platform. They showed that cooperative communication evolves when robot colonies consists of genetically similar individuals. In contrast, when the robot colonies were dissimilar the robots evolved deceptive communication signals.

Wagner and Arkin [18] used interdependence theory and game theory to develop algorithms that allow a robot to determine both when and how it should deceive others. More recent work at Georgia Tech is exploring the role of deception according to Grafen's dishonesty model [24] in the context of bird mobbing behavior [21].

Terada and Ito [16] demonstrated that a robot is able to deceive a human by producing a deceptive behavior contrary to the human subject’s prediction. These results illustrated that an unexpected change of the robot’s behavior gave rise to an impression in the human of being deceived by the robot.

Other research shows that robot deception behavior can increase users' engagement in robotic game domains. Work at Yale University [14] illustrated increased engagement with a cheating robot in the context of a rock-paper-scissor game. They proved greater attributions of mental state to the robot by the human players, when participants played against the cheating robots. At Carnegie Mellon University [17] a study showed an increase of user's engagement and enjoyment in a multi-player robotic game in the presence of a deceptive referee. By declaring false information to game players about how much players win or lose, they observed whether this behavior affects a human's general motivation and interest based on frequency of winning, duration of playing, and so on. These results indicate that deceptive behaviors are potentially beneficial not only in the military domain but also in a human's more everyday context.

3 Deceptive Behaviors in Food Hoarding

In this paper, we focus on the deception behavior of squirrels in terms of their food hoarding strategies. Food hoarding is an important behavior for many animal species, such as birds and rodents. Food-hoarding strategies are mainly comprised of two parts: caching and protecting the food. The deceptive component falls in the food protection phase.

Cache Formation

Food caching activity ranges widely from highly dispersed (scatter hoards) to highly clumped (larder hoards). Scatter hoarders cache a few items in many small/scattered caches. On the other hand, larder hoarders place most of the food in one or a few central locations referred to as middens. The evolution of the particular hoarding strategy for a species depends on the abilities of individuals to defend their caches against pilfering [5]. According to observation, animals use a larder hoarding cache strategy when their competitors are conspecifics or they are weaker than themselves; however, when potential competitors are heterospecific or stronger adversaries, animals tend to use a scatter hoarding strategy [5].

Cache Protection

After hoarding food items, animals begin to protect their resources from pilfering by patrolling the caches. First, animals move around the caching areas and check whether the cached food items are safe. However, animals generally change their behavior after they experienced pilfering.

For example, one general food protecting behavior of animals is changing the locations of its food items. According to Preston's experiments [11,12], after kangaroo rats experienced pilfering from conspecific or heterospecific competitors, they moved the location of their food items.

Of particular use in this study is an interesting deceptive behavior observed in the food protection strategy of certain squirrels [15]. Social context (i.e., presence or absence of competitors) appears to be pivotal to the expression of cache protection behaviors. Deceptive behavior in the tree squirrel has been observed with respect to food protection [15]. While patrolling, tree squirrels visit the cache locations and check on their food. However, if potential competitors are present nearby, tree squirrels visit several empty cache locations. This deceptive behavior attempts to confuse competitors about the food's location, so that they can protect against the loss of their hoarded food. After the potential competitors leave the territory, the tree squirrels move the location of their stored food items, if pilfering has occurred.

4 Computational Model and Implementation

A model of a bio-inspired behavior-based model [25] of squirrel caching and protecting behaviors for application to robotic systems is now presented. Simulations studies were performed in *MissionLab*[1], a software package developed by the Mobile Robotics laboratory at Georgia Tech [9]. *MissionLab* provides a graphical user interface that enables users to specify behavioral states and the control transitions between states easily, yielding a finite state acceptor (FSA), which can then be compiled down to executable code for both simulations and robots. The caching behaviors created for this project are combined with pre-existing behaviors such as avoiding obstacles, moving toward an object, or injecting randomness (noise).

Table 1. States and Triggers

State/Trigger	Description
Caching	Find and pickup food items and store them in safe caching locations
True Patrolling	Move around true caching locations and stay for a finite time based on the amount of food cached
False Patrolling	Move around empty caching locations and stay for a finite time based on the inverted probabilities of true patrol
Enough food cached	Activate when the number of items in a caching location is over the threshold
Select true place	The robot is probabilistically likely to move to a selected TRUE caching place
Select false place	The robot is probabilistically likely to move to a selected FALSE caching place

In this section, the computational model is described that determines how robots behave in resource caching and protecting scenarios inspired by squirrel's behaviors earlier. Like the squirrel's behavior, the model consists of two main parts - caching behavior and patrolling (protecting) behavior. The simulation is based on interactions between two robotic agents: a squirrel robot (resource storer) and a competitor robot (resource pilferer).

Caching Strategy

Many groups, including ours [27], have studied foraging behavior in robotics. In the caching simulation, one robot is required to store the scattered resources in safe

[1] *MissionLab* is freely available for research and educational purposes at: `http://www.cc.gatech.edu/ai/robot-lab/research/MissionLab/`

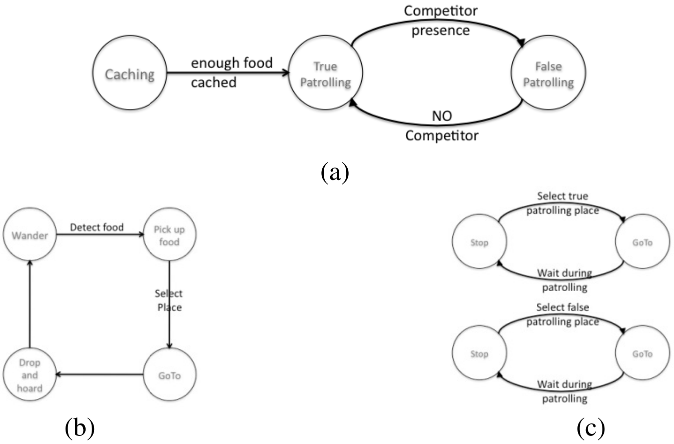


Fig. 2. (a) High-level FSA: caching behaviors of squirrels, (b) sub-FSA: food hoarding, and (c) sub-FSA: food patrolling

locations. Figure 2(a), illustrates the high-level model. The caching sub-state (Fig. 2b) consists of several states and triggers (Table 1). First, the robot wanders around searching for food items. When the robot detects a food item during foraging, it is picked up. Then, the robot selects the place to cache this item based on a pre-defined probabilistic distribution. After selecting a specific caching place out of several choices, the robot moves to the location and drops the item there. The robot repeats this strategy until the "enough food cached" trigger is activated.

Protecting Strategy

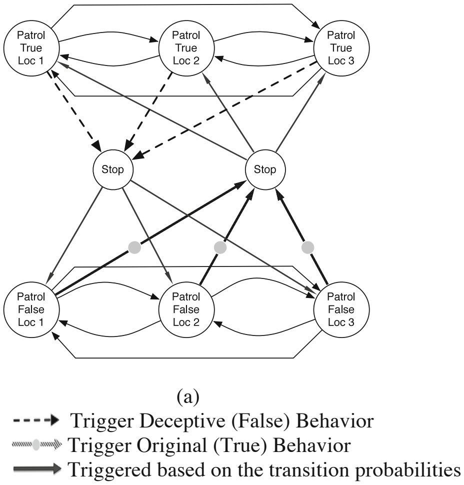
After caching is complete, the robot begins to move between the caching locations to patrol the resources. The behaviors of the robot include goal-oriented movement, selecting places, and waiting behavior (figure 2(c)).

Initially the robot employs the true patrolling strategy, when the "select true location" trigger is activated. This trigger calculates which of the many caching locations the robot should patrol in the current step. The calculation is a random selection based on the transition probabilities among the places. Probabilistic transitions between behavioral states have been used for successfully developing models of wolf pack predation [26]. Transition probabilities are determined by the number of cached items. If a place has more items, the probability to visit is higher. The transition probabilities are calculated by the following equation (1):

$$P_{ij} = \frac{\# items_j}{\sum\limits_{1 \le k \le n, k \ne j} \# items_k}$$

In this equation, P_{ij} is the transition probability from location i to location j, and n is the total number of locations. $\#item_x$ indicates the number of food items in location x.

In each state, the next patrol state is determined based on these transition probabilities. As shown below, the system generates a random number and determines the next location if the number is in certain range (equation (2)).


Patrol True Loc 1
Patrol True Loc 2
Patrol True Loc 3
Stop
Stop
Patrol False Loc 1
Patrol False Loc 2
Patrol False Loc 3
Trigger Deceptive (False) Behavior
Trigger Original (True) Behavior
Triggered based on the transition probabilities

(a)

	Loc 1	Loc 2	Loc 3
# items	10	1	1

(b)

	Loc 1	Loc 2	Loc 3
Loc 1		0.5	0.5
Loc 2	0.9		0.1
Loc 3	0.9	0.1	
Stop	0.8	0.1	0.1

(c) Among true locations

	Loc 1	Loc 2	Loc 3
Loc 1		0.5	0.5
Loc 2	0.5		0.5
Loc 3	0.5	0.5	
Stop	0.33	0.33	0.33

(d) Among false locations

Fig. 3. (a) Example FSA of protecting strategy with three true caching places and three deceptive caching places, (b) number of items in each true caching location, (c) transition probability between true location i and j, (d) transition probability between false location i and j

$$R = \text{random number between 0 and 1}$$

$$NextLocation_i = \begin{cases} Location1 & , \quad R < P_{i1} \\ Location2 & , \quad R < P_{i1} + P_{i2} \\ \quad\vdots & \end{cases}$$

Figure 3 shows an example of the robot's patrolling strategy when it includes three true and three false caching locations. In figure 3(a), the robot moves between the caching locations. The robot determines the transition among the true caching places based on the transition probabilities in figure 3(c). These transition probabilities among the true locations are calculated by equation (1) based on the number of items in each place as shown in figure 3(b).

When the squirrel robot detects the presence of competitor, deceptive behavior is triggered and the squirrel robot patrols the false (empty) caching locations to deceive the competitor. The selection of deceptive locations is also calculated by transition probabilities. Here, the transition probabilities among the false locations are set as uniform distributions (fig. 3(d)). These are not based on ethological observations as they were in the wolf pack case [26], as that data is unfortunately not available.

In each patrolling state in figure 3(a), the robot goes to the cache and remains there for a finite amount of time. The time spent at the cache is determined by the number of food items in that place. If a place contains *n* food items, the robot stays there for *n* seconds. At the end of the waiting phase, the robot selects the next patrolling locations based on equation (2) and goes to the next patrolling state.

Competitor Robot Behavior

A competitor robot has a simple mechanism in the current scenario (fig. 3). The competitor robot simply wanders around the map to try to find the squirrel robot.

When it detects the squirrel robot, it determines whether it is at the potential caching location or not. To recognize the caching area, the competitor robot observes how long the squirrel robot stays in place. Since the squirrel robot takes time to patrol the caching place proportional to the number of food items, the competitor robot can get an evidence of caching area based on the squirrel robot's staying time duration.

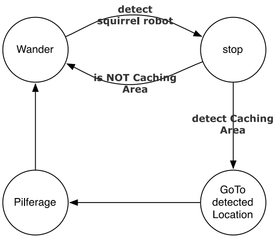
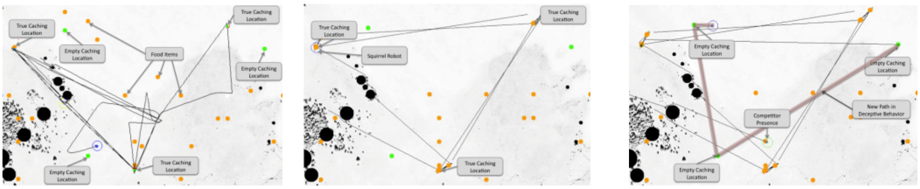


Fig. 4. FSA for the Competitor

Therefore, if the duration is over a threshold, set manually, it activates the "detect caching area" trigger. Then, the competitor robot goes to this location and remains until the end of pilfering. The duration of pilfering is determined by the number of cached items. If the duration is less than the threshold, the competitor determines that the current location of the squirrel robot is not the true cache. It then returns to "wander" state and repeats the detecting process again.



(a) (b) (c)

Fig. 5. Simulation Results. (a) Caching, (b) True patrolling, (c) Deceptive patrolling strategies

5 Simulation Results[2]

A simple scenario of the squirrel-like deceptive behavior was simulated in *MissionLab*. The simulation environment is shown in figure 4. Yellow-colored food items were randomly placed around the map. In this simulation, the robot detects these food items by discriminating colors. Three caching places and three empty places were chosen arbitrarily.

First, the robot finds a food item and stores it in the pre-defined caching places as shown in figure 4(a). When the number of the cached items is over a threshold for any of the caches, the state of the robot switches to the cache protection. If a competitor is not present, it patrols the true caching locations (fig. 4b) Otherwise, the deceptive patrolling strategy is activated, and the robot moves to empty caching places (fig. 4c).

To evaluate the approach, the performance was evaluated by measuring the time duration until the competitor robot detects the exact caching places and begins pilfering. The same scenarios without deceptive behaviors formed the baseline. Comparing the baseline results to the measured time when deception is active, serves

[2] Simulation videos are available at Simulation: `http://www.cc.gatech.edu/ai/robot-lab/hunt/squirrelProject.html`

Table 2. Simulation results of first ten of 30 trials: time duration until competitor successfully pilferages resources in contexts; (a) with deceptive behaviors and (b) without deceptive behaviors. (Measurements given in minutes).

Trials Condition	1	2	3	4	5	6	7	8	9	10
(a) With	8.76	12.73	5.92	9.25	12.33	10.24	10.97	7.8	15	11.79
(b) W/O	6.79	7.80	10.82	3.13	5.42	11.02	12.03	6.08	5.83	8.48

as an evaluation of its effectiveness. The simulation was run 30 times per each condition-with and without deceptive behaviors. In each trial, all the other conditions except deceptive behaviors are the same. Even though the number of cached items varies in each trial, it maintains the same two conditions - with and without deceptive behaviors.

Table 2 and figure 5 show the simulation results. In two, the average time to successful pilferage when the squirrel robot includes deceptive behavior is 10.4 minutes (std: 3.04), compared to the average time duration without deception is 7.69 minutes (std: 2.91). The statistical analysis yielded 0.0009 p-value (< 0.05) with the Student's t-test, a significant difference between the results of the two conditions.

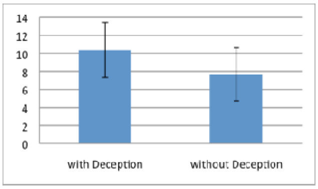


Fig. 6. Average time to pilferage

As a result, it can be concluded that the deceptive behavior affects significantly the robot's performance. With deceptive behaviors, the squirrel robot protects resources longer and performs significantly better than the one without deceptive behaviors.

6 Conclusions and Future Work

In this paper, a novel approach was presented for deception in robots, focusing on how to preserve resource gains. This approach was inspired from biological findings, i.e., deceptive behaviors of eastern grey squirrels during cache protection. Computational algorithms were developed applying these deceptive behaviors to robots. In the evaluation phase, several simulations were run on simple scenarios and it was found that the deceptive behaviors worked effectively and enabled robots to perform better with than without deception.

The current version of our algorithm only handles a scenario with one deceiver robot and one competitor robot. However, to be more realistic and reasonable, it should include multiple autonomous agents. In the foraging strategy, robots may need to determine the probabilistic distribution for storage locations based on their safeness instead of a manually pre-defined distribution. Furthermore, we have a plan to apply our simulations to real robot experiments later. These remain for future work.

As this research focuses on deceptive behaviors of robots in the military domain, where robots may hide and protect resources from humans or other autonomous agents, this deceptive behavior can be beneficial. We will potentially extend our research more towards human-friendly environments. To evaluate the performance, we will conduct Human-Robot Interaction studies with real human subjects.

Adding deceptive behaviors to robots leads to ethical questions, such as whether it is ethical for robots to deceive humans for any purpose. This requires considerable discuss in a broader community, which we actively encourage.

References

1. Vrij, A.: Detecting lies and deceit: the psychology of lying and the implications for professional practice. John Wiley & Sons, New York (2001)
2. Bond, C., Robinson, M.: The evolution of deception. Jour. of Nonverbal. Behav. (1988)
3. Cheney, D.L., Seyfarth, R.M.: Baboon metaphysics: The evolution of a social mind. University of Chicago Press, Chicago (2008)
4. de Waal, F.B.M.: Intentional Deception in Primates. Evolutionary Anthropology (1992)
5. Gerhardt, F.: Food Pilfering in Larder-Hoarding Red Squirrels (Tamiasciurus Hudsonicus). Journal of Mammalogy (2005)
6. Gouzoules, H., Gouzoules, S.: Primate Communication: By nature honest, or by experience wise? International Journal of Primatology (2002)
7. Hawthorne, L.: Military Deception. Joint Publication, JP 3-13.4 (2006)
8. Jenkins, S.H., Rothstein, A., Green, W.C.H.: Food Hoarding by Merriam's Kangaroo Rats: A Test of Alternative Hypotheses. Ecological Society of America (1995)
9. MacKenzie, D., Arkin, R., Cameron, J.: Multiagent Mission Specification and Execution. Autonomous Robotics (1997)
10. Meehan, W.J.: FM 90-2 Battlefield Deception. Army Field Manuals (1988)
11. Preston, S.D., Jacobs, L.F.: Conspecific pilferage but not presence affects Merriam's Kangaroo rat cache strategy. Behavioral Ecology (2001)
12. Preston, S.D., Jacobs, L.F.: Cache decision making: the effects of competition on cache decisions in Merriam's kangaroo rat. Journal of Comparative Psychology (2005)
13. Ristau, C.: Aspects of the cognitive ethology of an injury-feigning bird, the piping Plover. Cognitive Ethology: The Minds of Other Animals (1991)
14. Short, E., Hart, J., Vu, M., Scassellati, B.: No fair!!: an interaction with a cheating robot. In: Proc. 5th ACM/IEEE International Conference on Human-Robot Interaction (2010)
15. Steele, M.A., et al.: Cache protection strategies of a scatter-hoarding rodent: do tree squirrels engage in behavioural deception? Animal Behaviour (2008)
16. Terada, K., Ito, A.: Can a Robot Deceive Humans? In: Proceeding of the ACM/IEEE International Conference on Human-Robot Interaction (2010)
17. Vazquez, M., et al.: A deceptive robot referee in a multiplayer gaming environment. In: Collaboration Technologies and Systems (CTS) International Conference (2011)
18. Wagner, A., Arkin, R.: Acting deceptively: Providing robots with the capacity for deception. International Journal of Social Robotics (2011)
19. Wilcox, R.S., Jackson, R.: Spider-Eating Spiders. American Scientist (1998)
20. Floreano, D., Mitri, S., Magnenat, S., Keller, L.: Evolutionary Conditions for the Emergence of Communication in Robots. Current Biology (2007)
21. Davis, J., Arkin, R.C.: Mobbing Behavior and Deceit and its role in Bio-inspired Autonomous Robotic Agents. In: International Conference on Swarm Intelligence (2012)
22. Arkin, R.C.: Moral Emotions for Robots and the Ethics of Robotics Deception. In: 1st International Conference of the International Assoc. for Computing and Philosophy (2011)
23. U.S. Department of Defense, FY 2009-2034 Unmanned Systems Integrated Roadmap
24. Johnstone, R., Grafen, A.: Dishonesty and the Handicap Principle. Anim. Behav. (1993)
25. Arkin, R.C.: Behavior-based Robotics. MIT Press (1998)

26. Madden, J., Arkin, R.C., McNulty, D.: Multi-robot System Based on Model of Wolf Hunting Behavior to Emulate Wolf and Elk Interactions. In: Proc. IEEE International Conference on Robotics and Biomimetics (2010)
27. Balch, T., Arkin, R.C.: Communication in Reactive Multiagent Robotic Systems. Autonomous Robots (1994)
28. Joshi, S.S., Johnson, R., Rundus, A., Clark, R.W., Barbour, M., Owings, D.H.: Robotic Squirrel Models. IEEE Robotics and Automation Magazine (2011)
29. Wikipedia, `http://en.wikipedia.org/wiki/File:Eastern_Grey_Squirrel-black.jpg`

Automated Synthesis of Locomotion Controllers for Self-reconfigurable Modular Robots

Fernando Torres and Juan Cristóbal Zagal

Department of Mechanical Engineering, University of Chile
Beaufchef 850, 8370448, Santiago, Chile
{fertorre,jczagal}@ing.uchile.cl

Abstract. Previous studies on control of self-reconfiguring modular robots have shown how complex group behavior can be obtained from simple low level interactions. In this study we explore the power of Genetic Algorithms and NEAT to automatically produce group behavior such as locomotion with obstacles. We study the invariance of resulting rule set controllers with respect to different scenarios, scales, and initial robot configurations. Resulting GA controllers performed 17.88% better than NEAT controllers. The use of sequential mode of cell activation was critical for the evolvability of robot controllers.

Keywords: NEAT, Genetic Algorithm, Self-Reconfiguring Modular Robots, Programmable Matter.

1 Introduction

Self-reconfigurable modular robotic systems have the potential of being more versatile and robust than current robots. A group of modules may reconfigure itself into a tool shape, suitable to accomplish an unforeseen task. Eventually, members of the same group may replace damaged or lost components, giving the overall system the capability of self-repair [9].

Achieving unsupervised adaptive self-organization is fundamental on a wide variety of applications, and this type of systems promises to bring new light into the field of robotics and automation. However, there are a number of problems that must be addressed in the way of producing practical low cost modular self-reconfiguring machines.

Planning and control have been recognized amid the great challenges in modular self-reconfigurable robotics [9]. There is a need of algorithms for locomotion of modular systems under environments with obstacles. The later is of utmost relevance, since locomotion is a sub-task of reconfiguration and self-repair.

Various studies on generic locomotion algorithms for self-reconfigurable robots have been carried by Rus and colleagues [1,2]. They focused on methods inspired by cellular automata. The idea is that a set of geometric rules is used to control module actions by looking at their neighborhood. Modules operate in a lattice environment composed either by free space, obstacles or other modules. They are assumed to be able to sense the type of elements on their surroundings, and thus, they can decide where to move by applying a shared geometric rule set.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 412–420, 2012.

Although the rule set only defines low level interactions, it has been shown how complex global behavior arises as a consequence of these local interactions [5]. Unfortunately, there is no clear model of how local rules can be designed or tuned in order to achieve a desired high level behavior. So far, researchers have concentrated on testing manually designed rule sets with a potential for solving locomotion or self-repair tasks.

In [1] rule sets for locomotion with and without obstacles were analyzed. Tests were performed on a 2D lattice simulation environment. A set of five rules was sufficient to achieve locomotion without obstacles. Locomotion over certain type of obstacles was achieved with a set of eight rules. Three different rule evaluation models were tested.

At first, cells were evaluated in a cyclical order (D_0 model). Then, cells were evaluated at random, without allowing for repetition along an evaluation cycle (D_1 model). Finally, cells were evaluated completely at random, allowing for repetition of activation on a single cycle (D_∞ model). They observed how the later, most flexible model, requires more complex rule sets, although it allows for more variability and potential use on distributed systems.

Despite of their success at generating basic locomotion controllers, it was clear that the manual generation of such rules is an increasingly difficult task, especially as the complexity of environment increases and more challenging sophisticated behaviors are required. This encourages the development of automated methods to generate rule sets for self-reconfigurable modular robots.

Stoy [8] also uses cellular automation to guide the reconfiguration of a modular ensemble so as to reach a desired geometrical configuration in three dimensions. The local behavior is automatically generated over the basis of a gradient function constructed with a CAD model of the target geometrical shape. He provides successful examples on the autonomous construction of a configuration resembling a chair.

Promising results on evolving cellular automata rules were obtained by Koza and colleagues in another context [3]. Their rules were evolved for solving the Majority Problem, and achieved a performance better than the best known human-written solution (Gacs-Kurdyumov-Levin rule).

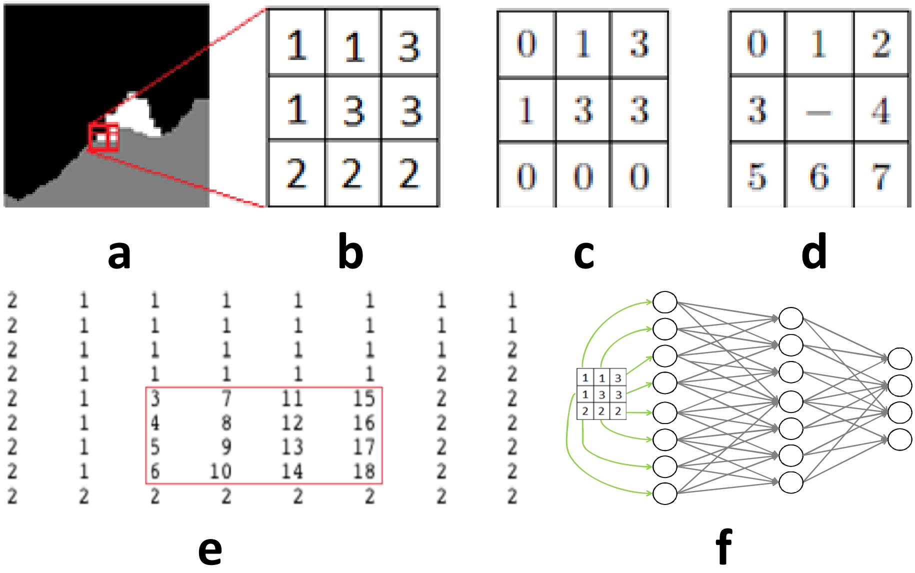
Our present work focuses on the automatic generation of rule sets for the locomotion of self-reconfigurable modular robots. We will explore the use of Genetic Algorithms (GA) [4] and Neuroevolution of Augmenting Topologies (NEAT) [7].

The remainder of this paper is as follows: In section 2 we present our experimental setup. Results of the adaptation process are shown on section 3. In sections 4 and 5 we explore the robustness of our solutions by varying the initial robot shape and scale of simulation. Finally, in section 6 we present the conclusions of our work.

2 Experimental Setup

2.1 Simulation

We used a matrix representation of a 2D lattice environment containing robot cells, background and obstacles. Figure 1 shows various elements of our simulation. An example of a robot and environment is shown in (a). A mask of identifiers centered on an activated cell is shown in (b). A rule input condition is shown in (c). Possible displacements are presented in (d). An example of an entire simulation is presented in (e). Obstacles are represented by “2”, background by “1” and robot cells by numbers greater or equal to “3”. A neural network used as an alternative mapping between geometric conditions and cell displacement is shown in (f).



1	1	3
1	3	3
2	2	2

0	1	3
1	3	3
0	0	0

0	1	2
3	–	4
5	6	7

2	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1
2	1	1	1	1	1	1	2
2	1	1	1	1	1	2	2
2	1	3	7	11	15	2	2
2	1	4	8	12	16	2	2
2	1	5	9	13	17	2	2
2	1	6	10	14	18	2	2
2	2	2	2	2	2	2	2

1	1	3
1	3	3
2	2	2

Fig. 1. a. Example of a robot (white) moving on an environment containing obstacles (grey) and free space (black). The red square represents a mask centered on an activated cell. **b.** The mask is used to evaluate if a rule input condition is satisfied. **c.** Example of a rule input condition where robot cells (“3”) and background elements (“1”) are relevant. In this example the input condition is met and a displacement is triggered. **d.** Eight possible displacements are represented by numbers (0-7). **e.** Matrix representation of a small simulation environment. Obstacles are represented by “2”, background by “1” and robot cells by numbers greater or equal to “3”. **f.** Neural network used as an alternative mapping between geometric condition and displacement.

2.2 Training and Test Environments

We produced complex training and test scenarios by segmenting natural images. Figure 2 shows our training (a) and test scenario (b). The later scenario is particularly challenging, since it usually forces a moving robot to split in two pieces. Simulation environments of different sizes were produced out of these images.

The idea was to test the robustness of resulting behaviors to changes in the scale of simulation. The complete learning process was carried on the training scenario, while the final experiments were done in both training and testing scenarios.

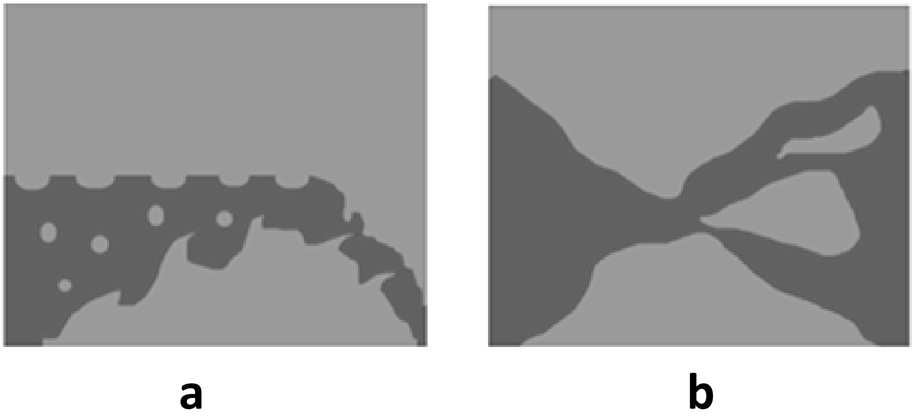


Fig. 2. Training and test scenarios used for our experiments. The darker area represents the background, while the gray area represents the obstacles. **a.** Training scenario. **b.** Test scenario.

2.3 Encoding

In the case of GA, we represented a locomotion controller by an individual binary string genome. The genome is implemented as a string containing pairs of input rule geometric conditions and action displacements. Each rule was represented by a total of 21 bits, where 18 bits are used encoding the input rule and 3 bits are used for representing the displacement. We considered a total of 30 rules for our representation.

In the case of NEAT, we used the numerical representation provided by the simulation itself. The eight outer values of the cell centered mask were fed into a neural network, as shown in Figure 1f. The four output neurons were considered as binary values and paired to represent the horizontal and vertical displacement separately.

2.4 Objective Function

The objective function was designed to maximize overall robot locomotion toward the right most portions of the scenarios while maintaining, at the same time, the connectivity of the robot. We defined the travelled distance d as the averaged position of each robot cell at the end of the simulation. We also defined the connectivity c as the (normalized) size of the largest 4-connected cell group averaged through every simulation step. Both quantities were bounded in the interval [0,1]. Then the fitness f for each candidate individual was defined as

$$f = d \cdot c \tag{1}$$

2.5 Testing Parameters

The training was executed with robots of 4 by 4 cells size. The training scenario was chosen to be 30 (height) by 50 (width) cells. Five learning trials were performed for GA and NEAT. The best performing individual was selected for further characterization on each method.

Further tests consisted on studying the invariance of resulting controllers with respect to changes on robot's initial shape and scale of the simulation. These tests were executed on both, training and test scenarios. Table 1 defines the configurations for initial shape invariance study and Table 2 defines the settings for simulation scale invariance analysis.

Table 1. Initial shape invariance settings. The scenario size was fixed in 30x50 cells.

Shape Index	Robot Height	Robot Width
I	4	4
II	3	5
III	5	3
IV	2	8
V	8	2

Table 2. Scale invariance settings. Each case corresponds to 1x, 2x, 5x and 10x the size of the initial learning configuration.

Scale Index	Scenario Height	Scenario Width	Robot Height	Robot Width
I	30	50	4	4
II	60	100	8	8
III	150	250	20	20
IV	300	500	40	40

3 Results

Figure 3 shows the learning curves for Sequential GA, Random GA and NEAT in terms of number of evaluations. We can observe that Random GA shows signs of premature converegence. NEAT shows the smaller standar deviation. Despite of this, its final fitness is lower than that reached by GA.

The final average fitness is 0.862 for sequential GA, 0.860 for random GA and 0.709 for NEAT. The overall standar deviation is 0.179 for sequential GA, 0.118 for random GA and 0.019 for NEAT. Averages were taken by considering five independent learning trials.

The maximum fitness obtained with GA was 0.936. On the other hand, the best NEAT individual‘s fitness reached 0.73. These were the individuals selected for further characterization.

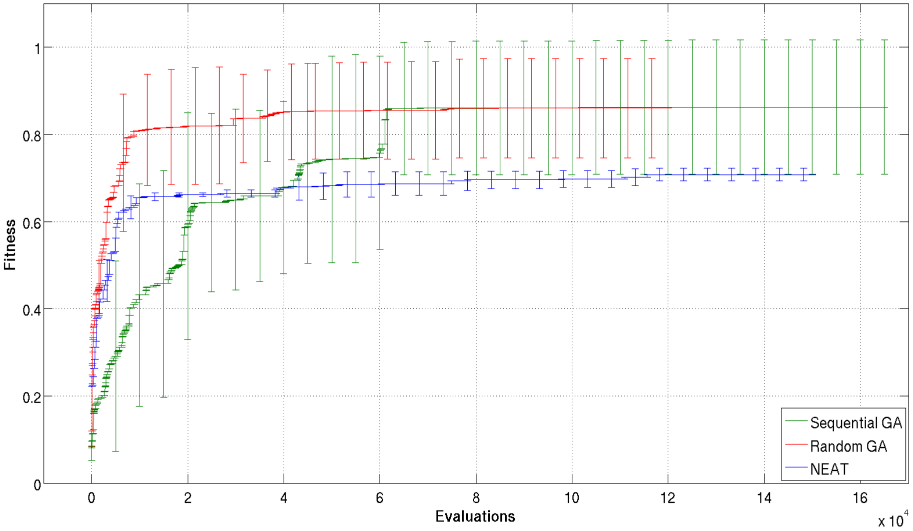


Fig. 3. Resulting learning curves obtained for sequential GA, random GA and NEAT. The final average fitness is 0.862 for sequential GA, 0.860 for random GA and 0.709 for NEAT. The overall standar deviation is 0.179 for sequential GA, 0.118 for random GA and 0.019 for NEAT. Averages were taken by considering five independent learning trials.

4 Initial Shape Invariance

Figure 4 shows the results obtained when varying the initial robot shape. For the case of GA, the fitness obtained was close to the one obtained during training, for all initial configurations and scenarios. On the other hand, the results shown for NEAT didn‘t show a clear relationship between initial shape and fitness.

5 Scale Invariance

Figure 5 shows the fitness obtained with different methods and scenarios while varying the simulation scale. The resulting NEAT behavior showed a higher invariace between scenarios than those obtained with GA, although they don’t show a clear relation between configurations. In the GA case, it can be seen an inverse relationship between the fitness obtained and the simulation scale. This might be due to the number of simulation steps required to cross the scenario. The later implies a disconnected locomotion to reconnect at the end of the scenario (See Figure 6).

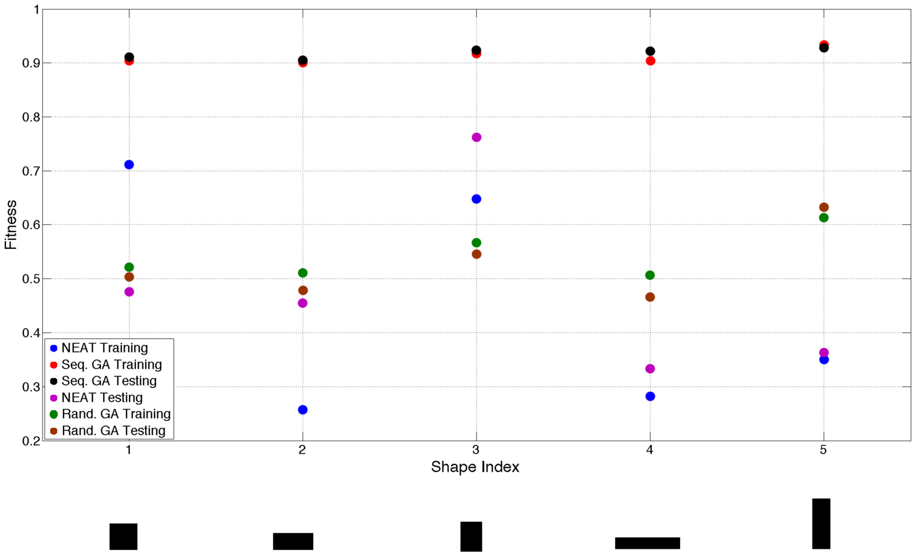


Fig. 4. Summary of fitness values obtained for different methods, scenarios and initial robot shape. Black and red dots are representing the level of fitness obtained when using Sequential GA. There is a clear similarity of results obtained under test and training scenarios. Results show how Random GA always gives lower results under testing, even during training configuration.

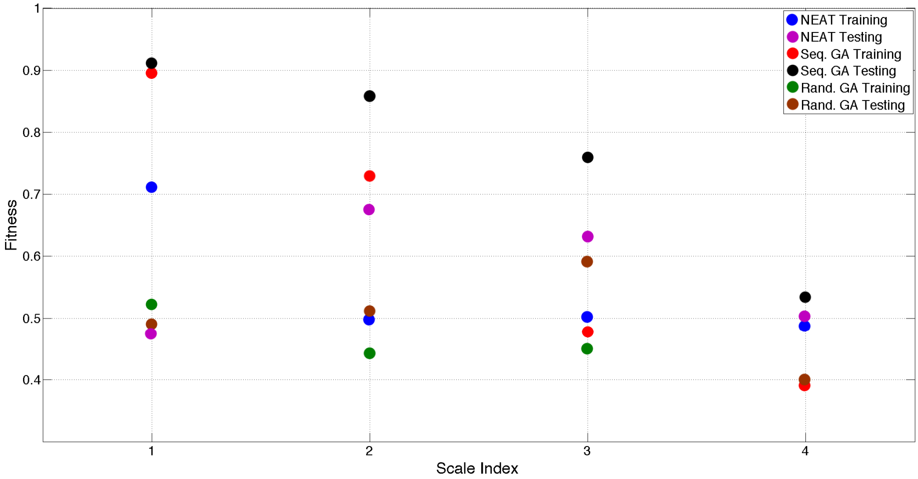


Fig. 5. Fitness values obtained with each evolutionary method in both scenarios and while varying the simulation scale index

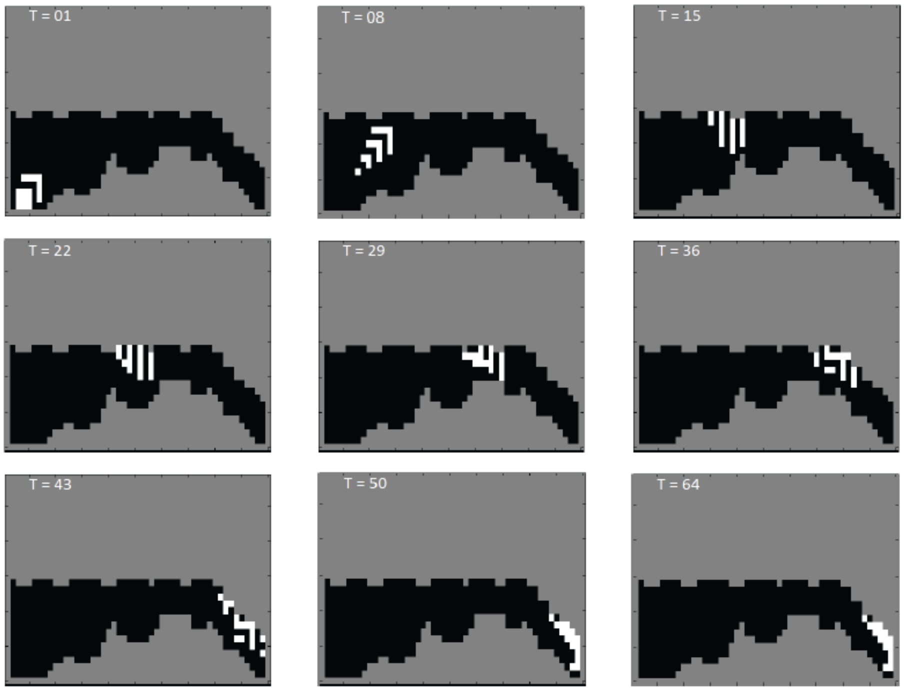


Fig. 6. Snapshots of the best performing individual generated with sequential GA in the training scenario. The gray zone corresponds to the obstacles while white dots are robot cells.

6 Conclusions

We have studied three methodologies to generate decentralized controllers to achieve locomotion of self-reconfigurable modular robots. We have tested the resulting controllers by varying the initial robot's shape and the simulation scale, using different scenarios.

We have observed the convenience of using secuential module activation, so each individual evolved could be characterized by one fitness value. In other hand, random activation resulted in noisy learning curves, where the resulting fitness were not even close to those resulting during the learning process.

Sequential GA results showed invariance to the initial shape and an inverse relation to the scale of simulation. This behavior seems to be independent of the simulation scenario. It was also observed that GA generated controllers with slight better performance in vertical initial robot shapes. On the other hand, NEAT showed no clear relationship through the configurations being tested.

The cellular automata representation used with GA is able to condense several situations into a single rule, while the neural network treats every local scenario as unique. This is true unless some operator is introduced to disable input nodes on the neural network, whereby a more global behavior could be achieved.

The amount of simulation steps explains the decrease in fitness shown on the scale invariance experiments. This variable was somehow uncontrollable, since there was no clear way to stablish stangnation due to the small local movements that disabled the connectivity in the fitness function. Robots were able to increase their scored connectivity index by travelling with a slight separation from each other, but gathering closely at the end.

We note that for all experiments, the desired direction of locomotion was achieved. This raises GA and NEAT as tools to synthesize decentralized controllers in even more complex tasks than those developed so far (including native 3D displacements and reconfigurations).

In a future work we plan to provide comparisons with previous hand made controllers. A fair direct comparison is not yet possible, since this entails adapting the rule domain to the exact settings used on other studies.

Aknowledgements. This research was funded by Fondecyt project number 11110353. We thank the thorough revision provided by anonymous reviewers.

References

1. Butler, Z., Kotay, K., Rus, D., Tomita, K.: Generic decentralized control for a class of self-reconfigurable robots. In: Proc. of IEEE International Conference on Robotics and Automation, ICRA, pp. 809–816 (May 2002)
2. Butler, Z., Kotay, K., Rus, D., Tomita, K.: Generic decentralized control for lattice-based self-reconfigurable robots. International Journal of Robotics Research 23(9), 919–937 (2004)
3. David, A., Bennett, F.H., Koza, J.R.: Evolution of intricate long-distance communication signals in cellular automata using genetic programming. In: Proc. of the Fifth Int. Conference on the Synthesis and Simulation of Living Systems, ALIFE, pp. 513–520 (1996)
4. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co. (1989)
5. Kubica, J., Casal, A.: Complex behaviors from local rules in modular self-reconfigurable robots. In: Proc. of IEEE International Conference on Robotics and Automation, ICRA, pp. 360–367 (2001)
6. Kotay, K., Rus, D.: Generic distributed assembly and repair algorithms for self-reconfiguring robots. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2362–2369 (February 2005)
7. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10(2), 99–127 (2002)
8. Stoy, K.: Using cellular automata and gradients to control self-reconfiguration. Robotics and Autonomous Systems 54(2), 135–141
9. Yim, M., et al.: Modular Self-Reconfigurable Robot Systems. IEEE Robotics and Automation Magazine 14(1), 43–52 (2007)

Towards Detecting Group Identities in Complex Artificial Societies

Corrado Grappiolo and Georgios N. Yannakakis

Centre for Computer Games Research
IT University of Copenhagen, Denmark
{cogr,yannakakis}@itu.dk

Abstract. This paper presents a framework for modelling group structures and dynamics in both artificial societies and human-populated virtual environments such as computer games. The group modelling (GM) framework proposed focuses on the detection of existing, pre-defined group structures and is composed of a reinforcement learning method that infers collaboration values from the society's local interactions and a clustering algorithm that detects group identities based on the learned collaboration values. An empirical evaluation of the framework in the social ultimatum bargain game shows that the GM method proposed is robust independently of the size of the society and the locality of the interactions.

Keywords: Group Identity Detection, Reinforcement Learning, Hierarchical Clustering, Artificial Societies, Complex Adaptive Systems, Complexity.

1 Introduction

The interactions among a population of social individuals (such as humans or simulated agents) — due to factors such as the agents' attitude, personality, cultural biases and stereotypes, and history of past interactions [7] — yield complex dynamics which lead to the formation of global patterns, intended as the behaviour of the system as a whole [7]. Global patterns, such as group identities (i.e. the ability of an individual to identify himself or herself with his or her group [3]) and norms, influence the behaviour of the individuals in an indirect feedback fashion [7], with the potential of generating and aggravating social conflicts, such as social discriminations and other forms of inequalities. A way to effectively detect group formations is through modelling the local-to-global transition (i.e. from agent interactions to group identities). However, doing so is not a trivial task due to the recurrent influence of the global structure in the society-system[1].

The focus of this study is the real-time detection of consolidated group identities in a society of believable, human-like artificial agents. We hereby assume

[1] Sometimes it is referred to as *emergence of complexity* [7]

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 421–430, 2012.

that the group structures have reached an equilibrium, i.e. no further group formation dynamics occur. Following the findings of Dawes and Messick [3], we aim to identify group identity structures by measuring and processing the levels of collaboration among agents. The overview of the proposed group modelling (GM) approach, as depicted in Figure 1, is composed of three key phases. The first phase regards the classification of the interactions occurring between the agents into two values, namely *in-group* and *out-group* — i.e. interacting agents do or do not belong to the same group, respectively [3]. In the second phase a reinforcement learning update rule [15] approximates values of cooperative interaction between the agents based on the *in/out-group* values obtained from phase 1; *collaboration* values [2] are then retrieved by calculating the agents' reciprocal altruistic level of cooperation [7]. Finally, in the third phase, a clustering algorithm [17] determines the number of group identities existing in the society and their corresponding agents.

In this paper we have conducted experiments based on simulations of societies of complex social agents of variant sizes. Social complex and believable agents interact with each other by playing the *social ultimatum bargain game* [1] and are constrained by local interactions which allow them to interact only with a subset of the society. Results demonstrate that probabilistic *in/out-group* group classification functions that approximate the level of favourability the provider agent shows with respect to each receiver agent it interacts with can detect consolidated, pre-defined group identities with high accuracy even in the most complex agent society.

This study is novel as, to the authors' best knowledge, there has not been any attempt to real-time detect group identities in artificial societies based on the sole observation of agents interactions. We argue that our real-time GM framework is applicable to multi-agent scenarios such as artificial societies or multiplayer games, with an emphasis on adaptive multiplayer games [19] in which the social interactions among the individuals are complex [20] and beyond the resource exchange interactions of the social ultimatum game covered in this paper.

2 Related Work

There is a number of studies investigating groups of agents and their behaviours In both artificial societies and virtual environments such as multiplayer games.

Nowak et al. [13], among others, focus on the evolution of collaboration by evolving the policies of artificial agents. The approach, however, neglects the impact of collaboration on group formation. Similar work was conducted by Hammond and Axelrod [6], though it was focused on the evolution of ethnocentrism. Among the studies on collective behaviour, Lerman and Galstyan [8] create mathematical models, through differential equations, of the collective behaviour of simple multi-agent systems, such as social insects. Their approach differs from ours in that they aim to build a generic model of an agent, based on observations, and then devise a mathematical model of it. Martinez et al. [10] investigated the use of rule-learning algorithms to predict group behaviours in

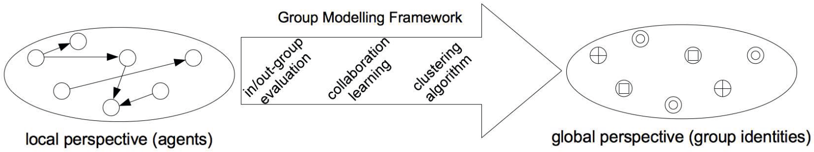

Group Modelling Framework
in/out-group evaluation
collaboration learning
clustering algorithm
local perspective (agents)
global perspective (group identities)

Fig. 1. Overview of the group modelling framework. The figure depicts an example society (ellipse) of 7 agents and their interactions (circles and arrows respectively) and the presence of three group identities (i.e. cross, circle and square) the 7 agents are labelled with.

artificial societies. Their method is based on historical data and even though we share a common goal (i.e. modelling of group dynamics), that study does not aim to model group identities in real time.

For what it concerns virtual environments, instead, Vogiazou and Eisenstadt [16] studied the influence of communication to the emergence of group behaviours; their qualitative findings, however, were not backed-up empirically. Ducheneaut et al. [4] analysed gameplay data from the popular massively multiplayer online game *World of Warcraft* (Blizzard Entertainment, 2004) in order to understand the relationships occurring between players grouping together in order to solve common guilds. Their work does not focus on the creation of spontaneous groups but rather examines pre-determined groups. El Nasr et al. [12] defined metrics of collaboration for collaborative games. Their approach involves expert knowledge and is based on well established console games such as *Guitar Hero* (Activision, 2006). Once again, the study does not focus on the formation of groups, although it provides metrics which could be adopted as interaction classifiers in our framework.

3 Social Ultimatum Bargain Game and Agent Policies

The test-bed game used for the interaction of agents involves a *scenario* $\mathcal{S}$, composed of T episodes, in which a population $\mathcal{P}$ of n agents $a_1, \ldots a_n$ interact with each other by means of the social ultimatum bargain game [1]. At each episode, the population $\mathcal{P}$ is randomly divided into m partitions of equal size P (depending on the n and m values, the partition sizes might differ by 1 agent). A fixed amount of money, M, is assigned to each partition. Then, all agents will play the social ultimatum bargain game [1], hereby described for a single partition size, P. An agent a_i is randomly selected to become the *provider agent* and the remaining $P-1$ agents become the *receiver agents*. The provider agent a_i will bargain an equal *endowment*, $e = M/(P-1)$, with all receiver agents. In particular, the provider agent will propose to each receiver agent a_j an *offer* $0 \leq o_{i,j} \leq e$; as a *response* to the offer, $r_{j,i}$, the a_j agent may *accept* or *refuse* the offer. At the end of each episode t, the gain for a_i is the sum of all accepted offers, $g_{i,t} = \sum_{j=1}^{P-1}(e - o_{i,j})$, whilst the gain for each receiver agent a_j is either $o_{i,j}$ (if $r_{j,i}$ is *accept*) or 0 (if $r_{j,i}$ is *refuse*).

Although in this study we assume consolidated groups and therefore no agent adaptation, we still aim to study believable agents with social preferences simulating human-like behaviours. The agents' social preferences are driven by the findings of Marzo et al. [11]: (1) the provider agent should always aim to maximise its own gain; (2) the provider's offer should tend to be more generous when it is interacting with a friend; (3) there should not be any apparent influence of the relationship types on the response of the receiver agents. The agents we have implemented follow the human-like, adaptive agent model of Chang et al. [1] for the social ultimatum game. Furthermore, in order to fulfil Marzo et al.'s second finding, three relationship types were defined, namely friendship, acquaintanceship and strangeness. On average, the highest offers will occur between friendly agents; moderate offers are expected in an acquaintanceship between two agents; and the lowest offers are expected between stranger agents.

4 Method: Group Identity Detection

For our group modelling (GM) framework (see Figure 1), we assume that the agent policies have the Markov property — i.e. the agents perform actions based only on their current internal *state* (e.g. cultural biases, aggressiveness, relationship values). As a consequence, we assume that the whole society has the Markov property. With respect to the canonical reinforcement learning terminology, our GM framework aims to learn the collaboration-function values, $\mathcal{C}^{\triangle}$, of the whole society $\mathcal{P}$; the society's state is represented by the pairs $(a_i, a_j)\,, \forall i, j \in [1, n]$ and the actions are represented by the pairs $(o_{i,j}, r_{j,i})\,\forall i, j \in [1, n]$.

4.1 Phase 1: *In/out-group* Interaction Classification

Following the findings of Dawes and Messick [3] with respect to the social ultimatum bargain game presented in Section 3, each interaction feeds an Interaction Classifier, $\mathcal{I}$, which returns either an *in-group* or an *out-group* value for the interaction $(o_{i,j}, r_{j,i})$. In this study, we consider five possible classifiers. The first two are deterministic:

- $\mathcal{I}_{RB}$ is the *receiver behaviour* classifier: *in-group* is returned if $r_{j,i}$ is *accept*, *out-group* is returned if $r_{j,i}$ is $refuse$.
- $\mathcal{I}_{PB}$ is the *provider behaviour* classifier: it maintains the average offer proposed by a_i throughout the whole experiment, $\hat{o}_i(t)$. This classifier returns *in-group* if $o_{i,j}(t) > \hat{o}_i(t-1)$ at episode t whereas it returns *out-group* if $o_{i,j}(t) < \hat{o}_i(t-1)$. If $o_{i,j} = \hat{o}_{i,t-1}$, *in-group* or *out-group* are picked randomly following a binomial distribution.

The remaining three classifiers are non-deterministic and based on all the $P-1$ offers, $O_{i,} = \{o_{i,1}, \ldots o_{i,i-1}, o_{i,i+1}, \ldots o_{i,P}\}$ made by the provider agent to all receiver agents in the partition. These interaction classifiers return *in-group* with probability $p(a_j \textit{ in-group } a_i)$ and *out-group* with probability $1-p(a_j \textit{ in-group } a_i)$ for each receiver agent a_j. The interaction classifiers and their corresponding probability functions are as follows:

- $\mathcal{I}_H$ is the *entropy* classifier based on the probability function p_H which is defined as the normalised Shannon's entropy and represents uniformity of offers [18]:

$$p_H(a_j \ \textit{in-group} \ a_i) = -\frac{1}{log(P-1)} \sum_{j=1}^{P-1} \frac{o_{i,j}}{O_i} log\left(\frac{o_{i,j}}{O_i}\right) \tag{1}$$

- $\mathcal{I}_{AP}$ is the *agent preference* classifier based on the probability function $p_{AP}(a_j \ \textit{in-group} \ a_i)$ which is calculated as the following min-max normalisation:

$$p_{AP}(a_j \ \textit{in-group} \ a_i) = \frac{o_{i,j} - \min(O_i)}{\max(O_i) - \min(O_i)} \tag{2}$$

- $\mathcal{I}_{HAP}$ is the *entropy-agent preference* classifier based on the probability function p_{HAP} which is calculated as the following linear combination:

$$p_{HAP}(a_j \ \textit{in-group} \ a_i) = p_{AP}(a_j \ \textit{in-group} \ a_i) \cdot p_H(a_j \ \textit{in-group} \ a_i) \tag{3}$$

The probability function p_H was selected as it represents the *fairness of the offers* O_i made — on average — by the provider agent a_i in its partition. In an unfair offer distribution situation (i.e. there is a disparity of treatment made by the provider agent towards the receiver agents — the provider is favouring a subset of the agents in P), the resulting p_H is low, meaning that — on average — the interactions will be likely to be classified as *out-group*. On the other hand, in a fair distribution situation (i.e. there is equality of treatment made by the provider agent towards the receiver agents — the provider is not favouring any subset of agents in P), the resulting p_H is high, meaning that — on average — the interactions will be likely to be classified as *in-group*. Further discussion about the selection of these probability functions is provided in Section 6.

4.2 Phase 2: Collaboration Learning

The second phase of our GM framework follows two steps. First, it interprets the *in/out-group* values of the interaction classifier as an immediate reward function that is used to approximate cooperation values in the society. The result is a cooperation matrix $\mathcal{C}$, of size $n \times n$, in which the value $\mathcal{C}_{i,j}(t)$ represents the up-to-date cooperation value between agents a_i (provider) and a_j (receiver) at the t-th episode. In this paper we consider the *constant-α Monte Carlo* update rule for non-stationary environments [15]:

$$\mathcal{C}_{i,j}(t) = \mathcal{C}_{i,j}(t-1) + \alpha\left[\mathcal{I}(t) - \mathcal{C}_{i,j}(t-1)\right] \tag{4}$$

where $\mathcal{I}(t)$ is the immediate reward (interaction classifier), for the t-th interaction between a_i and a_j and α is a constant step-size parameter. The update rule just defined encompasses well the notion of past interactions [7] which influence agent behaviours, it allows the estimation of up-to-date levels of cooperation between agents and, finally, it maintains a distinction about the directionality of the interactions (i.e. from the provider agent to the receiver agent).

The second step of this phase regards the retrieval of collaboration values from the learnt cooperation matrix; this is achieved by considering collaboration as reciprocal altruism [7]. The result is a (triangular) collaboration matrix, $\mathcal{C}^{\triangle}$, in which the collaboration value, at episode t, between a_i and a_j is calculated as $\mathcal{C}^{\triangle}_{i,j}(t) = \frac{1}{2}\left[\mathcal{C}_{i,j}(t) + \mathcal{C}_{j,i}(t)\right]$.

4.3 Phase 3: Clustering for Group Identification

The final phase of our GM framework interprets $\mathcal{C}^{\triangle}$ as a dissimilarity/distance matrix and uses it in order to assign group identities to agents. The clustering method we adopt in this paper is the agglomerative complete-link algorithm with the elbow-rule as stopping criterion [17].

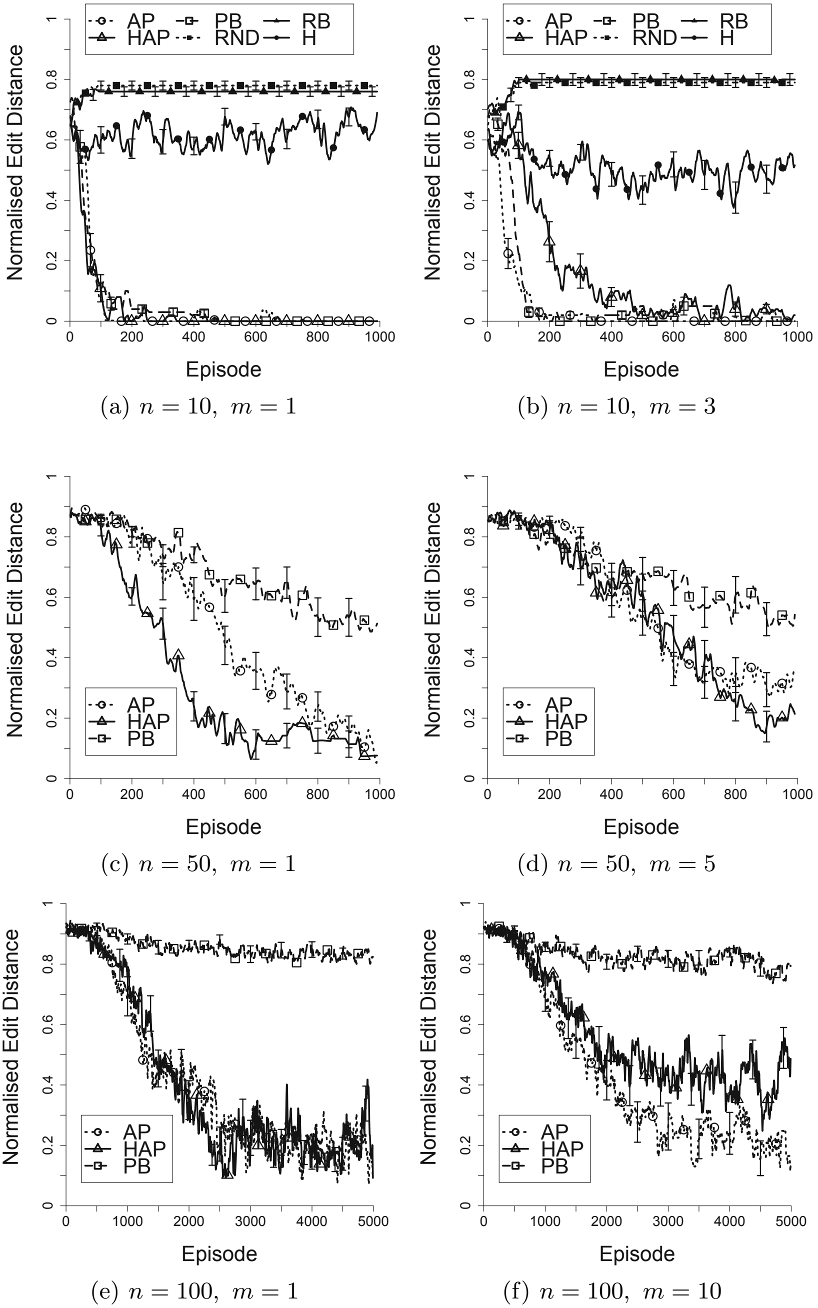
5 Results

We have tested the performance of our GM framework against three different society sizes ($n = 10, n = 50$ and 100), both in absence ($m = 1$) and in presence of agent interaction locality constrained by a number of population partitions ($m = 3, m = 5$ and $m = 10$ for each society size, respectively). Algorithm performance is measured as the ability of our GM framework to detect consolidated predefined group identities. In addition, we test the GM framework across all five interaction classifiers (see Section 4.1) and compare the results.

At the beginning of each experimental setup, *true* group identities are randomly generated within the population satisfying a number of constraints: (1) there must exist at least 2 group identities; (2) each group identity must have at least size of 2; and (3) agents belonging to the same group identity are friends whereas agents belonging to different group identities have a 50% chance to be either acquaintances or strangers. At the end of each episode, two integer vectors of length n — one for the true and one for the GM's group identities — are retrieved and the performance (error) of the GM framework is calculated through the normalised edit distance [9] between the two vectors.

In order to grasp the generic performance of the GM framework and reduce the non-deterministic nature of the interactions, we repeat the execution of each experimental setup 10 times. For each setup we set α to 0.1. The number of episodes T are set to 1000 when $n = 10$ and $n = 50$, whilst $T = 5000$ when $n = 100$. Furthermore, to smooth the noise of random fluctuations, Figure 2 depicts the performance of our GM framework based on a moving average window of 10 episodes. For the $n = 10$ case (see Figures 2(a) and 2(b)), the interaction classifiers are tested against a baseline random interaction classifier, $\mathcal{I}_{RND}$, which returns either *in-group* or *out-group* with probability 0.5. Note that $\mathcal{I}_{RND}$, $\mathcal{I}_{RB}$ and $\mathcal{I}_{H}$ depict similar performance in all agent scenarios but are not represented in the figures of the $n = 50$ (Figures 2(c) and 2(d)) and the $n = 100$ (Figures 2(e) and 2(f)) scenarios for illustration purposes.

Results show that when $\mathcal{I}_{AP}$ and $\mathcal{I}_{HAP}$ are used, our GM framework manages to detect the true group structures in all 6 scenarios. All probabilistic interaction



(a) $n = 10,\ m = 1$

(b) $n = 10,\ m = 3$

(c) $n = 50,\ m = 1$

(d) $n = 50,\ m = 5$

(e) $n = 100,\ m = 1$

(f) $n = 100,\ m = 10$

Fig. 2. Performance of the GM Framework with respect to population size (n) and number of partitions (m) based on different reward functions ($\mathcal{I}$): agent preference (AP), entropy-agent preference (HAP), provider behaviour (PB), random (RND), responder behaviour (RB), entropy (H).

classifiers, excluding $\mathcal{I}_H$, are robust with respect to n and m. The $\mathcal{I}_H$ interaction classifier does not allow to detect the true group identities because the $p_H(a_j \textit{ in-group } a_i)$ values are equal for all receivers agents in the partition (see Equation 1): even if the normalised entropy detects fair/unfair scenarios, the classifier is not able to detect the agents which receive a more altruistic offer. $\mathcal{I}_{AP}$ and $\mathcal{I}_{HAP}$ yield similar results (i.e. no significant difference) in the first 5 scenarios (see Figures 2(a)...2(d)). For the $n = 100, m = 10$ scenario (see Figure 2(f)), instead, $\mathcal{I}_{AP}$ outperforms $\mathcal{I}_{HAP}$; a t-test on this scenario shows a significant difference between the two (t-stat = 4.0705; p-value $<$ 0.0001). Comparing the two with respect to their convergence speed, $\mathcal{I}_{AP}$ appears to converge to zero error quicker than $\mathcal{I}_{HAP}$ in the $n = 10, m = 3$ case (see Figure 2(b)) whilst the opposite result is obtained when $n = 50, m = 1$ (see Figure 2(c)). Despite the difference in performance in the $n = 100, m = 10$ scenario, an explanation can be hardly found by focusing solely on the interaction classifiers. Since $p_{HAP} \leq p_{AP}$ for all interactions (p_{HAP} is a linear combination of $p_{AP} \leq 1$ and $p_H \leq 1$), this implies that p_{HAP} is more robust with larger (fairly diluted) partitions, and therefore could explain why $\mathcal{I}_{HAP}$ outperforms $\mathcal{I}_{AP}$ in the $n = 50, m = 1$ setup (see Figure 2(c)). However, the same effect does not occur that clearly in the $n = 100, m = 1$ case as the two interaction classifiers perform almost equally. Moreover, in small partition size setups (P is 10 in the $n = 50, m = 5$ and the $n = 100, m = 10$ setups) we observe the inverse picture. The differences in performance might be explained by the size of the cooperation martix, $\mathcal{C}$ (see Section 4.2), and the number of updates it requires in order to converge. More importantly, the elbow rule criterion appears to be sensitive with respect to small changes in the $\mathcal{C}$ values (i.e. the number of groups detected, between two consecutive episodes, was often fluctuating between n and $g \ll n$), as we have observed in many of the experimental runs we have conducted.

The $\mathcal{I}_{PB}$ interaction classifier manages to detect the exact group identities on small societies but it performs poorly on larger societies, not only with respect to the final average errors obtained but also with respect to the speed of convergence. Finally, $\mathcal{I}_{RB}$ performs as poorly as the random baseline interaction classifier ($\mathcal{I}_{RND}$), which is evidence that the agent policies of our society reflect well the social preference findings of Marzo et al. [11].

6 Discussion and Conclusions

The key findings of this paper suggest that the probabilistic agent-to-agents interaction classifiers proposed (based on both indicators of preference for particular agents and the general fairness of offer distribution) allow the detection of group identities under a group modelling (GM) framework which is robust independently of the size of an artificial society of human-like, believable, agents and the locality of interactions.

The GM approach proposed aims to learn a collaboration matrix which grows quadratically with respect to the society size, $O(n^2)$. This key limitation is already shown by the GM performance decrease observed in larger society sizes;

one would, therefore, expect that the GM framework would not be able to scale to very large societies composed by thousands of agents. A solution could be the reduction of the size of the collaboration matrix. For instance, one could use the group structures inferred by the framework up to the previous episode in a feedback loop and learn collaboration values among groups rather than agents. In this way, the new group-based cooperation matrix would have the size of $O(g \leq n)$, where g is the number of groups detected. On that basis, ongoing research on the definition of a group-based metric of fairness is being conducted. As fairness of offer distribution is a subjective and ambiguous term we designed several ad-hoc metrics of fairness based on key and generic elements of fair offers, such as the amount of the offer and the need of the receivers. We then tested those against human notions of fairness, by running a crowdsourcing experiment in which participants were asked to compare levels of fairness of dissimilar offer distribution scenarios in a resource management game environment [5]. Preliminary results suggest that some of our fairness metrics, such as the entropy of distributed offers (as used in this study) appears to be highly correlated with the human notion of fairness and suggests that such a metric is robust for group identification (as seen in the results of this paper).

The complete-link clustering algorithm used is conceptually sound: grouping agents by maximum $\mathcal{C}^{\triangle}$ values and calculating between-group distances by minimum $\mathcal{C}^{\triangle}$ values aligns well with the *in/out-group* concepts [3]. However, we believe that the use of more advanced validity methods rather than the elbow-rule will most likely lead to faster algorithm convergence.

As a direct consequence of the findings of our study, the GM framework can easily be extended to model group identities in human-based collaborative (virtual) environments, such as multiplayer games; moreover, due to its effectiveness to model identities in small-to-mid societies, the framework is ideal for collaborative serious games targeting small communities such as classrooms [20]. Furthermore, the interaction classifier is the only component of the framework which requires context-based information (i.e. the offers in our social ultimatum bargain game scenario). Nonetheless, by maintaining the concepts of general fairness and agent favourability (similarly to $\mathcal{I}_{AP}$ and $\mathcal{I}_{HAP}$), the framework can be extended to scenarios which are not merely based on the exchange of resources.

Future work will consider three main directions: we will consider more complex societies of agents which not only showcase adaptive behaviours but are also driven by their own emotions and the emotional states of others [14]; we will augment our GM framework by modelling group dynamics and group internal structures (e.g. agent leaderships); finally, we will approach the subsequent phase of providing influencing methods which would affect the global-to-local indirect influence in order to reach global-level goals, such as maximisation of collaboration for the agent society as a whole.

Acknowledgments. This work has been supported, in part, by the FP7 ICT project SIREN (project no: 258453).

References

1. Chang, Y.H., Maheswaran, R., Levinboim, T., Rajan, V.: Learning and Evaluating Human-Like NPC Behaviors in Dynamic Games. In: Proceedings of AIIDE (2011)
2. Cheong, Y.G., Khaled, R., Grappiolo, C., Campos, J., Martinho, C., Ingram, G.P.D., Paiva, A., Yannakakis, G.N.: A Computational Approach Towards Conflict Resolution for Serious Games. In: Proceedings of FDG, pp. 15–22 (2010)
3. Dawes, R.M., Messick, D.M.: Social Dilemmas. International Journal of Psychology 2(35), 111–116 (2000)
4. Ducheneaut, N., Yee, N., Nickell, E., Moore, R.J.: "Alone Together?": Exploring the Social Dynamics of Massively Multiplayer Online Games. In: Proceedings of CHI, pp. 407–416 (2006)
5. Grappiolo, C., Cheong, Y.G., Togelius, J., Khaled, R., Yannakakis, G.N.: Towards Player Adaptivity in a Serious Game for Conflict Resolution. In: Proceedings of VS Games, pp. 192–198 (2011)
6. Hammond, R.A., Axelrod, R.: The Evolution of Ethnocentrism. Journal of Conflict Resolution 50(6), 926–936 (2006)
7. Lansing, S.J.: Complex Adaptive Systems. Annual Review of Anthropology 32, 183–204 (2003)
8. Lerman, K., Galstyan, A.: Automatically Modeling Group Behavior of Simple Agents. In: Proceedings of AAMAS (2004)
9. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. Soviet Physics Doklady 10(8), 707–710 (1966)
10. Martinez, V., Simari, G.I., Sliva, A., Subrahmanian, V.: CONVEX: Similarity-Based Algorithms for Forecasting Group Behavior. IEEE Intelligent Systems 23, 51–57 (2008)
11. Marzo, F., Grosz, B.J., Pfeffer, A.: Social Preferences in Relational Contexts. In: Proceedings of Collective Intentionality (2005)
12. Nasr, M.S.E., Aghabeigi, B., Milam, D., Erfani, M., Lameman, B., Maygoli, H., Mah, S.: Understanding and Evaluating Cooperative Games. In: Proceedings of CHI, pp. 253–262 (2010)
13. Nowak, M.A., Tarnita, C.E., Antal, T.: Evolutionary Dynamics in Structured Populations. Philosophical Transactions of the Royal Society 365(1537), 19–30 (2010)
14. Pereira, G., Dimas, J., Prada, R., Santos, P.A., Paiva, A.: A Generic Emotional Contagion Computational Model. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, pp. 256–266. Springer, Heidelberg (2011)
15. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. Adaptive Computation and Machine Learning. The MIT Press (1998)
16. Vogiazou, Y., Eisenstadt, M.: Designing Multiplayer Games to Facilitate Emergent Social Behaviours Online. Interactive Technology and Smart Education 2, 113–126 (2005)
17. Webb, A.R.: Statistical Pattern Recognition. John Wiley & Sons (2002)
18. Yannakakis, G.N., Hallam, J.: Towards Optimizing Entertainment in Computer Games. Applied Artificial Intelligence 21(10), 933–971 (2007)
19. Yannakakis, G.N., Togelius, J.: Experience-Driven Procedural Content Generation. IEEE Transactions on Affective Computing 2, 147–161 (2011)
20. Yannakakis, G.N., Togelius, J., Khaled, R., Jhala, A., Karpouzis, K., Paiva, A., Vasalou, A.: Siren: Towards Adaptive Serious Games for Teaching Conflict Resolution. In: Proceedings of ECGBL, pp. 412–417 (2010)

Cost, Precision, and Task Structure in Aggression-Based Arbitration for Minimalist Robot Cooperation

Tanushree Mitra and Dylan A. Shell

Department of Computer Science and Engineering
Texas A&M University

Abstract. This paper reexamines a multi-robot transportation task, introduced and studied by Vaughan and his collaborators, in which constrained space induces inter-agent interference. Previous research demonstrated the effectiveness of an arbitration mechanism inspired by biological signaling where the level of aggression displayed by each agent effectively prioritizes the limited resources. This paper shows that apart from determining the correct fitness of an individual several other factors, such as signaling cost, precision of the outcome and properties of the resource and task are key to determine an effective arbitration technique. Based on these factors we present a taxonomy of the arbitration mechanisms.

The large signalling costs incurred by our simple robots using minimal set of sensors permit us to identify scenarios in which a dominance hierarchy outperforms, not only to no arbitration, but also aggression-based mechanisms. We identify how memory of past interactions can be used to the advantage of an agent, albeit with a trade-off between cost and outcome accuracy. We also show that the importance of a particular aggressive interaction to long-term task performance is not trivial to determine and depends on the task structure. Results help us identify instances where agents may manipulate interactions to alter the frequency and duration of aggressive encounters, affecting the overall task performance.

1 Introduction

Spatial interference is a common and important phenomenon in navigation tasks involving multiple robots; it is a particular instance of the general problem of resource competition amongst agents attempting to achieve their own ends while interacting with others. When autonomous agents have an incentive to cooperate, a worthwhile question is how best to mitigate the negative effects of resource contention. Motivated by the methods which animals employ to contest resources, Vaughan and his collaborators (*cf.* [5], [1], and [6]) have shown how displays of stylized aggression can effectively resolve resource conflicts in a multi-robot transportation task. That line of work has produced increasingly effective methods for assessing the level of aggression that an individual agent should exhibit in order to prioritize the limited resource effectively.

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 431–441, 2012.

This paper shows that determining the correct fitness of an individual at a particular time is only one of several aspects of effective conflict resolution. An important consideration is the cost of the aggressive signaling. In fact, the analogous biological mechanism is directly concerned with the interplay of signal precision and cost: aggressive displays allow animal to assess the strength of their resource competitors before they decide to engage in a costly fight [2]. Animals, after all, organize themselves into a dominance hierarchy which they can use to resolve future resource competition [4] in an inexpensive albeit static way.

In this paper, we examine the multi-robot transportation task domain that Vaughan and his collaborators have studied. Specifically, we study a two robot interference scenario, the goal of which is to cooperatively perform the maximum number of collective transportation tasks in a given time. We present a taxonomy of arbitration mechanisms for two-agent spatial interference, including a characterization of conflict resolution models, introduce a notion of outcome accuracy and explicitly consider interaction cost. Results from physical robot experiments and data-driven simulations led to following contributions:–

1. We show that there exist similar problem instances in which either dynamic aggression or a static dominance hierarchy are advantageous.
2. We also demonstrate how memory of past interactions with respect to the task structure and *properties of the resource* can result in improved future task performance.
3. The paper shows that varying the *properties of assigned task*, the frequency of spatial interference and the cost incurred in its resolution varies significantly.
4. A new "minimalist" resource arbitration method is introduced which produces dynamic outcomes—albeit with comparatively high costs—suitable for simpler robots (with fewer sensors) than heretofore known.
5. We identify how agents may manipulate interactions to alter the frequency and duration of aggressive encounters, affecting the overall task performance in repetitive tasks.

We begin by giving a brief overview of previous research of interference in multi-robot systems. Next, in Section 3 we propose a taxonomy of arbitration mechanisms for two-agent spatial interference. In Section 4, we provide a comparative study of the various arbitration models. Results of the study are based on physical robot experiments. Finally in Section 5, we designed a custom simulator based on our physical robot data from previous section and anlyzed interference under systemic variation of environment.

2 Related Work

Goldberg and Matarić [3] have suggested using interference as a tool for evaluating multi-robot controllers, *viz.* identifying trade-off between performance time and interference. Vaughan *et al.* [5] compared a dominance hierarchy to the aggression-based strategy in a multi-robot transportation task in a simulated environment. Our data below show that one strategy can be preferable to the other, but exactly which depends on the shared resource and also on the individual task dynamics. Brown *et al.* [1] introduced the concept of rational aggression where the level of aggression is determined by the investment made by Zuluaga and Vaughan [6] further improved on Brown *et al.*'s performance by basing the level of aggression on the investment in the shared resource. Details on the relevant strategies compared in this work are below.

Brown *et al.* [1] introduced the concept of rational aggression where the level of aggression is determined by the investment made by Zuluaga and Vaughan [6] further improved on Brown *et al.*'s performance by basing the level of aggression on the investment in the shared resource. Details on the relevant strategies compared in this work are below.

This paper uses the same controlled scenario (depicted in Fig. 1) as this previous line of research. Two robots perform repetitive transportation tasks by moving around cycles in the environment (shown as blue and green walls they must follow). Competition for space occurs at a narrow shared region, the girdle, where only one robot traverse at a time; different arbitration models determine who gives way.

Additionally, in [3] the authors suggested that changing the environment could play a role in altering interference properties. This manifest itself in [1], where changes in the (simulated) environment produced large standard deviations in their results. Part of our work is an attempt to analyze and mitigate interference under systematic variation of the environment.

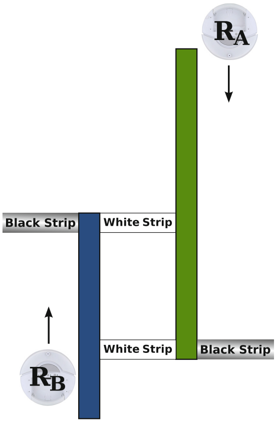


Fig. 1. The task involves robots navigating their respective regions (around blue and green walls, respectively) but also having to resolve a conflict in the shared space (between the "white strips").

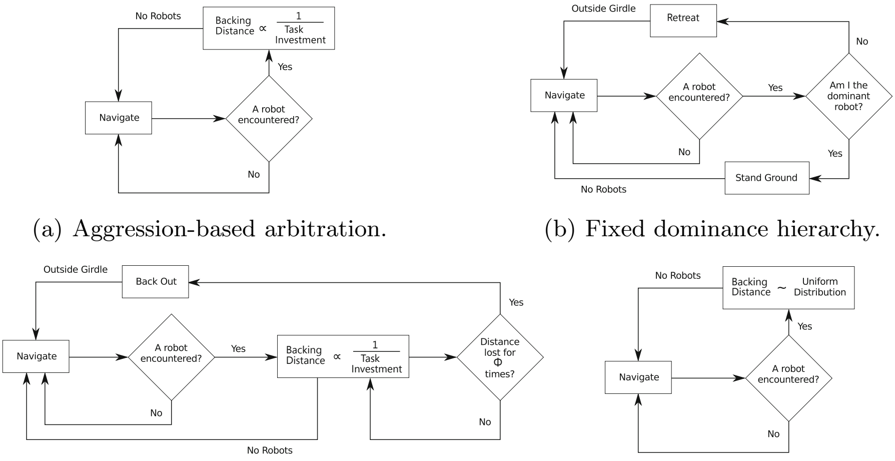
2.1 Arbitration Models

A resource arbitration model determines which of the two robots should have access to the shared resource. In aggression based models each robot determines a quantity, their level of aggression, and engages in a dynamic behavior involving giving or taking way in the shared space.

Vaughan's random aggression: Each agent picks a random aggression at each encounter, resulting in a random outcome; *i.e.,* the resource is gained by a random agent.

Vaughan's personal space method: The level of aggression is determined by the amount of free space visible behind the robot in the event of interference.

Rational aggression based on local task investment: This is modeled after [6], aggressive interaction based on local task investment, where the robot "displays its aggression" by moving backward a distance inversely proportional to the distance traveled so far in the constrained region, and then moving forward until it bumps again. The robot's controller is shown in Figure 2(a).



(a) Aggression-based arbitration.

(b) Fixed dominance hierarchy.

(c) Cutting one's losses.

(d) Random walk arbitration.

Fig. 2. Details for four different arbitration mechanisms

Linear dominance hierarchy: A fixed dominance is assigned to each robot before they start their spatial navigation and each one follows this at every encounter to determine who gets right-of-way. Figure 2(b) shows this.

Cutting your Losses: Some memory of local task performance is added to the rational aggression method. When a robot meets an opponent inside the girdle, it displays its level of aggression for at most ϕ attempts. At the same time it measures and remembers the cost it incurs in this display, by measuring the lose or gain in the shared space distance from the time it starts its aggressive display. Figure 2(c) gives this mechanism.

Random walk: As soon as a robot encounters an opponent, it backs a random distance. It then moves forward and if the opponent is still in the girdle, it again moves back by a random distance. The opponent also follows the same protocol. Eventually one of the robots is pushed out of the girdle. Figure 2(d) illustrates this mechanism.

3 Taxonomy of Spatial Conflict Resolution Models

We propose a taxonomy of conflict resolution models with the following axes:–

- Dynamic vs. static: An arbitration method is static if and only if it does not employ information about a particular encounter to resolve that conflict.
- Deterministic (DET) vs. Probabilistic (PROB): A method is deterministic if and only if, given the same scenario, the resource is always awarded to the same agent.
- High (HOA) vs. Low outcome accuracy (LOA): The former has higher probability of selecting the rational winner. The robot with the greatest local investment should gain the resource in rational interactions.
- Costly (HIGHCOST) vs. cheap (LOWCOST): Time, energy and other resources may be involved in an arbitration mechanism. Their utility depends on the comparative saving and/or trade-off of these costs.

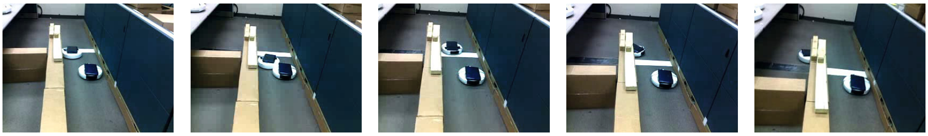


Fig. 3. An example of a robot giving way. After bumping each other, both robots move back and one exits the girdle to make room.

Different arbitration models can be denoted by a quadruplet:

Model	Classification			
Linear dominance hierarchy	STATIC	DET	LOA	LOWCOST
Vaughan's random aggression	STATIC	PROB	LOA	LOWCOST
Random walk	DYNAMIC	PROB	LOA	LOWCOST
Vaughan's Personal space	DYNAMIC	DET	HOA	HIGHCOST
Rational aggression	DYNAMIC	PROB	HOA	HIGHCOST
Cutting your losses	DYNAMIC	PROB	HOA	LOWCOST

4 Comparative Study

4.1 Implementation Details

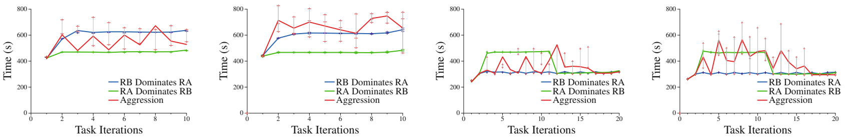
We imposed spatial interference on physical robots by making them navigate through an environment as shown in Figure 1. Interference occurs when two iRobot *Creates*'s R_A and R_B, 33cm in diameter attempt to cross a girdle ∼53cm wide from opposite directions as shown by the arrows in Figure 1. R_B's transportation task length is almost half that of R_A. R_A does 10 traversals, while R_B covers 20. We assigned these numbers so as to avoid situations where the robot performing the shorter task finishes all its trips while the other one keeps traversing an encounter-free region.

4.2 Aggressive Interaction and Linear Dominance

Both these models were executed in environments with different girdle lengths. The aim is to assess the role the shared resource plays on arbitration outcomes.

Varying Girdle Length: The utility of aggressive interaction is reduced when both robots have large, almost equal aggression levels, a phenomenon which happens when encounters are at the center of a large enough girdle. This can be observed in Figure 4, as the aggression strategy performs increasingly poorly with increasing girdle length. A dominance hierarchy, despite it not necessarily resolving the conflict toward the agent with the greater investment, proves to be a better arbitration method in such cases.

However for encounters at the ends of the girdle, the short aggressive interactions coupled with the ability to produce a rational winner makes aggression



(a) R$_A$,GL=3. 35 (b) R$_A$,GL=4.7 (c) R$_B$,GL=3.35 (d) R$_B$,GL=4.7

Fig. 4. Average task times of R$_A$ and R$_B$ with varying girdle lengths (GL) in meters, fixed task ratio 25:38. (Results are from three experiments averaged for each of the three strategies, for each of the three cases. We show 2 cases for space constraints.)

based arbitration beneficial over dominance. Vaughan *et al.* [5] provide an instance where choice of aggression level proves no better than random selection of aggression (Vaughan's random aggression). In fact, the outcome of such a random mechanism is an average drawn from the outcomes of following either extremes of the linear dominance hierarchy. In the data above the advantage of such a mechanism can be seen.

An important question is "how precisely can the outcome of the arbitration be predicted given the initial position of encounter?" When the robots have approximately equal local task investment, as in the setup above, the noise in the robot's interactions breaks the symmetry. In Figure 5 the mixed red and blue region near the center of the girdle (girdle proportion ∼0.5) shows that there are situations when R$_A$'s local investment is less than R$_B$, but R$_A$ wins or vice-versa. These are the few instances when the outcome of the aggressive encounter is neither particularly accurate nor rational and there is high cost involved, decreasing their utility in such situations. Moreover, in instances where encounters occur at girdle ends, the inaccuracy of dominance hurts only when the less dominant robot has higher local investment and still retreats. If the task ratios are appropriate, then such scenarios may occur only rarely, making dominance arbitration the superior arbitration model for such environments.

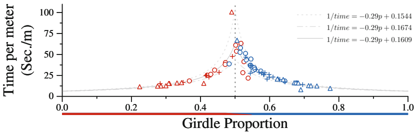


Fig. 5. Time to resolve an aggressive interaction of two physically grounded robots. The horizontal bar shows the robot which gets right of way when the point of encounter inside the girdle (length normalized) varies, red being robot R$_A$ and blue R$_B$

Varying Task Ratio: We examined how the properties of the task assigned to each agent influence aggressive encounters. This factor dictates the time when a robot starts its journey inside the girdle relative to the other and, thus, the initial position where they end up meeting. There is also the chance that they do not meet at all. The variation of the duration of aggressive interactions as shown in Figure 6 indicates the importance of task structure in aggression based arbitration. Task structure is reexamined through exhaustive simulation below.

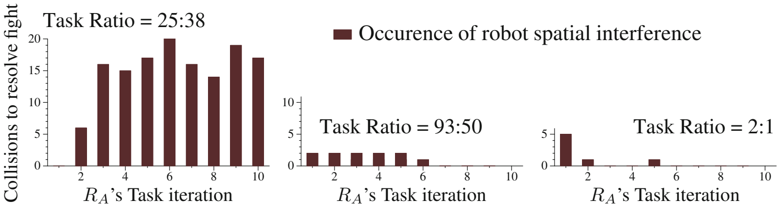
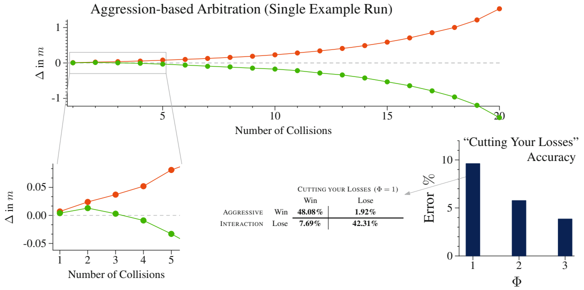


Fig. 6. The duration of aggressive interaction of two physically grounded robots changes as we vary task ratio, suggesting the importance of task structure in spatial interference. These experiments were performed on physical robots. We reexamine effects of task structure in spatial interference through exhaustive simulation in the next section.



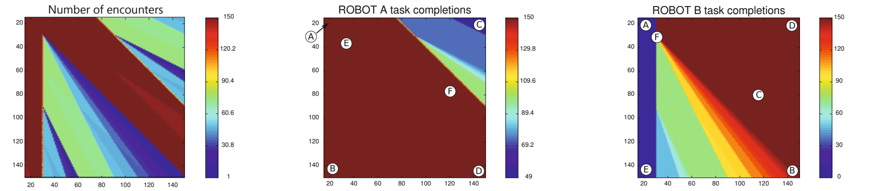
Cutting your Losses (Φ = 1)

		Win	Lose
Aggressive Interaction	Win	**48.08%**	**1.92%**
	Lose	**7.69%**	**42.31%**

Fig. 7. The intuition behind the "cutting your losses" strategy is illustrated via an example of the aggression-based interaction. The sign of the single-time gain (denoted Δ in the graphs) indicates a likely win or loss. Waiting longer before measuring $sgn(\Delta)$ reduces the estimate error due to the "escalation" dynamics. The results are from physical robot experiments.

Cutting your Losses: The utility of aggressive encounters can be improved by adding memory of recent performance. The robot measures and remembers the loss or gain in the shared space distance from the time it starts its aggressive display. If it repetitively loses distance, then it is unlikely to win the whole interaction. In such a situation it is beneficial to retreat. The greater the number of confirmations about the gain/loss in distance, the more accurate its decision. Figure 7 shows this decrease in error with an increase in the number of confirmations. The tradeoff here is whether to take an early

Random Walk: The attraction of random walk arbitration is its minimalism compared to other arbitration methods: robots do not need to sense or estimate their positions, since the position the bump takes place implicitly encodes the dynamic variable. It is perhaps somewhat surprising that a dynamic arbitration mechanism is possible despite neither of the robots having the means to record their investment in the task.



(a) Number of encounters (b) Laps completed by R$_A$ (c) Laps completed by R$_B$

Fig. 8. Dominance for girdle length = 30m, R$_A$ is dominator. The x-axis shows R$_A$'s task length, y-axis that of R$_B$. (a) Color bars shows the relative number of encounters,(b), (c) Color bar shows the relative number of laps finished when at least one of the robots completes 150 laps. Points A to F are detailed in Figure 9.

5 Task Ratio—A Macroscopic Study

Designed a custom simulator that uses physical robot data to model dominance and aggressive robot interactions for different girdle lengths across a range of task ratios. We ran the simulator, varying girdle lengths (GL) from 10m to 150m and, for each girdle length, task lengths varied from 15m to 150m. The results represent complementary foci: either minimizing absolute signal cost via a static arbitration, or incurring whatever cost to ensure a dynamic arbitration.

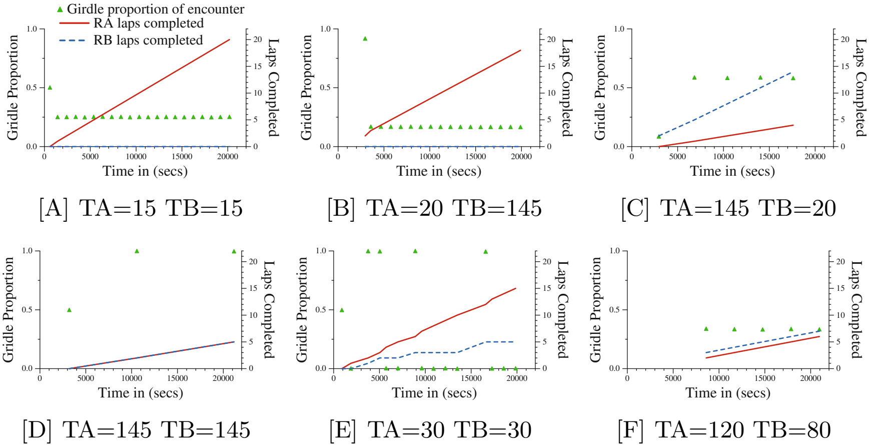
Dominance Model: Figure 8 shows the performance in a girdle length of 30m when R$_A$ is the dominator. Interesting regions from these plots were selected to investigate the interaction dynamics for the first 20,000 seconds (long enough to show long-term behavior). These are marked with A—F in Figure 9, and described in detail in that caption.

Aggression Model: The model was run to compare with the dominance method.

Collective best performance across varying girdle length: Certain combinations of task lengths takes significantly longer to complete 100 tasks (Figure omitted due to space restrictions). One might think that this is due to severe interference for these task ratios. However, plotting the interference count we find that this is not always true. The following considers one fixed girdle length.

Collective vs. individual best performance for fixed girdle length: There are regions of high interference corresponding to regions of low task completion time (Figure omitted due to space restrictions). These are the instances where the robots met often but engaged in less costly aggressive interactions. On the other hand, there also exist regions of low interference but with high task times. These regions involve high cost aggressive interactions.

We further investigate as to what happens for certain combinations of task lengths, similar to what we did for dominance model. Positions of first encounter for every task iteration completed in the first 20,000s are shown (Figure 10).



[A] TA=15 TB=15 [B] TA=20 TB=145 [C] TA=145 TB=20

[D] TA=145 TB=145 [E] TA=30 TB=30 [F] TA=120 TB=80

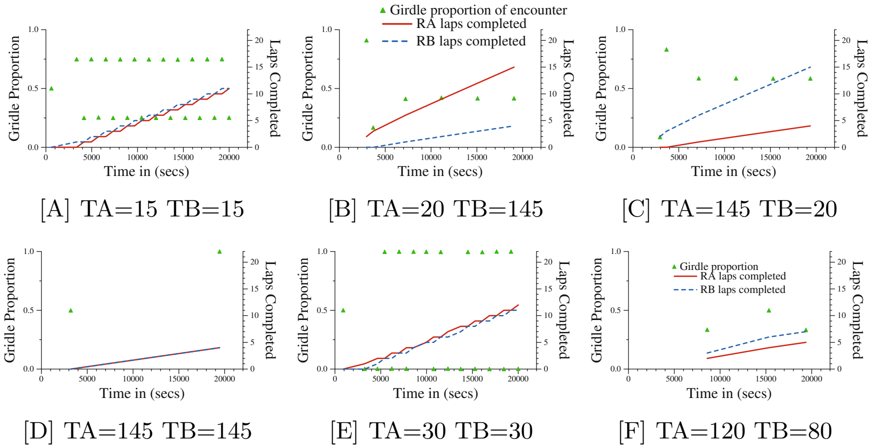
Fig. 9. Dominance for girdle length = 30m, R$_A$ is dominator. The girdle proportions are with respect to the position of robot R$_A$. TA corresponds to the task length of R$_A$, and TB to that of R$_B$. Results presented are based on simulation experiments. We observe the following at each point:

A & B— Every time R$_B$ makes an attempt to cross the girdle, it meets the dominator R$_A$ and is pushed out of the girdle, making no progress whereas R$_A$ finishes more than 15 laps during the alloted time. This is clearly a model of resource starvation.

E— In this case, R$_A$ and R$_B$ meet frequently and with R$_A$ being the dominator, R$_B$ is able to complete fewer trips than R$_A$ with such frequent spatial interference.

C— The encounter position is close to R$_B$'s girdle entry point, so even if R$_B$ retreats (R$_A$ being the dominator) the local investment made by R$_B$ is less. R$_A$ is the rational winner with aggression but at the cost of aggressive interaction time. Also they do not encounter one another every time R$_B$ enters the girdle. But R$_B$'s shorter task allows it to complete more trips than R$_A$.

F & D— The number of task iterations completed by both robots are almost equal. D belongs to the region where the number of encounters are less frequent (Figure 8(a)) and, if they occur at all, they are at the girdle end when R$_A$ is about to exit (see Figure 9[D]). In such a situation R$_A$ is the rational winner and the dominance hierarchy (with dominator R$_A$) is the best model to follow.



[A] TA=15 TB=15

[B] TA=20 TB=145

[C] TA=145 TB=20

[D] TA=145 TB=145

[E] TA=30 TB=30

[F] TA=120 TB=80

Fig. 10. Aggression GL30. The girdle proportion are w.r.t. the position of robot R_A.

A— R_A and R_B meet frequently. The position of encounter inside the girdle results in costly interactionsi.The number of task iterations completed by R_A and R_B is almost equal for the first 20,000 seconds of simulation and every time a rational winner is chosen. Compare this with how R_B performs when R_A is the dominator (Fig. 9[A] and 9[B]). R_B did not make any progress during the time allotted. Here the trade-off is to, either to engage in costly aggressive interactions obtaining a rational winner, giving a fair chance of winning to each robot or to resort to (cheap) dominance and bias towards one agent.

B— R_A and R_B do not meet as frequently here, but whenever they do, long aggressive interactions result: R_B is the rational winner in all cases. Compare this with Figure 9[B] where R_B hardly made any progress. The best resolution mechanism would be with R_B as dominator so that no aggression cost need be paid.

C— This is the reciprocal of case B and our earlier conclusion (but for R_A) holds true.

D— The number of task iterations completed by R_A and R_B are almost equal. We notice that D belongs to the region where the number of encounters are less frequent and they occur at the ends of the girdle when R_A is about to exit (Figure 9[D]). We had earlier concluded that, in such a situation R_A is the rational winner and the dominance hierarchy (with R_A being the dominator) is the best interference model to follow. We see that aggressive arbitration is also a reasonable interference resolution mechanism. The reason being that these regions have cheap interactions.

E— Compare Figure 10[E] with Figure 9[E]. The number of laps which R_B completes with aggressive signaling doubles compared with when it is dominated. However, the decrease in the number of laps of R_A is not that significant in both these modes of arbitration. We do see more frequent encounters in case of aggressive interactions, but all of these take place at the very ends of the girdle resulting in cheap arbitration, making it beneficial.

6 Conclusion

Several factors contribute to conflict resolution and its effectiveness:

Cost vs. precision of the arbitration mechanism—Time and energy cost are incurred in resolving resource conflicts. This influences the utility of aggressive displays in the first place.

Properties of the shared resource for which the agents are competing—This affects, among other things, the cost of communicating its aggression and what constitutes a worthwhile investment.

The task which each agent is assigned to perform—This can be coupled through the shared resource. This coupling, effects individual and collective dynamics.

The inherent noise in the "communication" channel—Noise plays a role in dynamic arbitration mechanisms: it can be beneficial in breaking symmetry, a situation which occurs when agents have identical aggression.

From all these facts we can conclude that there cannot be just one single best arbitration mechanism catering to all situations. We have also shown instances where a small variation of task ratio may cause a significant change in the task dynamics. With a prior knowledge of this entire task performance space, a single unfavorable interaction can be predicted beforehand, and by adding a wait to its task navigation, the robot can shift its performance to a more favorable region.

References

1. Brown, S., Zuluaga, M., Zhang, Y., Vaughan, R.T.: Rational aggressive behaviour reduced interferrence in a mobile robot team. In: Proc. International Conference on Advanced Robotics, Seattle, WA, U.S.A. (July 2005)
2. Enquist, M.: Communication during aggressive interactions with particular reference to variation in choice of behaviour. Animal Behav. 33(4), 1152 (1985)
3. Goldberg, D., Matarić, M.J.: Interference as a Tool for Designing and Evaluating Multi-Robot Controllers. In: Proc. AAAI National Conference on Artificial Intelligence, Providence, RI, U.S.A., pp. 637–642 (July 1997)
4. Stuart-Fox, D.: Testing game theory models: fighting ability and decision rules in chameleon contests. Proc. Rol. Soc. B: Biological Sciences 273(1593), 1555 (2006)
5. Vaughan, R.T., Støy, K., Sukhatme, G.S., Matarić, M.J.: Go ahead, make my day: Robot conflict resolution by aggressive competition. In: From Animals to Animats: Proc. Simulation of Adaptive Behavior, Paris, France, pp. 491–500 (August 2000)
6. Zuluaga, M., Vaughan, R.T.: Reducing spatial interference in robot teams by local-investment aggression. In: Proc. IEEE International Conference on Intelligent Robots and Systems, Edmonton, Alberta (August 2005)

Examining the Information Requirements for Flocking Motion

Benjamin T. Fine and Dylan A. Shell

Department of Computer Science and Engineering
Texas A&M University

Abstract. Flocking is an archetype emergent behavior that is displayed by a wide variety of groups and has been extensively studied in both biological and robotic communities. Still today, the exact requirements on the detail and type of information required for the production of flocking motion is unclear; moreover, these requirements have large potential impacts on biological plausibility and robotic implementations. This work implements a previously published flocking algorithm (Local Crowed Horizon) on a robotic platform and in computer simulations to explore the effects that the type and detail of information have on the produced motions. Specifically, we investigate the level of detail needed for the observation of flock members and study the differences between the use of pose and bearing information. Surprisingly, the results show that there is no significant difference in the motions produce by any observation detail or type of information. From the results, we introduce and define information-abstracted flocking algorithms, which are structured in such a way that the rule is agnostic to the observation detail and/or type of information given as input. Moreover, we believe our implementation of the Local Crowded Horizon flocking algorithm produces motions that require the least and most simplistic type of information (bearing only) which has been validated on robotic hardware to date.

1 Introduction

Flocking is a collective spatial behavior displayed by a wide variety of groups and has been studied in several research communities including biology, physics, and robotics. As an archetype emergent behavior, gaining an understanding of what the individual flock members observe and compute presents several challenges; one critical challenge is identifying how much detail is required when observing other flock members. It remains unclear how detailed the observations of other flock members need to be in order to produce flocking motion. For example, do individuals form a cohesive flock even if sub-groups or local densities (group-centric information) are observed rather than clearly discernible/identifiable individuals (member-centric information)? The results in this work prove that it is possible to produce flocking motion through the use of group-centric information.

The use of group-centric information shows that flock members do not need to detect individual flock members, which is difficult and error prone; instead, flock

T. Ziemke, C. Balkenius, and J. Hallam (Eds.): SAB 2012, LNAI 7426, pp. 442–452, 2012.

members need only detect groups of flock members. One conceivable implementation of a group-centric flocking algorithm is to design an agent which only detects *objects* that have a particular image-size. This would suggest that groups further away could still influence the flock members motions similar to the way a nearby flock member would. By reducing the complexity of the required sensing, a flocking algorithm can be implemented on a more simplistic agent (*e.g.,* doing some of the processing via an attentional mechanism), which makes the algorithm more biologically plausible and easier to successfully implement on a robotic system.

Another challenge in studying flocking, is understanding what type of information (*e.g.*, pose, bearing, velocity) must be gathered/sensed from the observations, and what implications the chosen information may have on biological plausibility and/or robotic implementations. The vast majority of the literature uses pose information (at least) to produce flocking motion [3,8,10,1], with relatively few works using only bearing information [9,6][1]. The bearing only algorithm studied here is not the first example of flocking motion produced by only using bearing information, but it is the first example of an algorithm that produces flocking motion which is equivalent when using either bearing or pose information. For the first time, we are able to compare the effects that different types of information have on the produced flocking motions through the implementation of a single algorithm.

To investigate the effects of the level of observation detail and the type of information sensed, we implemented the Local Crowded Horizon (LCH) rule [10] on a mobile robotic platform. The LCH was chosen for two reasons; (1) the rule has a unique structure that affords the ability to study the two aforementioned questions (level of observation detail and type of information) without modifying the algorithm and (2) Viscido *et al.* [10] shows the LCH produces flocking motions that are observed in biological flocks. In addition to the robotic implementation proof of concept, we implemented a computer simulation in order to better study the information requirements for flocking motion.

From the results of the robotic implementation and the computer simulations, we identify the notion of **information-abstracted flocking**, which refers to flocking algorithms that are structured in such a way that the resulting motion is agnostic to the detail of the observation and/or to the type of information sensed. In other words, if the motions produced by a flocking algorithm are equivalent under different types of information (*e.g.,* pose, bearing), then the algorithm is considered to be information-abstracted on type. The concept of information-abstracted flocking algorithms is rooted from the work of generic programming [7], where a single implementation of an algorithm can be *instantiated* with different data representations, an idea analogous to abstraction in abstract algebra. Information-abstraction is actually stronger, as it actually produces comparable emergent behavior despite being instantiated with different data representations.

[1] The model presented in [9] requires velocity; however, the velocity parameter is constant for the simulations; it cancels out leaving only the use of bearing information.

Additionally, the results in this work support the following claims:

- Some flocking algorithms may be structured so that the resulting behaviors are independent to the detail of the observation or the type of information given to the algorithm.
- Comparison of one flocking algorithm with another purely on the basis of the motion they produce is inadequate. This is especially the case when one is trying to make a claim about biological behavior on the basis of similar behavior generated through some computational means. The existence of information-abstracted flocking rules imply that, despite the output motion appearing equivalent, major pieces of the puzzle could remain underspecified.
- The vast majority of the literature assumes pose information is required to produce flocking behaviors, we further support the suggestion that biological flocking motion is possible using bearing information only.

Furthermore, we believe that our implementation of the LCH using only bearing information is the simplest biologically plausible, flocking algorithm to date.

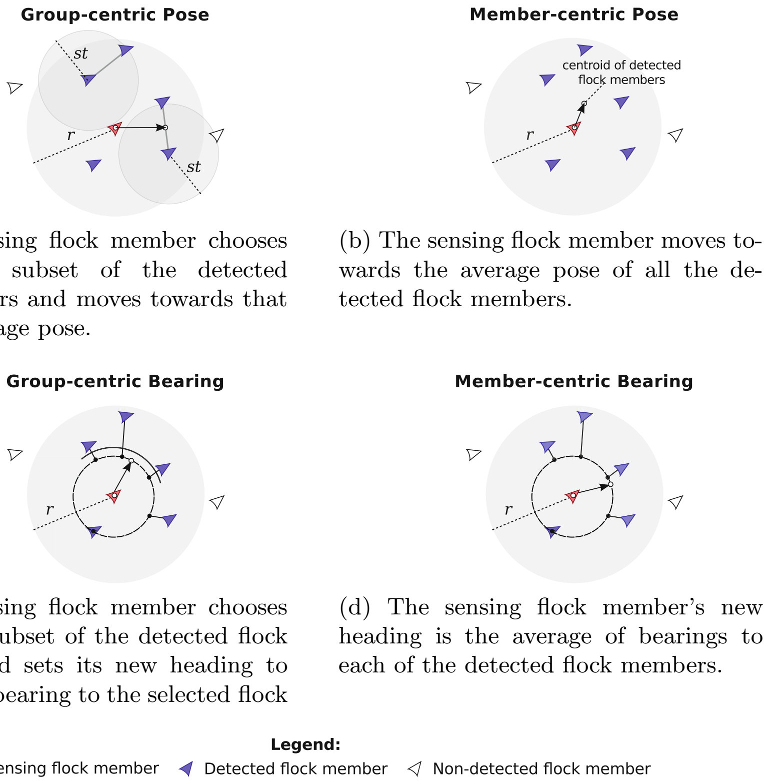
2 Local Crowded Horizon

The LCH was presented by Viscido *et al.* [10] as a biologically plausible explanation for flocking motion exhibited in the presence of a predator. In the original work there is a discrepancy between the theoretical design and the simulated version of the LCH. The authors describe the LCH by stating "[Flock members] use the density of the entire [flock] to determine their [next pose]"; however, their implementation has flock members "move toward the average [pose] of [all of the detected flock members.]" This discrepancy leaves the LCH with at least four different variations in regards to the information requirements; (1) group-centric pose, (2) group-centric bearing, (3) member-centric pose, and (4) member-centric bearing. The two **group-centric** variations use a subset of the detected flock members where the two **member-centric** variations use all of the detected flock members to compute the next pose. Figure 1 is a pictorial and prose description of the four different variations.

Both member-centric variations follow the same structure for the production of flocking motion. The member-centric variations take a set of all the detected flock members ($\mathbf{I}$) and moves one unit (d) towards the average of the feature vectors in $\mathbf{I}$. A **feature vector** contains all of the sensed information required for the computation of the next pose (*e.g.*, pose, bearing, velocity) for each flock member in $\mathbf{I}$. The only difference between the two member-centric variations is in how the averaging of the feature vectors are handled.

In the member-centric variation using pose (see algorithm named Variation 2) the function AveragePose($\mathbf{I}$) calculates the average pose of the set $\mathbf{I}$. To correctly average the flock member bearings (algorithm in full omitted, due to space), AverageBearing($\mathbf{I}$) replaces AveragePose($\mathbf{I}$) in variation 2 and calculates the bearing from the average unit vector from the flock member and all members in the set $\mathbf{I}$, *i.e.* 2-dimensional pose information is not required, only 1-dimensional bearing information.

The group-centric variations are identical to the member-centric variations, except that the motion command that is computed depends on only a subset ($\mathbf{I}'$) of the set $\mathbf{I}$. This models the idea that the motions follow some strict prioritization where attention need only be paid to some salient (or dense, or tightly-clustered) individuals. In both group-centric variations, the flock member computes the set $\mathbf{I}'$ based on density ($\mathbf{I}'$ contains all flock members that exist in the highest density cluster) and moves one unit towards the average of the feature vectors in the set $\mathbf{I}'$. Analogous to the case described above, different density selection functions are needed to handle the differences in the feature vectors. In our implementation of the group-centric variations, member-centric information is used to calculate the group-centric information to utilize the same detection process for all parameterizations. Of course, group-centric pose computations are only permitted to use group-centric information.



(a) The sensing flock member chooses the largest subset of the detected flock members and moves towards that group's average pose.

(b) The sensing flock member moves towards the average pose of all the detected flock members.

(c) The sensing flock member chooses the largest subset of the detected flock members and sets its new heading to the average bearing to the selected flock members.

(d) The sensing flock member's new heading is the average of bearings to each of the detected flock members.

Fig. 1. The four variations of the LCH flocking algorithm used in this study. Variations 1a and 1b require pose information where 1c and 1d use bearing information. Variations 1b and 1d utilize all of the detected flock members where 1a and 1c only use a subset of the detected flock members.

LCH Variation 1: Member-centric LCH Pose (Corresp. to Fig. 1(b))

```
Input: Set of detected flock members (I) in egocentric coordinate frame.
Parameters:
    $d \doteq$ distance to travel
    $\mathbf{r}_i \doteq$ current pose
Output: Desired pose in a member-centric coordinate frame.
if $|\mathbf{I}| = 0$ then
    return $[0, 0]$
else
    $\mathbf{v} \leftarrow$ AveragePose($\mathbf{I}$)
    return $\left[\mathbf{r}_{i^x} + \left(\frac{\mathbf{v}}{||\mathbf{v}||} * d\right), \mathbf{r}_{i^y} + \left(\frac{\mathbf{v}}{||\mathbf{v}||} * d\right)\right]$
```

LCH Variation 2: Group-centric LCH Bearing (Corresp. to Fig. 1(c))

```
Input: Set of detected flock members (I) in egocentric coordinate frame.
Parameters:
    $st \doteq$ splitting threshold (angle)
    $d \doteq$ distance to travel
    $\mathbf{r}_i \doteq$ current pose
Output: Desired pose in a member-centric coordinate frame.
if $|\mathbf{I}| = 0$ then
    return $[0, 0]$
else
    $\mathbf{CC} \leftarrow$ ConnComponents($\mathbf{I}, st$), where $\mathbf{CC} \geq 1$
    $\mathbf{I}' \leftarrow$ MaxConnComponent($\mathbf{CC}$)
    $\theta \leftarrow$ AverageBearing($\mathbf{I}'$)
    return $[\mathbf{r}_{i^x} + (\cos(\theta) * d), \mathbf{r}_{i^y} + (\sin(\theta) * d)]$
```

In both group-centric variations, the flock member groups the detected members based on a threshold (st) in the set **CC** (st could either be a distance or angular based threshold). The algorithm then selects the largest group from the set **CC** to compute the next pose. The function MaxConnComponent(**CC**) takes **CC** and returns the largest group of flock members. The details of the group-centric variation using bearing information can be seen in Variation 2 (the group-centric variation using pose is not shown here due to space).

3 Robotic Implementation

Each robot (flock member) is an iRobot Create [5] equipped with a ASUS Eee PC and a Hokuyo URG-04LX-UG01 [4] laser range-finder; see Figure 2a. The robots are wrapped in a highly reflective material and each robot has a modified Gearbox 9.07 driver [2] that allows for the detection of other robots. The trials conducted for this study consider four robots in the single-group starting formation (Figure 2b) in a obstacle free space that can be considered infinite for the presented trials.

We conducted several trials for each of the four LCH variations mentioned earlier; Figure 3 shows time series from a few of these trials. There is no significant difference observed in the motions produced by the four different variations of the LCH. We do observe that the motions produced by our robotic implementation do not collapse into a single group, as observed in the simulations of Viscido *et al.* [10]. However, the robots do indeed collapse into a single group, but because of the robot's limited field of view (FOV), the flock exhibits directional motion. This directional motion occurs when one or more robots do not

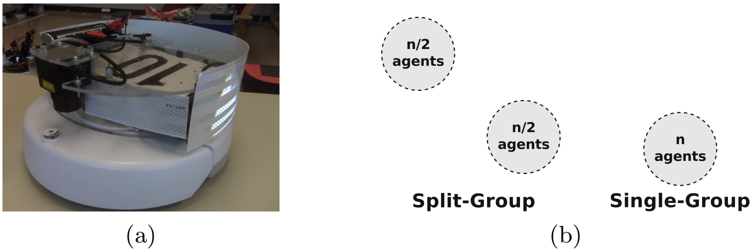


Fig. 2. Figure 2a shows a single iRobot Create with a Hokuyo laser and a ASUS Eee PC wrapped in high reflectance material for robot detection. Figure 2b shows the two starting formations used in this study. The exact poses of the flock members, within the formation, were chosen at random for each run.

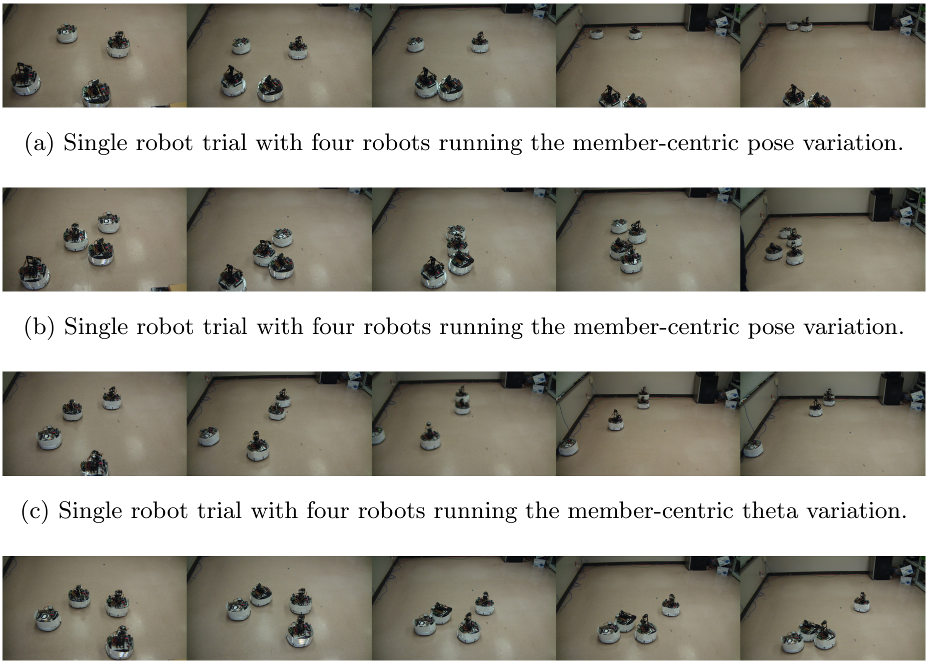
observe other robots in the flock, therefore they continue moving in their current direction; see Figure 4a. We show, through computer simulations, that the FOV limitations of the robots is the primary cause of the motion differences.

We should note that Figure 3 does not show any results from trials using the group-centric variation because the size of our system is not large enough to show the desired motions. When the there is only one robot in the selected group, the robots will exhibit motions that can best be described as a *follow the leader* behavior. Figure 4b is a pictorial representation showing how the *follow the leader* motions are generated.

3.1 The Effect of a Limited Field of View

Using MatLab (version R2011b) we implemented all four variations of the LCH similar to the implementation in [10]. Each flock member has access to the global information for every other flock member but each member is only able to *detect* flock members within the sensing radius (r). Each flock member will calculate their next pose ($\mathbf{r}_i$) according to the given LCH variation. Additionally, we included the flock members FOV into our implementation in order to account for the limited FOV of our robots.

Figure 5a shows the simulated motions of 50 flock members for the member-centric variation using pose with no limitations on their FOV. These motions reveal that our implementation of the LCH is similar to the LCH implementation in [10]. Furthermore, the motions in Figure 5b show the simulated motions for the 50 flock members with a limited FOV similar to the FOV of the robots. Comparing the motions in Figure 5b and Figure 3 we notice our simulation, with a limited FOV, can produce equivalent motion to the robotic implementation. Since our implementation can produce the motions of the original motivating work and the motions produced by a robotic implementation we feel the simulation is adequate for further investigation of information (type and detail) on flocking.



(a) Single robot trial with four robots running the member-centric pose variation.

(b) Single robot trial with four robots running the member-centric pose variation.

(c) Single robot trial with four robots running the member-centric theta variation.

(d) Single robot trial with four robots running the member-centric theta variation.

Fig. 3. Each time series shows the motions of our robotic system running one of the four LCH variations

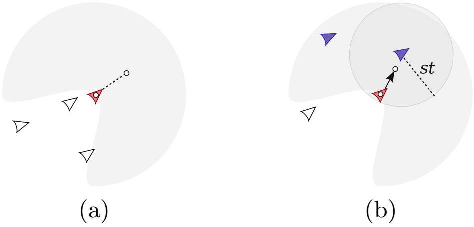
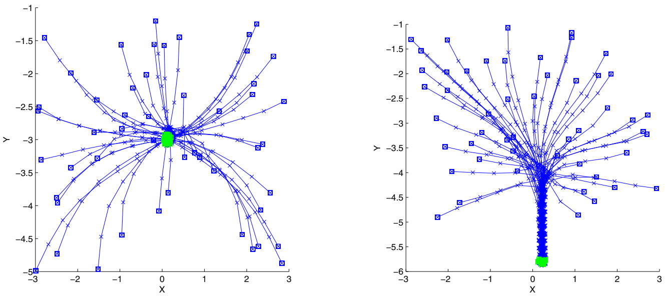


Fig. 4. Figure 4a shows that a flock member will continue in the same direction when no other flock members are observed. This behavior causes the resulting motion of the flock to be more directional than the computer simulations in [10]. Figure 4b shows how the *follow the leader* behavior is generated when there are only a few detected neighbors.



(a) No limitations on FOV (b) Limitations on FOV

Fig. 5. These two motion plots were generated from 50 simulated flock members using the member-centric pose variation of the LCH. The blue squares represents the starting formation, which was randomly generated within a squared region, and the green squares represent the ending formation of the flock members.

4 Information-Abstracted Flocking

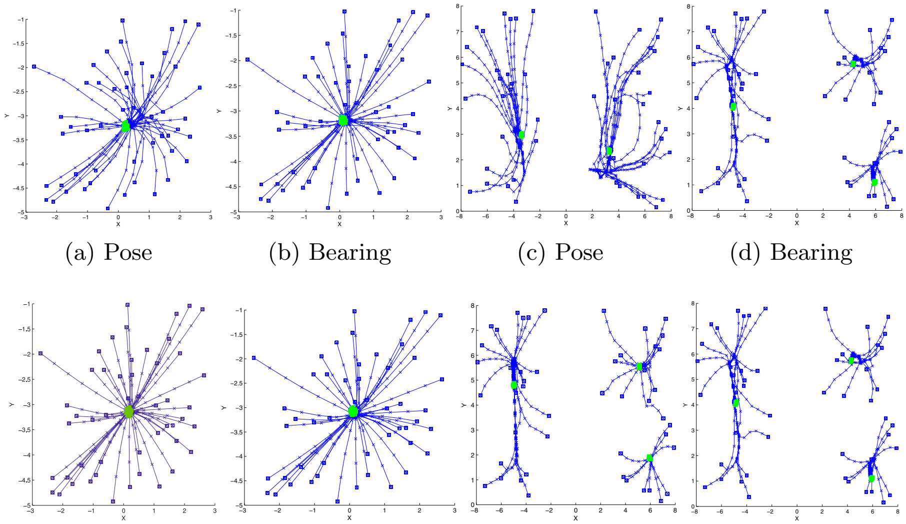
To investigate the existence of information-abstracted flocking algorithms, we conducted computer simulations for each of the 16 different parameterizations of sensing range (3000[2] and 3 units), starting formation (single-group and split-group), type of information (pose and bearing), and observation detail (member-centric and group-centric). For each parameterization we conducted ten trials, each with random starting positions for the flock members, resulting in a total of 160 simulations of 75 iterations for 50 flock members. To aid in the study of the underlying motions produced by the chosen variations, we do not limit the flock members' FOV.

Using motion equivalence to compare the results in Figure 6, we see very little difference in the simulated motions. Even when we look at the worst case, Figures 6c and 6g, there are no real differences in the resultant motion. Even though the motions are not identical, they still resemble the motions of the motivating biological flocks as presented in [10].

In addition to motion equivalence, Viscido *et al.* [10] proposed the use of mean median distance (MMD) from the center of the flock as a metric to describe the motions of a flock that compresses during a predator attack. To supplement the observations of motion equivalence made form the motion plots in Figure 6, we calculated the MMD for all of the parameterizations over all trials. The computed MMDs for the trials that used a sensing range of 3 units are not significantly different from the trials that used a sensing range of 3000 units, therefore we only report the MMDs from the trials that used a sensing range of 3 units; see Table 1.

When the flock starts in a single-group, Table 1 shows that there is no difference in the flock's MMD for any of the 16 parameterization. When the flock starts

[2] For these results 3000 units can be considered infinite.



(a) Pose (b) Bearing (c) Pose (d) Bearing

(e) Pose (f) Bearing (g) Pose (h) Bearing

Fig. 6. These motion plots are typical results of simulations conducted for this study. Each trial simulates 50 flock members over 75 iterations (only plotting every third pose per flock member), where st_{distance} is set to 0.5 units and st_{angular} is set to 10 degrees. The top row of motion plots (plots 6a,6b, 6c, and 6d) where generated using group-centric information and the bottom row of plots (plots 6e, 6f, 6g, and 6h) where generated using member-centric information. The blue squares represent the starting positions of the flock members, and the green squares represent the end positions of the flock members. Motion plots 6e, 6f, 6a, and 6b were simulated with the single-group starting formation and a sensing range of 3000 units, where plots 6g, 6h, 6c, and 6d were simulated using the split-group starting formation and a sensing range of 3 units.

Table 1. The MMD (in units) from all of the simulations that had a sensing range of 3 units. For each simulation, the median distance from the center of all 50 flock members is calculated from the ending formation. The medians from all ten trials were averaged to yield the MMD for the given parameterization.

Single-group

	Pose	**Bearing**
Group-centric	0.05	0.05
Member-centric	0.05	0.05

Split-group

	Pose	**Bearing**
Group-centric	3.23	4.94
Member-centric	4.95	4.94

in two separate groups, there is a slight difference in the computed MMDs. As we would expect from the motions reported in Figure 6, there is a slight difference between the MMDs of the group-centric pose and member-centric pose trials when the flock started in two groups. These results not only show that flocking motions are possible using group-centric information, but since the group-centric variation produces a smaller MMD, this suggests that group-centric information may be preferred in certain situations.

5 Conclusion

The preceding results show that the information available to a flock member, while very important from an implementation and biological modeling point of view, are not necessarily distinguishable in terms of the flocking motion they produce. Additional thought is required if one is to understand what individual flock members are sensing or computing. At least two distinct aspects are worthy of consideration: the detail involved in the observation and the type of information extracted from that observation. Moreover, metrics that are intended to evaluate the flock's motion (*e.g.*, MMD) are also insufficient in and of themselves. Therefore, future work may need to focus less on metrics designed to study the resulting group motion, and instead on the impact that the requisite information has on the biological plausibility, or the ability to implement the given algorithm on a robotic system, or both.

The ability to produce biologically motivated flocking motions using either group-centric or member-centric information suggests that flocking motions may be possible using a combination of the two. In other words, the flock member may observe individual flock members when they are nearby, but may still observe groups that are further away. Furthermore, if we consider the case when flock members only make observations based on image-size, it is plausible that the flock member may not be aware of the differences between the two types of information. If this holds true, this may offer an explanation to how multiple flocks merge an split over time as occurs in large flocks of birds.

References

1. Conradt, L., Krause, J., Couzin, I.D., Roper, T.J.: "Leading According to Need" in Self-Organizing Groups. The American Naturalist 173(3), 304–312 (2009)
2. Gearbox. The gearbox home page (February 2012), gearbox.sourceforge.net
3. Hamilton, W.D.: Geometry for the Selfish Herd. Journal of Theoretical Biology 31(2), 295–311 (1971)
4. Hokuyo. The hokuyo home page (February 2012), http://www.hokuyo-aut.jb
5. iRobot. The irobot create home page (February 2012), store.irobot.com/shop/index.jsp?categoryId=3311368
6. Jadbabaie, A., Lin, J., Stephen Morse, A.: Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules. IEEE Transactions on Automatic Control 48(6), 988–1001 (2002)

7. Musser, D.R., Stepanov, A.A.: Generic Programming. In: International Symposium on Symbolic and Algebraic Computation (ISSAC), pp. 13–25 (1988)
8. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. Computer Graphics 21(4), 25–34 (1987)
9. Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., Shochet, O.: Novel Type of Phase Transition in a System of Self-Driven Particles. Phys. Rev. Let. 75(6), 1226–1229 (1995)
10. Viscido, S.V., Miller, M., Wethey, D.S.: The Dilemma of the Selfish Herd: The Search for a Realistic Movement Rule. Jour. of Theor. Bio. 217(2), 183–194 (2002)

Author Index