Robotics and Autonomous Systems 76 (2016) 80–92

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Concurrent controller and Simulator Neural Network development for a differentially-steered robot in Evolutionary Robotics

Grant W. Woodford [a,*], Christiaan J. Pretorius [b], Mathys C. du Plessis [a]

[a] Department of Computing Sciences, Nelson Mandela Metropolitan University (NMMU), Port Elizabeth, South Africa

[b] Department of Mathematics and Applied Mathematics, Nelson Mandela Metropolitan University (NMMU), Port Elizabeth, South Africa

HIGHLIGHTS

- Controllers are developed for trajectory planning using Evolutionary Robotics and a differentially-steered mobile robot.
- A novel approach is proposed for the concurrent development of controllers and a simulator.
- Robot behaviours are simulated using Artificial Neural Networks.
- An extensive parameter comparison study of the proposed approach is conducted.

ARTICLE INFO

Article history:
Received 13 March 2015
Received in revised form
12 October 2015
Accepted 23 October 2015
Available online 19 November 2015

Keywords:
Evolutionary Robotics
Coevolution
Simulator
Artificial Neural Networks
Differentially-Steered
Mobile robotics

ABSTRACT

Evolutionary Robotics (ER) strives for the automatic creation of robotic controllers and morphologies. The ER process is normally performed in simulation in order to reduce the time required and robot wear. Simulator development is a time consuming process which requires expert knowledge and must traditionally be completed before the ER process can commence. Traditional simulators have limited accuracy, can be computationally expensive and typically do not account for minor operational differences between physical robots.

This research proposes the automatic creation of simulators concurrently with the normal ER process. The simulator is derived from an Artificial Neural Network (ANN) to remove the need for formulating an analytical model for the robot. The ANN simulator is improved concurrently with the ER process through real-world controller evaluations which continuously generate behavioural data. Simultaneously, the ER process is informed by the improving simulator to evolve better controllers which are periodically evaluated in the real-world. Hence, the concurrent processes provide further targeted behavioural data for simulator improvement.

The concurrent and real-time creation of both controllers and ANN-based simulators is successfully demonstrated for a differentially-steered mobile robot. Various parameter settings in the proposed algorithm are investigated to determine factors pertinent to the success of the proposed approach.

1. Introduction

Evolutionary Robotics (ER) is a field of study that involves the automatic artificial evolution and optimization of particular traits of autonomous robotic systems [1]. ER seeks to automate the development of robot controllers and morphologies through the use of Evolutionary Computation as an alternative to their manual creation. As robots, environments and tasks become more complex, the manual creation of controllers becomes more infeasible [2]. ER can be used to evolve robot behaviours such that path following, inverted pendulum stabilization, light following, obstacle avoidance and many others [3,4]. In ER, controllers need to be evaluated to quantify the relative performance of each controller at performing a given task. ER requires the evaluation of a large number of controllers in order to evolve better ones. However, evaluating a large number of controllers on a real-world robot is unrealistically time-consuming and can damage hardware through mechanical wear [5]. To overcome these issues, simulators serve as an alternative to reality for evaluating controller performance in the ER process [5].

* Corresponding author.
E-mail addresses: grant.woodford@nmmu.ac.za (G.W. Woodford), cpretorius@nmmu.ac.za (C.J. Pretorius), mc.duplessis@nmmu.ac.za (M.C. du Plessis).

http://dx.doi.org/10.1016/j.robot.2015.10.011
0921-8890/© 2015 Elsevier B.V. All rights reserved.

There are many different types of simulators. Simulators can be broadly classified into three categories, namely physics based simulators, empirical models (based on experimentally collected data) and hybrids which are a combination of both physics and empirical models [3]. Traditional methods for creating simulators are often time-consuming and complicated because it may require the construction and tuning of complex physics based models or the collection of large amounts of real-world data or both.

Much research in ER is concerned with overcoming the challenges in using simulators effectively [1]. Challenges in simulator design are inaccuracies and/or over-simplification in the modelling of certain phenomena. Oversimplified or inaccurate simulators may result in controllers that rely too heavily on peculiarities that exist only in simulation but are not present when the controller is evaluated in reality, commonly referred to as the *reality gap* problem [6]. Oversimplification can be avoided by using highly accurate simulators. However, even highly accurate simulators cannot model reality perfectly and will inevitably contain inaccuracies [7]. Highly accurate simulators can also be computationally expensive [8]. Scalability can become an issue where the time taken to evaluate controllers can grow substantially with increased complexity in the robotic system [1,9]. Thus, simulators ideally need to provide highly accurate representations of reality whilst not being too computationally expensive to operate.

Pretorius, du Plessis and Cilliers [10] have shown that Artificial Neural Networks (ANN) can be utilized as simulators in the ER process. A simulator created using ANNs shall be referred to as a Simulator Neural Network (SNN). SNNs have been shown to possess good prediction accuracy, noise-tolerance and generalization abilities in the modelling of certain robot behaviours [3]. SNNs are computationally efficient and can be created without the need for specialized knowledge about the dynamics of the robotic system. It has also been shown that SNNs can aid in the successful transference of controller behaviour from simulation to reality [3].

Previous approaches to SNN construction required the gathering of large quantities of data from real-world robot behaviour which can be used to train SNNs before the ER process begins. This development of simulators before controllers are evolved is time-consuming. The robot generates the behavioural data by evaluating randomly generated commands. However, if only certain behaviours need to be modelled then much of this behavioural data is unnecessary. A simulator could specialize in accurately modelling only the behaviour required to perform a given task and thus require mostly behavioural data specific to the desired behaviour.

As an alternative to the traditional approach to SNN creation, the current work aims to investigate the concurrent creation of SNNs and controllers in the ER process. This approach could potentially speed up the process by eliminating the need to precompute SNNs. The SNNs would be specialized to accurately model only behaviours required for a given task and therefore require less empirical data for their development than the previous approach. There could also potentially be further advantages warranting investigation such as the ability to adapt to changing environments and robots. Automatic damage recovery capabilities could also be possible since the simulator is no longer static.

The related work (Section 2) gives an outline of some of the work previously done in this area. The proposed approach of concurrent simulator and controller development is outlined in Section 3. The evaluation methodology and experimental procedure used for investigating the proposed approach are then discussed in Sections 4 and 5, respectively. The results of the experimental work are presented (Section 6) and finally conclusions are drawn and future work is suggested (Section 7).

2. Related work

This section reviews important approaches in the development of simulators and controllers for the ER process. Section 2.1 discusses related work with regard to concurrent controller and simulator development. Section 2.2 discusses the development and use of SNNs in ER.

2.1. Concurrent controller and simulator development

One approach to dealing with the *reality gap* problem described in Section 1 is to evolve controllers on real-world hardware only [11]. On the other hand, a combined approach can be taken where controllers are evolved in simulation then transferred to a real-world robot where the ER process continues in reality [12].

The one directional transference of a controller from simulation to reality can also be replaced by a more bidirectional approach [1]. There have been many proposals for bidirectional approaches that allow the optimization process to alternate between the simulator and real-world [5,13–15].

A robot's behaviour is affected not only by its controller but also by the robot's morphology. A coevolutionary approach can be taken where controllers and robot morphologies are improved together [16,17]. Evolving a robot's morphology usually necessitates changing the model of what the robot looks like and evolving the physics model parameters of the simulator [14,18]. If a robot's model is fixed, then only the physics model parameters of the simulator could be evolved. Work related to coevolutionary approaches to controller and simulator development are thus important.

One example of the coevolution of simulator model parameters and controllers is the Anytime Learning Algorithm [13]. Anytime learning proposes a population of model parameters and a population of controllers. These populations are evolved by means of a Genetic Algorithm (GA). A population of model parameters is maintained and judged against real-world data to determine their accuracy. The most accurate model parameters are used for controller evolution.

Bongard, Zykov and Lipson [14] proposed the Estimation–Exploration Algorithm (EEA) that integrates robotic self modelling into the ER process. The EEA is a hybrid coevolutionary algorithm for maintaining populations of models of the robot system and a population of test controllers used to explore the model search space. Once a sufficiently accurate model has been found, it is used to evolve controllers to achieve the required behaviour.

The Transferability Approach has also been proposed [15]. This method does not attempt to evolve the simulator but rather attempts to complement it. Simulators often have limitations in the accurate modelling of certain phenomena in order to increase computational performance [15]. If these limitations are known during the ER process, controllers could avoid relying on dynamics that have not been accurately simulated. The Transferability Approach proposes a multi-objective fitness function of two parts. One part estimates how well a solution may transfer from simulation to reality (called the transferability function) and another estimates how well the desired behaviour is achieved in simulation [15]. The transferability function can be implemented as an ANN or Support Vector Machine and may be trained during the ER process. Controllers evaluated on a real-world robot generate data that is utilized in the training of the transferability function. Behavioural features that can be measured in simulation and reality are identified and the transferability function is trained to estimate how well these behaviours are simulated.

The Back to Reality (BTR) Algorithm concurrently evolves controllers and simulators [5]. The BTR Algorithm collects the fitness of controllers evaluated in simulation and reality. The