Adapting Sensory and Controller Configurations in Robot Technologies

NAEEM GANEY, University of Cape Town - South Africa

This paper investigates effective methods to enable Artificial Intelligence in Autonomous Mobile Robot systems. We analyse the various Machine Learning techniques to find a model which can adapt the controller and morphology of an autonomous mobile robot to accomplish a specific task. The task is for mobile robots to cooperate and push boxes from one area to another in a simulated environment.

Contents

1. INTRODUCTION

Artificial Intelligence [Russel and Norvig 2010] is a field of study concerned with the intelligence of machines or software. Machine Learning, a subset of Computer Science and Statistics, makes use of large amounts of data to enable learning in a machine. Machine Learning techniques have been applied extensively in Robotics to assist in the training of the controllers Autonomous Mobile Robots. The intelligence required to navigate an Autonomous Mobile Robots is very complex and it would be impractical to design a controller by conventional means, hence the need for a machine learning approach [Quin et al. 2002].

This paper will be primarily concerned with Evolutionary Computing (EC) [Eiben and Smith 2003] as a methodology for training autonomous mobile robots. Because of the use of Evolutionary Computing in the field of robotics, the combination is often referred to as Evolutionary Robotics (ER) [Lund 2003; Wahde 2007; Nolfi 2007]. In addition to training the controller of a robot, we will also adapt the physical configurations of a robot - the sensors and actuators that allow a robot to be aware of its surrounding and move accordingly. The training and testing of the robots are done in a simulated environment to speed up the learning process. [Petrovic 1999] justifies the need for a simulation of the robot and its environment for two very important reasons : (1) it is less expensive because no components are damaged in the experimentation process and (2) it allows the researcher to focus on the design of the controller model rather than problems encountered in the Engineering process.

Autonomous Mobile robots can be trained to work collectively on a specific task. Co-operation between robots is required when a single robot cannot complete a task without the assistance of another mobile robot. This sort of system is referred to as a Multi-Agent-System [Woodridge 2009; Dongyong et al. 2004] and the training is different to that of single-robot systems.

2. MACHINE LEARNING

Machine Learning [Mitchell 1997], a field of study in Artificial Intelligence, incorporates methods from Computer Science and Statistics. Machine Learning makes use of algorithms and large data sets accurately model a system. The data allows a machine to learn from past experiences to make accurate decisions or predictions. There are 3 main Machine Learning approaches: Supervised Learning, Unsupervised Learning and Reinforcement Learning. We will analyse which of the three approaches (or combination of approaches) would be most appropriate for the configuring the controllers and morphologies of our autonomous robots.

2.1. Supervised Learning

In the Supervised Learning Approach, the system is provided with a set of input and correct outputs. Using this information, the system models a mapping from input to output for new (unseen) input values. The supervised learning approach is **not** suitable for the training of autonomous mobile robots because we do not have the data of correct inputs and outputs to supply to the learning algorithm. Furthermore, the controller of a robot is complex, with many permutations which makes it difficult for us to provide inputs and correct outputs for every permutation. Because of this complexity - Unsupervised and Reinforcement Learning methods are more common in the research of Artificial Life for Robotics.

2.2. Unsupervised Learning

Unsupervised Learning, unlike Supervised Learning, has only a set of inputs without the correct outputs. The most commonly used method in Unsupervised Learning is *clustering* : organising data into clusters with the hope of finding a hidden underlying structure in the data. Neuro-Evolution (which will be discussed later) is a popular method that tries to replicate the evolutionary process that created the brain. During the evolution process of creation of the brain, the brain was evolved without a 'teacher' and therefore used types of unsupervised learning to change based on their experience [Sher 2012].

2.3. Reinforcement Learning

Reinforcement Learning [Barto 1998; Russel and Norvig 2010] is a learning mechanism where an *agent* is trained by means of reinforcement - a reward or punishment. An *agent* executes an *action* a in *state* s to maximise *reward* r. The reinforcement learning method is an attractive method in our robotics application. Robot agents act in an environment and are rewarded for pushing the boxes into their correct positions and are punished for moving (power usage). A learning algorithm will train the robot system to maximise reward (push as many boxes into place) and minimise punishment (wasting energy by moving).

3. ARTIFICIAL NEURAL NETWORKS

3.1. Structure of an Artificial Neural Network

The design of an Artificial Neural Network [Schalkoff 1997] is inspired by the biological structure of animals in nature. An Artificial Neural Network consists of nodes (or neurons) and links between these neurons. Neurons are organised in layers : the input layer, the output layer and 1 or more *hidden* layers in between. Artificial Neural Networks with hidden layers are referred to as *multi-layer networks*. The input and output layers are typically decided by the system and the hidden layers are either carefully specified or can form part of the elements that a machine learning algorithm needs to optimize.

The general structure of an Artificial Neural Network (ANN) is one where every Node in layer n is connected to every other node in layer $n + 1$. This structure is commonly referred to as a fully-connected structure. A fully connected structure with a single hidden is considered to be a universal function approximator [Cybenko 1989]

Another structure that is common in Evolutionary Robotics is a recurrent network structure whereby a node from layer n can connect to a node in the previous layer. It is shown by [Spronck et al. 2004] that a recurrent network is more suitable for a controller of an autonomous mobile robot because it can 'direct the wheels with more subtlety'. It is important to have subtle wheel movements to carefully avoid obstacles and walls in a typical mobile robot application.

3.2. Weight Adaptation

An Artificial Neural Network (ANN) is considered to be trained when a desirable output is received for an input. In order to train an Artificial Neural Network, we need to configure the weights of the connections between neurons. Artificial Neural Networks can be used in all three Machine Learning Strategies where the strategy will decide how the weights are adapted. For example, to train an ANN with a supervised learning method, we would use back propagation. Evolutionary Algorithms have been proven to work in adapting the ANN of a Robot Controller by [Petrovic 1999; Quin et al. 2002; Lund 2003; Spronck et al. 2004] and many others. We will therefore be using Neuro-Evolution [Miikkulainen 2010] to train the Artificial Neural Network with an Evolutionary Algorithm.

3.3. Topology Adaptation

In addition to weight adaptation, we can also adapt the structure (or topology) of an Artificial Neural Network. The input and output layers of a system are typically defined by the system. However, The number of hidden layers and the number of nodes per hidden layer can be adapted to perfectly suit our application. A good method described by [Stanley and Miikkulainen 2002] is NEAT (Neuro-Evolution for Augementing Topologies) which uses Neuro-Evolution to adapt the topology of an Artificial Neural Network. Since the controller of a autonomous mobile robot is a complex system to model - it is worthwhile looking into evolving the topology and weights to yield a better result and more accurate model.

3.4. ANN for Robot Controller

[Petrovic 1999] uses an Artificial Neural Network with recurrent connections to model the controller of an autonomous mobile robot. The input nodes are the sensory reading from the sensors on the robot and the output nodes control the actuators of the robot. In another work by [Waibel et al. 2009], an ANN is also used to model the controller of the robot but it does not make use of a recurrent network configuration. We agree with [Spronck et al. 2004] to use a recurrent Artificial Neural Network as it can navigate the robot with more subtlety.

We agree that an Artificial Neural Network is suitable for modelling the controller of autonomous mobile robot. [Nolfi et al. 1994] explains the appropriateness of using an Artificial Neural Network for controller design for 2 main reasons : neural networks are resistant to noise and they can 'learn various forms of learning which can speed up the learning process'.

4. EVOLUTIONARY ALGORITHMS

Evolutionary Algorithms, a subfield of Evolutionary Computing [Eiben and Smith 2003] have been inspired the concepts of Darwinian evolution. An Evolutionary Algorithm has a set of candidate solutions which are refined through the process of selection, mutation and recombination. A fitness function evaluates the quality of each candidate solution. At each iteration, a new generation is created by selecting the fittest individuals from the previous generation, recombining sets of individuals to create new ones and mutating the resulting offspring. The new generation is then evaluated and the selection, recombination and mutation process is repeated until a satisfactory condition is met. [Parisi et al. 1991] stresses the importance of mutation to create candidates that are similar to their parents but not exactly the same.

There are four main types of Evolutionary Algorithms : (1) Genetic Algorithms, (2) Evolutionary Strategies, (3) Genetic Programming and (4) Evolutionary Programming [Spronck et al. 2004].

4.1. Neuro-Evolution

Neuro-Evolution [Miikkulainen 2010] is a concept similar to Dynamic Programming but makes use of Artificial Neural Networks. Neuro-Evolution adapts the weights of an Artificial Neural Network by using an Evolutionary Algorithm.

4.2. Genetic Algorithms

Genetic Algorithms [Holland 1975] fall under the category of Evolutionary Algorithms but differ in the way the information is represented or encoded. Genetic Algorithms make use of bit-string (a string of 1's and 0's) to represent an individual.

5. MORPHOLOGY OF A ROBOT

The sensors and actuators used in a robot configuration is referred to as the *morphology* of a robot. [Petrovic 1999] refers to the morphology of the robot as the 'body' and the controller as the 'brain' of a robot system and encourages the design of the 'brain' **and** 'body to achieve the best performance. The *morphology* of a robot can predetermined by the researcher or adapted by a learning algorithm.

5.1. Robot Sensors

In order for a robot navigate in an environment it is important for the robot to be fitted with sensors for it to be aware of its surroundings. [Borenstein et al. 1996] categorises sensors into 2 categories : (1) absolute position measurements and (2) relative position measurements. Borenstein et al. also suggests the use of sensors from both categories to maximise awareness of the environment.

5.2. K-Team Corporation - Khepera Robot

[Spronck et al. 2004] uses the Khepera robot for a 'box-pushing application' and has proven that the controller designed was able to direct a simulated and real-life version Khepera robot. The Khepera [K-Team Corporation 2014; Mondada 2005] robot has been used in many other applications for the development of Artificial Intelligence in Robot Technologies sensors from both categories to maximise awareness of the environment. The Khepera robot has many sensors that can be configured and has good processing power which makes it an attractive choice for research in Evolutionary Robotics.

6. COEVOLUTION OF MORPHOLOGY AND CONTROLLER

The focus of co-evolution is to adapt the morphology and controller of the robot simultaneously. [Lund 2003] refers to the morphology and controller as the brain and the body of a robot and adapts the controller by using Genetic Programming and a Genetic Algorithm for adapting the Morphology of the robot.

6.1. Adapting Morphology

The Khepera Robot has limited physical space, battery power and processing power to accommodate an infinite number of sensors. Hence, there will be constraints placed on the number of sensors and types of sensors required by the system. In the work done by [Lund 2003] on LEGO robots - the number of possible sensors, the number of possible positions of the sensors on the system were computed to defines the search space of possible morphology configurations. Population Based Incremental Learning [Baluja 2004] is a useful Genetic Algorithm for selecting appropriate sensors. The data structure that supports PBIL is a bit string (an encoding of 1's and 0's) which acts as a selector/deselector of various sensor types and positions. PBIL converges to a solution much faster than a Greedy-Search or Standard Genetic algorithm and would therefore not be as computationally expensive.

6.2. Adapting Controller

As mentioned earlier in this paper, Neuro-Evolution is a good choice for adapting the controller of the robot.

7. MULTI-AGENT SYSTEMS

Multi-agent systems [Woodridge 2009] is a system that involves more than one individual agent (or robot). The agents can either be *competitive* or *cooperative*. In a cooperative system - the agents succeed and fail as a team. In competition, an individual agent benefits at the expense of his another robot. Cooperation is required when an individual robot is not able to complete a task without the assistance of a fellow robot in the same system. For example, the box-pushing application [Petrovic 1999] requires individual robots to pair up with one or more robots in order to be able to have enough force to push a box.

Another key element to consider with multi-agent systems is whether the agents are *homogenous* or *heterogenous*. In a *homogenous* solution - all individuals in the simulation are configured with the same controller and the same morphology. In a *heterogenous system*, the individuals in a simulation have different controller and morphology configurations.

8. CONCLUSIONS

Using our findings above, we will implement a co-evolution learning algorithm to adapt the morphology and controller of an autonomous mobile robot. The Khepera III by the K-Team Corporation is the robot we are going to model our solution on. Neuro-Evolution will be tested for adapting the controller of the robot and PBIL will be tested as a method for adapting the morphology of the controller. A homogenous and heterogenous approach will be tested to investigate which method is more effective for our box-pushing and trash collection task. Machine Learning methods require that a system be trialled and tested with multiple solutions which can be time consuming and computationally expensive. We will make use of Parallel Computing and a High-Performance Computer to speed up simulations.

REFERENCES

Shumeet Baluja. 2004. Population-Based Incremental Learning : A method for Integrating Genetic Search Based Function Optimization and Competitive Learning. *Technical Report - CMU-CS-94-163* (2004).

Andrew Barto. 1998. Reinforcement Learning : An Introduction. *MIT Press* (1998).

J. Borenstein, H.R Everett, and D. Wehe. 1996. Mobile Robot Positioning - Sensors and Techniques. *Journal of Robotic Systems. Special issue on Mobile Robots* (1996).

George Cybenko. 1989. Approximations by Superpositions of a sigmoidal function. *Mathematics of control, signals and systems* (1989).

Yang Dongyong, Tang Qiong, Fang Luping, Zhou Xiao, and Jiang Jingping. 2004. Cooperative Multi-Agent Transport Based on Coevolution. *SICE Annual Conference in Sapporo* (2004).

A.E Eiben and James Smith. 2003. *Introduction to Evolutionary Computing*. Springer.

John Holland. 1975. Adaptation in Natural and Artificial Systems. (1975).

K-Team Corporation 2014. *Khepera iii - Specifications*. K-Team Corporation.

Henrik Hautop Lund. 2003. Co-Evolving Control and Morphology with LEGO Robots. (2003).

Risto Miikkulainen. 2010. *Neuroevolution*. Springer, 716–720.

Tom Mitchell. 1997. *Machine Learning*. McGraw Hill.

Francesco Mondada. 2005. Mobile Robots for a Synthetic Approach to Interdisciplinary Research. *LSA - STI - EPFL* (2005).

Stefano Nolfi. 2007. Evolutionary Robotics. (2007).

Stefano Nolfi, Dario Floreano, Orazio Miglino, and Francesco Mondada. 1994. How to evolve autonomous robots : Different approaches in evolutionary robotics. *Proceedings on the Artificial Life IV Conference* (1994).

Domenico Parisi, Stefano Nolfi, and Federico Cecconi. 1991. Learning, Behavior and Evolution. *Toward a practice of autonomous systems* (1991).

Pavel Petrovic. 1999. Overview of Incremental Approaches to Evolutionary Robotics. *Norweigen University of Science and Technology* (1999).

Matt Quin, Lincoln Smith, Giles Mayley, and Phil Husbands. 2002. Evolving Teamwork and Role-Allocation with Real Robots. *Artificial Life VIII Proceedings, MIT Press* (2002).

Stuart J. Russel and Peter Norvig. 2010. *Artificial Intelligence : A modern approach*. Pearson Education.

Robert J. Schalkoff. 1997. *Artificial Neural Networks*. McGraw Hill.

Gene Sher. 2012. *Handbook of Neuroevolution Through Erlang*. Springer.

Pieter Spronck, Ida G. Sprinkhuizen-Kuyper, and Eric O. Postma. 2004. Evolutionary Learning of a Neural Robot Controller. (2004).

Kenneth O Stanley and Risto Miikkulainen. 2002. Evolving Neural Networks through Augementing topologies. *Evolutionary Computation* (2002).

Mattias Wahde. 2007. Evolutionary Robotics : The use of Artificial Evolution in Robotics. *A tutorial presented at Ro-Man* (2007).

Markus Waibel, Laurent Keller, and Dario Floreano. 2009. Genetic team composition and level of selection in the evolution of cooperation. *Evolutionary Computation, IEEE Transactions* (2009).

Michael Woodridge. 2009. *An Introduction to Multiagent Systems*. John Wiley and Sons, Inc.