UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

# COMPUTER SCIENCE HONOURS
# FINAL PAPER
# 2015

Title: Adaptive Sensory Configurations in Robot Technologies [Co-Evolution]

Author: Naeem Ganey

Project Abbreviation: ASCiRT

Supervisor: Dr Geoff Nitschke

| Category | Min | Max | Chosen |
|---|---|---|---|
| Requirement Analysis and Design | 0 | 20 | 0 |
| Theoretical Analysis | 0 | 25 | 0 |
| Experiment Design and Execution | 0 | 20 | 17 |
| System Development and Implementation | 0 | 15 | 10 |
| Results, Findings and Conclusion | 10 | 20 | 18 |
| Aim Formulation and Background Work | 10 | 15 | 15 |
| Quality of Paper Writing and Presentation | 10 | | 10 |
| Adherence to Project Proposal and Quality of Deliverables | 10 | | 10 |
| Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*) | 0 | 10 | |
| **Total marks** | | **80** | |

# Adaptive Sensory Configurations in Robot Technologies

## [Co-Evolution]

Naeem Akbar Ganey
University of Cape Town
Cape Town, South Africa
naeem@ganey.co.za

## ABSTRACT

This work demonstrates a comparison between three different approaches in Evolutionary Robotics. A robot agent is made up of a controller and morphology, a body and a brain. Machine learning techniques can be used to model a (1) controller for a fixed morphology, (2) model a morphology for a fixed a controller and (3) simultaneous modelling of controller and morphology.

This work concentrates specifically on the Co-Evolution of controller and morphology of a robot agent and aims to conduct a comparison between **Morphology Evolution**, and **Co-Evolution** strategies for minimum, medium and maximum levels of cooperation between robot agents in a foraging task.

## 1. INTRODUCTION

A challenge in the design of robots in multi-agent robot teams is one of complexity. Human designers have to anticipate the behaviour of robots and design the logic of a robot controller accordingly. As the environment and interactions between robot agents become more complex so does the design of a robot. *Evolutionary Robotics* [13] assists in the design of robot agents. Robot controller configuration is placed under evolutionary control to find an optimal solution. Another advantage of Evolutionary Robotics is its robustness to design robots to a changing environment.

Another challenge in Evolutionary Robotics is to find an optimal morphology configuration for a given controller (or to find an optimal controller for a given morphology). This research aims to co-evolve morphology **and** controller simultaneously and measures its effectiveness by comparing it to the morphology-only approach.

### 1.1 Artificial Intelligence

Artificial Intelligence [16] is the intelligence of machines and software. These machines and software can perform tasks that would normally require human intelligence. Good Old Fashioned AI (GOFAI) [7] was the dominant AI technique until the 1980's. Due to the increasing complexity of tasks and the increase in the performance, Machine Learning and Biologically Inspired AI have become the dominant Artificial Intelligence approaches.

This phenomenon of Machine Learning, Biologically inspired Artificial Intelligence and Evolutionary Computing [5] complements the research done in Evolutionary Robotics. Robots were traditionally designed by hand and taught how to complete a task by defining actionable steps for each state in a discrete manner. Evolutionary Robotics allows for a learning algorithm to determine the behaviour of an autonomous mobile robot. This research furthers the use of Evolutionary Algorithms [1] in Robotics by placing the controller **and** morphology under evolutionary control.

### 1.2 Machine Learning

Machine learning [11] is an Artificial Intelligence technique which borrows from the disciplines of Statistics and Computer Science. Machine Learning algorithms learn from large amounts of data. Machine learning techniques have proven to be resistant to noise, to find solutions that would not normally have been found by a human and allows the machine to adapt to new environments. There are three Machine Learning approaches : (1) Supervised Learning, (2) Unsupervised Learning and (3) Reinforcement Learning.

#### *1.2.1 Supervised Learning*

In the Supervised Learning Approach, the system is provided with a set of input and correct outputs. Using this data, the system defines a model to accurately predict the output of a new (unseen) input. The data is typically split into training and testing data. Training data is used to train the learning algorithms and testing data (which is not used in training) is used to test the correctness of the model built by the learning algorithm.

#### *1.2.2 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.

#### *1.2.3 Reinforcement Learning*

Reinforcement Learning [2] is a learning mechanism where an agent is trained by means of reinforcement - a reward or punishment. An agent executes an action $a$ in a state $s$ maximise reward $r$.

### 1.3 Artificial Neural Networks

Artificial Neural Networks are very common in Machine Learning algorithms to represent logic. Artificial Neural Networks are inspired by the central nervous systems (network of neurons) found in the animal brain.

#### *1.3.1 Structure of an ANN*

ANN's consists of neurons and links between these neurons (called *synapses).* Neurons are organised in layers : the input layer, the output layer and one or more hidden layers in between. ANN's 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 optimise. The general structure of an ANN is one where every Node in layer $n$ is connected to every other Node in layer $n+ 1$. The output of each neuron is modulated by an activation function and fed into all the neurons connected to the current neuron. A fully connected ANN with one hidden layer has proven to be a universal function approximator [4].

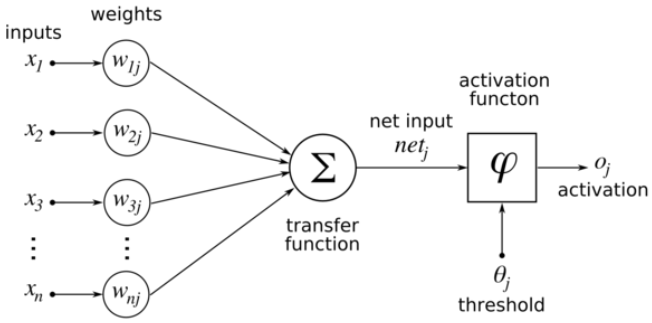


**Figure 1: General structure of an ANN**

### 1.3.2 Weight Adaptation

An Artificial Neural Network (ANN) is considered to be trained when a desirable out put 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.

### 1.3.3 ANN for Robot Controller

An ANN is suitable for modelling the controller of autonomous mobile robot. Nolfi et al. [14] explains the appropriateness of using an Artificial Neural Network for robot controller's due to the ANN's resistance to noise.

Waibel et al. [18] uses an ANN in a robotics application - the input nodes represented the sensory readings and the output nodes drove the right and left motors.

## 1.4 Evolutionary Algorithms

Evolutionary Algorithms [1], a subfield of Evolutionary Computing [5] have been inspired by 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 individuals 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.

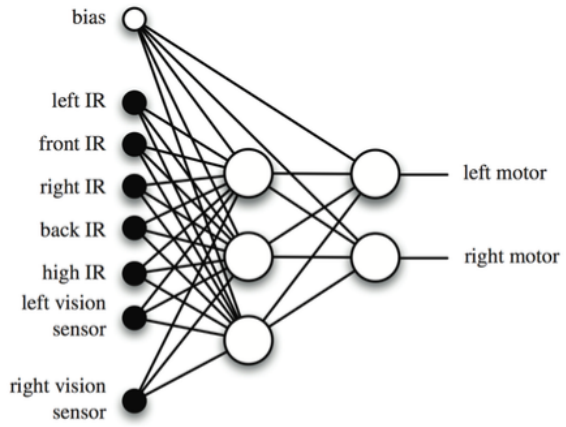


**Figure 2: Waibel's [18] use of an ANN in Robotics**

INITIALISE population with random candidate solutions
EVALUATE each candidate
REPEAT UNTIL (TERMINATION CONDITION is satisfied)

1. SELECT parents
2. RECOMBINE pairs of parents
3. MUTATE the resulting offspring
4. EVALUATE new candidates
5. SELECT individuals for next generation

**Figure 3: The pseudo-code of a general EA**

### 1.4.1 Neuro-Evolution

Neuro-Evolution [17] adapts the weights of an Artificial Neural Network by using an Evolutionary Algorithm. Rules of thumb can be applied to determine the number of hidden layers and the number of neurons in each layer or they can also be placed under evolutionary control. NEAT [17] is a machine learning technique to evolve the weights **and** topology of an ANN. As in the work by Hewland [9], because of the constant change in structure of an ANN due to the adding and removing of sensors, NEAT-M (Neuro-Evolution of Augmented Topologies and Morphologies) is used in the co-evolution of robot controller and morphology.

ANN + Evolutionary Algorithm = NeuroEvolution

## 1.5 Autonomous Mobile Robots

Mondada [12] speaks of the use of autonomous mobile robots which can mimmic the behaviour of living biological creatures. Autonomouxs Mobile robots can be used to do work normally required to by a humam (Amazon Robots). The use of autonomous mobile robots are popular in foraging tasks. Autonomous Mobile robots are fitted with sensors which are used as input into the learning algorithm, a complete learning algorithm will take in sensory information and drive the motors of an autonomous mobile robot in the right direction.

### 1.5.1 Khepera III Robot

The Khepera Robot [3] by the K-Team Corporation is used particularly in evolutionary robotics in many works including [9], [9], [12], [14] and [19]. The default Khepera configuration is fitted with 8 infrared sensors and has 2 wheels that drive the Khepera Robot and is shown in Figure 1. A simulated version of the Khepera Robot was used in the works by Hewland et al. [9] and its parameters are summarised in Table 1.

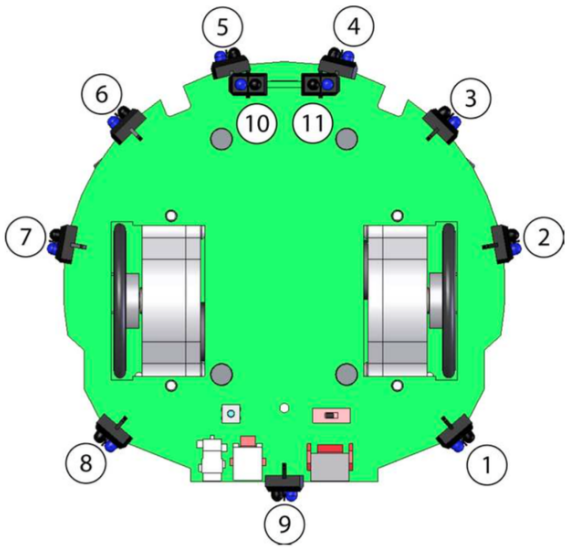


**Figure 4: Default morphology of a Khepera Robot**

| | |
|---|---|
| **Body diameter** | 30 cm |
| **Mass** | 0.7kg |
| **Wheel diameter** | 6cm |
| **Engine torque** | 0.075nm |
| **Maximum speed** | 0.5m· $s^{-1}$ |

**Table 1: Physical Dimensions of the Khepera Robot**

### 1.5.2 Robot Sensors

1. **Infrared (IR) Sensors**
   An infrared sensor's advantage is that it can sense objects close to the robot but cannot see very far and its field of view is limited. The advantage is that infrared sensors are generally cheaper than Ultrasonic or Long Range Infrared Sensors in terms of cost, weight and battery consumption.
2. **Ultrasonic Sensors**
   An ultrasonic's sensor can sense object far from it, but the sensor is unable to sense objects that are very close to it. The advantage of Ultrasonic sensors is that it has wide field of view.
3. **Long-Range Infrared (IR) Sensors**
   Long range infrared sensors have a very narrow field of view but are able to see very far and can sense objects that are far away from the robot.

Each of the three above mentioned sensors have their own strengths and weaknesses which complement each other. It would be interesting to see which combinations of the three sensors the learning algorithm finds to be fit. Sensors of different types need to work together to be effective as possible. For example, on one of the solutions found by the learning algorithm, an ultrasonic sensors and infrared sensor are implemented on the same position. The ultrasonic sensor can sense objects far and in a wide field of view and required an infrared sensor to sense objects that are very close to the robot.

### 1.5.3 Sensor Parameters

1. **Range**
   The distance a robot sensor can sense an object.
2. **Field of View**
   The value of the angle between the 2 edges.
3. **Bearing**
   The positioning of the sensor on the robot agent.
4. **Orientation**
   The anglular offset between the perpendicular line drawn from the centre of the circle and the centre of a sensor's vision

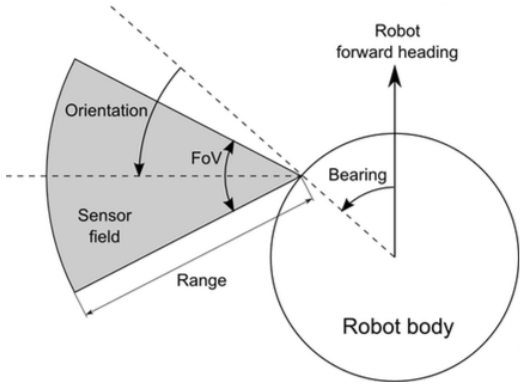


**Figure 5: Sensor parameters**

## 1.6 Evolutionary Robotics

Because of the very popular use of Evolutionary Algorithms in robotics applications. Evolutionary Robotics [13] has become a popular term to describe the intersection of these two fields of study. A robot agent consists of a controller and morphology (body and brain) and there are 3 different evolutionary robotics approaches : Morphology-only evolution, (2) Controller-only evolution and (3) Co-Evolution of Controller and Morphology.

### 1.6.1 Morphology evolution

Morphology evolution is the evolution of morphologies only, leaving the controller of a robot constant. Robot morphologies evolution is most commonly done by using a genetic algorithm. Robot configurations are represented in a string of bits and the genetic algorithm finds the best solution. The controller in the Morphology-only approach is a set of heuristics.

### 1.6.2 Controller evolution

Controller evolution looks at leaving the morphology constant. The controller adapts to the sensors that are configured on the autonomous mobile robot. The evolutionary algorithm finds a solution that is optimal to a specific robot morphology.

### 1.6.3 Co-Evolution

The beauty of evolutionary algorithms and machine learning in robot applications is that it does not require a design by hand. Machine learning techniques have show to outperform hand-designs in many applications. The machine learning algorithm can search through a large space and does not rely on intuition. Co-Evolution algorithms are useful because they can be used to easily adapt to a new environment, should a robot encounter a different environment - it can adapt without the need to redesign each robot for each environment.

The controller and morphology are both critical elements to the success of a robot's task. With a suboptimal controller or morphology - the fitness of a robot may be bad. Co-evolution is a technique to find a solution for both the controller and the morphology of a robot. Co-evolution in evolutionary robotics has been successful in the work by [9] and [10] and have outperformed a controller evolution only approach.

Hewland [9] uses an Artificial Neural Network as a controller and NEAT [17] as the machine learning method to evolve the ANN. The beauty of using NEAT is that the topology and structure of the ANN can be changed. Adding or removing a sensor requires that the number of input neurons into the ANN has to change. Hewland [9] developed a new method - called NEAT-M (Neuro Evolution for Augementing Topologies and Morphologies).

## 1.7 Multi-Agent Teams

Multi-agent systems [20] is a system that involves more than one individual robot agent. The agents can either be competitive or cooperative [6]. 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 [15] 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

In the application of this paper, autonomous mobile robots need to work together to push boxes to a designated home area. Without the element of cooperation - the boxes are too heavy for an individual robot to carry the box on its own and will require the assistance of the other robots.

## 1.8 Collective Behavior

Multi-Agent learning can be extended to many tasks, an important one is the foraging task. Multiple robots roam in a simulated environment to pick up resources and transfer them to a home area. In an experiment by Waibel et al. [18] - square robots push circular objects into a designated area.

# 2. METHODS

Neuroevolution of Augmented Topologies [17] is an evolutionary approach to evolve the weights and topology of an Artificial Neural Network. The NEAT-M [9] was the approach taken to evolve the weights, topologies and morphology of an Artificial Neural Network representing robot configuration and behaviour.

## 2.1 Simulator

A simulator was used to simulate the environment, the resources and the robots. The simulated environment allowed for experiments to be conducted efficiently.

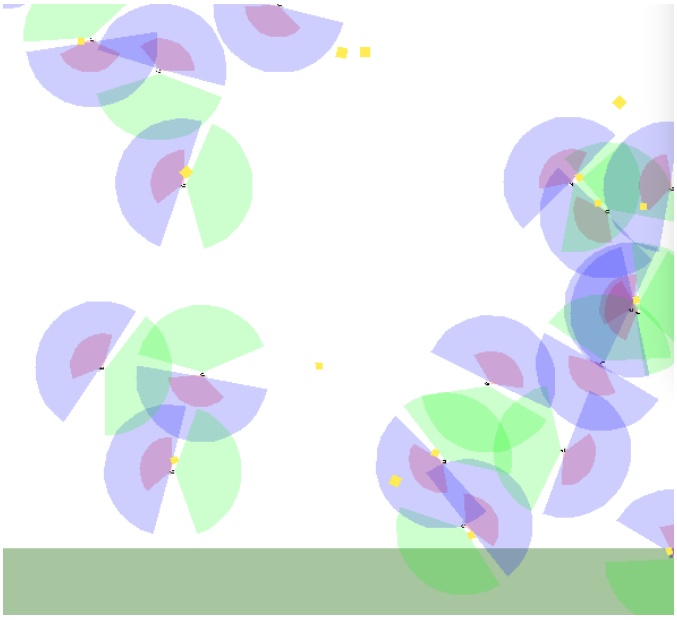

**Figure 6: Graphic Display of the simulator**

The robots are represented by the black circular shape. The sensors attached to the robot can be seen by the colours blue, red and green which are the long range IR sensors, infrared sensor and ultrasonic sensor respectively. Boxes are represented by yellow squares and the home area where boxes need to be dropped off are on the southern end of the simulated environment (See Table 2).

### 2.1.1 Box Resources

There are 3 different sizes of boxes in the environment : small, medium and large. Robots attach themselves to boxes

| Component | Dimensions (cm) | Mass (kg) |
|---|---|---|
| **Environment** | 20 x 20 | - |
| **Target area** | 20 x 4 | - |
| **Small resource** | 40 x 40 | 1 |
| **Medium resource** | 60 x 60 | 2 |
| **Large resource** | 80 x 80 | 3 |

**Table 2: The simulator component dimensions**

| Parameter | Valid Range |
|---|---|
| Infrared Sensor Range | 0.0 / 4.0 |
| Ultrasonic Sensor Range | 0.2 / 3.0 |
| Long Range IR Sensor Range | 0.0 / 6.0 |
| | |
| Infrared Sensor Field of View | 0.0 / $\pi$ Radians |
| Ultrasonic Sensor Field of View | 0.0 / $\pi/2$ Radians |
| Long Range IR Sensor Field of View | 0.0 / $\pi/3$ Radians |
| | |
| Infrared Sensor Orientation | 0.0 / $\pi$ Radians |
| Ultrasonic Sensor Orientation | 0.0 / $\pi$ Radians |
| Long Range IR Sensor Orientation | 0.0 / $\pi$ Radians |
| | |
| Infrared Sensor Bearing | 0.0 / $2\pi$ Radians |
| Ultrasonic Sensor Bearing | 0.0 /$2\pi$ Radians |
| Long Range IR Sensor Bearing | 0.0 /$2\pi$ Radians |

**Table 3: Range Parameters**

and push the box into the bottom segment of the environment. Small boxes require 1 robot to be pushed, medium boxes require 2 robots to be pushed and large boxes require 3 robots to be pushed. When a box has been pushed into the home area - the box disappears and the fitness for the environment is updated.

To encourage cooperation between robot agents, the robot team is rewarded twice as more for pushing a medium sized box in the home area and three times as much for pushing a large box into the home area.

## 2.2 Sensors

The 3 sensors defined for the robot team are : (1) Infrared Sensors, (2) Ultrasonic Sensors and (3) Long rage infrared sensors. Each sensor has its own unique strengths and the objective of the learning algorithm is to select the best combination of sensors to maximise fitness.

In addition to the selection of sensors, the learning algorithm searches the solution space for the correct parameters of each sensor. The parameters being evolved are **range**, **field of view**, **bearing** and **orientation**. The valid ranges for each of these parameters for their respective sensors are summarised in Table 3.

## 2.3 The Task

The task involved is for robot agents to work together, sense objects in the environment, attach themselves to a box and drive the box to the home area. Rewards are assigned to the robot team for the number of boxes collected and for efficient completion of the task.

| Parameter | Value |
|---|---|
| Population Size | 100 |
| Truncation Selection | Top 30% |
| Elitism | 30% |
| Generations | 250 |
| Simulation Runs (Epochs) | 3 |
| Experiment Runs | 15 |
| | |
| Infrared Sensor chance of selection | 40% |
| Ultrasonic Sensor chance of selection | 30% |
| Long Range IR Sensor chance of selection | 30% |

**Table 4: NEAT-M Parameters**

## 2.4 Controller Evolutions

### *2.4.1 NEAT*

ENCOG, a java-based machine learning platform contains an implementation of NEAT. The Encog library's implementation of NEAT is based on the paper by Stanely et al. [17].

The parameters for the NEAT implementation as synonymous in the following works [9], [17], [9] and the default ENCOG parameters for NEAT. The parameters are summarised in table 2.

### *2.4.2 NEAT-M*

NEAT-M [9] is an extension of the *Encog* library's implementation of NEAT. The adding and removal of sensors affects the structure of the Artificial Neural Network. NEAT implementation add and removes nodes with a probability of $p$. Because of the relative costs of the 3 sensors implemented in terms of cost, size and energy consumption. The sensors were selected with the following probabilities summarised in Table 4

## 2.5 ANN for Robot Controller

As in the works by Waibel et al. [18] and to support the implementation of NEAT-M and the Khepera robot, an ANN has been used to represent the configuration of the robot. Sensory readings are fed into the ANN through the input nodes and the ANN will compute a corresponding value for the 2 wheels of the Khepera Robot. The Khepera robot is driven by 2 wheels - left and right. The values applied to each of the wheels of the robot is in the range $-1 <= x <= 1$.

## 2.6 Guidance Heuristics

The following guidance heuristics were add to the robot controller. The robot's actions were determined by *schduler*, the scheduler would either delegate robot controller to the ANN or use a guidance heuristic.

- When a robot is close to an object - the robot automatically attaches itself to the resource object.
- When a robot is attached to an object, the guidance heuristic implemented was to drive the resource object to the home area.
- When a robot is attached to an object and the object cannot be pushed to the home area (because it requires

another robot's assistance) - it will wait for 5 seconds, and if the 5 seconds has elapsed it will detach itself from the resource.

- When a box that requires 3 robots to push has one robot attached to it, when a second one joins, the timer is reset and the two robots will wait for another 5 seconds for a third robot agent to assist with pushing the box the home area.

# 3. EXPERIMENTS

There are three experiments that will be configured, one will show the effectiveness of co-evolution on **minimal** levels of co-operation between bots, the other **medium** level of co-operation and the final will be one with **large** level of co-operation. The experiment configurations are summarised in Table 5.

| # | Level of Co-Operation | Small | Medium | Large |
|---|---|---|---|---|
| 1 | Minimum | 10 | 5 | 0 |
| 2 | Medium | 5 | 5 | 5 |
| 3 | Maximum | 0 | 5 | 10 |

**Table 5: Experiment Configurations**

This research aims to conduct a **comparison between Morphology Evolution and Co-Evolution strategies** for minimal, medium and maximum cooperation between robot agents in a foraging task.

The Morphology Evolution machine learning algorithm has defined a set of guidance heuristics to control the robot and the morphology (the sensors and its parameters) are evolved. The Machine Learning Genetic Algorithm aims to find a suitable Robot Morphology to maximise fitness for the task. The Co-Evolution, as stated above, will evolve the controller and morphology of a robot agent.

The experiments conducted will test the effectiveness of the two different approaches for the cooperative foraging task. A total of 3 experiments have been conducted. To achieve more consistent results, each experiment was run 15 times, each generation in the Evolutionary Algorithm was run 3 times and there were a total of 250 generations. The Evolutionary Algorithms for the Morphology Evolution and Co-Evolution were Genetic Algorithms and NEAT-M [9] respectively.

Morphology Evolution and Controller Evolution experiments were run in the simulated platform for a total of 10 000 iterations. The fitness calculated for each generation are calculated by the number of boxes returned to the home area and a bonus is given to the robot team for collecting all boxes in a short time. Because of the complexity of collecting medium and large sized boxes - the robot team are rewarded twice as much for collecting a medium and three times as much for collecting a large box compared to the reward collecting a small box.

The experiments were conducted on total of 20 computers, each running multiple experiment runs. The *Encog* Machine Library allowed the experiment runs to run in multithreaded mode and each computer was able to utilise each of its 4 cores to speedup the experimental process.

# 4. RESULTS

## 4.1 Results from the Co-Evolution Approach

The experiments for the co-evolution approach was completed after a period of 4 days. The average fitness at each generation for each of the experiments were obtained by finding the mean of the fitness at that generation across the 15 simulation results. The average fitness for each generations of the 3 experiments (Minimal, Medium and Maximum Co-operation) are plotted in a line graph. The Blue, Red and Green lines represent the minimal, medium and maximum experiments respectively.

It can be seen that each of the experiments line graphs are very similar to that of a logarithmic graph, there is much increase in fitness in the early generations and the graphs reach a plateau and converge to an optimal solution. A normality and log-normal test was conducted for each of the experiments. A Kolmogorov-Smirnov Test was used to test for normality. The KS-Test found experiments 1, 2 and 3 to be consistent with the normal distributions with the following $p$ values : 0.94, 0.89 and 0.92 respectively. Using a significance level of 5%, the results for experiments 1, 2 and 3 are **not consistent with the normal distribution**. However, because there are only 15 results in the dataset for each experiment, we can expect the results to be normally distributed if there is a significantly larger dataset.

It can also be seen that the co-evolution approach is more efficient with minimal levels of cooperation and the fitness level decreases as the required level of cooperation increases. Because of the bonus given to robot teams for collecting all the boxes in a shorter time, the small experiments thrives on this and therefore achieves a higher fitness.

Each Generation and its best genome is simulated for a total of three simulations. From this, the best and average results for that generation is recorded. It can be noted on the Box Plot on Figure 8 that the results for the Best results are obviously higher but the average results are more normally distributed, extreme outliers are eliminated and therefore the average results are more reliable. An outlier may exist due to a convenient random spawning of robots and the placement of objects - a simulation can achieve an unreliable results and hence the need for 3 simulation runs.

To achieve a more accurate and critical analysis of results - the average (and not the best results) will be used.

## 4.2 A Comparison between Co-Evolution and Morphology Evolution

The Crux of this paper is to conduct a comparison between Co-Evolution and Morphology Evolution for various levels of cooperation in a robot team to complete a foraging task.

The minimum and maximum for each of the experiments for Co-Evolution and Morphology Evolution are summarised in Table 6. The comparisons conducted are :

1. Co-Evolution vs Morphology Evolution : Min Cooperation
2. Co-Evolution vs Morphology Evolution : Med Cooperation

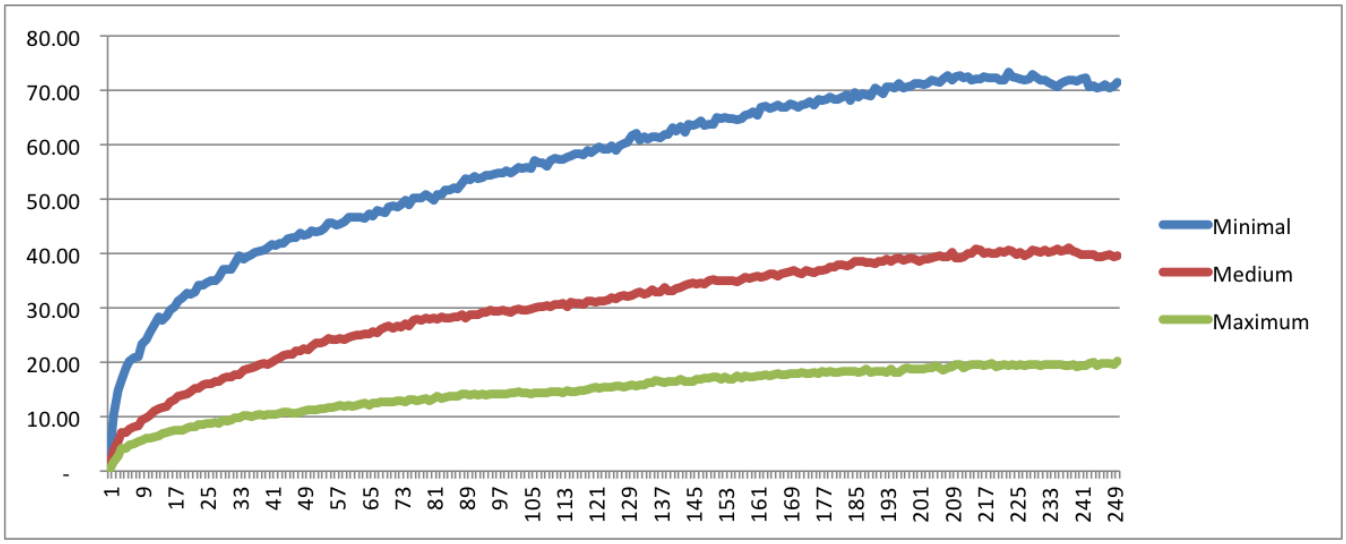


**Figure 7: All Sim Line graph**

3. Co-Evolution vs Morphology Evolution : Max Cooperation

#### 4.2.1 *Comparison #1 : Minimum Cooperation*

The Co-Evolution approach is compared to the Morphology Evolution for minimal cooperation. The Kolmogorov-Smirnov Test found both datasets to be inconsistent with the normal distribution at a significance level of 5%. Because both datasets are not normally distributed, the KS-Test was used again to test for a statistically significant difference between the two datasets. The KS-Test found the datasets to be statistically different with probability $p < 0.001$.

The Co-Evolution approach has outperformed the fitness of the morphology evolution for the minimal cooperation experiment. The maximum fitness recorded Co-Evolution Approach is 97.87 and the Maximum Fitness recorded for the Morphology Evolution is 51.29.

**Co-Evolution of Controller and Morphology achieves a higher fitness compared to Morphology Evolution in a minimal cooperation foraging task.**

#### 4.2.2 *Comparison #2 : Medium Cooperation*

The Co-Evolution approach is compared to the Morphology Evolution for medium cooperation. The Kolmogorov-Smirnov Test found both datasets to be inconsistent with the normal distribution at a significance level of 5%. Because both datasets are not normally distributed, the KS-Test was used again to test for a statistically significant difference between the two datasets. The KS-Test found the datasets to be statistically different with probability $p < 0.001$.

The Co-Evolution approach has outperformed the fitness of the morphology evolution for the minimal cooperation experiment. The maximum fitness recorded Co-Evolution Approach is 68.69 and the Maximum Fitness recorded for the Morphology Evolution is 24.15.

**Co-Evolution of Controller and Morphology achieves a higher fitness compared to Morphology Evolution in a medium cooperation foraging task.**

#### 4.2.3 *Comparison #3 : Maximum Cooperation*

The Co-Evolution approach is compared to the Morphology Evolution for maximum cooperation. The Kolmogorov-Smirnov Test found both datasets to be inconsistent with the normal distribution at a significance level of 5%. Because both datasets are not normally distributed, the KS-Test was used again to test for a statistically significant difference between the two datasets. The KS-Test found the datasets to be statistically different with probability $p < 0.001$.

The Co-Evolution approach has outperformed the fitness of the morphology evolution for the minimal cooperation experiment. The maximum fitness recorded Co-Evolution Approach is 33.30 and the Maximum Fitness recorded for the Morphology Evolution is 10.77.

**Co-Evolution of Controller and Morphology achieves a higher fitness compared to Morphology Evolution in a maximum cooperation foraging task.**

## 5. DISCUSSION

In addition to the co-evolution outperforming morphology evolution in the maximum fitness, it also outperforms the morphology evolution in minimum and average fitness for each of the experiments conducted. Co-Evolution has shown to be the superior Evolutionary Robotics technique for minimal, medium and maximum levels of cooperation.

A major motivation for using Evolutionary Robotics to design controllers and morphologies for robot agents is because it does not require the controller and morphology to be designed by hand. The quality of the morphology evolution approach relies heavily on the skills of the designer who programmes the heuristics enabling the robot agent to move and complete a task. Because the heuristics are designed specifically for a particular environment, morphology evo-

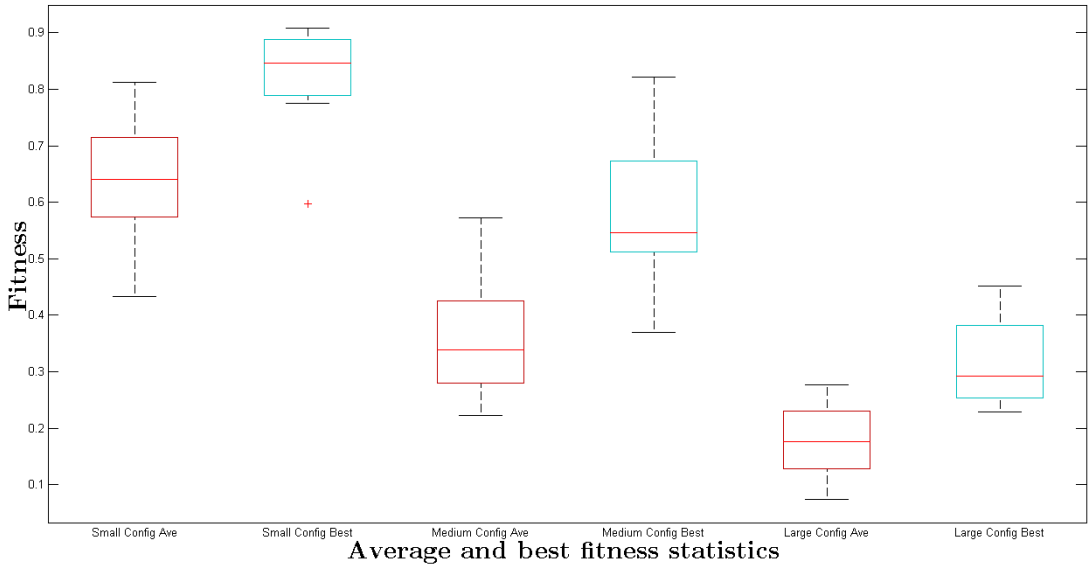


**Figure 8: Average and Best fitness of the Co-Evolution Approach Experiments**

| | Morphology Evolution | | | | Co-Evolution | | |
|---|---|---|---|---|---|---|---|
| | Small | Medium | Large | | Small | Medium | Large |
| Minimum | 28.130841 | 7.829699 | 0.736246 | | 51.983333 | 26.67789 | 11.466667 |
| Maximum | 51.292835 | 24.153686 | 10.771028 | | 97.518467 | 68.691151 | 33.308333 |
| Standard Deviation | 6.31070214 | 4.64953286 | 3.07774354 | | 11.8338345 | 11.79189 | 6.65126441 |
| Normality Test ($p$) | 0.57 | 0.9 | 0.68 | | 0.94 | 0.89 | 0.92 |
| Log-Normality Test ($p$) | 0.82 | 0.71 | 0.49 | | 0.76 | 0.97 | 0.98 |

**Table 6: The KS-Test was conducted to check for normality in the distributions of the fitness statistics for the two approaches.**

lution is not robust as it may not have heuristics defined for the new environment. Conversely, the Co-Evolution approach is more robust as it can learn the new environment and adapt its controller and morphology configuration in accordance with the new simulation configuration.

It can be seen in Figure 7 that the Evolutionary Algorithms start to converge (the line graph starts to plateau), Machine Learning theory states that long runs are not useful as once convergence is reached, then no new solutions can be found. Does the research need to consider structural changes to the Artificial Neural Network and Evolutionary Algorithm to find new solutions. For example, low levels of mutation in the Evolutionary Algorithm will converge towards a local maximum and lose sight of the global maximum.

## 5.1 Challenges

Due to constraints in time and resources, this research was unable to run very long simulations (500+ generations) to evaluate if the Co-Evolution Algorithm can escape a local minimum and converge towards a global minimum.

Running each generation for more simulation runs (5+) will avoid the machine learning algorithm finding a good solution due to a random coincidence of the placement of objects and spawning of robots and can therefore be a more accurate result. This was not complete due to time and resource constraints.

It would be useful to test the third Evolutionary Robotics approach - the controller evolution but the third member of the ASCiRT Group had limited resources and was unable not run a simulation for a environment similar to the one configured in this paper.

The High Performance Clusters, *Hex* in South Africa and *Lisa* in Japan was complex to configure and when the configuration was completed, the jobs would have taken too long to complete and the team resorted to running all the experiments across 20 computers in a Computer Lab.

A major challenge of this paper was to extend the Simulator and Experiment platform on which this research was built. Code was not well documented which made it a challenge to understand the code in order to extend it. Furthermore, the code was not designed in order for it to be extended.

The JBox2D physics engine is not very realistic to the real

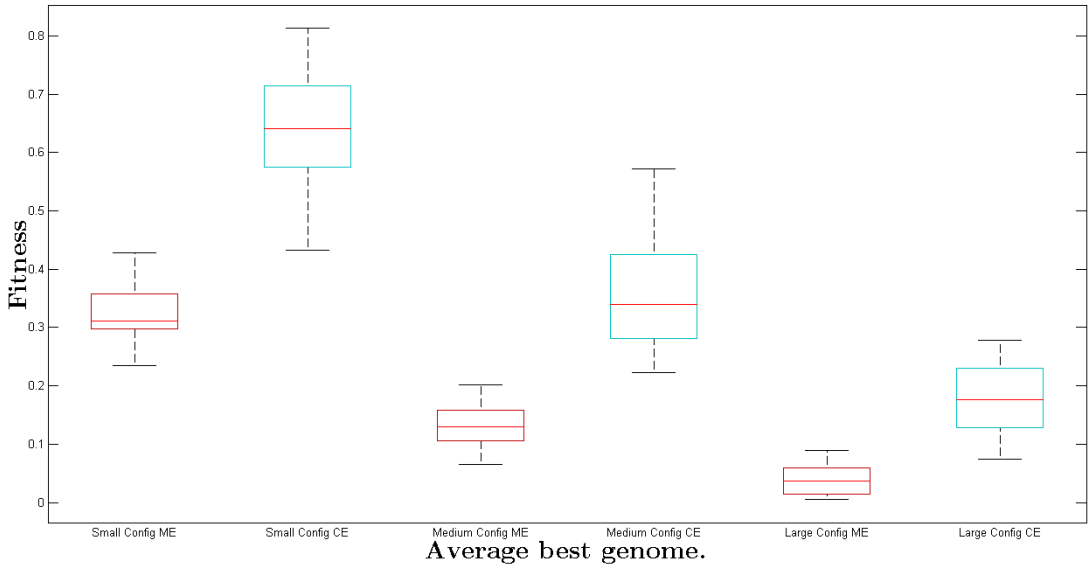


**Figure 9: Co-Evolution vs Morphology Evolution : A boxplot of results**

world simulation. For example, robots crashing into each other has no impact on the robot's movement.

## 5.2 Future Work

- Currently, there are only three sensors defined for a robot. More sensors, like a *High Definition CCR Camera*, can be added to the list of available sensors to the robot.

- Currently, there are only resources to be picked up by the robot and the robot is rewarded accordingly. The simulator platform can be extended to include objects that need to be avoided by the robot such that the robot will learn to collect the useful resources and avoid the dangerous ones.

- Currently, the robot team is composed of robots that share a common controller and morphology and this is referred to as a homogenous system. Another configuration which may yield better results is a heterogenous system, one where each robot in the robot team has a different controller. This research can be extended to conduct a comparison between homogeneous and heterogenous systems.

- Larger levels of cooperation between robot agents. Resources that require more than three robots to move can be added to the environment.

- The simulated platform doesn't cater for the battery of a robot agent. In the real world, robots have limited battery life and constantly require *recharging*. The platform can be extended to make the robot's lose charge and have a designated area in the simulated environment for robot to recharge.

- Another task that can be done with the same robots and environment is a collective construction task. Instead of the boxes disappearing after they have been placed in the home area, the robots are required to arrange them in a pre specified manner

- Refactoring the existing code to allow future researches to extend the code easily.

- In addition to comparing the fitness of Co-Evolution, Morphology Evolution and Controller Evolution, more experiments such as (1) speed of convergence and (2) adaptability of the controller and morphology to a new environment

- The experiments were limited to a circular robot with sensors around its perimeter - what if we let the machine learning algorithm also decided the size, shape and wheel position of the robot.

- Currently, the robot morphology is limited to a circular robot with sensors around its perimeter. The Machine Learning Algorithm can be extended to include size, shape and position of the wheels on a robot as part of the variables to find an optimal solution for

# 6. CONCLUSION

Evolutionary Robotics is a useful method for developing the controller and morphology for a robot agent. This paper has shown that placing the controller and morphology of a robot agent under evolutionary control finds better solutions than only placing only the morphology under evolutionary control. It has been shown in many other research papers such as [9] that the co-evolution of robot controller and morphology is more likely to find a good solution as a suboptimal controller for a fixed morphology or a suboptimal morphology for a fixed controller can lead to a suboptimal solution.

To further extend the research done in the Co-Evolutionary Robotics, research can be done in adapting morphology without the constraints of a robot with a specific size and shape - instead this should be determined by the evolutionary algorithm.

## 7. ACKNOWLEDGMENTS

I would like to extend my gratitude to Geoff Nitschke, Sonia Berman and Michelle Kuttel who have been excellent guides on this challenging journey. To Chuck and Jae, my project partners and to the ASCiRT Team of 2014 who have laid the groundwork for us to further their research.

## 8. REFERENCES

[1] T. Back. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* Oxford University Press, 1996.

[2] A. G. Barto. *Reinforcement learning: An introduction.* MIT Press, 1998.

[3] K. T. Corporation. Khepera iii specifactions.

[4] G. Cybenko. Approximations by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 1989.

[5] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing.* SpringerVerlag, 2003.

[6] D. Floreano. *Methods for Artificial Evolution of Truly Cooperative Robots.* PhD thesis, University of Lausanne Switzerland, 2009.

[7] J. Haugeland. *Artificial Intelligence: The Very Idea.* Massachusetts Institute of Technology, Cambridge, MA, USA, 1985.

[8] J. Hewland. Evolving robot morphologies in a collective foraging task using neat-m. Master's thesis, Univeristy of Cape Town, 2014.

[9] J. Hewland and G. Nitschke. The benefits of adaptive behavior and morphology for cooperation in robot teams, 2015.

[10] H. Lund. *Co-evolving Control and Morphology with LEGO Robots.* PhD thesis, University of Southern Denmark, 2003.

[11] T. M. Mitchell. *Machine Learning.* McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[12] F. Mondada. Mobile robots for a synthetic approach to interdisciplinary research, 2005.

[13] S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* MIT Press, 2000.

[14] S. Nolfi, S. Nolfi, D. Floreano, D. Floreano, O. Miglino, O. Miglino, F. Mondada, and F. Mondada. How to evolve autonomous robots: Different approaches in evolutionary robotics, 1994.

[15] P. Petrovic. *Overview of Incremental Approaches to Evolutionary Robotics.* PhD thesis, Norweigan University of Science and Technology, 1999.

[16] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Pearson Education, 2 edition, 2003.

[17] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evol. Comput.*, 10(2):99–127, June 2002.

[18] M. Waibel, L. Keller, and D. Floreano. Genetic team composition and level of selection in the evolution of cooperation. Evolutionary Computation, IEEE Transactions on, 13(3):648–660, 2009.

[19] J. Walker and S. Garret. Evolving controllers for real robots. Master's thesis, University of Wales, 2004.

[20] M. Wooldridge. *An Introduction to Multiagent Systems.* John Wiley and Sons, 2009.