Neuroevolution approach to multi-robot controller design for a collective construction task

Daniel Grindlay
University of Cape Town
Rondebosch 7701
Cape Town, South Africa
grndan010@myuct.ac.za

Josh Buchalter
University of Cape Town
Rondebosch 7701
Cape Town, South Africa
bchjos003@myuct.ac.za

Ruben Putter
University of Cape Town
Rondebosch 7701
Cape Town, South Africa
pttand010@myuct.ac.za

1. PROJECT DESCRIPTION

Neuroevolution (NE) is an effective machine learning approach to solving complex optimization and control problems [5] [10] [21] [20]. NE is based on applying Evolutionary Algorithms (EA) to Artificial Neural Networks (ANN) in order to find the best approximate solution to a problem [29]. It has been used in a number of real world applications with non-linear, dynamic environments where the details of the solution could not be specified beforehand [11] [12] [16]. One such challenging problem that NE has demonstrated its ability to solve is the automated controller design of multi-robot systems [6].

A multi-robot system (MRS) is defined as a group of robots that interact and cooperate with one another in the same environment to perform a specific task [7]. Multiple robots in a system are able to perform tasks that could not easily be achieved by a single robot. An MRS is often cheaper to implement, more robust to changes in the environment and not as susceptible to faults [1]. However there are challenges involved with designing the controllers for robots in an MRS. The coordination and interaction between entities creates additional complexity as well as a large and highly dimensional search space of possible solutions [15]. Designing controllers for robots that have to cooperate and show collective behaviour is a challenging task.

NE can be applied to the controller design of MRS because of its ability to find solutions in continuous, dynamic environments where the search space of solutions is large [11]. An advantage of NE is that the details of how a task must be accomplished does not need to be known or specified beforehand [25]. It is possible for NE to do this through the use of EAs and ANNs. EAs are a group of algorithms that use the principles of Darwinian evolution and natural selection to evolve a population of candidate solutions. The solutions are encoded as genotypes which map to a physical phenotype that specifies the behaviour of the solution. In NE the phenotype of a solution is in the form of an ANN. An ANN is a processing device that attempts to imitate the functions of a biological nervous system [8]. They consist of simple processing units called neurons that are divided into layers with connections between them allowing ANNs to be universal function approximators. ANNs that act as controllers for robots use inputs from sensors and produce outputs that drive the robot's behaviour [29]. The performance of the robot is evaluated using a fitness function giving an indication of how well the ANN carries out a specific task. EAs iteratively refine the population of solutions using the evolution mechanisms of recombination, mutation and selection in order to find an approximation of an optimal solution [29]. The aim of this project is to determine the most effective and robust NE approach to multi-robot controller design over a range of sensor configurations.

Being able to evolve a team of robots' controllers so that they successfully complete a collective construction task can lead to many real-world applications. Building structures for humans on a planet before they arrive to colonise it and building safety structures for people in storm-hit areas, that are inaccessible to rescue teams, are just two of the many possible use cases of collective construction tasks. Industry could be another frontier whereby teams of robots manufacture products. There are numerous applications for MRS that can only be performed sufficiently through the cooperation and coordination of multiple robots. Collective construction is a task where a system of robots could be used to build structures in an environment that is not suitable for humans [34].

Tasks like collective construction not only requires a complex, multi-faceted, non-trivial and possibly dynamic solution but also generates a severely complex and deceptive problem space. Such a problem space means that traditional gradient-based Machine Learning techniques are not capable of finding an optimal solution because the gradient information required is, almost always, unknown. The second issue of such a problem space is that there are many points of local optima which can trap the search for optimal solutions in a phenomenon called premature convergence. Another problem faced is the design of the robot teams. This includes choices on how much inter-robot communication can take place as well as designing the morphology (sensory configuration) of the robots themselves. Both of these design decisions could have major implications on the success of our experiments. Finally, time constraints are a challenge in itself. The time needed to evolve the simulation in order to assess a particular run's success is unbounded. Thus there will be a challenge in defining a cut off time that will allow success to be measured.

2. PROBLEM STATEMENT

It is incredibly challenging to design controllers to be used in a multi-robot system where cooperation and coordination are required. The system is far too complex for a designer to have to explicitly specify how a task must be performed by multiple robots. NE is one solution to the automated design of controllers but there is very little research comparing different NE methods. Our aim is to determine which NE approach out of a chosen three can be used to evolve controllers for robots which have to coordinate with one another.

Our research question is which chosen NE approach is the most effective and robust at evolving controllers for a multi-robot system that has to perform a collective construction task.

We define an effective NE approach as one that can evolve controllers for a multi-robot system, in a reasonable amount of time, which is able to demonstrate collective behaviour and successfully carry out their task. A robust approach is one which is capable of evolving controllers for a wide range of robot morphologies.

A comparison will be made between the effectiveness of multi-robot systems that have been evolved using three different NE approaches:

(1) Evolving controllers using NEAT with objective, novelty and hybrid search functions.

(2) Evolving controllers using HyperNEAT with objective, novelty and hybrid search functions.

(3) Evolving controllers using NEAT with multi-objective search optimisation.

3. PROCEDURES AND METHODS

Our project involves comparing three different NE approaches to evolving controllers for a multi-robot system. The robots will have to gather different sized blocks and arrange them together in specific sequences. This is a complex task with a large search space and will allow us to test the effectiveness of our chosen NE approaches at learning complex group behaviour.

The experiments will be implemented on an existing framework based on the MASON library which is programmed for multi-agent simulators. It provides a two-dimensional simulation environment in which the multi-robot teams are able to move and interact with predetermined construction blocks. The framework was implemented by UCT Honours students Naeem Ganey, Ntokozo Zwane and Jae Jang to investigate a similar multi-robot controller design topic. Each team member will have to implement their own selected NE method within their local version of the framework in order to set up their experiments. This includes implementing our chosen NE algorithm as well as the different search, fitness and objective functions to be used in each approach.

The simulated environment will consist of a team of homogeneous robots with a specific sensor configuration. 10 different configurations will be chosen beforehand in order to sufficiently test the robustness of our NE approaches with a diverse selection of morphologies. There will also be a number of different construction blocks. Every block is of a certain type and can only be attached to other blocks in predetermined sequences. The goal of the robot system is to gather the blocks, place them in a predefined building zone and construct them according to the specified sequences. Each robot is controller by an ANN which receives inputs from the robot's sensors.

Two different NE algorithms will be used in the project to evolve controllers. The first is Neuroevolution of Augmenting Topologies (NEAT) [31] and the second is Hypercube-based Neuroevolution of Augmenting Topologies (HyperNEAT) [30].

NEAT is a NE approach that evolves both the connection weights and topology of ANNs. It makes use of innovation indicators for historical information of nodes. NEAT ensures the protection of innovation through speciation and minimizes the structure of solutions by starting with a population of simple ANNs [29]. NEAT is capable of finding solutions to problems with highly dimensional search spaces and have been found to perform just as well or better than other NE approaches without having to specify the ANN topology beforehand [31]. The direct encoding method employed by NEAT allows for the construction of complex ANNs. The representations of the ANNs are dynamic and can be expanded with the addition of new neurons (nodes) and connections. Each genotype in the population consists of a number of node genes and connection genes. The node genes specify the input, output and hidden nodes that make up the network. Each connection gene contains information about a relationship between two neurons specifically the in-node, out-node, weight of the connection, an enable bit and innovation number (Figure 1).

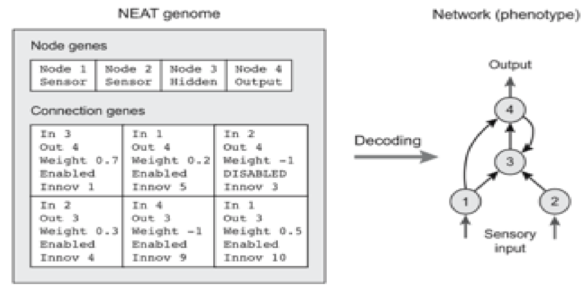

Figure 1: Example of NEAT encoding [8].

HyperNEAT is a NE method that implements a generative encoding scheme and evolves ANNs using the same principles of the NEAT algorithm[33] [32]. HyperNEAT attempts to abstract the process of natural development by using Compositional Pattern Producing Networks (CPPNs) to encode ANNs (Figure 2) [4]. This allows for the production of complex patterns by determining the phenotypic attributes of an ANN as functions of its geometric location[5]. The basis of CPPNs can be found in nature where it is possible to describe patterns as compositions of functions with each function representing a stage in development[5]. Each CPPN is a function that produces an output based on a specific input[3]. CPPNs allow for repeated structures in the genome to be represented by a single gene that gets reused when mapping from genotype to phenotype[30]. HyperNEAT has been shown to exploit intermediate problem regularity as modularity in multi-agent tasks[33] which makes it suitable for this collective construction task since the structures that are to be built are modular and regular[13].

Four different search functions will be implemented in the project to be used with NEAT and HyperNEAT. These are objective, novelty, hybrid and multi-objective search functions.

Objective search is the most straight forward approach. It uses a simple fitness function as an objective measure of a solution's performance and aims to improve the solutions by only selecting the most fit individuals for reproduction. A downside for objective search functions is that they often direct the search towards a dead end by getting stuck in a local optimum [19].

Lehman and Stanley proposed novelty search, rewarding solutions that exhibited vastly different (novel) behaviour

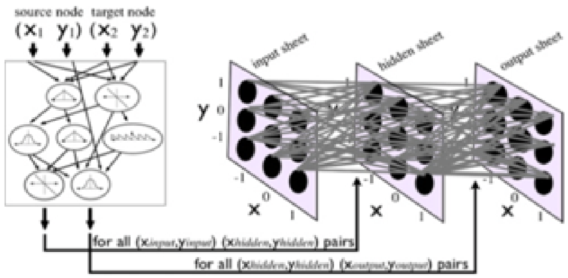

Figure 2: Example of HyperNEAT encoding [4].

from previous generations, effectively ignoring the objective/goal, as a means of avoiding premature convergence in deceptive problem domains [17]. By rewarding novel behaviour, diversity in the population of solutions is ensured and thus is a mechanism of diversity maintenance.

A hybrid search approach is a combination of objective and novelty searches where the next generation of solutions is chosen based on both search results. Half of the candidate solutions are chosen for how fit they are which is a measure of their performance of the given task and the other half is chosen based on how different they are from the rest of the population of solutions.

Multi-objective search algorithms are used to find the Pareto optimal solutions, a set of solutions that provide trade-offs between different objective functions [36]. Often complex tasks have multiple objectives that need to be simultaneously optimized but it is possible that objectives conflict with one another where better performance towards one leads to poor performance from another [9]. There are a number of ways that the fitness value in a multi-objective algorithm can be calculated including a weighted sum method [14] [28] and a vector based method [35].

Each of the three approaches will be implemented by a different team member and then evaluated based on the controllers it produces. The three approaches are:

(1) NEAT using objective, novelty and hybrid searches

(2) HyperNEAT using objective, novelty and hybrid searches

(3) NEAT using multi-objective search

Each of the search algorithms will be compared in terms of:

- How successfully the evolved phenotypes perform the collective construction task which includes time taken, blocks placed correctly and battery consumption.
- The probability that a successful solution can be found.
- How long it took the algorithm to find a solution that can satisfactorily perform the construction task.

4. ETHICAL, PROFESSIONAL AND LEGAL ISSUES

The simulator that will be used in this project was created by UCT Honours students Naeem Ganey, Ntokozo Zwane and Jae Jang using the MASON library for simulating multi-agent systems. The MASON library is a free open source project under a BSD license. The Intellectual Property policy of UCT is that UCT owns the copyright to all software produced during a project and that all papers must be produced under a Creative Commons license. There are no ethical or legal issues that have to be addressed.

5. RELATED WORK

It has been demonstrated that evolutionary algorithms can be applied to the design of controllers for multi-robot systems. Quin et al [26] evolved a controller for a system of robots that had to perform a movement tasks where they had to move away from a random starting position towards a target location without moving out of sensor range from one another. A conventional evolutionary algorithm was used to iteratively evolve the controller in a simulated environment before being applied to the physical robots. Nitschke et al [25] compare three different methods of co-evolving controllers for a team of simulated robots that have to perform a collective construction task. The simulated robots had to gather building blocks from their environment and attach them together in specific sequences in order to create structures. The three algorithms tested were Cooperative Co-Evolutionary Algorithm (CCGA), Multi-Agent Enforced Sub-Populations (MESP) and Collective Neuro-evolution (CONE). Each controller was made up of a recurrent ANN which received input from simulated sensors attached to the robots. It was found that CONE produced controllers with higher performance in the collective task.

Lehman and Stanley examined the performance of novelty search (in navigating a deceptive maze) and found that it was more effective than objective search [17] [27]. Risi et al. furthered the experiment by creating a maze system with tunable deception and compared the performance of NEAT with objective based and novelty search implementations. They found that the novelty search implementation of NEAT outperformed (in terms of number of evaluations required to find a general solution) objective based NEAT conclusively (needing on average of 48,235 evaluations as compared with 90,575 needed for objective based NEAT) [27]. Moriguchi and Honiden also found that combining novelty search with NEAT's niching scheme increased the NEA's performance for problems with large search domains and perceptual noise [22]. Notable implementations of novelty search in Evolutionary Robotics have been summarised by Mouret and Doncieux [24]:

- Navigation of a maze that includes an attractive dead-end in the direction of the goal by a wheeled robot with distance measured by the spatial distance between end points of each trial [17] [19] [23].
- Biped robot tries to walk as quick as possible with behaviour specified by the offsets of the centre of mass of the biped at one second intervals during the task. Distance is measured by Euclidean distance between the offsets [19].

- Wheeled robot tries to find a reward in a Maze by evolving its plastic nodes with behaviour evaluated as a vector of crashes and rewards per trial; Distance was measured by the Euclidean distance between the vectors [27].

In order to investigate the performance of HyperNEAT in evolving multi-agent behaviour, D'Ambrosio et al. conducted several predator-prey experiments in which HyperNEAT was used to evolve controllers for homogeneous and heterogeneous robot teams. In this task, the team of predators have to round up a team of prey and coordinate their behaviour in order to avoid pushing the prey away from each other. It was found that HyperNEAT was able to successfully encode behaviours that contributed to wards superior behaviour[5]. When investigating methods evolving controllers for coordinated quadruped gaits, Clune et al. compared the performance of HyperNEAT and FT-NEAT. Their results show that HyperNEAT was able to vastly outperform FT-NEAT in each generation in fitness and that HyperNEAT has a greater advantage during the early stages of evolution[3]. It was concluded that HyperNEAT's success was due to its ability to exploit the symmetry of a quadruped's gait cycle.

6. ANTICIPATED OUTCOMES

Our first anticipated outcome is to have a well configured simulation environment for each of our experiments. As mentioned before, this is done by taking the framework that was edited in the previous year and altering it slightly for our specific needs. The framework includes abstractions of the Khepera robot as well as various sensors. We expect to have a framework with a suitable simulation environment in which we can evolve and test robots based on the collective construction task.

By the end of this investigation, we expect to have successfully evolved a multi-robot team to cooperate in order to solve the collective construction task over a sufficient number of generations. Since we are each testing several variations of our chosen NE approaches across ten teams with different homogeneous sensor configurations, we expect to be able to determine which approaches are most effective in these various test cases.

Knowledge on which is the most effective NE approach could impact the design decisions of implementations where controllers for multi-robot systems need to be evolved. The project will have succeeded if even one of the NE approach was able to display collective behaviour and adequately perform the construction task.

7. PROJECT PLAN

7.1 Risks

7.1.1 Scope is too large

This is a common problem. It is difficult to determine how much work one can do within a given amount of time and people are often over-ambitious, biting off more than they can chew, so to say. It is possible that at some point during the project we realise that there simply is not enough time to finish all the necessary tasks.

In order to mitigate this, we have come up with carefully planned Gantt charts and timelines, along with project milestones and realistic deadlines.

7.1.2 Hardware Theft

A large portion of the coding for this project will be done on our own personal computers. If anything were to happen to a team member's computer, it would cause a huge disruption in the progress of the project.

This has been mitigated by using a private repository on GitHub for version control. Any data lost or framework breaking mistakes made can easily be recovered.

7.1.3 Difficulty adapting framework

The framework for projects such as this can often be difficult to use. It is not always easy understanding someone else's code and it could take a substantial amount of time just to figure out how things work.

There is a very low chance of this happening. The majority of the team members have had at least some experience with using frameworks to implement machine learning algorithms. We will also use the same framework that was used by a team from a previous year that investigated a similar topic. This means that the framework is already partially configured for some of our experiments.

7.1.4 Long training time

For this project, each team member will need to run each of their NE approaches for each of the ten different sensory configurations for the robot team. Simulations such as this are extremely resource heavy. If we had to run these on our local machines then there would be a good chance of the experiments not finishing in time.

Since this would be completely infeasible, we are going to be running our experiments using a high performance grid/cluster.

7.2 Timeline

Gantt Chart is attached in Appendix A.

7.3 Resources required

There are only a few resources that we require to complete our project. The first is the MASON library based simulator that will be used to model the multi-robot system and evolve solutions to the construction task. Each team member will be working on their own personal computers which will be used to run initial tests and simulations. Lastly as evolving and searching through a large candidate population of solutions in parallel is computationally expensive, Nightmare will be used for additional resources.

7.4 Deliverables

The following are honours project deliverables that are included in the Gantt chart:

Deliverable	ID	Due date
Literature review	1	24/04/2016
Project proposal	2	11/05/2016
Project presentation	3	22/05/2016
Revised proposal	4	04/06/2016
Web presence	5	08/06/2016
Initial software feasibility demonstration	10	18/07/2016
Theory section of report	11	22/07/2016
Report scaffold	15	29/08/2016
Initial implementation	16	20/09/2016
Final prototype	17	29/09/2016
Final implementation	18	04/10/2016
Outline of final report	19	11/10/2016
Complete draft of final report	20	18/10/2016
Complete final report	21	28/10/2016
Complete code	22	31/10/2016
Final project demonstration	23	31/10/2016
Poster	24	07/11/2016
Final web page	25	11/11/2016
Reflection paper	26	14/11/2016

7.5 Milestones

The following milestones represent the work and goals required to complete the CASAIRT project:

Sensor configurations - Choose 10 different sensor configurations to be modeled for testing the robustness of our three NE approaches.

Simulator complete - Finish adjusting the simulator so that it can meet the requirements of our project. This includes correctly modeling the chosen sensors and all aspects of the construction task.

Neuroevolution approach implemented - Each member must implement their chosen NE algorithm within the simulator in order to start evolving solutions.

Initial tests - Once the NE approach is completed, initial tests and simulations must be undertake to determine if solutions are being found.

Parameter tuning - The next step is finding the best possible parameters to use which produce the most effective solutions. This will require evolving and finding solutions with different sets of parameters.

Start running simulations - Run many simulations over a long period of time in order to find the best possible solutions.

Evaluate results - Analyze the results of the simulations to determine how effective they were and compare the three NE approaches.

Milestone	ID	Due date
Sensor configurations	6	13/06/2016
Simulator complete	7	27/06/2016
Neuroevolution approach implemented	8	11/07/2016
Initial tests	9	18/07/2016
Parameter tuning	12	16/08/2016
Start running simulations	13	03/09/2016
Evaluate results	14	04/09/2016

7.6 Work allocation

The whole team will collaborate on adapting the simulator to our project. This includes modifying the framework, choosing 10 different sensor configurations and modeling them correctly. Everything to do with the construction task will also be implemented together by the team. Once the simulator is complete each team member will program their NE approach on top of their own version of the simulator. After finding optimal solutions for the collective construction task the team will compare the results of the three approaches.

8. REFERENCES

[1] Y. U. Cao, A. S. Fukunaga, and A. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous robots*, 4(1):7–27, 1997.

[2] P. A. Castillo, J. J. Merelo, M. G. Arenas, and G. Romero. Comparing evolutionary hybrid systems for design and optimization of multilayer perceptron structure along training parameters. *Information Sciences*, 177(14):2884–2905, 2007.

[3] J. Clune, B. E. Beckmann, C. Ofria, and R. T. Pennock. Evolving coordinated quadruped gaits with the HyperNEAT generative encoding. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pages 2764–2771. IEEE, 2009.

[4] J. Clune, K. O. Stanley, R. T. Pennock, and C. Ofria. On the performance of indirect encoding across the continuum of regularity. *Evolutionary Computation, IEEE Transactions on*, 15(3):346–367, 2011.

[5] D. B. D'Ambrosio and K. O. Stanley. Generative encoding for multiagent learning. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 819–826. ACM, 2008.

[6] A. E. Eiben, G. S. Nitschke, and M. C. Schut. Collective specialization for evolutionary design of a multi-robot system. In *Swarm Robotics*, pages 189–205. Springer, 2006.

[7] A. Farinelli, L. Iocchi, and D. Nardi. Multirobot systems: a classification focused on coordination. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(5):2015–2028, 2004.

[8] D. Floreano, P. Dürr, and C. Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.

[9] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary computation*, 3(1):1–16, 1995.

[10] F. Gomez, J. Schmidhuber, and R. Miikkulainen. Efficient non-linear control through neuroevolution. In *Machine Learning: ECML 2006*, pages 654–662. Springer, 2006.

[11] F. J. Gomez and R. Miikkulainen. Active guidance for a finless rocket using neuroevolution. In *Genetic and Evolutionary ComputationâĂŤGECCO 2003*, pages 2084–2095. Springer, 2003.

[12] F. J. Gomez and R. Miikkulainen. *Robust non-linear control through neuroevolution.* Computer Science Department, University of Texas at Austin, 2003.

[13] M. Hausknecht, P. Khandelwal, R. Miikkulainen, and P. Stone. HyperNEAT-GGP: A HyperNEAT-based

Atari general game player. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 217–224. ACM, 2012.

[14] W. Jakob, M. Gorges-Schleuter, and C. Blume. Application of genetic algorithms to task planning and learning. In *PPSN*, pages 293–302, 1992.

[15] C. Jones and M. J. Matarić. Automatic synthesis of communication-based coordinated multi-robot systems. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 381–387. IEEE, 2004.

[16] N. Kohl, K. Stanley, R. Miikkulainen, M. Samples, and R. Sherony. Evolving a real-world vehicle warning system. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1681–1688. ACM, 2006.

[17] J. Lehman and K. O. Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336, 2008.

[18] J. Lehman and K. O. Stanley. Revising the Evolutionary Computation Abstraction: Minimal Criteria Novelty Search. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'10)*, (Gecco):103–110, 2010.

[19] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011.

[20] R. Miikkulainen. Neuroevolution. In *Encyclopedia of Machine Learning*, pages 716–720. Springer, 2011.

[21] R. Miikkulainen, B. D. Bryant, R. Cornelius, I. V. Karpov, K. O. Stanley, and C. H. Yong. Computational intelligence in games. *Computational Intelligence: Principles and Practice*, pages 155–191, 2006.

[22] H. Moriguchi and S. Honiden. Sustaining behavioral diversity in neat. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, GECCO '10, pages 611–618, New York, NY, USA, 2010. ACM.

[23] J.-B. Mouret. Novelty-based multiobjectivization. In *New horizons in evolutionary robotics*, pages 139–154. Springer, 2011.

[24] J.-B. Mouret and S. Doncieux. Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evolutionary computation*, 20(1):91–133, 2012.

[25] G. S. Nitschke, M. C. Schut, and A. Eiben. Evolving behavioral specialization in robot teams to solve a collective construction task. *Swarm and Evolutionary Computation*, 2:25–38, 2012.

[26] M. Quinn, L. Smith, G. Mayley, and P. Husbands. Evolving controllers for a homogeneous system of physical robots: Structured cooperation with minimal sensors. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 361(1811):2321–2343, 2003.

[27] S. Risi, S. D. Vanderbleek, C. E. Hughes, and K. O. Stanley. How novelty search escapes the deceptive trap of learning to learn. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 153–160. ACM, 2009.

[28] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st international Conference on Genetic Algorithms*, pages 93–100. L. Erlbaum Associates Inc., 1985.

[29] K. O. Stanley. Efficient evolution of neural networks through complexification. 2004.

[30] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial life*, 15(2):185–212, 2009.

[31] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

[32] P. Tonelli and J.-B. Mouret. On the relationships between generative encodings, regularity, and learning abilities when evolving plastic artificial neural networks. *PloS one*, 8(11):e79138, 2013.

[33] J. Watson and G. Nitschke. Evolving Robust Robot Team Morphologies for Collective Construction. In *Computational Intelligence, 2015 IEEE Symposium Series on*, pages 1039–1046. IEEE, 2015.

[34] J. Wawerla, G. S. Sukhatme, and M. J. Matarić. Collective construction with multiple robots. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2696–2701. IEEE, 2002.

[35] D. Wienke, C. Lucasius, and G. Kateman. Multicriteria target vector optimization of analytical procedures using a genetic algorithm: Part i. theory, numerical simulations and application to atomic emission spectroscopy. *Analytica Chimica Acta*, 265(2):211–225, 1992.

[36] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

APPENDIX

ID	Task Name	Start	Finish	Duration
1	Literature review	2016/04/19	2016/04/24	6d
2	Project proposal	2016/04/27	2016/05/11	15d
3	Project presentation	2016/05/18	2016/05/22	5d
4	Revised proposal	2016/05/25	2016/06/04	11d
5	Web presence	2016/05/30	2016/06/08	10d
6	Sensor configurations	2016/06/11	2016/06/13	3d
7	Simulator complete	2016/06/14	2016/06/27	14d
8	Neuroevolution approach complete	2016/06/28	2016/07/11	14d
9	Start running initial tests	2016/07/12	2016/07/18	7d
10	Initial software feasibility demonstration	2016/06/11	2016/07/18	38d
11	Theory section of report	2016/07/17	2016/07/22	6d
12	Parameter tuning	2016/07/23	2016/08/16	25d
13	Start running simulations	2016/08/10	2016/09/03	25d
14	Evaluate results	2016/08/28	2016/09/04	8d
15	Report scaffold	2016/08/25	2016/08/29	5d
16	Initial implementation	2016/08/30	2016/09/20	22d
17	Final prototype	2016/09/21	2016/09/29	9d
18	Final implementation	2016/09/30	2016/10/04	5d
19	Outline of final report	2016/10/05	2016/10/11	7d
20	Complete draft of final report	2016/10/12	2016/10/19	8d
21	Complete final report	2016/10/18	2016/10/28	11d
22	Submit completed code	2016/10/29	2016/10/31	3d
23	Final project demonstration	2016/10/29	2016/10/31	3d
24	Poster	2016/10/31	2016/11/07	8d
25	Final web page	2016/11/08	2016/11/11	4d
26	Reflection paper	2016/11/11	2016/11/14	4d

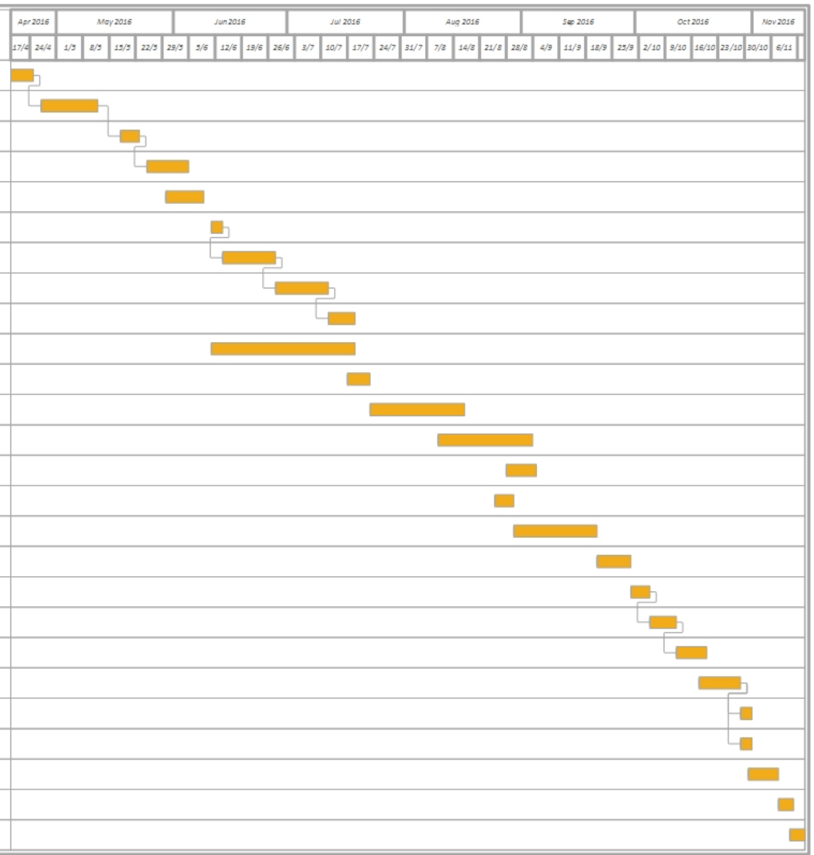

Figure 3: Project Gantt Chart