Evolving Indoor Navigational Strategies Using Gated Recurrent Units In NEAT

James Butterworth
University of Liverpool
Liverpool, UK
j.butterworth2@liverpool.ac.uk

Rahul Savani
University of Liverpool
Liverpool, UK
rahul.savani@liverpool.ac.uk

Karl Tuyls
University of Liverpool
Liverpool, UK
k.tuyls@liverpool.ac.uk

ABSTRACT

Simultaneous Localisation and Mapping (SLAM) algorithms are expensive to run on smaller robotic platforms such as Micro-Aerial Vehicles. Bug algorithms are an alternative that use relatively little processing power, and avoid high memory consumption by not building an explicit map of the environment. In this work we explore the performance of Neuroevolution - specifically NEAT - at evolving control policies for simulated differential drive robots carrying out generalised maze navigation. We compare this performance with respect to one particular bug algorithm known as I-Bug. We extend NEAT to include Gated Recurrent Units (GRUs) to help deal with long term dependencies. We show that both NEAT and our NEAT-GRU can repeatably generate controllers that outperform I-Bug on a test set of 209 indoor maze like environments. We show that NEAT-GRU is superior to NEAT in this task. Moreover, we show that out of the 2 systems, only NEAT-GRU can continuously evolve successful controllers for a much harder task in which no bearing information about the target is provided to the agent.

CCS CONCEPTS

• **Computing methodologies** → **Intelligent agents**; **Mobile agents**; **Neural networks**; **Artificial life**; **Genetic algorithms**;

KEYWORDS

genetic algorithms, neuroevolution, NEAT, navigation, maze solving, gated recurrent units, memory

ACM Reference format:
James Butterworth, Rahul Savani, and Karl Tuyls. 2019. Evolving Indoor Navigational Strategies Using Gated Recurrent Units In NEAT. In *Proceedings of Genetic and Evolutionary Computation Conference Companion, Prague, Czech Republic, July 13–17, 2019 (GECCO '19 Companion),* 2 pages.
https://doi.org/10.1145/3319619.3321995

1 INTRODUCTION

Smaller robotic platforms such as Micro-Aerial Vehicles (MAVs) have the potential to carry out tasks in indoor environments that are often too dangerous or time consuming for humans to do. Simultaneous Localisation and Mapping (SLAM) is the process whereby an agent constructs a map of the environment and uses this explicit map representation as a means to navigate. Despite the success of SLAM, it is a relatively computationally expensive algorithm leading to problems with memory and processing speed on smaller robotic platforms. An alternative suite of algorithms known as Bug Algorithms aim to navigate through an environment without building an explicit representation of the map. These agents react to local sensor readings such as proximity sensors to avoid objects. The agents also often know the distance and/or the relative position (the azimuth angle) to the goal but are not aware of the overall structure of the environment.

Even though Bug Algorithms are quite successful, they are all hand designed. This raises the question as to whether there exists more efficient and effective algorithms or control policies for these environments that have not yet been conceived of. We explore this by testing the performance of Neuroevolution of Augmenting Topologies (NEAT) with respect to one particular Bug Algorithm known as I-Bug which is particularly suitable to real robotic tasks. We also introduce an algorithm, NEAT-GRU, that includes Gated Recurrent Units (GRUs) in the NEAT networks thereby introducing a form of long term memory.

Previous work [6] considered the problem of evolving generalised maze solvers. Our contribution is to consider the performance of the algorithms with relation to a greater number of metrics; explore the advantages of using specific long term memory components; and introduce a much harder task in which no bearing information is available to the network.

2 EXPERIMENTAL SETUP

NEAT-GRU inserts GRU units into the networks as well as normal hidden nodes via node mutations. The weights of the GRUs are modified via link mutations identical to those used in normal NEAT. For simplicity, crossover is not used in NEAT-GRU. NEAT-GRU is very similar to NEAT-LSTM introduced in [5] however there are less parameters to optimise and we test NEAT-GRU in a continuous control domain.

The robotic simulator ARGoS [4] is used in this work for both the baseline I-Bug experiments and for the training and testing of the evolved solutions. In this work we use the simulator's Foot-Bot model [2] since it is equipped with 24 local proximity sensors, a range and a bearing sensor, which are the same sensors that are required for the I-Bug algorithm.

Bearing Experiments. We tested I-Bug, NEAT and NEAT-GRU on their ability to produce controllers that can navigate through 209 randomly generated 3D test environments of size $14m \times 14m$ that contain walls and obstacles. These environments were the same as those originally used in [3]. Each agent had 300 simulated seconds

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3321995

to navigate through each environment. The success percentage was recorded as the number of mazes in which the target was found as a percentage of the total number of mazes in the test set. Furthermore, the agents' trajectory lengths were recorded and normalised by divided through by the A* path length, which represents the shortest possible path through the maze. For I-Bug, the sensor configuration in [3] was used. In the experiments for NEAT and NEAT-GRU, only 12 (compared to 24) proximity sensors were used and their size was reduced to $0.2m$. The network outputs for the evolved solutions were the speeds of the left and right wheels of the Foot-Bot.

The environments used in training were randomly generated according to the same algorithm that generated the test set; details of which are available in [3]. During training a new set of 10 mazes were randomly generated every generation. The entire evolutionary run lasted 1000 generations with a population size of 150. Every 25 generations, the 3 best genomes from the current generation, the best 2 genomes from previous generation and the best genome from two generations ago are tested on the test set and their performance is recorded. 20 runs are performed using both NEAT and NEAT-GRU. The following fitness function, f_1, was used in training:

$$f_1 = \begin{cases} \frac{1}{l^{0.5}} & \text{if } maze\_solved \\ 0 & otherwise \end{cases} \quad (1)$$

where l is the trajectory length per A* length taken to find the target. If the agent crashed into a wall the final fitness was divided by 10.

No Bearing Experiments. A much harder task was designed to test the cognitive abilities of NEAT-GRU further. This task reduces the number of sensors leaving just the distance to target as input. This sensor configuration removes the ability of the agent to know its relative orientation towards the target, therefore finding the target must be done by accumulating distance measurements and performing significant cognition in order to ascertain the direction in which to travel. Only NEAT and NEAT-GRU were evaluated on this task due to the fact that I-Bug cannot function without access to the bearing sensor.

Given that this task is much more difficult, an environment of size $10m \times 10m$ was used that contained no obstacles. During training each agent is evaluated 5 times with different starting orientations. Each agent is given 80 simulated seconds to find the target. NEAT and NEAT-GRU were ran 10 times each for 5000 generations per run with a population size of 150. The fitness function $f_2 = (L-d)^3$ was used where L is the diagonal length of the arena (the maximum distance the agent can be from the target) and d is the final distance between the agent and the target at the end of the run.

3 RESULTS

I-Bug. On the 209 test environments I-Bug achieved a success rate of 195/209 = 93.3%. The mean and median of the trajectory lengths per A* lengths over all the environments were 2.4174 and 1.69 respectively.

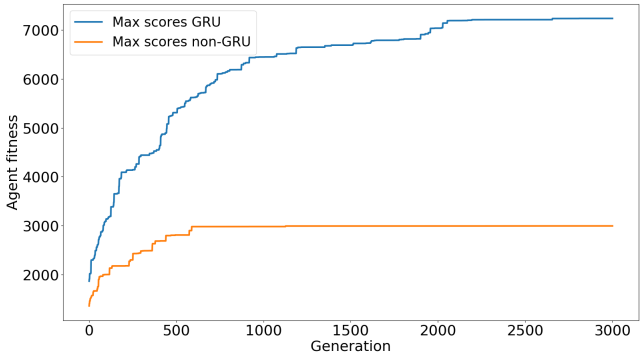
Evolved Solutions. For the experiments *with* the bearing sensor, Table 1 highlights the number of runs that produced a genome (out of those evaluated on the test set) that outperformed I-Bug in 2 metrics: success percentage and the *mean* of the trajectory length per A* length, and in 3 metrics: success percentage and the *mean and median* of the trajectory length per A* length. Some of the solutions produced were significantly better than I-Bug ($p < 0.0001$ based upon trajectory lengths), for example, one solution named 'G89' had a success rate of 196/209 (93.7%), a trajectory length mean of 1.9024 and a median of 1.5459. There also exist solutions that have a larger success rate but at the expense of having longer path lengths.

	2 metric winners	3 metric winners
NEAT	3/20	0/20
NEAT-GRU	**10/20**	**2/20**

Table 1: Number of evolutionary runs in which at least one genome outperformed I-Bug on the 209 test environments.

For the experiments *without* the bearing sensor, out of the 10 evolutionary runs for NEAT-GRU, all 10 produced a solution capable of solving the task in all 5 orientations. In contrast, out of the 10 runs using NEAT, 0 of them produced solutions that could solve the task in all 5 orientations. Figure 1 shows the maximum fitness so far for the population during training for the no bearing task. It shows the dramatic increase in performance due to the inclusion of GRUs into the NEAT networks. More detail on all aspects of this work exists [1] and a video demonstrating some of the most interesting evolved solutions is available at https://youtu.be/8EqyeuX_lR0

Figure 1: The maximum fitness so far for the population during training for both GRU and non-GRU versions of the no-bearing experiment. The results are averaged over 10 runs.



REFERENCES

[1] J Butterworth, R Savani, and K Tuyls. 2019. Evolving Indoor Navigational Strategies Using Gated Recurrent Units In NEAT. *CoRR* (2019). arXiv:1904.06239
[2] M Dorigo et al. 2013. Swarmanoid: A Novel Concept for the Study of Heterogeneous Robotic Swarms. *IEEE Robotics Automation Magazine* 20, 4 (2013), 60–71.
[3] K McGuire, G de Croon, and K Tuyls. 2018. A Comparative Study of Bug Algorithms for Robot Navigation. *CoRR* (2018). arXiv:1808.05050
[4] C Pinciroli et al. 2012. ARGoS: A Modular, Parallel, Multi-Engine Simulator for Multi-Robot Systems. *Swarm Intelligence* 6, 4 (2012), 271–295.
[5] A Rawal and R Miikkulainen. 2016. Evolving Deep LSTM-based Memory Networks Using an Information Maximization Objective. In *GECCO 2016 (GECCO '16)*. 501–508.
[6] D Shorten and G Nitschke. 2015. Evolving Generalised Maze Solvers. In *Applications of Evolutionary Computation*, Antonio M Mora and Giovanni Squillero (Eds.). Springer International Publishing, Cham, 783–794.