Evolutionary Predator-prey Robot Systems: from simulation to real world

Gongjin Lan
Vrije Universiteit Amsterdam
g.lan@vu.nl

Jiunhan Chen
University of Amsterdam
chen.jiunhan@gmail.com

A.E. Eiben
Vrije Universiteit Amsterdam
a.e.eiben@vu.nl

ABSTRACT

We present a feasibility study on evolving controllers for a group of wheeled robot predators that need to capture a prey robot. Our solution method works by evolving controllers in simulation for 100 generations, followed by 10 generations on real robots. The best controllers are further evaluated by their sensitivity for the initial positions. The results demonstrate the practical feasibility of this approach and give an indication of the time required to develop good solutions for the predator-prey problem.

CCS CONCEPTS

• **Computer systems organization → Evolutionary robotics**;

KEYWORDS

Evolutionary robotics, Predator-prey system, Reality Gap

ACM Reference Format:
Gongjin Lan, Jiunhan Chen, and A.E. Eiben . 2019. Evolutionary Predator-prey Robot Systems: from simulation to real world. In *Proceedings of the Genetic and Evolutionary Computation Conference 2019 (GECCO '19 Companion).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3322033

1 INTRODUCTION

Evolving behaviour strategies for robots in a predator-prey scenario is one of the classic themes in evolutionary robotics [3]. After many years of being an academic research subject, the problem of designing successful prey capturing strategies for robots is becoming practically relevant by the recent advances in drone technology. Inspired by such (future) applications we conduct a feasibility study in a simplified scenario to assess the practicability of an evolutionary approach and answer two questions: 1) Will it work? and 2) How long will it take?

2 SYSTEM DESCRIPTION

Our approach consists of three stages, 1) evolution in simulation for many generations, 2) evolution on the real robots for fewer generations, 3) evaluation of best evolved controllers' sensitivity to different starting conditions. By design, we use exactly the same evolutionary algorithm in simulation and in the real world. In the real world, we use a set of Thymio II robots[1] extended with a Raspberry Pi (that can handle wifi) and an extra battery. The field for pursuit and evasion is a 2m by 2m square arena without obstacles. Instead of adding cameras to the robots, we use an overhead camera above the arena to provide location information. The camera can distinguish each of the robots and recognize their positions and directions. The simulation environment is built on Gazebo[2], where we construct the models of Thymio II robot and a 2m by 2m field without obstacles.

While the 'bodies' of predators and the prey are identical, their controllers are different. For the prey, we use a smart but fixed evasion strategy based on a 2D Gaussian function to model the danger zone around the predators and a 1D Gaussian function to model the danger zone close to the walls. Using this danger zone map (continually updated during a chase), the prey tries to navigate towards less dangerous places to avoid the predators.

The predator controllers are neural networks with three input nodes, one hidden layer with four neurons and two outputs nodes that drive the two wheels. The first input is the inverse of the distance from the predator to the nearest predator, the second input is the angle between the orientation of predator and the direction of prey and the third input is the distance between the predator itself and the prey. Both the hidden and the output layers use hyperbolic tangent as the activation function.

The fitness of a predator controller is determined by testing it during a test episode. Equation (1) shows the fitness function, where d_{it} is the distance between predator i and prey at time t and r_{it} is the distance between predator i and nearest predator at time t. N_P is the number of predators. For each predator we compute the average distance $\frac{1}{T}\sum_{t=1}^{T} d_{it}$ to the prey and the average distance to the nearest predator $\frac{1}{T}\sum_{t=1}^{T} r_{it}$.

$$\frac{1}{N_P}\sum_{i=1}^{N_P}\left[\frac{1}{\frac{1}{T}\sum_{t=1}^{T} d_{it}}\right] + \frac{1}{N_P}\sum_{i=1}^{N_P}\frac{1}{T}\sum_{t=1}^{T} r_{it} \quad (1)$$

The evolutionary algorithm we use is the CMA-ES with population size 13 and the recommended parameter values from [1].

3 EXPERIMENTS

Comparing 3, 4, and 5 predators in preliminary experiments we found 5 too many, 3 and 4 performing comparably. Hence we choose 3 for practical reasons. In the beginning of a test episode of 60 seconds the prey is placed at the center of the square and the three predators are lined up along one of the edges of the arena. The inputs for the robot controllers are provided through wifi based on an overhead camera that collects the coordinates, orientation and

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

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3322033

[1] https://www.thymio.org/home-en:home
[2] http://gazebosim.org/

unique IDs of the robots. Figure 1 shows the fitness development during the real world evolution. Generation 100 and 101 contain the same set of controllers, but the fitness values are determined in simulation (gen 100) or on real robots (gen 101). The data show a clear reality gap here, the average fitness drops from 2.94 to 2.12, while the maximum fitness drops from 3.77 to 2.48. The red, green and blue crosses designate the 3 best controllers with fitness 4.55, 4.54, 4.11, the corresponding trajectories of the test episodes are shown in Figure 2.

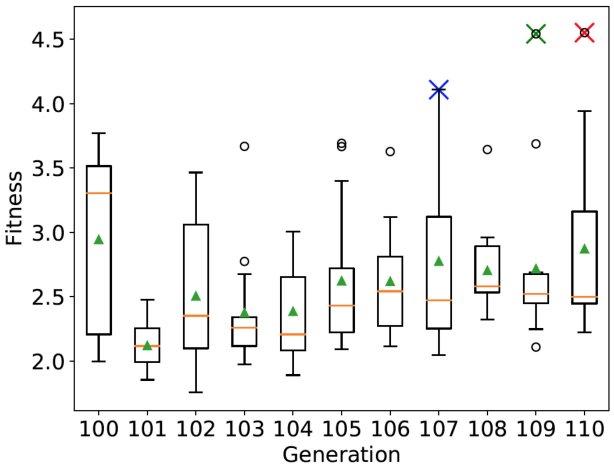
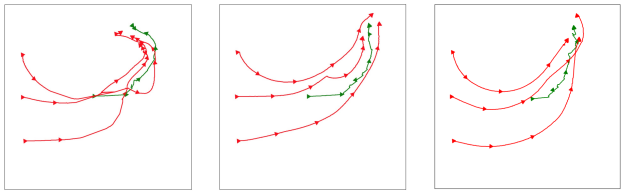


Figure 1: Evolution in the real world and the reality gap.



(a) Fitness = 4.55 (b) Fitness = 4.54 (c) Fitness = 4.11

Figure 2: Trajectories of the top three individuals evolved in the real world. The green line is the trajectory of prey.

To evaluate how well these controllers work in different situations, we perform a robustness test on the three best controllers. (For the red, green, and blue controllers as identified in Figure 1). We generate 1000 random initial positions for the robots and run a test episode for each of these in simulation. The results are exhibited in the histograms of Figure 3 that show the number of test episodes ending with a given fitness level .

4 DISCUSSION AND CONCLUSIONS

The "elephant in the room" in evolutionary robotics studies is always the reality gap [2]. Our system suffers from this too; as exhibited in Figure 1, the fitness drops by approximately 30% when we switch from simulation to real robots (average fitness: 28%, maximum fitness: 34%). The differences in execution times are even greater. Evolution in the simulated world took about twenty minutes to run 100 generations with population size 13 on an Intel

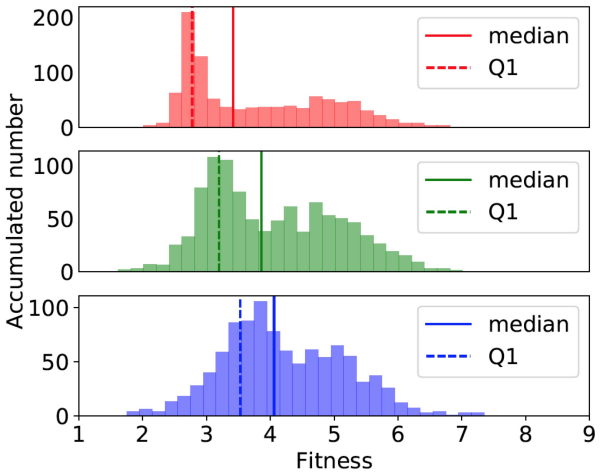


Figure 3: Robustness histograms of the best three controllers evolved on real robots.

Core™ i5-5200U CPU @ 2.20GHz × 4. Running 10 generations with the same population size on the real robots took about 2.5 hours. Thus, evolution in the real world for 100 generations would take about 25 hours to complete – approximately 70 times slower. Execution times are of course subject to many practical details, such as the computers clock speed, number of cores, simulation accuracy, reset time of the physical robots, etc., but these numbers give an indication of the time vs. quality trade-off inherent to evolutionary robotics experiments.

Is summary, we found that the two-stage evolutionary system was able to produce predators that successfully captured the prey in the real world. The fast simulations allow testing different algorithmic options, e.g., vary the number of predators or compare several EA variants. In the meanwhile, running evolution on the real robots can mitigate the reality gap problem. The main disadvantage we encountered was the time needed for the hardware experiments that were 70 times slower than the simulations. Using bigger populations we expect that this ratio becomes even worse. On the positive side, 10 real world generations turned out to be enough to reach the fitness level achieved in 100 simulated generations. Thus, we can consider the simulations as a good way to kick-start evolution in hardware and to reduce the total time needed to evolve a solution that works in the real world. With our current setup (Thymio robots, 2m × 2m arena, overhead camera) the combined software-hardware evolutionary system needed less than 24 hours clean run-time. Of course, there is an overhead of implementing and adjusting the simulator and setting up the physical arena, but the total duration of this project was less than three months.

REFERENCES

[1] Nikolaus Hansen and Stefan Kern. 2004. Evaluating the CMA evolution strategy on multimodal test functions. In *International Conference on Parallel Problem Solving from Nature*. Springer, 282–291.
[2] Nick Jakobi, Phil Husbands, and Inman Harvey. 1995. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life*. Springer, 704–720.
[3] Stefano Nolfi. 2012. Co-evolving predator and prey robots. *Adaptive Behavior* 20, 1 (2012), 10–15.