TU Delft

Delft University of Technology

Document Version
Accepted author manuscript

Citation (APA)
Scheper, K., & de Croon, G. (2016). Abstraction as a Mechanism to Cross the Reality Gap in Evolutionary Robotics. In E. Tuci, A. Giagkos, M. Wilson, & J. Hallam (Eds.), *From Animals to Animats 14* (Vol. 9825, pp. 280-292). (Lecture Notes in Computer Science; Vol. 9825). Springer. https://doi.org/10.1007/978-3-319-43488-9_25

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Abstraction as a Mechanism to Cross the Reality Gap in Evolutionary Robotics

Kirk Y.W. Scheper and Guido C.H.E. de Croon

Micro Air Vehicle Laboratory, Delft University of Technology, 2629 HS Delft, The Netherlands
k.y.w.scheper@tudelft.nl

Abstract. One of the major challenges of Evolutionary Robotics is to transfer robot controllers evolved in simulation to robots in the real world. In this article, we investigate abstraction on the sensory inputs and motor actions as a potential solution to this problem. Abstraction means that the robot uses preprocessed sensory inputs and closed loop low-level controllers that execute higher level motor commands. We apply abstraction to the task of forming an asymmetric triangle with a homogeneous swarm of MAVs. The results show that the evolved behavior is effective both in simulation and reality, suggesting that abstraction can be a useful tool in making evolved behavior robust to the reality gap. Furthermore, we study the evolved solution, showing that it exploits the environment (in this case the identical behavior of the other robots) and creates behavioral attractors resulting in the creation of the required formation. Hence, the analysis suggests that by using abstraction, sensory-motor coordination is not necessarily lost but rather shifted to a higher level of abstraction.

1 Introduction

Evolutionary Robotics (ER) is a field of research which uses Evolutionary Computation techniques to solve robotic tasks without explicit interaction from a human designer. This approach requires a roboticist to define the problem to be solved and the evolutionary optimization determines the behavior required solve it. Early work in this field made quick progress showing that ER could automatically solve tasks such as: obstacle avoidance [10], phototaxis [11] and chemotaxis [2]. Work was not restricted to the evolution of the brain but was also used to evolve the physical body of the robot in a form of co-evolution [4,17]. A comprehensive overview of this early work in ER can be found in the book from Nolfi et. al. [21].

Despite this early sprint, the pace of development slowed as researchers attempted more complex tasks. Some of the major challenges encountered include the reality gap, reducing optimization time, fitness function design and behavior representation [3]. Although all these issues must be addressed for ER to be truly successful, in this paper we would like to address the reality gap.

ER typically utilizes simulated environments to evaluate the performance of generated candidate solutions. Although these faster-than-real-time simulations reduce the total optimization time, differences between simulation and reality often result in reduced performance in reality when compared to that seen in simulation. This difference is referred to as the *reality gap*.

The apparent coupling seen between the perceived environment and emergent behavior that causes this reality gap is partly the result of Sensory-Motor Coordination (SMC). This inherent coupling of perception and action with embodied agents results in the observation that an agent can actively influence the perceived environment through its actions [20]. Typical implementations of ER utilize raw sensor inputs to generate low level control commands to the robotic platform. This approach has been shown to be effective in the development of behavior which can solve non-trivial tasks with SMC [1,20]. The evolved SMC will exploit the properties of the low-level sensors and motor actions in the simulated environment. Unfortunately, these properties in general are quite different from those of the real robotic platform, causing a significant reality gap.

Much progress has been made towards solving the reality gap problem most notably by Jakobi et. al. [15], Koos et. al. [16] and Eiben et. al. [8]. Jakobi suggests hiding unnecessary features of the simulation in noise through *minimal simulations*. Koos includes the transferability of the robotic behavior to the agent's fitness evaluated by intermittently testing the simulated behavior on real robots during evolution. Eiben promotes on-line embodied evolution on a swarm of real robotic platforms to remove the reality gap altogether. Some recent work has also suggested that improved insight into the optimized behavior can enable the roboticist to actively reduce the reality gap after optimization [24].

One method which has not been investigated in much detail is the use of *abstraction* to make the robotic control more robust to the reality gap. Generally, real robotic platforms are controlled using closed loop control systems. These systems receive a desired set-point as input and drive the output to reduce the perceived error. Closed loop control has been mathematically proven to make the eventual control system more robust to external disturbances or changes to the environment [18]. With the use of a closed loop low-level control system, evolution would abstract away from the low-level sensor inputs and actuator outputs. Some resent work has shown promising results in bridging the reality gap using abstracted methods [5,7]. This however may come at a price, as abstraction "hides" the properties of the raw sensory inputs and motor actions to the controller, it may have as a disadvantage that the potential for SMC by the robot is reduced.

In this paper we investigate whether abstraction can lead to an easy transfer of an evolved robot controller from simulation to the real world. Moreover, we look into the open question of how abstraction affects the ability of the robot to exploit its environment to solve a seemingly complex task. What happens to SMC when the agent doesn't have access to the raw sensor inputs and the ability to directly control the raw motor outputs?

To investigate this we implement an experiment based on the work of Izzo et. al. [14] to generate an asymmetric formation of a swarm of robots using a homogeneous control system. The task will be discussed in more detail in Sect. 2. The implementation and results of the evolutionary optimization of the robotic behavior is presented in Sect. 3. A brief description of the flight hardware shown in Fig. 1 is given in Sect. 4 followed by a discussion of the behavior on a swarm of real flying robots in Sect. 5. Sect. 6 dives a bit deeper into the optimized behavior and the effect of abstraction on the SMC. Finally, we summarize and make some conclusions in Sect. 7.

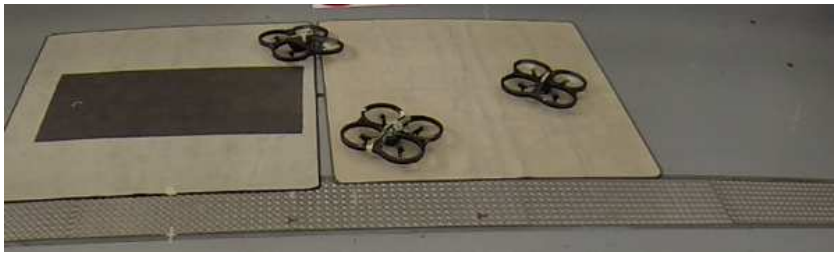


Fig. 1. Swarm of homogeneous ARDrone 2 quadrotors autonomously used to form an asymmetric triangle using evolved behavior.

2 Task

In this paper we would like to demonstrate the power of using high level control cues with an underlying closed loop control system to reduce the reality gap. The use of closed loop control systems helps to reject disturbances due to noise or small a mismatches between the dynamics in simulation and that in reality.

To this end we have selected the asymmetric formation flight as demonstrated in simulation by Izzo et. al [14]. That paper described a homogeneous swarm of three SPHERES spacecraft flying in a triangular formation where each side was a different length. Methods have been developed to autonomously form symmetric formations using homogeneous swarms but asymmetric shapes have proven to be more difficult [13]. The design of asymmetric formations using a distributed control system without explicit roles in the formation is a non-trival task for most human designers making it an ideal task for automatic optimization.

The work of Izzo et al. was confined to a simulated environment, in this paper we would like to move to reality to demonstrate the effect of abstraction on crossing the reality gap. Due to the lack of availability of the SPHERES vehicles on the International Space Station, the formation control is implemented on a swarm of Micro Air Vehicles (MAVs). Notably, we constrain the problem to two dimensions to facilitate a more straightforward analysis of the resultant behavior.

The goal of the swarm will be to achieve an asymmetric triangular formation with sides of lengths: 0.7, 0.9 and 1.3m. The MAVs can observe the relative position of all other members in the formation. The control system should use this information and provide the MAV with a velocity set-point. As in [14], we utilize a single hidden layer Artificial Neural Network (ANN) with three inputs and two outputs as shown in Fig. 2. A *tanh* activation function was used in the neurons. Additionally, a bias neuron is added to the input and hidden layer. The output of the network can be linearly scaled to the limits of the vehicle, which in the case of this paper was set as $\pm 0.5\mathrm{m/s}$. The inputs, outputs and ranges of all parameters were chosen to be as similar to the original values used by Izzo et. al. [14] to facilitate a fair comparison of our results with the original work.

The inputs to the ANN are the sum of the Cartesian components of the relative positions of the other members of the formation ($\mathbf{r}$) and the sum of absolute distances

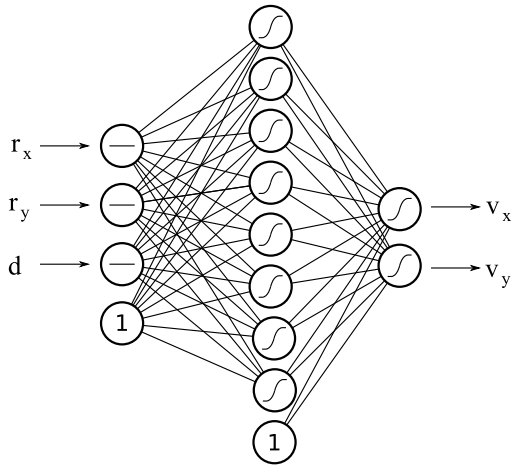


Fig. 2. Single hidden layer Neural Network topology used in this paper. Inputs are the summed Cartesian components of the relative positions to the other vehicles in the formation (**r**) along with the summed absolute distances (d).

(d) as given by (1) and (2). Note that **r** is mathematically equivalent to triple the distance from the ownship to the centroid of the formation (**c**).

$$\mathbf{r} = \sum_{i=2}^{k} (\mathbf{p}_i - \mathbf{p}_1) \tag{1}$$

$$d = \sum_{i=2}^{k} |\mathbf{p}_i - \mathbf{p}_1| \tag{2}$$

Where **p** is the position vector of a vehicle and k is the total number of vehicles in the swarm. These inputs are computed for each vehicle where $\mathbf{p_1}$ is the ownship location. Fig. 3 illustrates a possible solution to the formation problem. Combining the positions of the other vehicles in this way is essential to ensure the input is invariant to permutations of the vehicles. Adding a separate input for each vehicle would implicitly encode a unique formation identifier for each vehicle. Additionally, only the relative positions are required for the input rather than the absolute position, this would facilitate a wide range of real world sensors to be used to provide this information. It should be noted that as this input is from the point of view of each vehicle, a given set of inputs describes a unique formation and is not rotationally invariant.

3 Evolutionary Optimization

There are many forms of evolutionary optimization in literature but they all have a few things in common: a population of candidate solutions; a measure of fitness; a way of evaluating the individuals on this fitness function; and a method to change individuals to create the next population [9]. In this paper, we use a population of 100 candidate

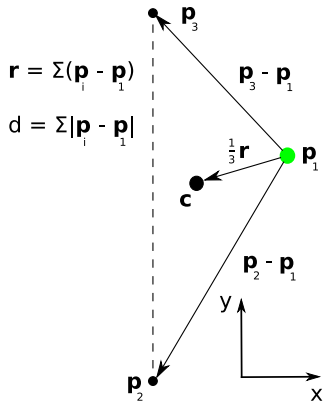


Fig. 3. Illustration of a possible formation. Vehicles are represented by a filled dot with the ownship in this example highlighted.

solutions expressed as ANNs. The fitness of an individual is determined with the use of a multi-objective sorting algorithm based on Nondominated Sorting Genetic Algorithm II (NSGA-II) [6]. Multiple objective optimization was used to promote effective exploration of the fitness landscape. Again, to facilitate a fair comparison, the objective functions used in this paper are based on those used in [14] and are given in (3).

$$
\begin{aligned}
f_1 &= \sum_{n=1}^{3} |L_n - l_n| \\
f_2 &= \sum_{n=1}^{3} |\mathbf{v}_n|^2 \\
f_3 &= \begin{cases} 0, & |\mathbf{c}| < 2 \\ 1, & \text{else} \end{cases}
\end{aligned}
\tag{3}
$$

Where $\mathbf{L}$ is the vector of the required distances, $\mathbf{l}$ is the vector of the distances between the vehicles at the end of the simulation sorted in ascending order. Sorting the distances makes the computation of the fitness it invariant to the vehicle order. $\mathbf{v}$ is the velocity vector of the MAV at the end of the simulation and $\mathbf{c}$ is the location of the centroid of the triangle. The first fitness function tries to have the MAVs end up in the correct formation. The second promotes individuals that have a low final velocity. The final function is an augmentation to the original set from [14] and promotes behavior that results in the centroid of the formation remaining inside of a 2m radius of the origin of the axis system. This requirement was added due to the practical limitation that the vehicles must operate in a constrained 8×8m flight arena.

A simulation was used to assign a fitness to the candidate solutions. A simple Euler integration based kinematic simulation was implemented to ensure that computational requirements of each simulation would remain low, speeding up the optimization. This simulator captures the approximate kinematics of the real MAVs with a simple low pass

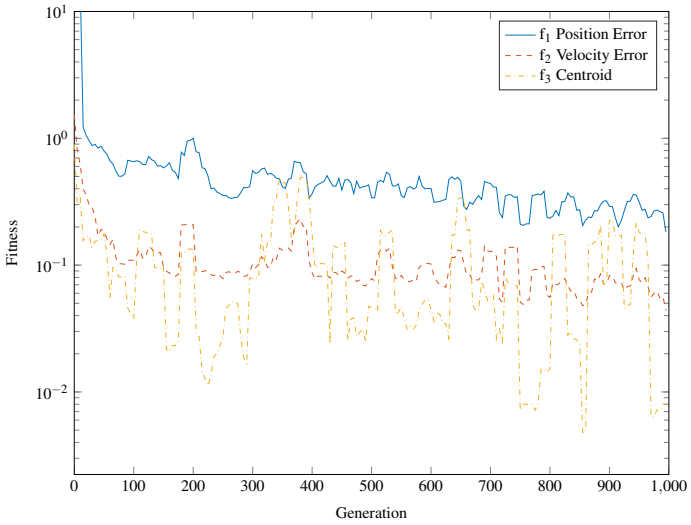
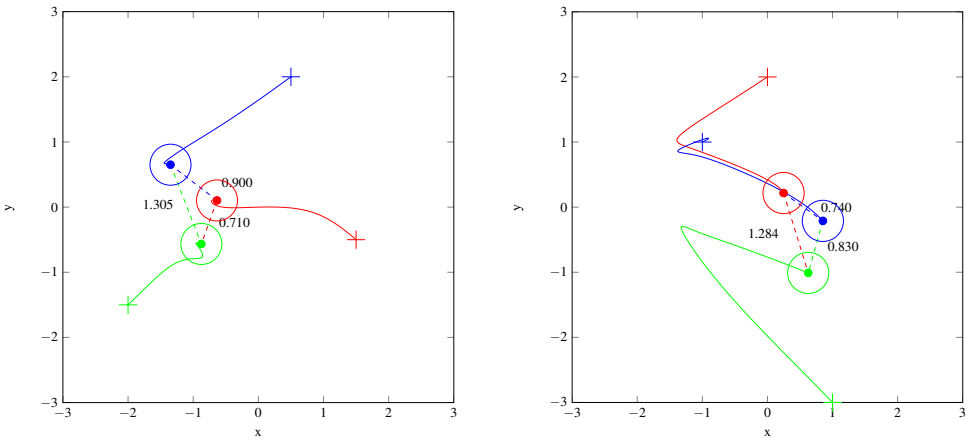


Fig. 4. Progression of the performance of the best individual during evolution validated using 250 initialization points.

filtered velocity response with a time constant determined by performing real world flight tests. Each simulation was initialized with the three MAVs at a stand still at random locations in a 2×2m area with at least 1m separating each vehicle and the centroid of the initial orientation located at the origin of the axis system. The vehicles were then allowed to traverse the room in the $x-y$ plane for a maximum of 50s. The simulation can be prematurely terminated if the MAVs come within 30cm of each other as this would constitute a collision on the real vehicles. At the end of the simulation run, the final position and speed of the MAVs is used to assign a value to each fitness function as given in (3).

Once the population is evaluated, they are sorted using the NSGA-II algorithm. The best individuals are stored in an archive of 100 members. This archive is also used as the mating pool which is used to generate the population of the next generation. Members are selected from the mating pool using a tournament selection technique with a size of 8 randomly selected members from the mating pool and the best individual averaged over all fitness dimensions returned as a champion. Mutation was the only evolutionary operator used in this paper as some works have shown that mutation only evolution to be effective [25]. Each weight in the ANN was considered for mutation with a probability of 5%. Mutation consisted of a random perturbation of the previous value by selecting a new value based on statistical acceptance based on the previous value constrained on the range [-1,1].

Fig. 4 shows the performance of the best individual from each generation of the evolutionary optimization for this problem. Each individual was evaluated using 250 different combinations of initial conditions with a maximum simulation time of 50s. This figure shows that evolution gradually reduces the error in the final vehicle distances



(a) Collision free (b) Collision at (-1.25,1)

Fig. 5. Ground tracks of a collision free flight and a flight that would have ended in a collision of three ARDrones performing the asymmetric formation task. The length of each side is shown in text, + marks the start location and the circle with the dot in the center marks the end location with the diameter of the vehicle to scale.

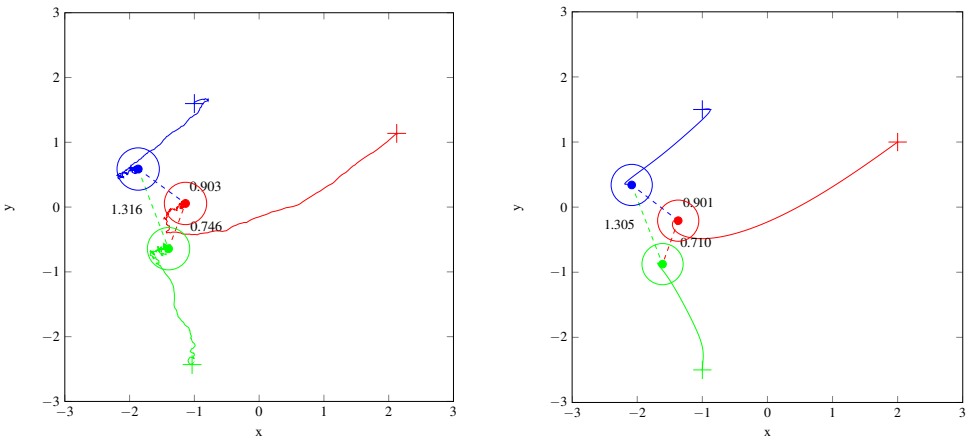
and the final velocity. This figure also shows that the behavior does not guarantee a collision free flight for all initial conditions. After 1000 generations the average error of each side of the formation is about 5cm.

The member with the lowest average score over the three fitness functions from the last generation of the evolutionary optimization was selected for further analysis and implementation on the real swarm of three MAVs. Here we will first analyze the behavior exhibited by the ANN controller to gain some insight into the solution to the problem.

The behavior was evaluated by a validation run of 250 different initial conditions. During the validation run, the simulation was not cut short if a collision occurred. A formation is considered accurate when the summed error of the lengths is below 0.15m or 5cm average error. The results show that 98% of runs resulted in a successful triangle formation within 50s of which 14% would have incurred a crash. Of these successful runs, the mean error was 0.0222m with a standard deviation of 0.0262. In 2% of the runs the triangle was not formed within 50s. Fig. 5 shows one of the successful runs of the formation behavior and one case where a collision would have occurred.

4 Flight Hardware

The flight tests performed in this paper were conducted using the 420g Parrot ARDrone 2 quadrotor MAV. This vehicle is equipped with a 1GHz 32 bit ARM Cortex A8 processor running an embedded Linux operating system [22]. The default flight software provided by Parrot was overwritten by custom flight software implemented using Paparazzi, an open-source flight control software [12,23].



(a) Real world (b) Simulation

Fig. 6. Ground track of the real world flight test and the simulated flight for the same initial positions. The length of each side is shown in text, + marks the start location and the circle with a dot in the center marks the end location with the diameter of the vehicle to scale.

5 Flight Tests Results

Moving from the simulated environment to the real world, the behavior shown above was implemented on a swarm of three ARDrones. Flights were performed in an 8×8m flight arena and the flight path of the vehicles was captured using an Optitrack motion camera system [19]. The position of all vehicles were broadcast at 5Hz so all vehicles know the relative position of the other swarm members. For the first set of tests, as in simulation, the three vehicles were initialized at random in a 3×3m area in the flight arena with the centroid of the initial formation at the origin of the arena. Fig. 6 shows the result of one test performed.

It was observed that the quadrotors were so close to each other that the downwash from one quadrotor would interact with the other vehicles causing significant external disturbances. Fig. 7 shows the commanded velocity of the ANN and the true vehicle velocity along with the result of the simulation. In contrast with the simulation, the real-world quadrotors have clear errors in tracking the desired velocities, in part due to the aerodynamic interactions. These tracking errors represent a significant reality gap.

Despite this apparent mismatch between simulation and reality, the observed behavior is very similar to that seen in simulation with the correct formation achieved with an accuracy of ± 0.034.

6 Analysis of the Sensory-Motor Coordination

To analyze the effect of abstraction on the extent to which evolved robots exploit their environment and make use of SMC, we must first diver deeper into the optimized behavior. In Izzo et al., an analysis of the evolved behavior was performed. As a part of

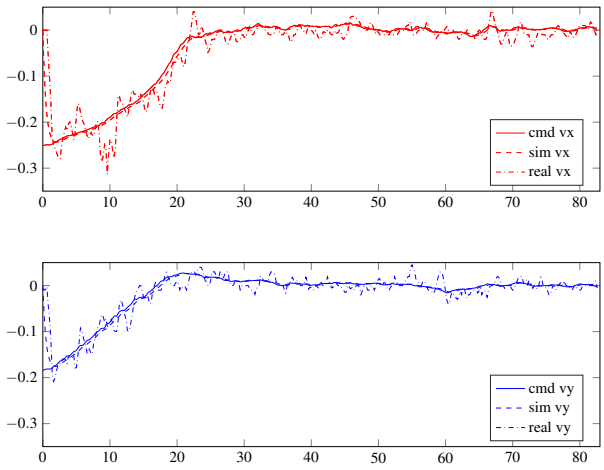


Fig. 7. Tracking performance of the velocity controller on the simulated and real ARDrone 2 given the same velocity command highlighting the reality gap.

the analysis, two robot satellites were fixed at one of the desired distances. The third satellite was left free to move but did not settle into a position which completed the formation. This led them to an interesting hypothesis: perhaps the asymmetric formation could only be reached if all three satellites were free to move. Here we will investigate if we can observe a similar phenomenon for our specific evolved solution, and evaluate whether SMC plays a role in successful formation flight.

In the flight tests performed in this paper, it was observed that all successful formations resulted in a triangular formation with the same rotational orientation to the Cartesian axis system. The orientation can be seen in Fig. 5 and Fig. 6. As there is a unique set of inputs to describe every possible rotational orientation of the triangle formation, this alludes us to the possibility that the ANN is trying to solve the formation for a fixed set of sensory inputs (r_x, r_y, d) rather than a set of linear combinations that would define a rotationally independent formation. This demonstrates how influential the fitness function is to the final solution. The function used in this paper requires the formation of a triangle with three fixed length sides but makes no definition of the required orientation. Given that freedom, the optimization finds the simplest solution to the problem, which in this case is a unique formation.

This solution also suggests a level of inherent environmental exploitation, namely that the other vehicles will comply and also move in such a way to solve the problem. We can test this by fixing two of the vehicles in an orientation different to that converged upon when they are all free. If we initialize the third vehicle at various locations around the other two and allow the vehicle to move for 500s we should be able to identify the basin of attraction this configuration. Fig. 8 shows a basin of attraction for the situation when the longest side of the formation is fixed along the x-axis. This figure shows the magnitude of the commanded velocity of the free vehicle at all locations in around the other two fixed vehicles. It also shows that the velocity vector field has

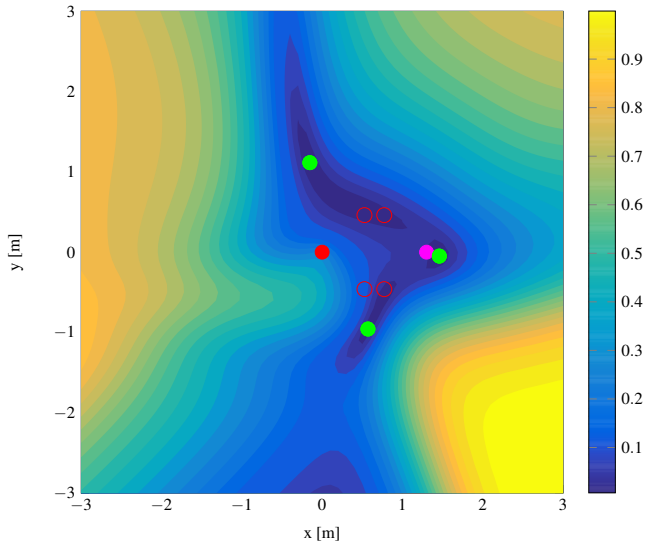


Fig. 8. Basin of attraction showing the velocity magnitude for one vehicle given the other two vehicles (dark dots) are fixed in space. The hollow circles highlight the possible solutions to the formation problem and the light dots show the stable attractor points.

three attractor points, none of which are correct locations to complete the formation. Notably, although the highlighted spots are stable points when the two other vehicles are fixed, the commanded velocities of the other two vehicles is non-zero showing that this formation itself is not stable.

If we repeat this for the case when the two fixed vehicles form the angle which is converged upon when all vehicles are free we are left with the basin of attraction seen in Fig. 9. This analysis shows that in this new configuration there is only one stable attractor in the velocity map which would indeed solve the formation problem. In this location the fixed vehicles have near zero velocity set-points.

We also performed real flight tests with two vehicles fixed along the correct orientation and distance of one side of the formation. Fig. 9 shows that the ground tracks of the real flights overlap almost exactly with this velocity field.

Nolfi et. al. suggest that the emergence of behavioral attractors is indicative of SMC [20]. Given this, the behavior we have shown here would seem to exhibit some form of SMC albeit on a more abstract level. Although the evolution has no access to the low level control or sensory inputs, the resultant behavior was still able to exploit the implicit knowledge that the other members of the swarm would unintentionally cooperate to solve the task.

This result also sheds some light onto the result of Izzo et. al.. It may not have been necessary for all the vehicles to be moving to achieve the formation but rather the relative orientation of the members must facilitate the behavioral attractor.

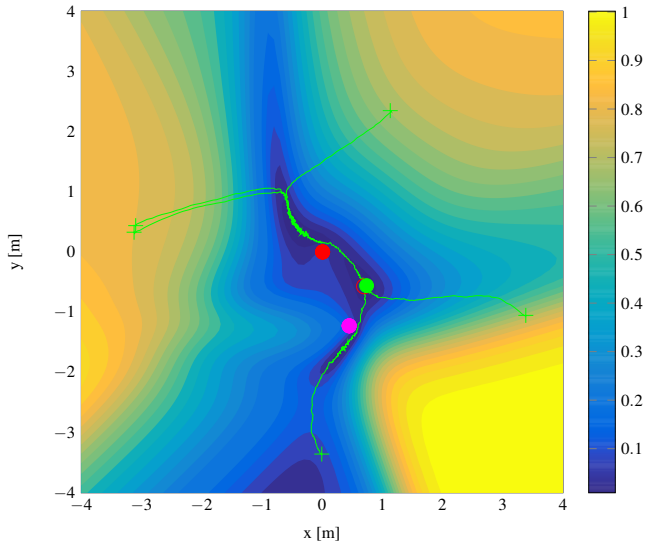


Fig. 9. Basin of attraction showing the velocity magnitude for one vehicle given the other two vehicles (dark dots) are fixed in space. The hollow circle highlights the solution to the formation problem. Overlain are ground tracks of all real world flights with the light dot showing the location the real robots converged toward. This shows that the real world performance mirrors what is expected from simulation quite well despite a clear reality gap.

7 Conclusions

In this paper we investigated the application of abstraction on the inputs and outputs of a neural network controller within the Evolutionary Robotics paradigm. The evolutionary optimization was tasked with forming an asymmetric triangle with MAVs. The optimized behavior was effective both in simulation and reality suggesting that abstraction can be a useful tool in making evolved behavior robust to the reality gap. We also showed that sensory-motor coordination which is a typical emergent phenomenon of reactive agents is not necessarily lost when abstracting away from the raw inputs and output but is rather shifted to a higher level of abstraction. Future work will implement the task presented here with the control on a lower level of abstraction to more explicitly investigate the influence of abstraction through direct comparison.

References

1. Agmon, E., Beer, R.D.: The evolution and analysis of action switching in embodied agents. Adaptive Behavior 22(1), 3–20 (2013)
2. Beer, R.D., Gallagher, J.C.: Evolving Dynamical Neural Networks for Adaptive Behavior. Adaptive Behavior 1(1), 91–122 (1992)
3. Bongard, J.C.: Evolutionary Robotics. Communications of the ACM 56(8), 74–83 (2013)
4. Bongard, J.C., Zykov, V., Lipson, H.: Resilient machines through continuous self-modeling. Science 314(5802), 1118–21 (2006)

5. Cully, A., Clune, J., Tarapore, D., Mouret, J.B.: Robots that can adapt like animals. Nature 521(7553), 503–507 (may 2015), http://dx.doi.org/10.1038/nature14422http://10.1038/nature14422http://www.nature.co
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
7. Duarte, M., Costa, V., Gomes, J., Rodrigues, T., Silva, F., Oliveira, S.M., Christensen, A.L.: Evolution of collective behaviors for a real swarm of aquatic surface robots. PLoS ONE 11(3), 1–25 (2016)
8. Eiben, A.E., Kernbach, S., Haasdijk, E.: Embodied artificial evolution: Artificial evolutionary systems in the 21st Century. Evolutionary Intelligence 5(4), 261–272 (2012)
9. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Springer-Verlag, Berlin Heidelberg, 2 edn. (2015)
10. Floreano, D., Mondada, F.: Automatic Creation of an Autonomous Agent: Genetic Evolution of a Neural-Network Driven robot. In: Cliff, D., Husbands, P., Meyer, J.A., Wilson, S. (eds.) Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animals to Animats 3. pp. 421–430. MIT Press, Cambridge, MA (1994)
11. Floreano, D., Mondada, F.: Evolution of homing navigation in a real mobile robot. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 26(3), 396–407 (1996)
12. Hattenberger, G., Bronz, M., Gorraz, M.: Using the Paparazzi UAV System for Scientific Research. In: International Micro Air Vehicle Conference and Competition 2014. pp. 247–252. IMAV, Delft, Netherlands (2014)
13. Izzo, D., Pettazzi, L.: Autonomous and Distributed Motion Planning for Satellite Swarm. Journal of Guidance, Control, and Dynamics 30(2), 449–459 (2007)
14. Izzo, D., Simões, L.F., de Croon, G.C.H.E.: An evolutionary robotics approach for the distributed control of satellite formations. Evolutionary Intelligence 7(2), 107–118 (2014)
15. Jakobi, N.: Minimal Simulations For Evolutionary Robotics. Ph.D. thesis, University of Sussex (1998)
16. Koos, S., Mouret, J.B., Doncieux, S.: The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics. Transactions on Evolutionary Computation 17(1), 122–145 (feb 2013)
17. Lipson, H.: Evolutionary Robotics: Emergence of Communication. Current Biology 17(9), 129–155 (2007)
18. Love, J.: Process Automation Handbook. No. 800 in Production & Process Engineering, Springer, London, 1 edn. (2007)
19. Natural Point Inc: Optitrack (2014), www.naturalpoint.com/optitrack/
20. Nolfi, S.: Power and limits of reactive agents. Neurocomputing 42, 119–145 (2002)
21. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence and Technology. MIT Press, Cambridge, MA (2000)
22. Parrot: ARDrone 2, www.ardrone2.parrot.com/
23. Remes, B., Hensen, D., van Tienen, F., de Wagter, C., van der Horst, E., de Croon, G.: Paparazzi: How to make a swarm of Parrot AR Drones fly autonomously based on GPS. In: Proceedings of the International Micro Air Vehicle Conference and Flight Competition. pp. 17–20. IMAV, Toulouse, France (2013)
24. Scheper, K.Y.W., Tijmons, S., de Visser, C.C., de Croon, G.C.H.E.: Behaviour Trees for Evolutionary Robotics. Artificial life 22(1), 23–48 (2016)
25. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)