Proceedings of the 2018 IEEE
International Conference on Robotics and Biomimetics
December 12-15, 2018, Kuala Lumpur, Malaysia

Identifying the Reality Gap between Abstract and Realistic Models using Evolved Agents and Simulated Kilobots

Jane Holland, Ciaran Gallagher, Josephine Griffith and Colm O'Riordan

***Abstract*— A common challenge in evolutionary swarm robotics is the transfer of simulated results into real-world applications. This difficulty can arise in a variety of real-world settings and problems such as sensory differences in robots and changes in the environment. We identify this reality gap at a simulation level by comparing the evolved behaviours of simulated Kilobots in two different models with different levels of abstraction. Our aim is to identify the reality gap that occurs at simulation level by increasing the task difficulty and noting differences in outcomes. Insights gained in this process may help rule out any further causes of reality gap when moving to experiments with physical robots.**

I. INTRODUCTION

Although simulation in evolutionary swarm robotics is often a desirable approach given its inexpensive nature in comparison to real world experimentation, it can be subject to a reality gap. The reality gap in question refers to crossing the gap between simulations and reality; how can one transfer behaviour seamlessly into physical robots? Modelling in simulation may help us understand naturally occurring phenomena, but sometimes important natural features such as physical embodiment and behavioural interactions may differ when modelled in artificial systems. This gap is even more evident in systems with greater complexity, where minuscule differences between simulation and reality can lead to largely disparate behaviours.

As there is always potential for a reality gap, models need to be accurate and retain as many of the robots features as possible; controllers need the right amount of noise in a carefully modelled simulation of the robot and its environment. Despite these efforts, one must keep in mind that simulations cannot be perfect and it can be very difficult to simulate the actual dynamics of a real world system [1]. Furthermore, in terms of hardware, we need to consider the fact that the communication responses can be inaccurate and although physical sensors and actuators may appear identical, they can perform differently due to small discrepancies in their electronics [2]. It is due to these issues that using an evolutionary approach in swarm robotics has yet to demonstrate the ability to scale in complexity, or provide solutions for realistic applications [3].

In this paper we explore the distinction between the reproduction of behaviour and the performance in evolutionary swarm robotics in order to identify the reality gap accurately. The reproduction of behaviour is defined as a task being performed in the same way in each simulation model as it is performed in reality. In terms of performance, we refer to the efficiency of the task being performed in the simulations and in reality. In identifying the reality gap between abstract and realistic models, we hope to discover any possible discrepancies and, in future work, diminish them in order to more accurately measure the reality gap in real-world experimentation. Comparing abstract and realistic simulation models is a necessary first step into investigating the reality gap; by observing the gap at a this level, between a clean simulation environment and a noisy simulation environment, it can give us an indication of where the noise occurs and where it could potentially occur in real-world experiments.

We start with a simple simulation model; an abstract 2D model with discrete movement and no noise. Once the behaviours are successfully evolved using this simulator, we repeat the process on a more realistic model using a simulation tool that allows experiments to be carried out in a 3D world which takes physical laws into account. This process is repeated for two tasks of increasing difficultly in order to investigate whether there is some correlation between the level of difficulty of a task and the presence of a reality gap. However, in order to do this, we need to make sure that the controller running on the abstract simulation and the controller running on the realistic simulation are as similar as possible. If this is not done correctly, we cannot take the next step and run the same controller in both simulation and reality.

We also investigate the effect of noise on the simulated robots' behaviours. We do this in two ways: first, we introduce noise after the robots' behaviours have been evolved in a noise-free environment; and second, we test behaviours that were evolved in a noisy environment, in a noisy-free environment. This experimentation with noisy and noise-free environments is undertaken to further study potential sources of the reality gap.

The outline of the paper is as follows: initially, we discuss related work in this area in section 2. Next, an overview is given of our methodology. This includes subsections on the problem definition, the simulation models, as well as an outline on our evolutionary mechanisms and evaluation tasks. In section 4, we discuss the experiment setup in each of our models: abstract and realistic, and we present our results and discuss them in detail in section 5. Lastly, conclusions and future work are explored in section 6.

*This work was supported by the HEA Ireland, through the ComputerDISC fund at the National University of Ireland, Galway.
The authors are with the College of Engineering and Informatics at the National University of Ireland, Galway. {j.holland3, c.gallagher41, josephine.griffith, colm.oriordan}@nuigalway.ie

II. Related Work

Over the past three decades there have been a number of experiments conducted to transfer control systems from simulation to reality, each with varying degrees of success. For instance, Miglino et al. [2], compare the fitnesses of the robots' controllers from the simulation to the controllers in reality. They argue that although the performance of the robots decrease in real-world experimentation, robust solutions can still be obtained by continuing the evolutionary process for a few more generations and thus, the closer the fitnesses become between simulation and reality, the better the transfer.

Jakobi et al. [4] take a different approach, whereby they observe whether the controllers behave qualitatively similar in both simulation and reality by adding different amounts of noise to the simulation. They found that in a zero noise environment, strategies tend to evolve incredibly well, or astonishingly poorly. However, when too much noise is added, the same genotypes achieve completely different scores on two identical evaluations. They concluded that a balance between these two levels of noise achieved the best results and that coincidently, the level of noise that favours the evolution of robust behaviours in simulation is also the level of noise that achieves the best transfer to reality. In contrast, Francesca et al. [3] found that by adding uniform noise to the robots' proximity, ground and light sensors, as well as wheel actuators, the robots' behaviours were strongly affected by interference among robots; robots created clusters, but due to the presence of noise, the robots dissipated after a while. When they transferred their simulation to reality they found that the robots interfered with each other less in simulation than in reality and that circular trajectories tended to have a larger radius in simulation than in real-world experimentation.

In contrast to these more general approaches, Jakobi et al. [5], [6] focus only on behaviours that they are interested in modelling, such as light-seeking behaviours, rather than in how every controller evolves. In other words, the experimenter only implements real-world features that are considered necessary for a successful transfer. In a similar vein, Pollack et al. [7] evolve the robots partly in simulation, but evolve the last generations on real robots. This approach assumes that the solutions' real behaviour and simulated behaviour are related, and in doing so suggests that the reality gap is quite small [8].

Hartland and Bredeche [9] propose a generic control architecture that relies on an anticipation module. In other words, the robots undergo online learning to anticipate locomotion perturbations in a simulated environment. Then, the error of the model is approximated with regard to the robot's behaviour in the real world in order to adapt correct motor outputs to comply with physical dynamics in the real world. Koos et al. [10] develop this concept further by proposing a transferability approach whereby the goal is to learn the differences between simulation and reality and to limit evolution to avoid behaviours that do not cross the reality gap successfully. This approach was successful in finding solutions in one hundred or less generations, but it failed to scale successfully if hundreds of generations were required. Silva et al. [11] carried out a similar approach, whereby they cut short the evaluation of poor controllers which proved effective in experiments with a high selective pressure, but arguably decreased diversity amongst the population.

Although a number of different approaches have been proposed to cross the reality gap, there has been no solution found that does this effectively and efficiently. This gap is a significant issue in evolutionary swarm robotics, preventing the smooth transition to real-world experimentation without the need for ad hoc tuning [12]. Thus, we wish to take a step back and look at the reality gaps observed in abstract and realistic simulation models in order to get an insight into how to minimise this problem on a larger scale; that is through real-world implementation.

III. Methodology

A. Problem Definition

In contrast to other approaches, we identify the reality gap by comparing two types of simulation models with varying levels of abstraction: abstract and realistic. The idea here is to carry out two different experiments with increasing difficulty and observe any differences found between these two models in order to identify the reality gap at simulation level.

Furthermore, we analyse the effect of noise on the realistic simulation model in two ways. First, we introduce noise after the robots' behaviours have been evolved in a noise-free environment; and second we explore the alternative setup where the behaviours are initially evolved in an environment with noise and then tested in an environment without noise. This is done to observe how the robots perform when evolved in an environment different to the resulting environment.

B. Evolutionary Algorithm

Evolutionary algorithms are very useful for synthesising efficient self-organizing behaviours that are simple, robust and flexible to changing environments [13]. These properties make problem solving somewhat easier as they can decrease the computational cost, and the time required to solve the problem. Furthermore, it is arguable that evolutionary algorithms work better in real-world optimisation, where constraints can be non-linear, conditions non-stationary and environments noisy [14].

Generally, a control architecture is used that has a high representational power, such as a neural network to fine tune the dynamics of the interaction between the robot and its environment. However, in uncertain and dynamic conditions where there are multiple robot-to-robot interactions, neural networks can prove counter-productive [3]. We use a look-up table (LUT) with a vector of different actions (speeds between 0-1cm/s), which are linked to the robot's actuators to evolve the robot's behaviours. More specifically, the left and right motor speeds are evolved using a genetic algorithm; these values determine how our controller reacts in the environment. In the abstract model, the LUT uses a vector of directions (left, right, straight) to evolve the agent's behaviours. The controller of the robot can be seen as a phenotype of the robot and it determines the robot's behaviour; it is this behaviour which is assigned a fitness value. The phenotypes are represented by genotypes that are subject to evolutionary operators. When the robot carries out a task it looks up a specific action in the LUT in order to respond to the current situation, and so, each row in the table represents a response to the stimuli.

The initial population is composed of 50 random genotypes, which are encoded into, and mapped to, the simulated robot's controller. Each run of the experiment lasts 50 generations. We use a simple generational model with tournament selection, elitism, crossover and mutation. Once every genotype of the population has been evaluated by means of a fitness function, two individuals are picked at random from the population and the fittest of the pair is chosen to act as a parent. The offspring are formed through a process of uniform crossover (crossover rate is 30%) and are subject to mutation at a rate of 2%. The process is repeated until suitable behaviours are evolved.

C. Simulation Models

In our experiments we use two models with different levels of abstraction, both of which use a simulator; one abstract and the other realistic. The reason we chose an abstract, or minimal simulation, is to demonstrate that we can evolve behaviours using a simulator that is easy to design and build. Furthermore, it uses a minimal set of features that a simulator needs to have in order to evolve a particular behaviour. Our abstract model comprises a simple 2D grid in which agents can achieve collective behaviours in discrete time steps, and in a clean environment without the disruption from noise. In this model, the agents' behaviours are evolved to determine the correct action to the encountered situation.

Our realistic model runs on ARGoS [15], an open-source 3D world, multi-robot simulator. This simulator is able to take into account physical laws relating to mass, friction and acceleration, and it can accurately simulate different sensor devices such as infra-red proximity and LED sensors. This tool has Kilobot [16] hardware compatibility, it provides Kilobot models with all the functionality of real Kilobots. We will use simulated Kilobots to model our agents. The simplicity of this robot can be used in our favour as simple forms of communication are enough to accomplish collective behaviours, and can also be easily scaled up with the number of agents involved. Furthermore, by using this simulator we can add noise to our sensors and therefore, make it more comparable to a real-world experiment.

1) Differences between Models: In order to accurately evaluate and compare each model, we have designed our models in such a way that there are as few differences between them as possible. However, we still need to be aware of any differences that may exist between them (Table 1). By doing this comparison, we can potentially find the source of any reality gaps that may occur, why they occur, and how to minimise them based on the different parameters identified.

TABLE I
OVERVIEW OF MODELS: PARAMETER SETTING COMPARISONS BETWEEN THE ABSTRACT AND REALISTIC MODEL.

	Abstract Model	**Realistic Model**
Arena:	2D	3D
Arena Size:	100cm×100cm	100cm×100cm
Environment:	Clean	Noisy
Movements:	Discrete	Continuous
Random Movement:	Uniform	Non-uniform
Time:	Discrete	Continuous
LUT Sizes:	2, 7	2, 7
Perception Shape:	Circle	Circle
Perception Range:	10cm	10cm

In the abstract model, we use a 2D grid to demonstrate a minimal set of features in a clean environment in order to minimise any potential noise variations in both models. Due to this simplicity, the agents' movements are discrete and uniform; agents move one grid space per iteration. This is in contrast to the realistic model where the movement is continuous and agents exhibit potentially imprecise movement.

In the realistic model, simulated robots move in a non-uniform manner. We vary the LUT size in each experiment for both models to two and seven to allow for a more diverse range of behaviours. Furthermore, due to the Kilobot's limited perception range of 10cm, we consider extended neighbourhoods in both models for the post-evolutionary analysis of the clusters formed; number of clusters, cluster size, and number of outliers. That is, if $agentA$ is within 10cm of $agentB$, they are direct neighbours. However, if $agentC$ is a neighbour of $agentB$, but not of $agentA$, $agentC$ is considered an extended neighbour of $agentA$ and thus, in the same cluster.

D. Evaluation Tasks

We define two evaluation tasks (achieved by modifying the fitness function) with increasing difficulty. The aim of choosing two tasks is to investigate whether there is some correlation between the level of difficulty of a task and the presence of a reality gap.

1) In our first task, a simple fitness function is adopted which rewards agents for being a member of a cluster. No attention is paid to the shape or size of the cluster. The agent/simulated robot is awarded a fitness score of one if it is in a cluster and is awarded a score of zero otherwise. Thus, the LUT size in this experiment is two; one index that represents the action corresponding to the detection of a neighbour, and another index that represents an action corresponding to no neighbour detection. The fitness of the population of N agents is given as:

$$F1 = \sum_{i=1}^{N} = 1|i \in cluster, \tag{1}$$

Thus, the selective pressure will be on the formation of clusters.

2) The second task develops the first fitness function further; in this task agents are rewarded based on the size of their neighbourhood rather than just being in a cluster. This promotes larger cluster sizes. This time, the size of our LUT is seven in order to account for the possibility of six neighbours[1].

$$F2 = \sum_{i=1}^{N} |neigh(i)|, \quad (2)$$

we denote the neighbourhood of agent i as $neigh(i)$. We count how many interactions i receives from neighbouring robots, and robots with a larger neighbourhood size achieve higher fitness scores.

IV. The Experimental Setup

This section describes the experimental set up for each model and task. Both subsections outline the technical aspects involved such as experiment length, number of iterations used, and agent position and orientation. Furthermore, in the realistic model, the addition of noise is described.

A. Abstract Model

In our abstract model, agents are randomly placed (in terms of both position and orientation), at the beginning of each generation, in a 2D 50×50 (equates to 100×100cm) grid arena. Each experiment lasts 50 generations with 250 iterations in each generation. Agents move one cell per iteration. The agent density over the entire grid influences how often agents encounter other agents which in turn affects the results. Thus, it is important that there is not too much of a difference in density between the two models. As Kilobots have a maximum speed of 1cm/s, the realistic model requires a 100cm×100cm grid size. Therefore, in this model, a grid size of 50×50, with an iteration every 2 seconds was used as it has the closest density to the realistic model.

Due to the random initial placing of the agents, we run each experiment ten times and chose the trial with the best evolved behaviours, according to the average group performance, to analyse further. The behaviours resulting from the best trial is then run a further ten times without evolution to determine the quality of the evolved behaviours. Furthermore, as this is an abstract model and we wish to maintain a clean environment, no noise is added to the evolutionary runs.

B. Realistic Model

In this model we also randomly seed the position of the robots at the beginning of each generation in a 100cm×100cm enclosed arena. We run the evolutionary process ten times and chose the best performing run (in terms of overall group behaviour); this gives a population that has obtained a high fitness. However, this gives us little insight into the actual phenotypic behaviour exhibited by the population.

With different seed values, we run the simulator (without evolution) a further ten times to explore the quality of the evolved behaviours; we visualise the behaviour and measure aspects of the resulting emergent behaviours (e.g. cluster size).

Gaussian noise is also used to simulate the imprecise motion of Kilobots in real word experimentation (σ = 0.4). This also helps us explore whether our approach leads to the evolution of robust behaviours that perform well in noisy environments.

[1]We use six neighbours to account for the possibility of busy communication channels/signal blocking.

V. Results and Discussion

We evaluated our experimental models using two methods. First, we look at the average group performance over ten individual runs. The group performance is the aggregation of the performance of the individual agents in each generation. The evolved behaviours from the best evolutionary run resulting in the best group performance are then explored further by running a number of simulations and measuring the resulting outcomes. Our population at any given time may have a diverse range of behaviours and so, by measuring the group's performance rather than an individual's performance, it demonstrates the group as a whole is robust to individuals with diverging behaviours.

Second, we measure the resulting outcomes (clusters) by using three different metrics: number of clusters present, the number of outliers (agents not in any cluster), and the average cluster size. These metrics give a quantitative measure of the clustering behaviour of the robots. This allows us to compare the outcomes across the two models. We can compare the resulting clusters formed in environments containing noise or environments with no noise. Moreover, we can compare the cluster formation caused by behaviours evolved in noisy and noise-free environments.

A. Task One

In this task, agents are rewarded for being in a cluster; no reward is given for the size of the cluster or the distance between each agent. The evolved behaviours for the population of agents in the abstract model converged to two behaviours: in the absence of a neighbour, randomly explore, and in the presence of a neighbouring agent, do not move. In contrast to this, in the realistic model, the absence of a neighbour led to high left and right motor speeds, albeit not equal, which allowed them to traverse the arena. When a neighbour was present, both motor speeds dramatically decrease, causing the simulated robots to move as a cluster.

Both models stabilise very early on in the evolutionary run, all before the 15th generation. Despite this stabilisation, the abstract model appears to do quite poorly in the first task (Fig.1 and Fig.2 (a)), with an average group performance 56% less than the realistic model with no noise, despite having the largest number of clusters (Fig.1 (b)). This is due to two reasons. First, the agents in the realistic model receive messages every second (in contrast to the abstract

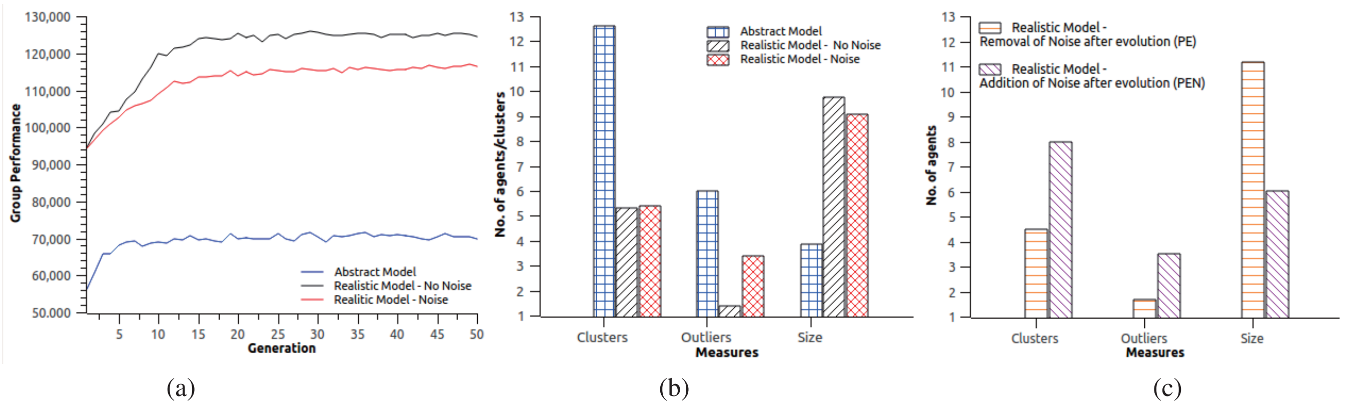

Fig. 1. Task 1: (a) Average Group Performance over 50 generations. (b) Comparison of models over different measures for best run: number of clusters, number of outliers, and average size of clusters. (c) Comparison of the addition/removal of noise post-evolution in the realistic model.

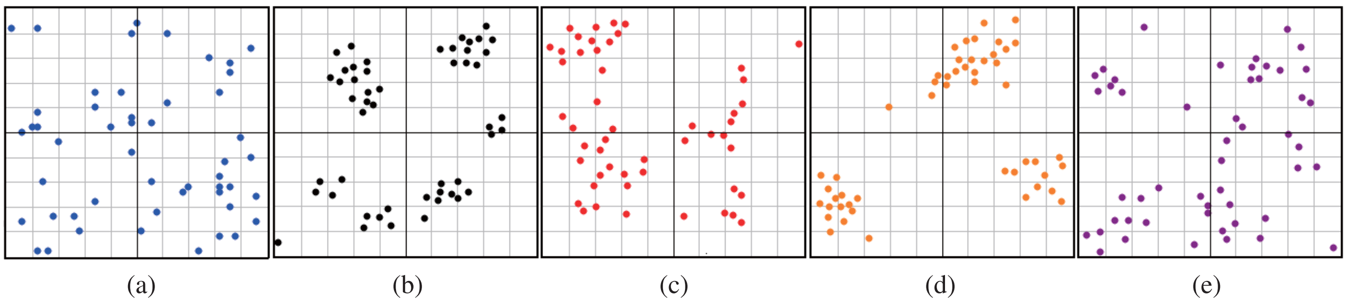

Fig. 2. Clusters formed at end of trials for all models in Task 1: (a) Abstract Model, (b) Realistic Model - No Noise, (c) Realistic Model - Noise, (d) Realistic Model - Removal of Noise after evolution (PE), (e) Realistic Model - Addition of Noise after evolution (PEN).

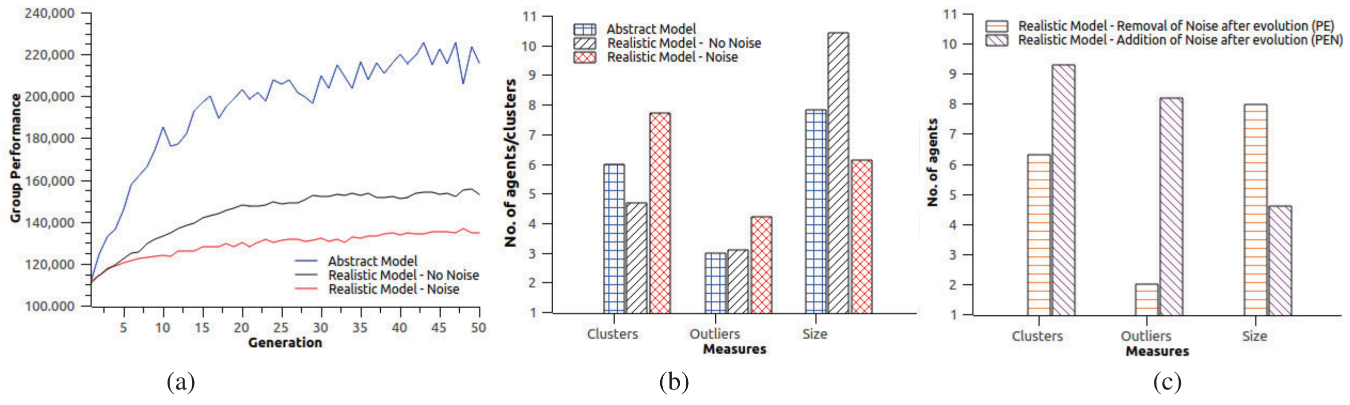

Fig. 3. Task 2: (a) Average Group Performance over 50 generations. (b) Comparison of models over different measures for best run: number of clusters, number of outliers, and average size of clusters. (c) Comparison of the addition/removal of noise post-evolution in the realistic model.

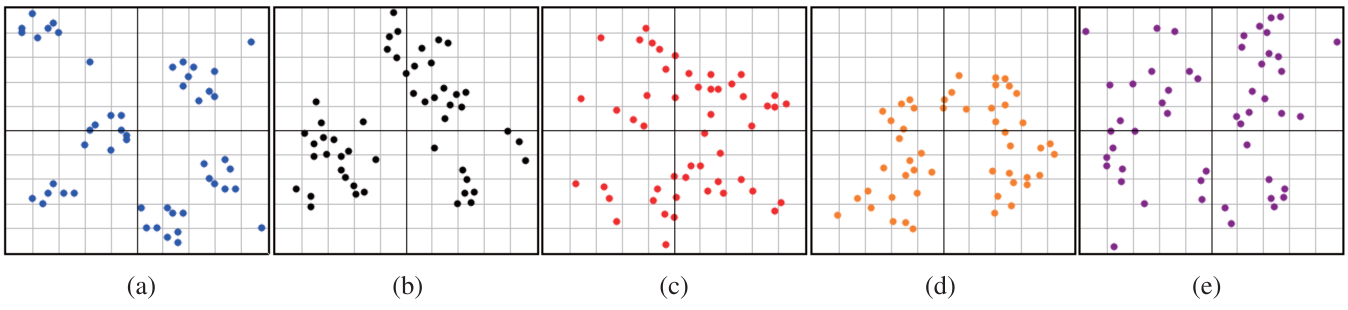

Fig. 4. Clusters formed at end of trials for all models in Task 2: (a) Abstract Model, (b) Realistic Model - No Noise, (c) Realistic Model - Noise, (d) Realistic Model - Removal of Noise after evolution (PE), (e) Realistic Model - Addition of Noise after evolution (PEN).

model, which has an iteration every two seconds), and so, has a higher sampling frequency. This in turn leads to a higher group fitness score as due to the limitations of the sensing abilities of real Kilobots, a Kilobot can only sense its neighbours and not their relative position or orientation. So, at any time interval we may overestimate the fitness value, as if an agent is a member of a cluster of size N over a period of time, we may count that agent as having membership to $N-1$ clusters. Hence the analysis of the clusters following a run is far more meaningful.

Secondly, in the abstract model, agents stop moving when they find a neighbour. In contrast, in the realistic model, agents decrease their left motor speed dramatically, but still continue to move. This would suggest that clusters generally move, and due to the further exploration, have a higher probability of gathering a larger neighbourhood size (Fig.1 and 2 (b)). As the clusters are larger in the realistic model, agents have a higher probability of finding another neighbour when exploring the arena, in contrast to the realistic model where agents stop moving when a neighbour is found. Thus, it could be argued that if the experiment length were to be longer, the agents in the abstract model would have a greater chance of finding a neighbour, which would in turn, lead to a smaller number of outliers. Similarly, the noise in the realistic model also affects the number of outliers exhibited; motors are slower for the index of the LUT that corresponds to having no neighbour. This limits the amount of exploration that the agents can achieve, and thus, leads to a higher number of outliers.

Lastly, we experimented with noise in this task in two ways. First, we evolved the robots behaviours within an environment with noise, but then tested the evolved behaviours in an environment without noise (Fig.1 (c) and Fig.2 (d)). From now on we will refer to this experimental setup as *PE*. Then, we evolved the robots behaviours in an environment without noise, and then tested the resulting evolved behaviours in an environment with noise (Fig.1 (c) and Fig.2 (e)). From now on we will refer to this experimental setup as *PEN*. As expected, the behaviours exhibited in *PE* and *PEN* performed the best and worst overall, respectively. That is to say, the robots that were subject to noise during their evolution performed the best when the noise was removed post-evolution as they had learned to deal with a noisy environment. The behaviours in *PE* outperform the realistic model with no noise (Fig.2 (c)) on all counts; number of clusters/outliers and cluster size. On the other hand, the robots that were not subject to noise during their evolution performed poorly when noise was added; the number of outliers doubled and cluster sizes decreased by 35.7% on average.

B. Task Two

In the second task, agents are rewarded based on the size of their neighbourhood, and so, ideally we wish to see a small number of clusters, few outliers and a large neighbourhood size. The agents in the abstract model have the highest group performance score, 1.55 times greater on average, and appear to be still evolving by the 50th generation (Fig.3 (a)). However, it appears that this model falls short in one case: average cluster size (Fig.3 (b)) despite evolving the behaviours with the highest rewarded score by the fitness function.

This is due to two reasons. First, just as with *F1*, the agents in the abstract model stop moving after they have reached their maximum neighbourhood size. This results in an average of six clusters with a size of eight agents, but clusters still remain further away from each other as agents are no longer under any evolutionary pressure to explore the arena any further. Again, agents in the realistic model have evolved behaviours whereby their speed is decreased with the more neighbours they have and they move slowly towards other clusters due to a higher left motor speed, resulting in larger clusters. This is interesting, because it appears that the jump from abstract to realistic model leads to very similar evolved behaviours, but substantially different emergent artefacts (clusters).

Second, there is a potential to over estimate cluster sizes. That is, Fig. 4 (b-e) are made up of groups of small clusters that result in one larger cluster; that is *agentC's* neighbour is *agentA's* neighbour through extension. In saying that, it can be argued that due to busy communication channels and blocking of signals, the simulated Kilobots are unable to receive more than four or five signals at any one time. This would also explain the average number of clusters at any given time (Fig. 3 (b)).

The evolved behaviours of *PE* and *PEN* perform similarly to the behaviours with *F1*. Although *PE* has the least amount of outliers on average, it does not have as high a cluster size as the realistic model with no noise (Fig.3 (b)). On the other hand, the evolved behaviours exhibited in *PEN* are very poor, with an average of 8.2 outliers, and a cluster size of 4.6. Furthermore, in comparison to the realistic model with noise, the amount of outliers are 50% less and the cluster size is 26% larger. This demonstrates that the evolved behaviours in this task do not perform well when noise is added to the environment after the evolution of the simulated robots' behaviours. This confirms that even the smallest addition of noise affects the evolved behaviours, and that a small step-up in task difficulty plays a large part in the reality gap at a simulation level.

VI. CONCLUSIONS

In this paper we explored the distinction between reproduction of behaviour and performance in abstract and realistic simulation models. We demonstrate that our approach successfully evolves clustering behaviours in simulated Kilobots in both models, and demonstrates that they are robust when evolved with noise. However, as expected, the evolved behaviours give the best and worst performance when evolved in an environment different to the resulting environment. We have identified reality gaps between abstract and realistic models through the evolution of clustering behaviours with tasks of varying levels of difficulty. This not only illustrates that the gaps are present, even at very high level, but it also demonstrates that there is a correlation between the level of difficulty of a task and the presence of a reality gap.

A possible avenue to pursue in the future is to compare the evolved robots' behaviours with noise in simulation to evolved behaviours without noise in real-world experimentation in order to identify any further gaps. We would also like to compare the two models presented in this paper with a real-world Kilobot model in order to explore the reality gap exhibited in this paper further. Lastly, we wish to move from the abstract model to the realistic model by increasing the notion of realism.

REFERENCES

[1] R. A. Brooks, "Artificial life and real robots," in *Proceedings of the First European Conference on artificial life*, 1992, pp. 3–10.

[2] O. Miglino, H. H. Lund, and S. Nolfi, "Evolving mobile robots in simulated and real environments," *Artificial life*, vol. 2, no. 4, pp. 417–434, 1995.

[3] G. Francesca, M. Brambilla, A. Brutschy, V. Trianni, and M. Birattari, "Automode: A novel approach to the automatic design of control software for robot swarms," *Swarm Intelligence*, vol. 8, no. 2, pp. 89–112, 2014.

[4] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *European Conference on Artificial Life*. Springer, 1995, pp. 704–720.

[5] N. Jakobi, "Evolutionary robotics and the radical envelope-of-noise hypothesis," *Adaptive behavior*, vol. 6, no. 2, pp. 325–368, 1997.

[6] N. Jakobi, "Minimal simulations for evolutionary robotics," Ph.D. dissertation, University of Sussex, 1998.

[7] J. B. Pollack, H. Lipson, S. Ficici, P. Funes, G. Hornby, and R. A. Watson, "Evolutionary techniques in physical robotics," in *International Conference on Evolvable Systems*. Springer, 2000, pp. 175–186.

[8] S. Koos, J.-B. Mouret, and S. Doncieux, "Crossing the reality gap in evolutionary robotics by promoting transferable controllers," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, 2010, pp. 119–126.

[9] C. Hartland and N. Bredeche, "Evolutionary robotics, anticipation and the reality gap," in *Robotics and Biomimetics, 2006. ROBIO'06. IEEE International Conference on*. IEEE, 2006, pp. 1640–1645.

[10] S. Koos, J.-B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 1, pp. 122–145, 2013.

[11] F. Silva, L. Correia, and A. L. Christensen, "Leveraging online racing and population cloning in evolutionary multirobot systems," in *European Conference on the Applications of Evolutionary Computation*. Springer, 2016, pp. 165–180.

[12] F. Silva, M. Duarte, L. Correia, S. M. Oliveira, and A. L. Christensen, "Open issues in evolutionary robotics," *Evolutionary Computation*, vol. 24, no. 2, pp. 205–236, 2016.

[13] V. Trianni and S. Nolfi, "Engineering the evolution of self-organizing behaviors in swarm robotics: A case study," *Artificial life*, vol. 17, no. 3, pp. 183–202, 2011.

[14] D. B. Fogel, "The advantages of evolutionary computation." in *BCEC*. Citeseer, 1997, pp. 1–11.

[15] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, *et al.*, "Argos: a modular, multi-engine simulator for heterogeneous swarm robotics," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 5027–5034.

[16] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3293–3298.