THE EFFECT OF CHANGE IN EVOLUTION PARAMETERS ON EVOLUTIONARY ROBOTS

Shanker G R Prabhu*, Richard Seals^
University of Greenwich
Dept. of Engineering Science
Chatham, Kent, UK, ME4 4TB.
+44 (0) 1634 88 3838*, 3953^
S.RadhakrishnaPrabhu*,R.Seals^@gre.ac.uk

ABSTRACT
One of the recent trends of robot design involves the evolution of morphology and controller of robots using techniques from evolutionary computation. In this co-evolution process, the evolution system utilises the stochastic and heuristic nature of artificial evolution to evolve robots for various tasks. Inspired by natural evolution, a population of initial solutions is randomly created and selected parents are mated to produce offspring. Based on the performance or fitness of individual solutions including children, next generation is chosen and this process continues until a solution of satisfactory performance is reached. Among various methods of evolution, Genetic Algorithms (GA) are commonly used for co-evolution. In this paper, the effect of change in various evolution parameters in the GA on the final solution is studied. Parameters such as size of population, number of generations evolved and several variation parameters are varied. Robots are evolved from a specific set of parts which includes various structural components, active and passive joints and sensors.

Keywords
Evolutionary robotics; Co-evolution; Morphology; Controller design; Evolutionary-aided design

1. INTRODUCTION
Evolutionary Algorithms (EAs) are a part of Evolutionary Robotics. It is the area of robotics that deals with application of the concept of biological evolution process to search for solutions in robotics [1]. Robotics found the first application of EAs for sensor positioning on a mobile robot in the early 90s [2]. Since then, they were regularly used to evolve the robot body plan or referred to as robot morphology, robot controller or both morphology and controller. While more than 95% of reported applications were in designing a controller for the robot, only about 1% seemed to show positive findings while using EAs for the co-evolution process [3]. Further, the latter has only been able to evolve robots purely for locomotion with simple obstacle avoidance.

Developed in 2014, RoboGen is an open-source evolution platform that can evolve mobile robots for primitive locomotion tasks [4]. It is a package capable of handling the co-evolution process of evolving complete virtual robots. It runs an evolution engine and simulation engine side by side with data transferred multiple times during the evolution process. The evolution engine performs the primary steps involved in the evolution process and the simulation engine estimates the performance of each evolved individual.

Due to multiple factors such as costly computation requirements, large number of adjustable variables, and their random nature, the applications of EAs have been constrained to highly targeted design and optimisation problems. For instance, EAs were applied to perform only wing design [5], optimise robot arm lengths [6] and vary robot shape parameters [7]. The key applications and advantages of EAs in robotics are discussed in [8], [9], [10], [11]. It can be safely stated that for improving the evolution process to evolve buildable robots, we need to have a better understanding of the process itself. Therefore, in this paper, efforts have been undertaken to study and analyse the behaviour of evolved robots under various conditions.

2. THEORY AND EXPERIMENT SETUP
A co-evolution process involves evolution of the robot morphology and controller for a specific application. In this work, robots are evolved to evade obstacles and cover as much distance as possible in a chosen time frame. The software package, RoboGen evolves a robot from a list of available parts namely, a core component brick that houses an IMU (Inertial Measurement Unit), controller and battery, a fixed brick, a parametric bar joint with variables to configure the arm length and tilt angle, an active servo motor driven joint, a passive hinge, an IR (Infrared) based distance sensor and a light sensor. The core and fixed component can connect up to four parts while all other parts allow only connections on both sides. Robot body plans in Table 2 and part details in Table 3 can be referred for obtaining a better understanding of how the robots are visualised. As per input parameters, it evolves robots and with the help of a physics based simulator, each robot is evaluated individually. The 3D printable robot part files and controller code are generated finally to physically test the robot. The simulations were performed on a Linux PC with an Intel i7 dual core 2.50GHz processor.

The EA used to evolve morphology in RoboGen is a Genetic Algorithm (GA) with a tree based representation of the phenotype. A phenotype refers to the physical representation of the robot where the observable characteristics of the robot are seen and a genotype refers to the internal representation of genetic information just as in biology. The GA works by randomly initialising a fixed population of parents (μ) and evaluating them as per application through a fitness function. After the fitness evaluation process is completed, the population is randomly divided into λ groups of two and the best individual in each group is chosen as a parent. This selection method is called a deterministic tournament strategy. Later, on these selected parents, various mutation operators are applied. Mutation refers to the changes applied through several ways to the parent to evolve an offspring. As per the set probabilities and Gaussian distribution in the evolution configuration file, operations such as addition and deletion of parts, modification of parametric variables, duplication, swapping and removal of sub-tree are performed on the robot tree. The mutated children are then added on to the population and the entire population is ranked according to their fitness and the best μ individuals are retained and the rest are deleted from the population. This method is a (μ+λ) evolution strategy where μ is the parent size and λ is the number of children [12].

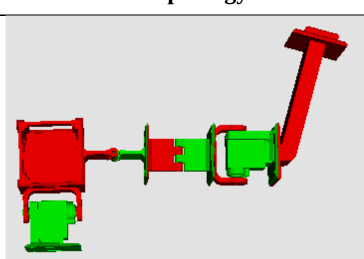
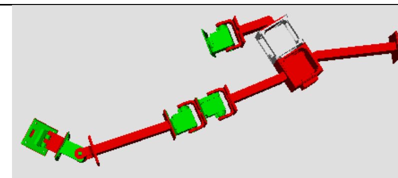
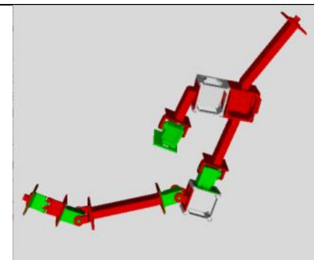
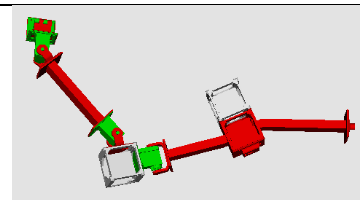
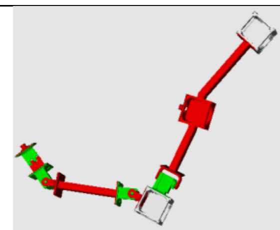
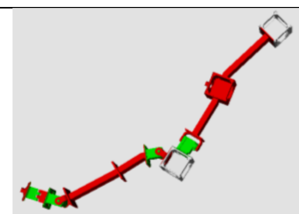
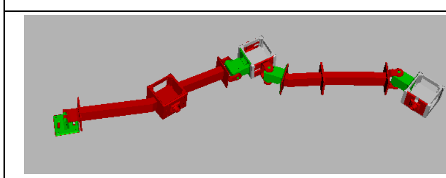
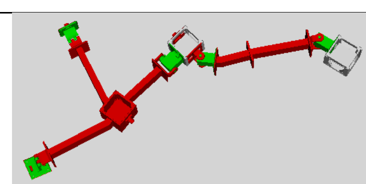
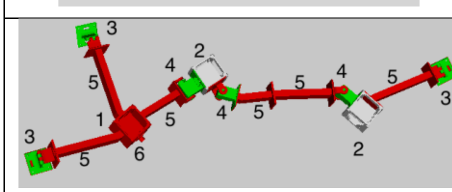
A similar process is performed on the neural network oscillator controller to evolve controller for each evolved body. Oscillatory neuron controller has been previously proved to be better than standard Artificial Neural Network (ANN) controller in the co-evolution process to evolve controllers to robots [13]. An oscillatory neural network is a variation of a standard artificial neural network with oscillators acting as signal generators along with inputs from sensors while generating motor control signals. Probability and bounds are set for mutation of period, phase and amplitude of oscillator, neurone bias and weights. Sample evolution configuration parameters which are used as default values during the experiment are listed in Table 1.

Table 1. Evolution parameters.

Parameter	Value
Population size	20
Number of evolved children	20
Number of generations	100
Probability of brain mutation	0.3
Sigma value of brain	0.7
Brain Bounds	3:3
Minimum and maximum number of initial parts	2:10
Probability of node insertion	0.1
Probability of sub-tree removal	0.1
Probability of duplicating sub-tree	0.1
Probability of swapping sub-tree	0.1
Probability of node removal	0.1
Probability of modifying parameters	0.1

The fitness function evaluation begins by recording the velocities and distance sensor values in every simulation step. An increase in movement speed is encouraged while proximity to obstacles is discouraged. At the end of the simulation, the recorded values are used to calculate the final fitness value which eventually is the best fitness calculated from the list of individual stepwise fitness calculations. This calculation is performed on all individuals of the solutions population. This is the most computationally demanding step as each robot is evaluated separately in a virtual 3D environment.

Table 2. Stages of evolution.

Morphology	Details
	Generation- 1 Fitness- 0.0001 No. of parts- 7
	Generation- 2 Fitness- 0.13 No. of parts- 13
	Generation- 7 Fitness- 0.39 No. of parts- 15
	Generation- 19 Fitness- 0.61 No. of parts- 12
	Generation- 61 Fitness- 0.91 No. of parts- 12
	Generation- 79 Fitness- 1.06 No. of parts- 13
	Generation- 3098 Fitness- 1.80 No. of parts- 14
	Generation- 3203 Fitness- 1.84 No. of parts- 15
	Generation- 37570 Fitness- 2.72 No. of parts- 16

The effectiveness of the evolver is only as good as its fitness evaluating platform. Consequently, the simulator plays an important role in the entire process. In the experiments performed unless mentioned, each robot was run for eight seconds in the virtual environment with a flat surface and readings are recorded every 0.005s to avoid any considerable loss of data. Though the experiments could have been performed on a more accurate scale by increasing the step resolution and simulation time, this combination seemed sufficient to generate reasonable output results.

Table 3. Part details.

Sl. No.	Part type	No. of occurrences	Colour
1	Core brick	1	Red
2	Fixed brick	2	Grey
3	Passive joint	3	Red-Green
4	Active joint	3	Red-Green
5	Parametric bar	6	Red
6	Light sensor	1	Red
7	IR distance sensor	0	

To confirm the physical buildability of the evolved robot, multiple design constraints are applied during the evolution process. They are, discarding robots whose parts intersect with each other, include only one core part as there needs to be only a single controller and to also satisfy the maximum I/O ports requirements of the controller board by allowing only up to three sensors and eight motors during the evolution process.

3. EXPERIMENTS

The evolution parameter values applied are listed in Table 1. As per each experiment, μ, λ, number of generations and the maximum initial parts available for evolution were suitably modified. Multiple experiments were designed specifically to observe the effects of variations of individual parameters in the fitness of the robot.

3.1 Generations

To study the effect of generations on the fitness value, the population size was fixed at 20 and with a maximum of 20 initial parts, robots were allowed to evolve on a flat surface for 37,000 generations. Fig. 1 shows the improvement in best individual's fitness and average fitness of the entire population as the generations progressed. It was observed that the evolution of robot morphology was extremely slow with the final robot shape (the last robot shown in Table 2) remained so in the last 32000 generations where the fitness increased from around 1.8 to 2.7. The number of parts were 16 and 7 in the last and first generations respectively. The changes observed to the morphology as the generations progressed is shown in Table 2. Even though generations evolved for more than 37,000 the morphology change was observed only 9 times. To help comprehend various parts of the evolved robot, the parts and their positions in the final evolved robot (last figure in Table 2) is listed in Table 3.

In the next set of experiments, the number of evolved generations was varied keeping all other parameters constant. Multiple experiments with the maximum generations doubling in every experiment from 100 to 35,000 was performed. It was observed that the fitness values were repeatable until about 10,000 maximum generations after which, there was a drastic difference between corresponding fitness values from experiments run for less than 10,000 generations and more than 10,000 generations. However, there seemed to show repeatability of output when the same experiment was run multiple times.

Being a stochastic process, the evolution is initiated by a seed number for the random number generator. In all the experiments above, the seed was set at 1. However, it was also found that changing the seed meant loss of repeatability of the experiments which is an advantage of evolutionary algorithms. Though multiple experiments were performed to see the effect of change in the seed number to the evolution process, the results exhibited complete randomness and therefore those observations are not reported here.

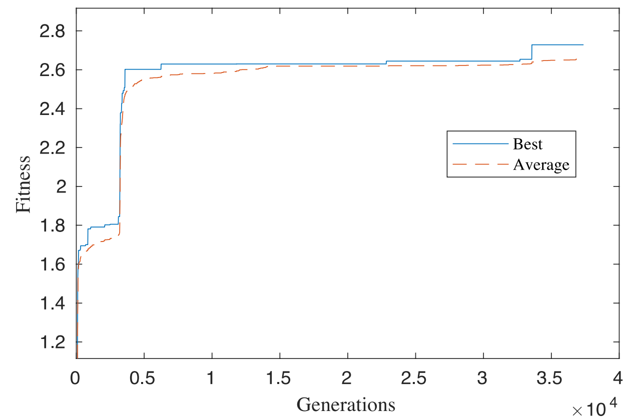


Figure 1. Fitness of best robot and average fitness of population versus generations

The average fitness of all the members in the population showed an expected deviation from the best individuals initially as seen in Fig. 1. But, as the fitness of the best individual settled, the average fitness also moved towards the best fitness. There were multiple occasions during the experiments where the standard deviation of the population converged to zero. This meant that all the individuals in the population were identical.

3.2 Initial Parts

The initial number of parts allowed to be used for robot building also played a role in behaviour of the robot. In the experiments conducted, the range of maximum number of initial parts were varied from 10 to 100. Even though they seemed to have a clear impact on the fitness progression, the plots did not exhibit any patterns. As shown in Fig. 2, experiment with 80 initial parts showed the least increase in fitness rate over the period of the experiment. It was followed by experiment with 100 initial parts and experiment with 50 initial parts showed the best overall rate of increase.

In the initial 50 generations, the rate of increase of fitness exhibited a different pattern with experiment evolving from 70 initial parts showed fastest speed followed by experiments with 60 and 90 initial parts. The lowest rate of change was exhibited by experiment with 10 initial parts followed by 80 initial parts. Experiment with 60 initial parts was fastest to settle down in ±5% of its final fitness followed by experiments with 90 and 100 initial parts. Despite allowing the use of a particular number of initial parts, the experiments performed showed random initial parts in the actual evolved robot in the first generation. There was also random increase or decrease of parts on the robot as the generations progressed. This can be noted from the data in Table 4. The best fit robot in the set of experiments were seen in the experiment with

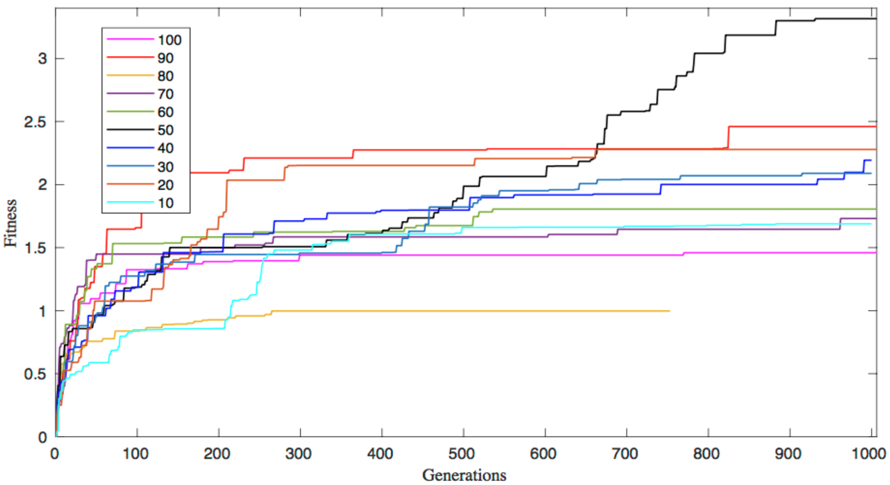


Figure 2. Fitness change to initial number of parts.

Table 4. Parts numbers and fitness in different experiments.

Max. initial parts allowed	Initial number of parts	No. of parts at 1000 gens.	Best fitness at 1000 gens.
10	10	7	1.6
20	7	21	2.3
30	17	22	2.1
40	17	16	2.1
50	37	32	3.3
60	23	14	1.7
70	20	19	1.6
80	5	70	1
90	20	22	2.4
100	74	19	1.4

50 initial parts and worst performing individuals were found in experiment with 80 initial parts.

3.3 Population size

The population size was varied from 20 to 100 members and experiments were run for over 10,000 generations. As expected, parts with 100 individuals stabilised first to a fitness value of 2.1 in about 2000 generations while the 20-member sized population needed the maximum time to reach close to 2.1. It can be seen from the curves in Fig. 3.

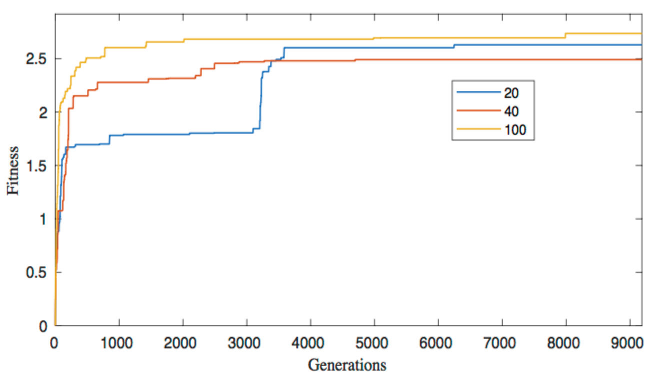


Figure 3. Fitness variation to population size.

3.4 Obstacles

To understand how robots behave when they are placed in a different environment with multiple obstacles, robots were first evolved with just a few obstacles as shown in Fig. 4 (a). This was expected to help the robot evolve with the obstacle sensors. It's travel route was then recorded (red lines in Fig. 4 (a)) and in the next experiment, the same robot was placed in a different setup with new obstacles. It was observed that the robot was able to perform minor course corrections. The corrected course along with new obstacle positions are shown in Fig. 4 (b).

To evolve robot in a complicated arena, an experiment was designed to evolve a robot in a maze shaped arena. Though the robots were allowed to evolve for 3000 generations with a 100 seconds window for every robot to cover the arena, the best fit individual showed a fitness of 0.64 and could just exit the central area. It was unexpected to noted that the evolved robot did not appear to have any distance sensors. The route taken by the best robot to solve the maze is shown in Fig. 4 (c).

3.5 Child population size

The number of children evolved at the end of each generation was varied to see its effect on the fitness. The child population was incremented from 10 to 40 with the parent population fixed at 40 individuals. The best fitness and average fitness of the population in each case is marked by the curves in Fig. 5. The slowest to increase the best fitness value was the population generating 20 children. It was followed by 10 and 30 child populations. The best performance was shown by population evolving 40 children. The tendency for the average fitness value of the entire population to gradually touch the best fitness value is also seen as in previous cases.

4. DISCUSSIONS

The experiments reported above offer multiple suggestions and insights to the evolution process. Despite the evolution process was performed on a 4-thread processor with a thread handling evolution and the other three threads performing the fitness analysis in parallel, the experiments took a few hours to even days in most instances. The population size, number of parts, number of generations run and simulation time were the major factors in determining the time taken. Ultimately, they underline that evolution is a time consuming and computationally expensive process.

Among remarks pertaining to the evolution parameters, the part number had a clear relation with the corresponding fitness of the robot (evident from Table 4). The initial number of parts available for building the first population had an effect on the progress of the population. Though the relationship is not exactly clear, there seems to be a correlation between the fitness, part number and population size.

(a) **(b)** **(c)**

Figure 4. Obstacle avoidance trajectories of evolved robots.

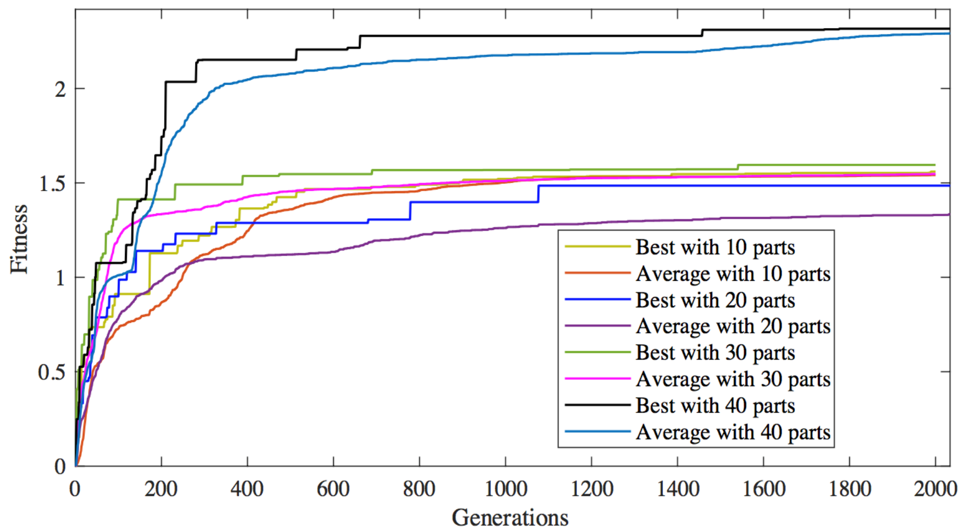


Figure 5. Fitness variation to child population size.

The evolution process showed focus mainly on evolving controller than evolving morphology. In the entire evolution procedure, the robot morphology was altered 4 to 5 times in the first 100 generations and only a few times later depending on how far the evolution was run. This may be due to the low probability values set for body mutations. Further, the same might have also caused the population to evolve identical individuals that later gradually reduced the diversity of the population and ultimately reduce the standard deviation of the population to zero. In the current setup, robots are always evaluated as a single entity without looking at the body plan and controller separately. This is a widely-accepted technique and has advantages. However, it could be time to explore other practices.

The fitness of populations exhibited a step by step improvement in all the experiments. The same trend was followed by the average fitness curves too. There have even been cases where the standard deviation of the population was consistently equal or close to zero. This suggests a lack of diversity among individuals in the population despite it is mathematically possible to have extremely high possible combinations of part connections depending on the parts limit set. To avoid solutions being stuck in the local maximum, various probabilities involved could also be altered.

The experiments also indicate that the best value for the number of children evolved at the end of every population was equal to the population size itself. Further, it should be noted that in the population updating strategy, the offspring and parent population are equally considered for transfer to the next generation which is unlike in biological systems where parents are always discarded. Among the experiments performed, the poorest performing experiment was the one which evolved half its population size.

After extensive simulations, it can be stated without a doubt that, there is strong need for more research to be performed to improve the effect of EAs on the co-evolution process. Every aspect of the evolution process from population initialisation, controller type selection, fitness function design to EA applied should be individually studied and optimised or modified to reduce the time consumed and evolve better results.

While the process of simultaneously evolving the robot body and controller has been attempted since 1994 [14], the effectiveness of the process is still questionable. After days of evolution, the best robot evolved to transverse through the maze shown in Fig. 4 (c) was just able to move out of the centre. It also lacked obstacle sensors which were a primary requirement to detect the obstacles. Instead of that, the robot focussed on remembering the trajectory than taking decisions based on sensory feedback. The trend of repeating trajectory was also observed in other experiments (Fig. 4(a) and (b)). About the evolved controller, though the oscillatory controller helped the evolved robots to start moving from early generations itself, there did not seem any improvement in the obstacle avoidance capabilities of the evolved controller. However, it may be argued that due to not choosing optimum parameters of the EAs, the output did not seem to be satisfactory. In cases where suitable sensors were added, there did not seem to be any guarantee on whether they were being used or not. This is a common problem with artificial intelligence based controller methods where it is extremely tough to interpret the internal wiring of the controller. All these demonstrate that EAs work and can evolve solutions to problems but not necessarily intelligent solutions.

5. CONCLUSIONS

EAs are known for evolving unintuitive solutions to problems and have been helping designers arrive at solutions to complex problems and there are multiple advantages of the using EAs in robotics. Further, in this paper, they have exhibited satisfactory results in evolving robots to perform repetitive actions like following a same set of steps with minute changes allowable in real time. But the question to be asked is if such an evolution process can outperform the current system of individual manual robot programming. Even though the answer to it may not be positive at least for now, it can be hoped that the full benefit of artificial intelligence based EAs for the co-evolution process is something to look forward to in the future.

6. FUTURE PLAN

In the immediate future, the first step will be to 3D print the evolved robot bodies and test their performance with the virtually evolved robot to confirm the accuracy of the simulations. Steps will also be taken to perform controller only evolution of robots with the morphologies evolved in the above experiments. This should shed light in evaluating the performance of a purely EA and ANN based control system. Among other tasks, the evolution probabilities will be altered as an attempt to improve the evolution process and the use of High Performance Computing (HPC) for evolving the robots will also be explored.

7. REFERENCES

[1] Nolfi, S., and Floreano, D. 2000 *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines,* MIT Press, Cambridge, MA, USA. DOI= 10.1162/106454601317297031.

[2] Cliff, D., Husbands, P., and Harvey, I. 1993 Explorations in Evolutionary Robotics, *Adapt. Behav*. 2 (1993) 73–110. DOI=10.1177/105971239300200104.

[3] Nichele, S. 2015. The coevolution of robot controllers (" brains") and morphologies (" bodies")–challenges and opportunities, (2015). DOI= http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.696.2391.

[4] Auerbach, J., Aydin, D., Maesani, A., Kornatowski, P., Cieslewski, T., Heitz, G., et al., 2014. RoboGen: Robot Generation through Artificial Evolution. In *Proceedings of the Alife*, (2014) The MIT Press, 136–137. DOI= 10.7551/978-0-262-32621-6-ch022.

[5] Shim, Y. S., Kim, S. J., Kim, C. H., 2004. Evolving flying creatures with path-following behavior. In *Proceedings of International Symposium on Computational Intelligence in Robotics and Automation* (2004), 125–132.

[6] Rubrecht, S., Singla, E., Padois, V., Bidaud, P., de Broissia, M., 2011. Evolutionary Design of a Robotic Manipulator for a Highly Constrained Environment, In *New Horizons in Evolutionary Robotics*, Springer Berlin Heidelberg, Berlin, Heidelberg, (2011), 109–121. DOI= 10.1007/978-3-642-18272-3_8.

[7] De Beir, A., and Vanderborght, B., 2016. Evolutionary method for robot morphology: Case study of social robot probo, In *Proceedings of ACM/IEEE International Conference on Human-Robot Interaction*, (Apr. 2016) 609–610. DOI= 10.1109/HRI.2016.7451880.

[8] Eiben, A.E., Smith, J., 2015. From evolutionary computation to the evolution of things, *Nature*. 521 (2015) 476–482. DOI= 10.1038/nature14544.

[9] Bongard, J. C., 2013. Evolutionary robotics, *Com. of the ACM*. 56 (2013) 74–83. DOI= 10.1145/2493883.

[10] Nolfi, S., Bongard, J., Husbands, P., Floreano, D., 2016. Evolutionary Robotics, In *Springer Handbook of Robotics*, Springer International Publishing, Cham, (2016) 2035–2068. DOI= 10.1007/978-3-319-32552-1_76.

[11] Bongard, J., 2014. Why Morphology Matters, In *The Horizons of Evolutionary Robotics*, MIT Press, MA, USA, 125–152. DOI= 10.1017/cbo9781107050419.010.

[12] Eiben, A.E., Smith, J.E., 2015. *Introduction to Evolutionary Computing*, Springer Berlin Heidelberg, Berlin, Heidelberg. DOI= 10.1007/978-3-662-44874-8.

[13] Auerbach, J. E., Heitz, G., Kornatowski, P., Floreano, D., 2015. Rapid Evolution of Robot Gaits, In *Proceedings of Genetic and Evolutionary Computation Conference*. ACM Press, New York, USA, (2015), 743–744. DOI= 10.1145/2739482.2764894.

[14] Sims, K., 1994. Evolving virtual creatures, In *Proceedings of Siggraph*, ACM Press, New York, New York, USA, (1994) 15–22. DOI= 10.1145/192161.192167.