The Fifteenth International Symposium on Artificial Life and Robotics 2010 (AROB 15th '10),
B-Con Plaza, Beppu,Oita, Japan, February 4-6, 2010

A Framework for Embodied Evolution with Pre-evaluation Applied to a Biped Robot

Junichi Nakai Takaya Arita
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
E-mail: nakai@alife.cs.is.nagoya-u.ac.jp, arita@nagoya-u.jp

Abstract: "Embodied evolution (EE)" is a methodology in evolutionary robotics, in which, without simulations on a host computer, real robots evolve on the basis of the interactions with actual environment. However, we had to accept robot behavior with low fitness especially in the early generations when adopting EE. This paper introduced pre-evaluation into the EE framework for a biped robot in order to restrain robot behavior of which fitness is estimated to be low, especially, falling down to the ground. We provide a comparative discussion on the conventional simulate-and-transfer method, the original EE method and the proposed one in terms of calculation time, cost of fitness evaluation and cost of simulation or modeling based on the evaluation experiments. We believe that the EE framework with pre-evaluation is applicable to a wide variety of optimization tasks in which the cost or risk of fitness evaluation is not negligible.

Keywords: evolutionary robotics, embodied evolution, biped robot, coevolution, genetic algorithm

1 Introduction

Evolutionary robotics is a challenging technique for creation of autonomous robots based on the Darwinian principle of natural selection [1]. In the conventional evolutionary robotics, the "simulate-and-transfer" method has been adopted, in which a controller is developed using an evolutionary algorithm on a host computer and then the solution is transferred to a physical robot. However, some issues are increasingly problematic for the method: performance of evolved behavior tends to be less than expected owing to the gap between simulation and the real world. Also, it is necessary to model the environment every time when a new task is given.

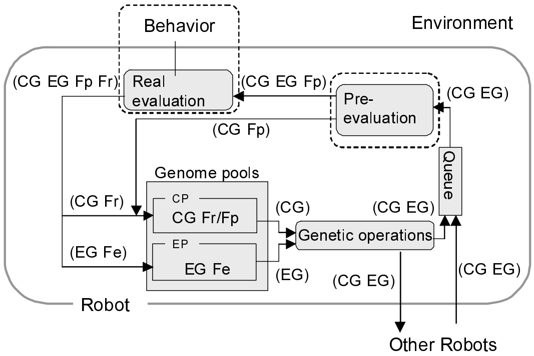
Pollack et al. proposed "embodied evolution (EE)" for solving these issues, in which real robots evolve based on the interactions with actual environment without simulation on a host computer [2]. Usui and Arita extended the framework by providing each robot with an evolving population of virtual individuals in order to reduce the dependence of the number of the robots and of the frequency of encounter with other robots on the speed of evolution [3]. However, these studies brought in new issues. We have to accept robot behavior with low fitness especially in the early generations. Also, evaluation based on robot behavior needs longer convergence time in general.

This paper introduces pre-evaluation into the EE framework so as to restrain robot behavior whose fitness is estimated to be low and evaluates the proposed architecture that is applied to a humanoid robot. The mechanism for pre-evaluation is introduced as a coevolutionary system in the proposed architecture. In other words, both the robot behavior and evaluation of the robot behavior evolve while interacting with each other. We expect introduction of pre-evaluation to decrease the number of falls in the early stage of evolution of biped-walking.

2 Framework

Fig. 1 shows the framework for Embodied Evolution with pre-evaluation. A set of information specifying the robot controller (Controller Genome, CG) and its fitness value evaluated by robot behavior (Fr) or estimated by pre-evaluation (Fp) are coupled and stored as an individual in the Controller Pool (CP). A new individual is generated by selecting (copying) two CGs from the CP based on the roulette wheel selection, and then uniform crossover and mutation are operated. The new CG is then coupled with an Environment Genome (EG) on which genetic operators have also been performed similarly and put into the genome queue. Pre-evaluation of a CG is done using its coupled EG before the robotic evaluation in the real world. In case the pre-evaluation value is less than a threshold described below, it skips robotic evaluation and returns to the CP coupled with the pre-evaluation value. Otherwise, robotic evaluation is conducted and returns to the CP coupled with the fitness value. This study does not consider transmission of individuals among robots, although migrations of CGs among robots are allowed in the framework.

EG, which is a set of information specifying the environment in pre-evaluation, is also coupled with its fitness value and stored into the Environment Pool (EP). The fitness of an EP is the agreement rate between the fitness value by robotic behavior and the estimated value by pre-evaluation. The design of the evolutionary system for pre-evaluation depends on the requirements of users. If knowledge about the robot and the task are not built-in at all, then the evolutionary system corresponds to genetic programming that constructs a function of a EG specifying the fitness value of it. On the other hand, if most possible information is built-in, then it corresponds to a kind of parameter tuning. There are important trade-offs here, which will be investigated thoroughly in later sections.



(a) The proposed architecture.

(1) Selection of a pair of CG and EG based on Fr, Fp, Fe
(2) Uniform crossover and mutation on the pair
(3) Pre-evaluation of the CG using EG (Fp is obtained)
(4) If Fp is less than T then go to (8)
(5) Robotic evaluation of CG (Fr is obtained)
(6) CG accompanied with Fr returns to CP
(7) Computation of Fe from Fp and Fr
(8) EG accompanied with Fe returns to EP
(9) Go to (1)

(b) The loop of the architecture.

CG: Controller Genome
EG: Environment Genome
CP: Controller Pool
EP: Environment Pool
Fr: Fitness of the CG based on robotic evaluation
Fp: Fitness of the CG based on pre-evaluation
Fe: The fitness of EP (Agreement rate between Fr and Fp)
T: Threshold value for bypassing the robotic evaluation

Figure 1: A framework for embodied evolution with pre-evaluation.

3 Implementation

We compared the original EE method with the one with pre-evaluation. We adopted two methods for pre-evaluation: A pre-defined function with parameters is optimized (PE1) and a neural network is optimized (PE2) both by evolution. The latter has a larger search space of pre-evaluation. We used a humanoid robot with two four-degree-of-freedom arms and two six-degree-of-freedom legs (HRP-2m Choromet), and biped walking was evolved. Walking distance was measured by using built-in power sensors and used as a fitness value.

Fig. 2 shows walking pattern for the biped robot. N was the walk count constituting one trial (walking pattern). The trajectory of legs was generated by a cosine function. $a(0 \le a \le 0.05)[m]$ represents the length of the movement of the center of gravity by inclining the legs to the left or right, $c(0 \le c \le 0.05)[m]$ represents the length of the movement of the center of gravity by moving the legs to the front, and $d(0 \le d \le 0.02)[m]$ represents the height to which legs are raised. Each CG is composed of these three parameters.

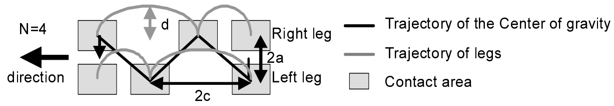


Figure 2: The following sequence constitutes a trial ($N = 4$): 1) Both legs are parallel with half-sitting (state 0), 2) Move center of gravity to the left by inclining the legs, 3) Move right leg to the front (action 1), 4) Move center of gravity to the right-front by inclining the legs (action 2), 5) Move left leg to the front, 6) Move center of gravity to the left-front by inclining the legs, 7) (action 1), 8) (action 2), 9) Move left leg to the front , 10) (state 0).

F_r was calculated as follows: $F_r = c\sum_{i=1}^{N} Z_i$, where Z_i was the value output from the built-in power sensors.

When adopting PE1, the pre-evaluation value was calculated as follows:

$$F_p = crP_a^{a_1}P_c^{c_1}P_d^{d_1} \quad (1)$$
$$P_a = -a'(a-a_2)^2+1 \quad (2)$$
$$a' = \begin{cases} 1/a_2^2 & (a_2 \ge a_{max}/2) \\ 1/(a_{max}-a_2)^2 & (a_2 < a_{max}/2) \end{cases} \quad (3)$$
$$P_c = -c'(c-c_2)^2+1 \quad (4)$$
$$c' = \begin{cases} 1/c_2^2 & (c_2 \ge c_{max}/2) \\ 1/(c_{max}-c_2)^2 & (c_2 < c_{max}/2) \end{cases} \quad (5)$$
$$P_d = -d'(d-d_2)^2+1 \quad (6)$$
$$d' = \begin{cases} 1/d_2^2 & (d_2 \ge d_{max}/2) \\ 1/(d_{max}-d_2)^2 & (d_2 < d_{max}/2) \end{cases} \quad (7)$$

in which a_1, a_2, c_1, c_2, d_1, d_2, and r were the genes in EG with initial values of 3, 0.03, 3, 0.05, 3, 0.015, and 18000, respectively, and the upper limits of a, c and d are represented as a_{max}, c_{max} and d_{max}, respectively. This function represents a fitness landscape in which when each value moves from the optimal value, the function value decreases rapidly (which corresponds to the increase in the number of falling down). The mutation rate was set to 0.3.

When adopting PE2, the neural network for identifying the fitness function that evaluates each EG was defined as follows:

$$y_j = sig(\sum_{i=1}^{3} x_i w_{i+3(j-1)} - \theta_j) \quad (8)$$

$$z = sig(\sum_{i=1}^{4} y_i w_{i+12} - \theta_5) \quad (9)$$

$$sig(x) = \frac{1}{1+exp(-x)} \quad (10)$$

$\theta_1 \sim \theta_5$ were the thresholds of the neurons. The input of the neural network was $x_1 = a/a_{max}$, $x_2 = c/c_{max}$ and $x_3 = d/d_{max}$, respectively and the output

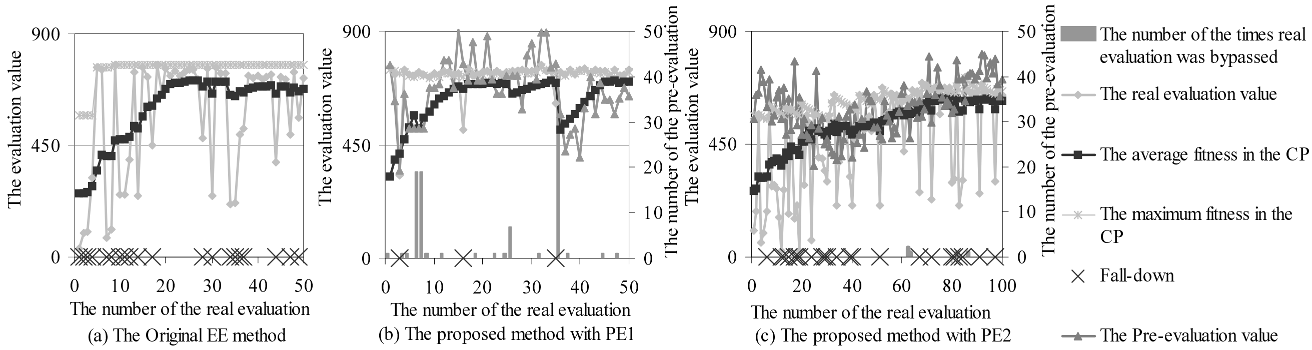


Figure 3: The index values of CG.

of it, pre-evaluation value, was $F_p = z \times 10^3$. EG was composed of 21 weights of the neural network ($w_1 \sim w_{16}$). The initial values were set at random in the range of -1.0 to 1.0. This mutation rate was set to $0.7 - 0.4 \cdot \frac{\sum_{i=1}^{P} F_{ei}}{P}$ which changed according to the average value of the fitness of EG. P was the size of CP and EP.

The fitness of each EG was defined as the agreement rate between the real evaluation value and the pre-evaluation value: $F_e = \begin{cases} \frac{F_p}{F_r} (F_r \geq F_p) \\ \frac{F_r}{F_p} (F_r < F_p) \end{cases}$.

T was the threshold calculated as the average CG value times the average EG value: $T = \frac{(\sum_{i=1}^{P} F_{ri})(\sum_{i=1}^{P} F_{ei})}{P2}$. In case the pre-evaluation value was below the threshold, CG skipped the evaluation based on robot behavior and returned to CP. Some kind of deterioration was introduced in such a way that the fitness of individuals in both CP and EP decreases by $w = 1\%$ every when CG or EG returns to their pool.

4 Evaluation

We evaluated the proposed architecture applied to a biped robot, in which one of the two methods was adopted for pre-evaluation as described in the previous section: function optimization (PE1) or neural network optimization (PE2) both by coevolution. We compared them with the original EE method. In addition, we evaluated the simulate-and-transfer method for comparison. The size of CP and EP was 10. The number of initial individuals was 10 in each of CP and EP. The mutation rate was set to 0.3, and N was set to 4. $\theta_1 \sim \theta_5$ were set to 0.

Fig. 3 shows the transition of fitness in CG, real evaluation values, pre-evaluation values and the times of pre-evaluations in the case of EE (a), PE1 (b), or PE2 (c). Also, x-marks in the graph indicate the real evaluations in which the robot fell down to the ground, and in case of PE1 and PE2 each bar graph indicates the number of the times the real evaluation was bypassed according to the low pre-evaluation values.

It is shown that the average fitness in PE1 increased more rapidly in the early evaluations than the one in the EE, while the one in the PE2 increased less rapidly than the one in the EE. We observe the same tendency with the transitions of the real evaluation value, although they fluctuated significantly. The good performance of PE1 was attained by bypassing the real evaluation of the individuals whose pre-evaluation was low. However, in case of PE2, the evolution of EG is harder than the one in case of PE1, and therefore the speed of the evolution of CG was reduced.

We see that the number of falls was significantly decreased by introducing pre-evaluation in case of PE1, especially in the early evaluations (approximately from 10 to 2 in the first 20 real evaluations). This was attained also by pre-evaluation without real evaluation, which happened 25 times during the first 20 real evaluations. In case of PE2, introduction of pre-evaluation reduced the total number of falls, but the effect was far less than in case of PE1.

The pre-evaluation values moved roughly in accordance with the real evaluation values. This means the coevolution had been successfully applied in the architecture. However, they tended to be a bit larger than the real evaluation values. This is due to the unique property of the evolution of EG. There are two kinds of selection pressure. The explicit one is to increase the fitness, in other words, by definition, to decrease the difference between the real evaluation value and the pre-evaluation value. The implicit pressure is to increase the pre-evaluation value because the EG does not return to the EP when the pre-evaluation value is smaller than the threshold and thus the EG bypasses the real evaluation (Fig. 1).

Fig. 4 shows the transitions of the average and maximum fitness in EP and the fitness of EG in case of PE1 (a) and PE2 (b). Their values were relatively large even in the early stages and fluctuated significantly even in later stages. This is because the fitness of EG is not defined absolutely but relatively as the difference between the real and the pre-evaluation values.

We further evaluated a kind of the simulate-and-transfer method, in which the same simulation frame-

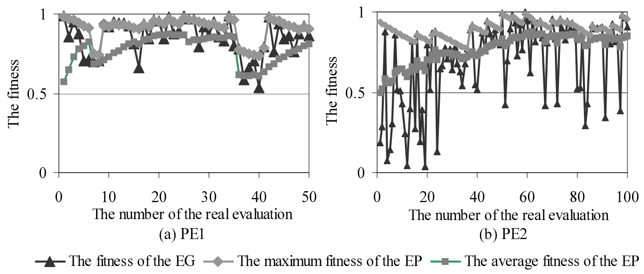


Figure 4: The fitness of EG.

work as the EE method was used except that the robotic evaluation was replaced by simulation carried out using the robotic platform OpenHRP. Table 1 shows the all of the evolved individuals (CGs) in CP after 50 evaluations. The first column shows the fitness in simulation environment. Each CG was then transferred into the robot and evaluated. The second column shows the fitness based on robotic behavior, and the third column shows whether the robot fell down to the ground or not in robotic evaluation. It is shown that nine out of ten individuals suffered from the decreased fitness caused by falling in real world. This result shows a typical gap between simulation and real world ("reality gap").

Table 1: Reality gap between simulation and real world.

Fitness (simulation)	830	814	806	797	781	769	766	751	743	735
Fitness (robotic behavior)	167	141	149	149	135	149	752	182	166	166
Falling down (With: ×, without: ○)	×	×	×	×	×	×	○	×	×	×

5 Conclusions

We introduced pre-evaluation into the embodied evolution (EE) framework for a biped robot in order to restrain robot behavior of which fitness contribution is estimated to be low, specifically falling down to the ground. We believe that the EE framework with pre-evaluation is applicable to a wide variety of optimization tasks by evolution in which the cost or risk of fitness evaluation is not negligible. We adopted two methods for pre-evaluator construction: PE1 and PE2. We comparatively evaluated the proposed architecture based on one of these two methods, the original EE, and a conventional simulate-and-transfer method.

Table 2 summarizes the characteristics of these methods based on the results of the evaluation experiments in terms of reality gap, calculation time, the cost for evaluation in real world (e.g. the risk of robot falling), and the cost of modeling or simulation. Each evaluation result is represented by the size of a circle or the thickness of a line in the table.

Table 2: The comparison between proposed method and other two methods.

	Original EE	EE with Pre-evaluation prior implementation of pre-evaluator (Min → Max)	Sim. & Trans.
Reality Gap	▪	▪	○
Calculation Time	○	○ → ∘	∘
Real Evaluation Cost	○	○ → ∘	∘
Simlulation/modeling Cost	∘	∘ → ○	○

The design of the evolutionary system for pre-evaluation depends on the requirements of users as described in Section 2. If we implement the evolutionary system for pre-evaluation to the utmost extent before evolution, the framework will become equal to the system with simulate-and-transfer except for the robotic evaluation of fitness. Generation of the pre-evaluator by PE1 corresponds to function optimization by evolution, and the architecture is close to the one in case of simulate-and-transfer. On the other hand, if we let the evolution generate the system for pre-evaluation to the utmost extent, each evaluation result will get closer to the case with EE. However, in this case the search space via evolution will get larger and thus the good solutions might be difficult to be obtained, as we observed the decrease in performance in case of PE2. The most important thing in implementing the proposed architecture is to consider the tradeoffs shown in this table.

Humans evaluate the behavior in advance using their internal models of the world before they actually do it. At the same time, receiving the feedback from the real world, they build and refine their internal models. It might be interesting to investigate the parallelism between the evolution of human intelligence and the evolution in the proposed architecture.

References

[1] Nolfi S, Floreano D (2000), *Evolutionary Robotics*, MIT Press.

[2] Watson R A, Ficici S G, Pollack J B (2002), Embodied Evolution: Distributing an Evolutionary Algorithm in a Population of Robots, *Robotics and Autonomous Systems*, 39(1): 1-18.

[3] Usui Y, Arita T (2003), Situated and Embodied Evolution in Collective Evolutionary Robotics, *Proc. of the Eighth International Symposium on Artificial Life and Robotics*, 212-215.

[4] Watson R A, Ficici S G, Pollack J B (1999), Embodied Evolution: Embodying an Evolutionary Algorithm in a Population of Robots, *Proc. of the 1999 Congress on Evolutionary Computation*, 335-342.