Autonomous Learning of Ball Passing by Four-legged Robots and Trial Reduction by Thinning-out and Surrogate Functions

Hayato KOBAYASHI [a,1], Kohei HATANO [b], Akira ISHINO [a], and Ayumi SHINOHARA [a]

[a] *Graduate School of Information Science, Tohoku University, Japan*
[b] *Department of Informatics, Kyushu University, Japan*

Abstract. This paper describes an autonomous learning method used with real robots in order to acquire ball passing skills in the RoboCup standard platform league. These skills involve precisely moving and stopping a ball to a certain objective area and are essential to realizing sophisticated cooperative strategy. Moreover, we propose a hybrid method using "thinning-out" and "surrogate functions" in order to reduce actual trials regarded as unnecessary or unpromising. We verify the performance of our method using the minimization problems of several test functions, and then we address the learning problem of ball passing skills on real robots, which is also the first application of thinning-out on real environments.

Keywords. Autonomous Learning, Four-legged Robot, RoboCup,

1. Introduction

For robots to function in the real world, they need the abilities to adapt to unknown environments, that is *learning* abilities. In particular, legged robots must acquire or learn many basic skills such as walking, running, pushing, pulling, kicking, and so on, in order to accomplish sophisticated behaviors such as playing soccer. In this paper, we address the learning of ball passing skills by four-legged robots as an instance of basic skills. Ball passing skills are used in RoboCup[2] and are essential to realizing sophisticated cooperative strategy.

We regard learning as optimization of an unknown score function $f : X \to \boldsymbol{R}$ on an n-dimensional search space $X \subseteq \boldsymbol{R}^n$, i.e. $\min_{x \in X} f(x)$ or $\max_{x \in X} f(x)$. In the case of robot learning, we must usually treat a noisy, high-dimensional, nonlinear function. Although such functions are surely difficult to optimize, it is more serious from a practical standpoint that each *trial*, which means evaluation of the function for a sampled point $x \in X$, brings the following various costs.

- Servo motors are damaged, and so robots themselves can be broken easily.

[1] Corresponding Author: Hayato Kobayashi; E-mail: kobayashi@shino.ecei.tohoku.ac.jp.

[2] RoboCup [1] is a competition for autonomous robots which play soccer and also an interesting and challenging research domain, because it has a noisy, incomplete, real-time, multi-agent environment.

- Human resources are often needed, e.g., we must restore a kicked ball for the next trial.
- It obviously takes more time compared to computations being done only on PCs.

There are two approaches in order to reduce those costs. One is an approach from experimental methodology, and the other is an approach from machine learning.

Approach from experimental methodology

Experiments in simulation environments are useful in the sense that each trial does not damage robots nor need human resources. Zagal and Solar [2] studied the learning of ball shooting skills by four-legged robots in a simulation environment. Kobayashi et al. [3] also studied the similar learning problem in another simulation environment and discovered sophisticated shooting motions. However, since simulation environments can not produce complete, real environments, we need to train robots in the real environment where basic skills heavily depend on complex physical interactions.

In experiments with real robots, *autonomous learning*, by which robots acquire some skills on their own without human intervention, is practically efficient, especially in environments that change frequently, such as RoboCup competitions. There have been many studies conducted on the autonomous learning of quadrupedal locomotion [4,5,6,7], which is the most basic skill for every movement. However, the skills used to control the ball are often coded by hand and have not been studied as much as quadrupedal locomotion. Fidelman and Stone [8] presented an autonomous learning method of ball acquisition skills that involves controlling a stopped ball by utilizing chest PSD sensors. Kobayashi et al. [9] studied the reinforcement learning to trap a moving ball autonomously. The goal of the learning was to acquire a good timing to initiate its trapping motion, depending on the distance and the speed of the ball, whose movement was restricted to one dimension.

Approach from machine learning

Memory-based learning is often utilized in robot learning, since the number of trials are highly restricted, which are up to and at most several hundred trials in many cases. Memory-based learning stores past evaluations in history and samples a new promising point based on the history. In memory-based learning, we can efficiently reduce the number of trials, even if computational complexity of the algorithm is too large.

Sano et al. [10] proposed Memory-based Fitness Evaluation GA (MFEGA) for noisy environments. They estimated more proper scores (or fitness values) by the weighted average of neighboring scores, so as to reduce the number of trials compared to multiple sampling methods, which evaluates a score function several times in each trial. Ratle [11] proposed an acceleration method of GA by utilizing *surrogate functions*, which are approximation models of original score functions. He chose kriging interpolation as surrogate functions and used it for evaluating some of the next generations. In our previous work [3], we took another approach for reducing the number of trials, based on the idea that it is able to theoretically determine whether or not selected points are worth evaluating if score functions are g-Lipschitz continuous described in Section 3.1. If a sampled point is unlikely to improve the results obtained so far, the trial for the point is not performed and just skipped over. We call this method *thinning-out* in search seedlings (or candidates), which contrasts to *pruning* in a search tree. One advantage of our method is that it is guaranteed to thin-out unnecessary candidates without any errors, if we know a proper Lipschitz function g. We can arbitrarily prepare the function g depending on the problems to be solved and also infer the function g for complex problems that are difficult for us to come up with it.

In this paper we propose an autonomous learning method of ball passing skills as an approach from experimental methodology as well as a hybrid method using thinning-out and surrogate functions as an approach from machine learning. This work is also important as the first application of thinning-out in the real world.

The remainder of this paper is organized as follows. In Section 2, we begin by illustrating the execution mechanism of passing motions and our training equipment for autonomous learning, and then formalizing our learning problem. In Section 3, we illustrate our proposed method and describe existing thinning-out and kriging interpolation utilized as a surrogate function. In Section 4, we apply the proposed method for the minimization of several mathematical test functions and learning of ball passing skills by four-legged robots. Finally, Section 5 presents our conclusions.

2. Preliminaries

2.1. Passing Motion

Ball passing is realized by accurate shooting motions that precisely move and stop a ball to an objective area, such as where a teammate is waiting. In this paper, we use AIBO as a legged robot, which is one of the robots permitted for use in the standard platform league in RoboCup. Shooting motions of AIBO are achieved by sending *frames*, consisting of the 15 joint angles for its head and legs, to OVirtualRobot every 8 ms. OVirtualRobot is a kind of proxy object that is defined in the software development kit OPEN-R for AIBO. In our framework, these frames are generated from *key-frames*, which are characteristic frames shaping the skeleton of each motion. For example, a kick motion needs at least two key-frames, since robots must pull and push its leg when executing it. We indicate the number of interpolations for each key-frame, so that whole frames can be generated by using a linear interpolation method. Thus, our motion takes $8n_i$ ms, where n_i is the total number of interpolations.

2.2. Training Equipment

Acquiring passing skills autonomously is usually difficult, because robots must be able to search for a kicked ball and then move the ball to an initial position. This requires sophisticated, low-level programs, such as an accurate, self-localization system, a ball acquiring skill, and a movement behavior with holding the ball. In order to avoid additional complications, we simplify the learning process a bit more. We can restrict ball passing to an anterior direction since our robot has rotational locomotion with holding the ball and can pass omni-directionally using only a forward passing motion. In other words, we can ideally treat our problem, which is to learn passing skills, one-dimensionally. In actuality though, the problem cannot be fully viewed in one-dimension because the ball might curve a little due to the grain of the grass.

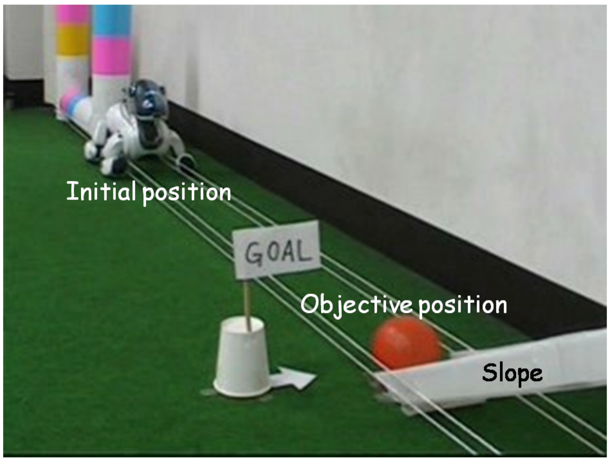

Figure 1. Training equipment for learning passing skills.

For autonomous learning of passing skills in one-dimension, we prepared almost the same training equipment as our previous work [9] as shown in Fig. 1. The equipment has rails of width nearly equal to an AIBO's shoulder-width. These rails are made of thin rope or string, and their purpose is to restrict the movement of the ball as well as the quadrupedal locomotion of the robot, to one-dimension. At the edge of these rails, there is a goal flag that shows the objective position of a passed ball. A slope just behind the flag can return the ball when passing is too strong, in the same way as an "automatic golf putting machine". Without the slope, the kicked ball would go beyond the objective position when the robot kicked it too strong, and it would take too much time for the robot to return the ball to the initial position where the robot first kicked it. Using the equipment, our robots can simply learn passing skills autonomously by kicking the ball straightforward and measuring the distance of the kicked ball on their own.

2.3. Problem Formulation

We directly utilize key-frames for learning of passing skills. It is possible to realize flexible search in the neighborhood of the skeleton without modeling the movement and setting extra-parametrization. All we do is create a sketchy motion as an initial motion, i.e., to indicate the key-frames for the motion. By adjusting the values of the key-frames of the initial motion, our robots learn an appropriate shooting motion, which is neither too strong nor too weak, so that the kicked ball will stop at an objective position.

We can apply the mirroring technique in the same way as Latzke et al. [12], since shooting motions are restricted to an anterior direction. In the mirrored situation, we can identify the angles of its right legs with those of its left legs and regard the horizontal "pan" angle of its neck as zero, although AIBO has 15 joint angles for its head and legs. As a consequence, the search space of the learning has $8n_k$-dimensions, where n_k means the number of key-frames.

Our learning is formalized as a maximization of a score function $f : X \rightarrow \boldsymbol{R}$ on an $8n_k$-dimensional search space $X \subseteq \boldsymbol{R}^{8n_k}$. The value $f(x)$ expresses the goodness of the shooting motion specified by motion parameters $x \in X$, that is, how closely the kicked ball stops at the objective position. The score is experimentally evaluated as follows:

First make our robot kick the ball by performing the motion, and then make the robot measure the distance d_{ball} from the robot to the point where the ball actually stopped. Since it obviously contains some noise in real environments, we treat the median $\hat{d}_{ball}$ of 5 executions of the trials for each x as the score $f(x) = \hat{d}_{ball}$. The score has the maximum value D_{goal}, which is the distance from the robot to the objective position shown by the goal flag in Fig. 1, because the ball will be returned back to the robot by the slope just behind the goal flag if it is kicked too strong. Moreover, for a constant D_{bounce}, we regard the score as zero if the returned ball is stopped within the distance D_{bounce}, because the ball might bounce off of its chest, and the estimated distance might be erroneous. In this paper, we set $D_{goal} = 800$ mm and $D_{bounce} = 300$ mm.

The distance 800 mm is used as the nearest distance between robots in the passing challenge, which was one of the technical challenges in the RoboCup four-legged league in 2007.

3. Learning Method

The score function in the previous section is obviously unknown, and so we utilize meta-heuristics for addressing maximization of the function. Meta-heuristics means heuristic algorithms that are independent of problems, such as Genetic Algorithm (GA), Simulated Annealing (SA), and Hill Climbing (HC). In this paper, we choose GA that was successfully utilized for the learning of shot motions by virtual four-legged robots in Kobayashi et al. [3].

For reducing the number of trials of robot learning, we utilize *thinning-out*, which will be described in Section 3.1. Thinning-out can judge whether or not a point sampled by meta-heuristics is promising according to the history of past trials. For the sake of discussion for now, we describe a point sampled by GA as a *candidate*. If a sampled candidate is regarded as unpromising, we can thin-out (or skip over) the candidate without evaluating it. In meta-heuristics with thinning-out only, when a sampled candidate is thinned-out a new candidate is simply resampled with random perturbation, i.e. mutation in GA, as shown by the left of Fig. 2. Therefore, samples with lower scores tend to be evaluated even in a later phase of learning, since samples in an unknown area where candidates have not been sampled yet can be regarded as promising. In order to make the resampling process more efficient, we propose a new method combined with a surrogate function $\hat{f}(x)$, which is an approximation model of $f(x)$. In this method, when a sampled candidate is thinned-out, the candidate is evaluated by the surrogate function instead of the original score function, and a new candidate is resampled by meta-heuristics, as shown by the right of Fig. 2. In this paper, we utilize kriging interpolation, which will be described in Section 3.2, as a surrogate function. We anticipate that the method has the advantage of accelerating the learning process, since meta-heuristics may be more efficient than random perturbation, and the disadvantage of increasing the possibility of convergence to a local optima, since the surrogate function might be wrongly approximated.

3.1. Thinning-out

In this section, we formalize our thinning-out technique for reducing unnecessary or unpromising trials. First, we assume that our score function is continuous and, to some

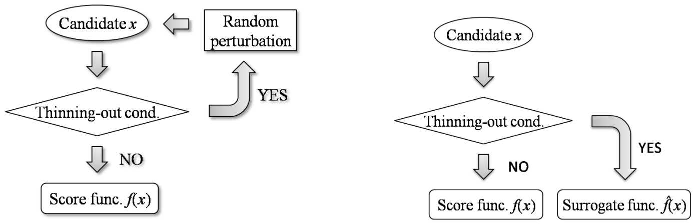

Figure 2. The left figure shows the concept of thinning-out only, and the right figure shows the concept of thinning-out with surrogate functions. When a sampled candidate is thinned-out, the former resamples a new candidate, while the latter evaluates a surrogate function instead of a score function.

extent, smooth over the search space. Our assumption seems to be reasonable, because each robot's movement is continuous, and thus small changes of parameters will not affect the score significantly. Based on this assumption, we can find out unpromising candidates theoretically by utilizing the degree of smoothness of the score function.

Now we define the local smoothness of the score function in terms of Lipschitz condition, which is found in standard textbooks on calculus. We use g-Lipschitz continuous for some function g, as a natural extension of c-Lipschitz continuous for some constant c in textbooks.

Definition 3.1 (Lipschitz condition) *Let $\boldsymbol{R}$ be the set of real numbers, X be a metric space with a metric d, and $f : X \to \boldsymbol{R}$ be a function on it. Given a function $g : \boldsymbol{R} \to \boldsymbol{R}$, f is said to be g-*Lipschitz continuous*, if it holds for any $x_1, x_2 \in X$ that*

$$|f(x_1) - f(x_2)| \leq g(d(x_1, x_2)).$$

The function g is called Lipschitz function*.*

From now on, we use d as the Euclidean metric. Supposing that a score function f is g-Lipschitz continuous, if a candidate satisfies the following thinning-out condition, it is guaranteed to safely thin-out the candidate, which will never become better than the current best score.

Definition 3.2 (Thinning-out condition) *Let x_b be the point with the current best score $f(x_b)$, x_c be a candidate point to try, and x_n be the nearest neighbor of the candidate. Given a Lipschitz function $g : \boldsymbol{R} \to \boldsymbol{R}$, x_c is said to* satisfy the thinning-out condition with respect to g*, if it holds that*

$$f(x_n) + g(d(x_c, x_n)) \leq f(x_b).$$

Since a Lipschitz function g is often not given in practical problems, we need infer the function g from the history of past trials. The naïve inferring method is Max Gradient (MG) method, which is defined by $g(x) = cx$ utilizing the maximum value c in the gradients between any two points in the history. We also proposed Gradient Differences (GD) inferring method. GD is intuitively a weighted average method of gradients of the score function using weights based on the inverse of the distance between any two points in the history of past evaluations. It will become a good approximation after enough evaluations, since a line connecting two close points can approximate the gradient of the function nearby the points.

3.2. Kriging Interpolation

Kriging [13] is one of the function interpolation or approximation methods of an unknown real function f, which is initially developed in geostatistics, and recently, many researchers successfully utilized kriging as a model of surrogate functions. Although there are several types of kriging, we choose *ordinary kriging* that is the most commonly used type of kriging. In ordinary kriging, its interpolator $\hat{f}$ at a point x^* is represented by a weighted linear combination as follows,

$$\hat{f}(x^*) = \sum_{i=1}^{n} w_i f(x_i),$$

where $f(x_1), ..., f(x_n)$ are the observed values of the function at some other points $x_1, ..., x_n$. In order to estimate the weights $w_1, ..., w_n$, we assumes that the observed values are the realization of a stochastic process with *second-order stationarity*, which means that the expected values are constants, i.e., $E[f(x_i) - f(x_j)] = 0$, and the covariances are dependent only on the distances, i.e., $\mathit{Cov}[f(x_i), f(x_j)] = C(d(x_i, x_j))$. The function C is called a *covariance function* and describes a correlation between any two points in terms of the distance. In this paper, we define $C(x) = \sigma^2 \exp(-\theta x^2)$ utilizing an isotropic Gaussian function, where $\sigma^2 (= C(0))$ is the variance $\mathit{Var}[f(x_i)]$, based on the heuristics that the closer points correlate outputs more positively.

The weights for x^* is chosen such that the error variance $V_e = \mathit{Var}(\hat{f}(x^*) - f(x^*))$ is minimized subject to $\sum_{i=1}^{n} w_i = 1$, which is given by the unbiased condition of the interpolator, i.e., $E[\hat{f}(x^*) - f(x)] = 0$, and the property of second-order stationarity. Utilizing a Lagrange multiplier λ, we can solve it by setting $V_e' = V_e + 2\lambda(\sum_{i=1}^{n} w_i - 1)$ and calculating $\frac{\partial}{\partial w_i} V_e' = 0$. Thus, the weights are given by the following $n+1$ equations.

$$\begin{cases} \sum_{i=1}^{n} w_i C(d(x_i, x_j)) + \lambda = C(d(x_i, x^*)) & \text{for } j = 1, ..., n, \\ \sum_{i=1}^{n} w_i = 1. \end{cases}$$

4. Experiments and Results

4.1. Minimization of Test Functions

For verifying the performance of our proposed method, we first address the minimization of the 6 mathematical test functions, Rastrigin, Schwefel, Griewank, Rosenbrock, Ridge, and Ackley. They have often been used for the performance evaluations of metaheuristics and are also utilized in Kobayashi et al. [3]. Rastrigin, Schwefel, Griewank, and Ackley have multiple peaks, although Griewank and Ackley have a single peak with a global view. Griewank, Rosenbrock and Ridge have the design variables' dependency.

In this section, SGA means a simple, real-coded GA with uniform crossover, whose rate is 0.3, Gaussian mutation with mean of 0 and variance of 10% of the domain size of each dimension, whose rate is 0.2, and roulette selection with elite strategy, and its

Table 1. This table shows the minimization results of 10-dimensional 6 test functions by SGA, GAT and GATS. Where, "min", "trial", and "error" represent the minimum scores in 100 trials, the trial rates in 100 candidates, and the error rates in 100 candidates, respectively. The trial rate means $100\times$#(tried candidates) / #(all candidates), and error rate means $100\times$#(wrongly thinned-out candidates) / #(all thinned-out candidates). Each value is the average over 100 experiments. As for GA, the trial rates and error rates are 100% and 0% (or undefined), respectively.

function	SGA	GAT			proposed GATS		
	min	min	trial (%)	error (%)	min	trial (%)	error (%)
Rastrigin	260	165	54.20	0.80	152	38.67	0.40
Schwefel	3583	1817	62.84	0.87	1305	42.63	0.17
Griewank	621	211	48.24	0.09	112	35.81	0.00
Rosenbrock	17472	3326	54.75	0.06	2265	39.34	0.00
Ridge	5.7e9	6.4e8	55.42	0.04	2.3e8	38.58	0.00
Ackley	21	21	60.37	0.92	21	43.26	0.05

population size is 20. GAT means SGA with ϵ-*thinning-out*, where $\epsilon = 0.01$, with MG inferring method. ϵ-thinning-out forcibly evaluates a candidate with probability ϵ, and otherwise, it skips over the candidate, so that it can hold out the possibility for evaluating candidates that are wrongly thinned-out once and avoid never halting by thinning-out all candidates. GATS means our proposed method, that is GAT with a surrogate function of ordinary kriging interpolation described in Section 3.2. The parameter θ of the covariance function is estimated by maximum likelihood estimation.

Table 1 shows the minimum scores, trial rates, and error rates of SGA, GAT, and GATS. The trial rate means the rate of tried candidates in all candidates, and the error rate means the rate of wrongly thinned-out candidates, whose scores (calculated by only for the error rate) were actually better than the current best score, in thinned-out candidates. Wrongly thinned-out candidates are examined by calculating thinned-out scores only for error rate. Note that even if candidates are thinned-out at random, the error rate becomes very small. All values of the minimum scores, trial rates, and error rates become better, as they become smaller. This section focuses on the comparison between GAT and GATS, since the comparison between SGA and GAT was already discussed in Kobayashi et al. [3]. As for GATS, we anticipated that the trial rates become higher, and the error rate become lower, because the efficient resampling method should choose promising candidates that can not be thinned-out. However, against our anticipation, the trial rate of GATS is lower than those of GAT, while, as expected, the error rates of GATS are lower than those of GAT. This was because evaluated points with good scores should make easy situations to thin-out new sampled candidates, although GATS surely tends to evaluate resampled candidates. Consequently, the fact that both trial rates and error rates are decreased leads to the good result that the minimum scores of GATS are better than those of GAT over all test functions.

4.2. Learning of Passing Skills

In this section, we address the learning of ball passing by four-legged robots, i.e., maximization of the score function formalized by Section 2.3. SGA, GAT, and GATS are almost the same in the previous section. The only following parameters are different. SGA utilizes discrete mutation, which randomly adds one from $\{-r, 0, +r\}$ in each dimension, where $r = \pi/36$ radians, and the population size is 10. GAT utilizes GD inferring method, which was effective for score functions over high dimensional spaces in our previous experiments [3].

The initial motion for learning is a motion of pushing a ball with its chest, which is actually used as a shooting motion in games in RoboCup. The motion is performed not only by moving the chest ahead, but also by moving the whole body so as to enhance the power of the shot. As the shooting motion involves the movement of almost all joint parts of the robot, it is not easy to even adjust the motion. Therefore, it is quite difficult for humans to design the shooting motion so that the ball goes to the objective position. The motion takes 482 ms because it has 54 total interpolations ($n_i = 54$), and can move the ball to approximately 1500 mm. Since the motion consists of 6 key-frames ($n_k = 6$), the search space of this learning problem is 48-dimensions.

Fig. 3 shows a comparison of the learning processes obtained by running SGA, GAT, and GATS for 50 trials. Unlike the previous experiment, the scores on these processes become better, as they become higher. Since the score of each trial is calculated by the median of 5 motion executions, our robot, in fact, needs 250 motion executions in total for 50 trials. Each experiment took more than 3 hours, and we must have exchanged more than 6 batteries since our robot needs one battery every 30 minutes. The figure indicates that GATS and GAT are obviously more efficient than SGA, and GATS seems to be slightly better than GAT. Unfortunately, we could not conduct any more experiments, since we realized that the learning of shooting motions damages or breaks servo motors of robots more frequently than expected. At least in this section, however, we could verify that our thinning-out method was successfully applied to the learning problem in the real world for the first time. Our robot finally acquired the best score 777 by utilizing GATS, which means an accuracy of 23 mm, since the target distance is 800 mm in this experiment. The acquired motion was not much different from the initial motion. The fact implies that the adjustment of passing motions is difficult for humans. All movies of the earlier and later phases of our experiments are available on-line [3].

5. Conclusions and Future Work

In this paper, we proposed an autonomous learning method of ball passing skills and a hybrid method using thinning-out and surrogate functions. The former realizes that robots learn ball passing skills without human intervention, except for replacing discharged batteries. The latter reduces the number of trials efficiently with few errors and improves the result of learning. By using the proposed two methods, our robot successfully acquired an appropriate passing motion with an accuracy of 23 mm on their own. Our experiments were also the first application of thinning-out in the real world.

Future work includes extending passing skills into two-dimensions and toward arbitrary objective distances. We can acquire more practical passing skills by replacing the score function with a new score function that refers the average (or median) and variance of the position of each kicked ball in two-dimensions. We can also acquire more flexible passing skills by utilizing the interpolation of some passing motions.

[3] `http://www.shino.ecei.tohoku.ac.jp/~kobayashi/movies.html#passing`

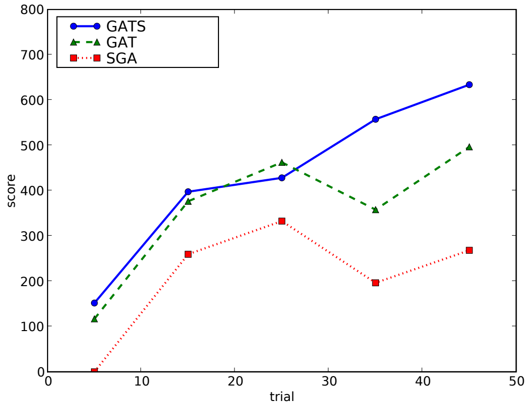

Figure 3. Learning processes of SGA, GAT, and GATS in terms of trials. The solid line, dashed line, and dotted line show the learning processes of GATS, GAT, and SGA, respectively. Each point means the average of each population, e.g. 10 trials.

References

[1] RoboCup Official Site. `http://www.robocup.org/`.

[2] Juan Cristóbal Zagal and Javier Ruiz del Solar. Learning to Kick the Ball Using Back to Reality. In *RoboCup 2004: Robot Soccer World Cup VIII*, volume 3276 of *LNAI*, pages 335–347. Springer-Verlag, 2005.

[3] Hayato Kobayashi, Kohei Hatano, Akira Ishino, and Ayumi Shinohara. Reducing Trials Using Thinning-out in Skill Discovery. In *Proceedings of the 10th International Conference on Discovery Science 2007 (DS2007)*, volume 4755 of *LNAI*, pages 127–138. Springer-Verlag, 2007.

[4] Gregory S. Hornby, Seichi Takamura, Takashi Yamamoto, and Masahiro Fujita. Autonomous Evolution of Dynamic Gaits With Two Quadruped Robots. *IEEE Transactions on Robotics*, 21(3):402–410, 2005.

[5] Min Sub Kim and William Uther. Automatic Gait Optimisation for Quadruped Robots. In *Proceedings of 2003 Australasian Conference on Robotics and Automation*, pages 1–9, 2003.

[6] Nate Kohl and Peter Stone. Machine Learning for Fast Quadrupedal Locomotion. In *The Nineteenth National Conference on Artificial Intelligence (AAAI2004)*, pages 611–616, 2004.

[7] Joel D. Weingarten, Gabriel A. D. Lopes, Martin Buehler, Richard E. Groff, and Daniel E. Koditschek. Automated Gait Adaptation for Legged Robots. In *IEEE International Conference on Robotics and Automation*, 2004.

[8] Peggy Fidelman and Peter Stone. Learning Ball Acquisition on a Physical Robot. In *2004 International Symposium on Robotics and Automation (ISRA)*, 2004.

[9] Hayato Kobayashi, Tsugutoyo Osaki, Eric Williams, Akira Ishino, and Ayumi Shinohara. Autonomous Learning of Ball Trapping in the Four-legged Robot League. In *RoboCup 2006: Robot Soccer World Cup X*, volume 4434 of *LNAI*, pages 86–97. Springer-Verlag, 2007.

[10] Yasuhito Sano, Hajime Kita, Ichikai Kamihira, and Masashi Yamaguchi. Online Optimization of an Engine Controller by means of a Genetic Algorithm using History of Search. In *Proceedings of the 3rd Asia-Pacific Conference on Simulated Evolution and Learning (SEAL)*, pages 2929–2934, 2000.

[11] Alain Ratle. Accelerating the Convergence of Evolutionary Algorithms by Fitness Landscape Approximation. In *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature (PPSN V)*, volume 1498 of *LNCS*, pages 87–96. Springer-Verlag, 1998.

[12] Tobias Latzke, Sven Behnke, and Maren Bennewitz. Imitative Reinforcement Learning for Soccer Playing Robots. In *RoboCup 2006: Robot Soccer World Cup X*, volume 4434 of *LNAI*, pages 47–58. Springer-Verlag, 2007.

[13] Georges Matheron. Principles of geostatistics. *Economic Geology*, 58(8):1246–1266, 1963.