Reducing Trials by Thinning-Out in Skill Discovery

Hayato Kobayashi[1], Kohei Hatano[2], Akira Ishino[1], and Ayumi Shinohara[1]

[1] Graduate School of Information Science, Tohoku University, Japan
[2] Department of Informatics, Kyushu University, Japan
kobayashi@shino.ecei.tohoku.ac.jp, hatano@i.kyushu-u.ac.jp,
ishino@ecei.tohoku.ac.jp, ayumi@ecei.tohoku.ac.jp

Abstract. In this paper, we propose a new concept, thinning-out, for reducing the number of trials in skill discovery. Thinning-out means to skip over such trials that are unlikely to improve discovering results, in the same way as "pruning" in a search tree. We show that our thinning-out technique significantly reduces the number of trials. In addition, we apply thinning-out to the discovery of good physical motions by legged robots in a simulation environment. By using thinning-out, our virtual robots can discover sophisticated motions that is much different from the initial motion in a reasonable amount of trials.

1 Introduction

Skill discovery is a task to obtain an ability to perform well in a target domain, through *trial and error*. It can be formulated as an optimization problem, in which the goal is to find a solution x in a vast search space X that maximizes (or minimizes) a score function $f : X \to \boldsymbol{R}$. When the shape of f is not known and there is no satisfactory problem-specific algorithm or heuristic, meta-heuristic guides to pick up the next candidate, based on the previous candidates and their evaluated scores. Various meta-heuristics are proposed and experimented, such as Genetic Algorithms (GA) and Simulated Annealing (SA). These meta-heuristics in general contribute to reduce the number of trials, in order to find a good solution x in X. Nevertheless, for some problem domains, the number of trials is still too large, especially when each trial consumes a considerable amount of time and costs. Skill discovery in robot movements, which we treat as a target application in this paper, is an instance of them.

When GA or SA picks up the next candidate to try, it is often duplicated: it could be already experimented in the past trials. Ratle [1] proposed an efficient method to avoid such duplications using function approximation. He showed that the method can reduce the number of actual function calls by creating an approximate model of the score function using kriging interpolation and using the model instead of the original score function for evaluating some of the next generations. When the score function can not be regarded as a deterministic function because of the noisy environment, duplicated trials are meaningful to increase the certainty of the evaluations. Sano et al. [2] proposed Memory-based

V. Corruble, M. Takeda, and E. Suzuki (Eds.): DS 2007, LNAI 4755, pp. 127–138, 2007.

Fitness Evaluation GA for noisy environment. They estimated more proper fitness values (or scores) by weighted average of neighboring scores, so as to reduce the number of trials more than multiple sampling methods (i.e., to evaluate fitness values several times in each trial.)

In this paper, we take another approach to reduce the trials, based on the idea that we can theoretically determine whether or not the selected candidates are worth evaluating, if the gradient of the score function is given. If the candidate x is unlikely to improve the results obtained so far, we do not perform the trial and just skip it. We call this method *thinning-out*, which contrasts to *pruning* in a search tree. One advantage of our method is that we can naturally combine the thinning-out with any search methods including GA and SA as well as random search. In preliminary experiments, we combined it with a simple random search method, and observed that the resulting number of trials is usually reduced to logarithmic with respect to the number of candidates. In this paper, we show that our thinning-out method significantly reduces the number of trials with a small failure rate with a combination with GA.

We address skill discovery by legged robots in a simulation environment as its application. For legged robots to function in the real world, they must need the ability to acquire such basic skills as walking, running, pushing, kicking, and so on. The ability for robots to learn some skills is known as *skill learning*, and is regarded as important. For several years, there have been many studies conducted on skill learning by legged robots. Kim and Uther [3] studied the learning of fast quadruped locomotion skills by modeling the locus of their gait as a quadrangle. Kohl and Stone [4] also studied the learning of stable quadruped locomotion by modeling the locus of their gait as a semi-ellipsoid. Fidelman and Stone [5] proposed a learning method for acquiring the ball-grasping skill. The learning task consists of two layers, the first for walking and the second for pinching the ball by its chin. Kobayashi et al. [6] studied the reinforcement learning to trap a moving ball. The goal of the learning was to acquire a good timing to initiate the catch motion, depending on the distance and the speed of the ball, whose movement was restricted to one dimension.

In this paper, we make robots to discover good shot motions, as Zagal and Solar [7] also addressed in their work. Compared with other tasks mentioned above, discovery of good shot motions could be more challenging, because it is difficult to construct a good model for shot motions. Although our work is similar to Zagal and Solar's, our parameterization is more flexible. On the other hand, flexibility of parameters implies that it takes many trials in the discovery process. Lee et al. [8] successfully realized flexible movements of legged robots to climb over a variety of obstacles by reinforcement learning with supervised information. Since we can not prepare supervised information for good motions other than an initial motion, we must find good motions in large search spaces. Our thinning-out method can reduce the number of trials in such problem and realize the flexibility in feasible trials.

The remainder of this paper is organized as follows. In Section 2, we describe the concept of thinning-out and propose two inferring methods for it. In

Section 3, we evaluate our method using the minimization problem of mathematical test functions. In Section 4, we apply our method to discovery of good shot motions by legged robots in a simulation environment. Finally, Section 5 presents our conclusions.

2 Thinning-Out

In this section, we treat the maximization problem of unknown score function. We assume that the score function is continuous and, to some extent, smooth over the search space. Our assumption seems reasonable because each robot movement is continuous and thus small changes of parameters will not affect the score significantly. Based on this assumption, we infer local shapes of the score function. Given a candidate point to try, we estimate the score of the candidate point by using the distance from the nearest neighborhood whose score is known. We thin-out the candidate point if the estimated upperbound of the score is lower than the current highest score. Note that our method does not take into account the distance from the current highest point. In other words, we do not assume that the true highest point lies near the current highest one. Therefore an expected point with high score, even it is far from the current highest point, has a chance to be tried. In summary, our method is robust and it is unlikely to get stuck in local maxima.

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 natural extension of c-Lipschitz for some constant c in the textbooks.

Definition 1 (Lipschitz condition). *Let $\boldsymbol{R}$ be the set of real numbers, X be a metric space with metric d, and $f : X \to \boldsymbol{R}$ be a score 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 $x_1, x_2 \in \boldsymbol{R}$ that*

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

The function g is called Lipschitz function*.*

Suppose that a score function f is g-Lipschitz continuous. Then, for any points x_1 and x_2, an upperbound of $f(x_1)$ is obtained by

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

Our thinning-out strategy is to infer a proper Lipschitz function which characterizes the score function f, so as to obtain an upperbound of the score of a candidate point. If the upperbound is smaller than the current best score, we do not have to try the candidate point. We will explain the details of our methods to infer Lipschitz functions soon. Our thinning-out condition is formally described as follows:

Definition 2 (Thinning-out condition). *Let x_b be the point whose score $f(x_b)$ is the current best. Let x_c be a candidate to try. Let x_n be the nearest neighbor of the candidate. Given an inferred function $\hat{g} : \boldsymbol{R} \to \boldsymbol{R}$, x_c is said to satisfy the thinning-out condition with respect to $\hat{g}$, if it holds that*

$$f(x_n) + \hat{g}(d(x_c, x_n)) \leq f(x_b)$$

Now we propose two methods to infer Lipschitz functions.

Max Gradient Method

Suppose that we know the maximum gradient

$$c = \max_{x_1, x_2 \in X,\ x_1 \neq x_2} \frac{|f(x_1) - f(x_2)|}{d(x_1, x_2)}$$

of the score function f, over any two different points x_1 and x_2 in X. Then for the function defined by $g(d) = c \cdot d$, it is easy to verify that f is g-Lipschitz continuous. Thus g can be used to thin-out candidates without errors. Since c itself is unavailable in practice, we substitute the maximum gradient from every two points in past trials so far, which will become a good approximation of c after enough trials. We call it *Max Gradient* (MG) method. This method may have small error rate, because it deals with the worst case scenario. However, it can hardly thin-out candidates in rough score functions obviously, since the estimated value of the Lipschitz function is too conservative in many cases.

Gathering Differences Method

Meta-heuristics picks up many samples from an interesting region expected to have the best score. We can get the shape of the interesting region by using information of points which are densely packed in past trials so far. Thus we infer Lipschitz functions by gathering the differences of the scores, from the smallest one in ascending order of the distance between the points, until the summation of the distances become greater than the distance between x_c and x_n, as shown in Algorithm 1. It will become a good approximation after enough trials, since a line connecting fairly close two points can approximate the gradient of the function nearby the points. We call it *Gathering Differences* (GD) method. This method can thin-out many candidates, since it may infer the local shape of the score function in the interesting region. However, it may wrongly thin-out them, because it is just heuristics and does not have any theoretical propriety.

3 Performance Evaluation of Thinning-Out

We need efficient sampling methods for picking up candidates, since our thinning-out method in the previous section just skips over the candidates. In this paper, we utilize Genetic Algorithm (GA), which is one of the *meta-heuristics* methods, because we intend to address discovery problems in which the score function is

Algorithm 1. Gathering differences method

input : distance and the set Hist of pairs $(x, f(x))$ observed so far
output : An inferred value of the Lipschitz function g(distance)

initialize Diff as a map from $\mathcal{R}$ to $\mathcal{R}$;
foreach x_1, $f(x_1)$ *in* Hist **do**
  foreach x_2, $f(x_2)$ *in* Hist **do**
    Diff $[d(x_1, x_2)] \leftarrow |f(x_1) - f(x_2)|$;
  end
end
sum_diff_point $\leftarrow 0$;
sum_diff_score $\leftarrow 0$;
foreach diff_point, diff_score *in* Diff *in ascending order w.r.t.* diff_point **do**
  sum_diff_point $\leftarrow$ sum_diff_point + diff_point ;
  sum_diff_score $\leftarrow$ sum_diff_score + diff_score ;
  if sum_diff_point $\geq$ distance **then**
    return sum_diff_score ;
  end
end
return ∞ ;

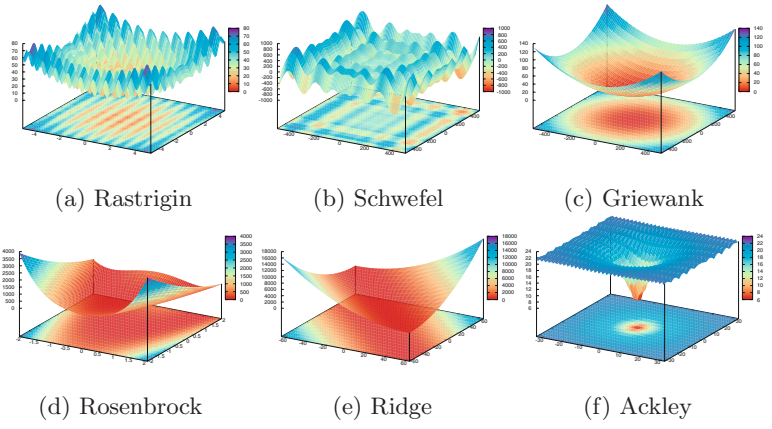
Algorithm 2. Evaluation of a candidate with thinning-out.

input : candidate
output : score

while candidate *satisfies the thinning-out condition* **do**
  candidate $\leftarrow$ a random perturbation of candidate ;
end
score $\leftarrow$ `Evaluate(candidate)`;
return score ;

unknown. We can combine GA and thinning-out by designing the evaluation function of a candidate as shown in Algorithm 2. Since thinning-out is meta strategy, we can easily utilize other meta-heuristics methods in the same way as GA. Although we have experimented other methods such as hill climbing, simulated annealing, and policy gradient and verified that they worked well with thinning-out, we omitted them here because of space limitations.

We use the minimization problem of mathematical test functions for verifying the performance of our thinning-out method. The evaluation by test functions is commonly performed for verifying the performance of meta-heuristics. In this paper, we use Rastrigin, Schwefel, Griewank, Rosenbrock, and Ridge functions, which are used by Hiroyasu et al. [9]. In addition, we add Ackley function [10], because we think our thinning-out method is not good at the function with deep rapid valleys. Characteristics of these functions are as follows. Rastrigin, Schwefel, Griewank, and Ackley functions have multiple peaks, although Griewank and Ackley functions have a single peak with a global view. Griewank, Rosenbrock



(a) Rastrigin (b) Schwefel (c) Griewank

(d) Rosenbrock (e) Ridge (f) Ackley

Fig. 1. The shape of test functions in 2 dimensions

Table 1. Thinning-out rate and error rate of GA+MG and GA+GD in 6 test functions in 2 dimensions. Each rate is the average over 100 experiments using 100 candidates.

function	GA+MG		GA+GD	
	thinning (%)	error (%)	thinning (%)	error (%)
Rastrigin	81.47	0.23	78.60	1.99
Schwefel	82.98	0.18	78.97	1.42
Griewank	82.82	0.24	78.98	1.01
Rosenbrock	82.23	0.05	78.71	0.69
Ridge	81.89	0.00	80.40	0.75
Ackley	79.91	2.77	71.48	2.94

and Ridge functions have the design variables' dependency. Fig. 1 shows the shapes of these functions in 2 dimensions.

We compared the performance of our methods by three different viewpoints: the kind of test functions, the number of candidates, and the dimension of test functions. In each experiment, the step size parameter (for mutation and perturbation) of GA is 1% of the domain size of each dimension. We used ϵ-*thinning-out* in the same way as ϵ-*greedy* in reinforcement learning [11], because our methods can not always thin-out candidates safely. ϵ-thinning-out evaluates a candidate with probability ϵ, and otherwise, it skips over the candidate. Consequently, ϵ-thinning-out can hold out the possibility for evaluating candidates that are wrongly thinned-out once and avoid never halting by thinning-out all candidates. We set $\epsilon = 0.01$.

Firstly, we compared the performance by the kind of test functions. Table 1 shows thinning-out rate and error rate of GA+MG and GA+GD in each test function. The thinning-out rate means the rate of thinned-out candidates in all

Table 2. Results of minimization by GA, GA+MG, and GA+GD in 6 test functions in 2 dimensions. Each result means the minimum score of 50 actual trials and is the average over 100 experiments.

function	GA	GA+MG	GA+GD
Rastrigin	24	13	19
Schwefel	712	435	439
Griewank	43	32	33
Rosenbrock	418	330	296
Ridge	11542427	8233764	8878178
Ackley	19	18	18

candidates, and the error rate means the rate of wrongly thinned-out candidates in thinned-out candidates. Both MG and GD totally reduced the number of trials by more than 70 % with low error rates. As anticipated, both MG and GD have slightly higher error rates in Ackley function. Table 2 shows the results of minimization by GA, GA+MG, and GA+GD. Both GA+MG and GA+GD always got better results than GA in all test functions. This is because a small number of errors (i.e., wrongly thinned-out candidates) will not affect final results as shown in Fig. 2. Contrary to our expectation, these tables indicate that

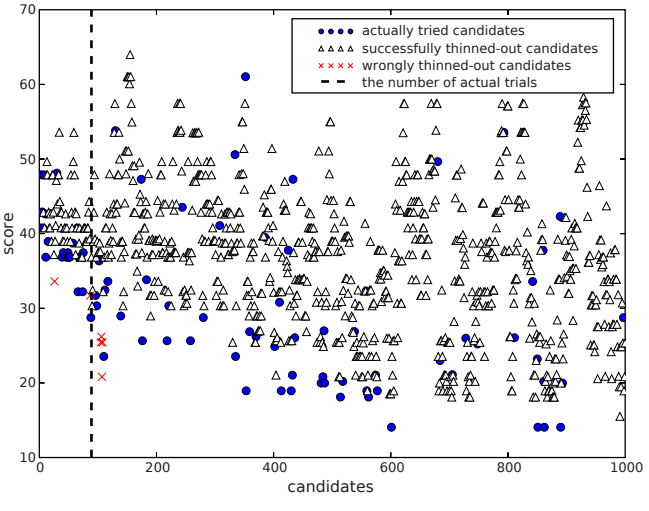


Fig. 2. This graph shows an example of minimization process with thinning-out. The example is in an experiment using 1000 candidates in 2 dimensional Rastrigin function. Circles, triangles, and crosses represent actually evaluated candidates, successfully thinned-out candidates, and wrongly thinned-out candidates, respectively. The dashed line indicates the same number of candidates as actual trials with thinning-out. The minimum score in the left side of the line means the minimization result without thinning-out.

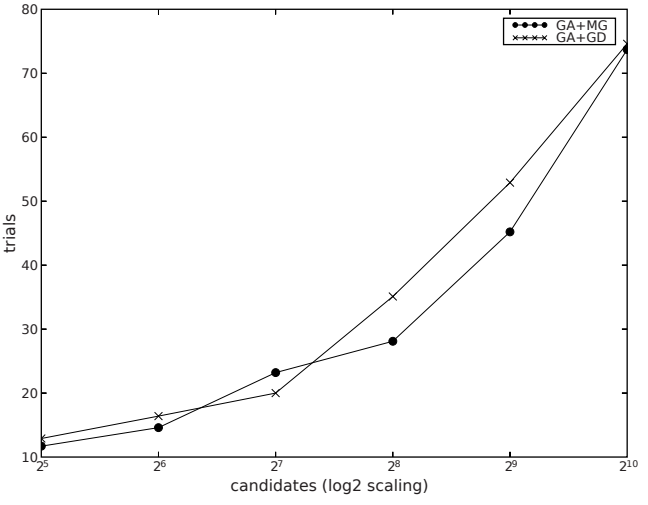


Fig. 3. This graph shows the number of actual trials over the number of candidates by GA+MG and GA+GD in Rastrigin function in 2 dimensions. Each result is the average over 10 experiments

the thinning-out rate of MG is higher than that of GD. The later experiment, however, finds that the thinning-out rate of GD is higher than that of MG in higher dimensions.

Secondly, we compared the performance by the number of candidates. Fig. 3 shows the relationship between the number of trials and that of candidates. The graph indicates that the thinning-out rate gets higher as the number of candidates gets larger. For example, the results of 80 trials by both GA+MG and GA+GD is almost the same as that of $2^{10} = 1024$ trials by GA, if there are no critical errors. We theoretically analyzed the number of trials with respect to the number of candidates for a simplified case. For a score function $f(x) = x$, let n_c be the number of candidates by random search. Then the number of trials is reduced to $O(\log(n_c))$ by our thinning-out method. If we can prove it in more practical functions (e.g., $f(x) = x^n$), the result has practical significance, because random sampling with thinning-out worked better than GA in low dimensions. In this paper, we picked up GA since random search can hardly suggest good candidates in high dimensions.

Finally, we compared the performance by the dimension of test functions. Table 3 shows thinning-out rate and error rate of GA+MG and GA+GD in various dimensions. The table indicates that MG can hardly thin-out the large number of candidates in high dimensions. On the other hand, GD can also thin-out almost the same number of candidates in high dimensions as that in low dimensions. This result indicates that GD is better than MG in high dimensions, if there are no critical errors. In the next section, we show that there are no critical errors in our intended problem, skill discovery.

Table 3. Thinning-out rate and error rate of GA+MG and GA+GD in Rastrigin function in 2, 5, 10, 50, and 100 dimensions. Each rate is the average over 100 experiments using 100 candidates

dimension	GA+MG		GA+GD	
	thinning (%)	error (%)	thinning (%)	error (%)
2	82.07	0.41	77.44	2.44
5	59.83	0.16	77.26	4.08
10	45.52	0.12	75.08	5.09
50	35.23	0.19	70.84	6.89
100	35.70	0.13	69.89	6.12

4 Discovery of Good Shot Motions

4.1 Creation of Initial Motions

In this paper, we use AIBO, which was developed by Sony Corporation, as a legged robot. Physical motions of AIBO are realized 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*. The key-frames are the 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.

4.2 Discovery Process

We directly utilize the key-frames for discovering good shot motions. All we do is to create sketchy motions, that is to indicate the key frames for the motion, and it is possible to realize flexible search in the neighborhood of the skeleton without modeling the movement and setting extra-parameterization. We fix the number of key-frames and interpolations. In other words, the search space of our discovery process has $15n_k$-dimensions, where n_k means the number of key-frames. In the process, we sample $x \in \boldsymbol{R}^{15n_k}$, make our robot perform a shot motion generated from x, and calculate scores as the formula $r_b \cdot (1 - |\theta_t - \theta_b| / \theta_c)$, where r_b and θ_b mean the distance and angle to the kicked ball, and θ_t means the target direction for shots. The formula linearly reduces the value in inverse proportion to the difference between θ_b and θ_t. θ_c is a constant for the degree of reducing, and we set $\theta_c = \pi/4$.

4.3 Experiments and Results

We applied our methods to discovery of good shot motions by legged robots in a simulation environment. We slightly extended the 3D simulator developed

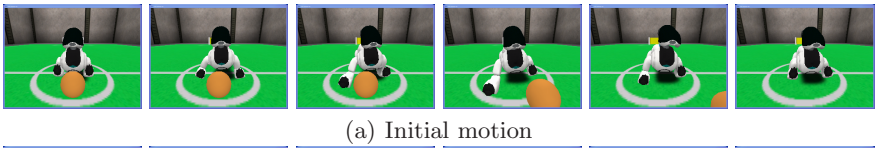
Table 4. Results of discovery of good motions by GA, GA+MG, and GA+GD. Each result means the maximum score of 50 actual trials and is the average of 10 experiments.

GA	GA+MG	GA+GD
936	940	1058

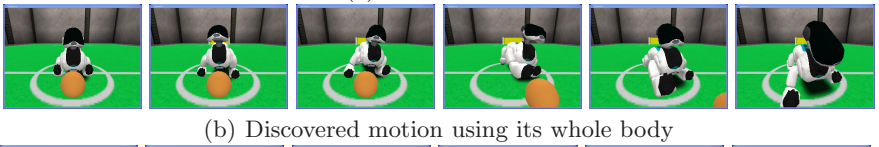
by Zaratti et al. [12] and used it. Although this simulator can absolutely not produce complete, real environments, it is suitable for verifying the performance of such new methods, because we can perform reproducible measurement without annoying real noise, as well as without damaging our robots.

Our experiments require much more time since the simulation of physical motions itself requires complex computation, even though our discovery processes lies in the simulator. Actually, each experiment in this section took a couple dozens hours. Therefore, thinning-out can make discovery processes more efficient in a simulation environment as well as real environments, because it can reduce time-consuming trials themselves.

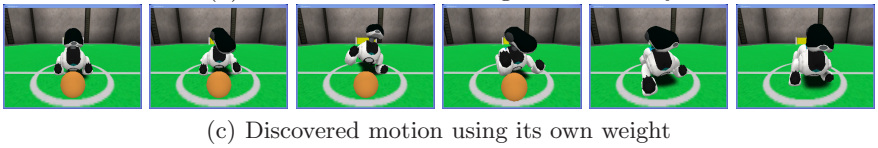
We experimented using the motion for shooting a ball to a left oblique direction with its right leg, as an initial motion. The search space is 75(= 15 joint angles * 5 key frames)-dimensions. The step size parameter of GA is $\pi/36$ in each dimension. Table 4 shows the results using GA, GA+MG and GA+GD. The table indicates that GA+GD get better results than GA and GA+MG. Although the difference is small, it should be noted that GA+GD used several hundred candidates. In other words, the result of GA+GD is almost the same result of several hundred trials in GA. The result ought to be improved by using a better sampling method. Fig. 4 shows the initial motion and two better motions which were discovered by using our methods. The motion (b) uses its whole body, although the initial motion (a) uses almost only



(a) Initial motion



(b) Discovered motion using its whole body



(c) Discovered motion using its own weight

Fig. 4. Initial motion and discovered motions

its right leg. The motion (c) uses its own weight, swinging down its right leg. It should be noted that the motion (c) is much different from the initial motion (a). This result can not be achieved by modeling the initial motion and adjusting the parameters of the model. These results, especially the motion (c), indicate that our skill discovery method with key-frames has flexibility suitable for practical use. The movies of these discovered motions are available online (`http://www.shino.ecei.tohoku.ac.jp/~{}kobayashi/movies.html`).

5 Conclusions and Future Work

In this paper, we proposed the concept "thinning-out", which is effective for the problems that take much more evaluation time in each trial. We proposed two methods (MG and GD), which infer Lipschitz functions for thinning-out. By the experiments on the minimization problem of several test functions, we verified that MG can safely thin-out few candidates, and conversely GD can fearlessly thin-out many candidates, especially in high dimensions. The results of test functions also suggests that thinning-out can be utilized widely in other different problems. In addition, we applied our methods to discovery of good shot motions by legged robots in a simulation environment. Our virtual robots discovered sophisticated motions that is much different from the initial motion in a feasible number of trials.

From now on, the experiments using real robots will be needed to verify that thinning-out can treat real noise. Discovery of good shot motions in real environments, however, will be unrealistic, because we must estimate the distance to the kicked ball and restore the ball to the initial point carefully with each trial for themselves. Therefore, we plan to make our robots perform *autonomous learning* in the same way as Kobayashi et al. [6]. Autonomous learning of forward shots is readily achievable in much the same way as the method of them, and that of other shots may be possible by utilizing ceiling cameras.

We also need to more accurate inference methods for Lipschitz functions, because the two methods proposed in this paper have both merits and demerits. Although we came up with several ideas, which include a method using the average, median, and weighted average of gradients, other than the two methods, they did not work well. For example, we may be able to infer more proper values by utilizing heuristics depending on each problem.

References

1. Ratle, A.: Accelerating the Convergence of Evolutionary Algorithms by Fitness Landscape Approximation. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H-P. (eds.) Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature (PPSN V). LNCS, vol. 1498, pp. 87–96. Springer, Heidelberg (1998)

2. Sano, Y., Kita, H., Kamihira, I., Yamaguchi, M.: 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), pp. 2929–2934 (2000)
3. Kim, M.S., Uther, W.: Automatic Gait Optimisation for Quadruped Robots. In: Proceedings of 2003 Australasian Conference on Robotics and Automation, pp. 1–9 (2003)
4. Kohl, N., Stone, P.: Machine Learning for Fast Quadrupedal Locomotion. In: The Nineteenth National Conference on Artificial Intelligence (AAAI2004), pp. 611–616 (2004)
5. Fidelman, P., Stone, P.: The Chin Pinch: A Case Study in Skill Learning on a Legged Robot. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 59–71. Springer, Heidelberg (2007)
6. Kobayashi, H., Osaki, T., Williams, E., Ishino, A., Shinohara, A.: Autonomous Learning of Ball Trapping in the Four-legged Robot League. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 86–97. Springer, Heidelberg (2007)
7. Zagal, J.C., del Solar, J.R.: Learning to Kick the Ball Using Back to Reality. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 335–347. Springer, Heidelberg (2005)
8. Lee, H., Shen, Y., Yu, C.H., Singh, G., Ng, A.Y.: Quadruped Robot Obstacle Negotiation via Reinforcement Learning. In: Proceedings of the 2006 IEEE International conference on robotics and Automation (ICRA2006) (2006)
9. Hiroyasu, T., Miki, M., Sano, M., Shimosaka, H., Tsutsui, S., Dongarra, J.: Distributed Probabilistic Model-Building Genetic Algorithm. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U-M., Beyer, H-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 1015–1028. Springer, Heidelberg (2003)
10. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press (1996)
11. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
12. Zaratti, M., Fratarcangeli, M., Iocchi, L.: A 3D Simulator of Multiple Legged Robots based on USARSim. In: Lakemeyer, G., et al. (eds.) RoboCup 2006. LNCS (LNAI), vol. 4434, pp. 13–24. Springer, Heidelberg (2007)