Dimensionality Reduction of Scene and Enemy Information in Mario

Hisashi Handa
Graduate School of Natural Science and Technology Okayama University
Okayama 700–8530, JAPAN
Email: handa@sdc.it.okayama-u.ac.jp

***Abstract*—Mario AI is one of competitions on Computational Intelligence. In the case of video games, agents have to cope with a large number of input information in order to decide their actions at every time step. We have proposed the use of Isomap, a famous Manifold Learning, to reduce the dimensionality of inputs. Especially, we have applied it into scene information. In this paper, we newly extend to enemy information, where the number of enemies is not fixed. Hence, we introduce the proximity metrics in terms of enemies. The generated low-dimensional data is used for input values of Neural Networks. That is, at every time step, transferred data by using a map from raw inputs into the low-dimensional data are presented to Neural Networks. Experimental results on Mario AI environment show the effectiveness of the proposed approach.**

I. Introduction

Recently, video games have been used as benchmark problems on Computational Intelligence. Competitions on video games such as Mario, Ms.Pacman and Car Racing, have been held at Computational Intelligence Conferences [1], [2], [3]. Real-time strategies are needed to constitute agents for such games [4]. They also provide more complex state space than conventional board games. Due to such complexity, conventional AI approaches where researchers try to extract and transfer experts' knowledge into rules or programs may be no longer enough to describe video game agents. In our previous study, Isomap, one of Manifold Learning algorithms, is used to reduce the dimensionality of scene information of Mario [5]. This method works well if the difficulty of Mario is the easiest, i.e., 0. Because, the enemy information provided to agents in the proposed method is the relative position of only two enemies. It was enough for that the difficulty parameter: 0. For the difficulty parameter > 0, however, there are more than two enemies in one scene.

In this paper, we employ two maps, which are generated by Isomap, to constitute input values of Neural Networks: The first map transfers the scene information into low dimensional data. This map is the same as in [5]. The second map proposed in this paper is to cope with the enemy information. In order to apply the Isomap to the enemy information, we introduce a similarity measure between sets of enemies in two scenes. In the proposed method, Neuroevolution with these maps is carried out: A single play is examined for the evaluation of the fitness of individuals. At each time step in the game, scene information and enemy information are extracted by calling APIs included in the competition library. Low dimensional data are separately acquired by using scene and enemy maps and the corresponding information. The low dimensional data is given to the Neural Network, i.e., the individual. Mario can take actions by referring to the output of the Neural Network. The weight value of the Neural Network is optimized by Particle Swarm Optimization.

Related works are described as follows: Dimension reduction techniques including SOM are often used in conventional reinforcement learning community and as genetic operations or visualization tools of individuals in Evolutionary Optimization [6], [7], [8], [9], [10]. In the case of Evolutionary Learning, there is few research. We can guess some reasons of this: One of main stream of applying Evolutionary Learning to robotics is of Learning Classifier Systems (LCS) [11]. In the case of LCS, schemata are quite important notion of them. If we use dimension reduction techniques, it would be difficult to constitute effective schemata. In robotics, Manifold Learning have attracted much attention for generating Maps [12]. Our research can be regarded as an extension of this study to Evolutionary Learning. The researches regarding Mario AI are enumerated as follows: [13] introduced Mario AI competition and examined not only Neuroevolution by Evolutionary strategies but also HyperGP. REALM is proposed for playing Mario recently in [14]. The REALM is a rule base approach such as Classifier Systems. The merit of this approach is that knowledge can easily be incorporated into the rule sets. Besides, actions can be defined as more sophisticated one by using designer's heuristics.

The organization of the remainder of the paper is as follows: Section II briefly summarizes Isomap by referring to the algorithm, and the application to Mario AI. The general calculation procedure of Particle Swarm Optimization, which is used as an Evolutionary Algorithm for evolving Neural Networks, is briefly introduced in section III. Computer simulations on the Mario AI Competition are carried out in Section IV.

II. Dimensionality Reduction of Scene and Enemy Information

A. Isomap

The first generation of Manifold Learning algorithms, i.e., Locally Linear Embedding and Isomap, is proposed in 2000 [15], [16], [17]. These have attracted much attention especially in image processing community since these can

978-1-4244-7835-4/11/$26.00 ©2011 IEEE

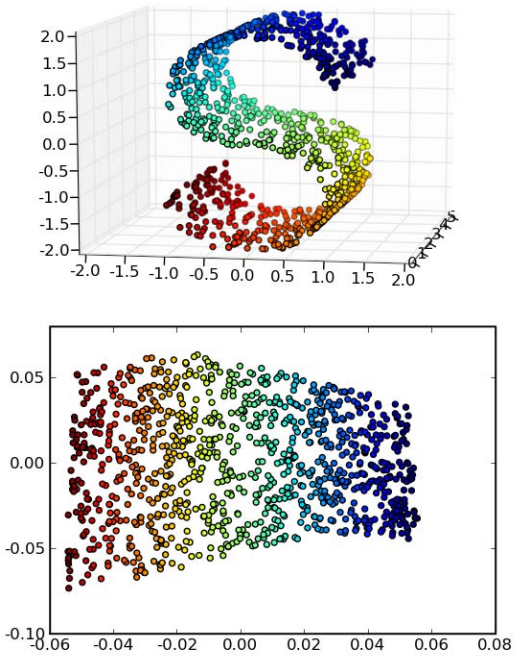

Fig. 1. S-shaped data (UPPER) and the typical result by Manifold Learning (LOWER)

embed the relationship among a large number of images into two dimensional space naturally. Figure 1, for instance, depicts the S-shaped data, which is often used to explain the effectiveness of the Manifold Learning. The upper graph in this figure denotes original data in a three-dimensional space, which are sampled from a two-dimensional manifold. Note that colors of points have no special meanings. They are just for ease of understandings. The lower graph in the figure is a typical result by Manifold Learning for the original data. The order of color sequence is maintained in this resultant two dimensional data. In this paper, we employ basic Manifold Learning methods, i.e., Isomap. In the Isomap, the geodesic distance on Manifolds is used instead of the Euclidean distance. The procedure of the Isomap is described as follows:

1) K-Nearest Neighbor method is adopted all the input data x_i. Then, a neighborhood graph x is constructed such that nodes in the graph is connected if they are of neighbor in the sense of K-Nearest Neighbor method. Distance $d_G(i,j)$ of edge among connected nodes is set to be $d_x(i,j)$, i.e., the distance between the input data x_i and x_j by a given distance metrics[1].
2) For all the pairs of (x_i, x_j) of input data, the shortest path distance $d_G(i,j)$ on the neighborhood graph G are calculated.
3) A low dimensional projection is generated by calling a metric MDS (Multi Dimensional Scaling) and by using the the shortest path distance $d_G(i,j)$.

[1]Usually, we employ Euclidean distance for $d_x(i,j)$.

B. Dimensionality Reduction of Scenes

A large number of scene informations, which exclude enemy information, are collected from one hundred games by using a modified ForwardJumping Agent in [18]. The modified ForwardJumping is re-designed for avoiding some enemies. This is not still sophisticated but is enough to collect various kinds of scene informations. One thousands of scene information is randomly sampled from these collected scene informations.

The distance between two scenes i, j is defined as follows:

$$d_x(i,j) = \sum_{k=1}^{22}\sum_{l=1}^{22}(1-(|k-11|+|l-11|)/22)\times \alpha_k \times J(x_i(k,l), x_j(k,l)),$$

where $x_i(k,l)$ denotes a value of a cell if the k^{th} column and the l^{th} row in scene information i. α_k means a coefficient which takes 4 if $k > 11$. Otherwise it takes 1. The cell in the 11^{th} column and the 11^{th} row means the center of scene information, i.e, the position located Mario. The function $J(\cdot,\cdot)$ for judging if the corresponding cells between two scenes i, j is defined as follows:

$$J(a,b) = \begin{cases} 0 & a \text{ is equal to } b \\ 1 & \text{otherwise.} \end{cases}$$

Hence, this distance metrics is weighted for the front and the adjacent cells of Mario.

By using the above distance metrics and Isomap, the dimension of scene informations is reduced as delineated in Figure 2. This is the result such that the dimension of the reduced space is 2. The plotted points in the center graph in this figure mean the scene informations in the reduced space. The depictions surrounding the graph denote scenes which correspond to circles in the graph. As illustrated in this figure, similar depictions, e.g., the left two depictions at upper row, is transferred into near points in the graph. Note that the right depiction at upper row is similar scene with the above two depictions, but Mario is jumping in the right depiction so that transferred point for this depiction away from points of the left two depictions. This is caused by the definition of scene informations, which is set to be relative to Mario.

C. Dimensionality Reduction of Enemy Information

Another Isomap for enemy information is constructed as follows: Suppose that the coordination and velocity of an enemy i are represented by x_i and v_i, respectively. Beside, there are I and J enemies in two scenes k, l. The proximity metrics $m(k,l)$ in terms of enemies between these two scenes is defined as follows:

$$m(k,l) = \sum_{i}^{I}\sum_{j}^{J}\exp(-(|\boldsymbol{x}_i - \boldsymbol{x}_j| + \alpha \times |\boldsymbol{v}_i - \boldsymbol{v}_j|)/\beta)$$

where α and β denote the coefficient for balancing the affects by the position and the speed. They are set to be 20 and 10, respectively. This metrics is not a distance metrics since it

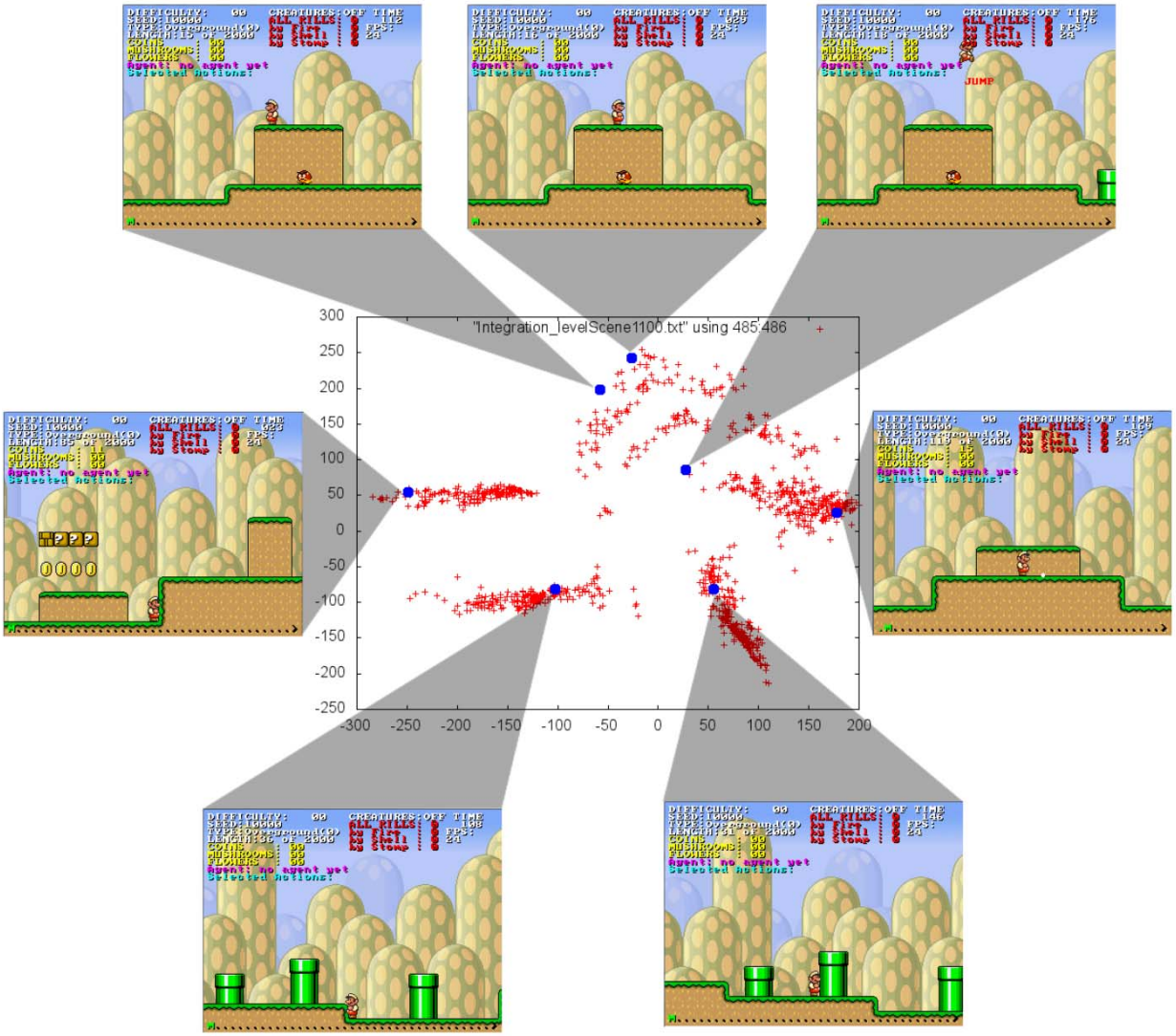

Fig. 2. A typical result of the dimensionality reduction by using Isomap for scene information of Mario, excluding enemy information

does not hold the axiom of distance metrics. This metrics, however, is employed for Isomap in this paper.

As in the scene information, a large number of enemy informations are collected from one hundred games by using a modified ForwardJumping Agent in [18]. One thousands of enemy informations are randomly sampled from these collected enemy informations. Figure 3 shows a typical result of dimensionality reduction by Isomap for enemy information.

III. Particle Swarm Optimization

Neuroevolution is evolving Neural Networks by using Evolutionary Computation. There are two trends of the neuroevolution. The first one is that the structures and the weights of Neural Network, such as NEAT [19]. By using the NEAT, we do not have to care about the structure of Neural Networks due to automatic design of the structure. Another one is the use of Particle Swarm Optimization for training Neural Networks [20], [21]. PSO is used in not only neuroevolution but also supervised learning of Neural Networks. PSO has been recognized as a powerful tool for learning Neural Networks. In the learning track in the Mario AI competition, the number of game plays in the competition is fixed. Therefore, we choose PSO as learning algorithms of Neural Networks.

We employed Clerc's constriction factor method as PSO algorithms [22], [23]. The procedure of this PSO is described

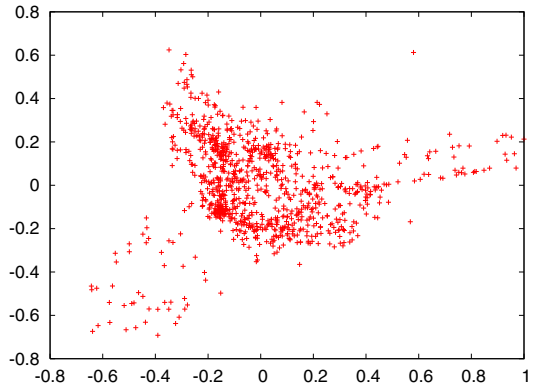

Fig. 3. A typical result of the dimensionality reduction by using Isomap for enemy information of Mario

as follows:

1) Initialize individuals and their velocities.
2) Each individual is evaluated.
3) Set $pbest$ for each individual and $gbest$.
4) Change the velocity and individual according to following equations:

$$\begin{aligned} v &\leftarrow K * [v + c_1 * rand() * (pbest - x) + \\ &\qquad c_2 * rand() * (gbest - x)] \\ x &\leftarrow x + v \end{aligned}$$

5) Go back to step 2. until a criterion is met.

The learning parameters c_1 and c_2 are set to be 2.05.

$$K = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} = 0.729,$$

where $\phi = c_1 + c_2$.

IV. Experiments

A. Proposed Method

The proposed method consists of two stages (Figure 4). In the Isomap stage (cf. section II), two maps for the scene informations and the enemy informations are constituted. In the neuroevolution stage, games are iterated for evolving neural networks. As an evolutionary algorithm, Particle Swarm Optimization is utilized. The details of the neural networks are described in the following subsection. In the evaluation, one game play is examined for each individual. The fitness F of individuals is defined in equation (1).

$$\begin{aligned} F &= d * w_d + s * w_s + m * w_m + c * w_c + \\ &\quad k_t * w_{k_t} + k_s * w_{k_s} + k_f * w_{k_f} + k_{sh} * w_{k_{sh}} + \\ &\quad c_h * w_{c_h} + t * w_t, \end{aligned} \tag{1}$$

where the meanings of variables and the value of corresponding weights are summarized in Table I.

At each time step in games, a scene information is presented to agent. In accordance with the map, corresponding low dimensional data is given to Neural Networks. The output vector of the Neural Networks is used to decide the actions of agents.

TABLE I
Variables and values of weights used in fitness function in learning track competition

Symbols of var.	*Meanings of var.*	*Corresponding Weight Value* $w_\star$
d	passed distance	1
s	state (win or loss)	1024
m	mode (normal, large, or fire)	32
c	coins gained	16
k_t	total kills	42
k_s	kills by stomp	12
k_f	kills by fire	4
k_{sh}	kills by shell	17
c_h	hidden coins gained	24
t	time left	8

In this paper, the Neural Network of the Mario agents has three output neurons: The output value o_1 of the first neuron is used to decide the direction such that Mario should go to. The Mario agents go right if $o_1 > 0$. Otherwise, they go left. The second output decides whether the DOWN button is pressed or not. 0 is threshold to decide it. Even if the Neural Network decides to press the DOWN button, it is neglected when the Mario Agents on the ground. SPEED button had been pressed during episodes. According to the third neuron, Mario agents jump by pressing the JUMP button during that the value of the third neuron is positive. The inputs of the Neural Network module consists of the low dimensional data by scene and enemy maps, and boolean values of MarioOnGround and MarioAbleToJump, which are provided by APIs.

B. Experimental Results

In the first experiments, we compare the proposed method with two kinds of Neural Networks as in [13] and Neuroevolution with Isomap only for scene information as in [5]: Two Neural Networks employ raw inputs surrounding the Mario agents. For the smaller Neural Network, scene and enemy informations on one cell around the Mario agents are recognized. For the larger one, three cells around the Mario agents can be perceived. The number of generations is set to be 200. For all the Neural Networks including the proposed method, PSO explained in section III is used as evolutionary algorithm.

Figure 5 shows the experimental results, which are averaged over the result of 50 different levels, where the difficulty of these levels is the same but they generated with different seeds. For each level, 5 runs are examined with different seeds for PSO. That is, totally, 250 runs are executed for each algorithm. The lines in the figure shows the average fitness in each algorithm. "Dual Isomap" stands for the proposed method, i.e., two maps of scene and enemy informations. Meanwhile, "Single Isomap" stands for Neuroevoluion with a map for scene informations. In the case of the "Single Isomap," the relative positions of enemy informations are given to agents. The proposed method outperforms other algorithms. The number of weights in the smaller and the

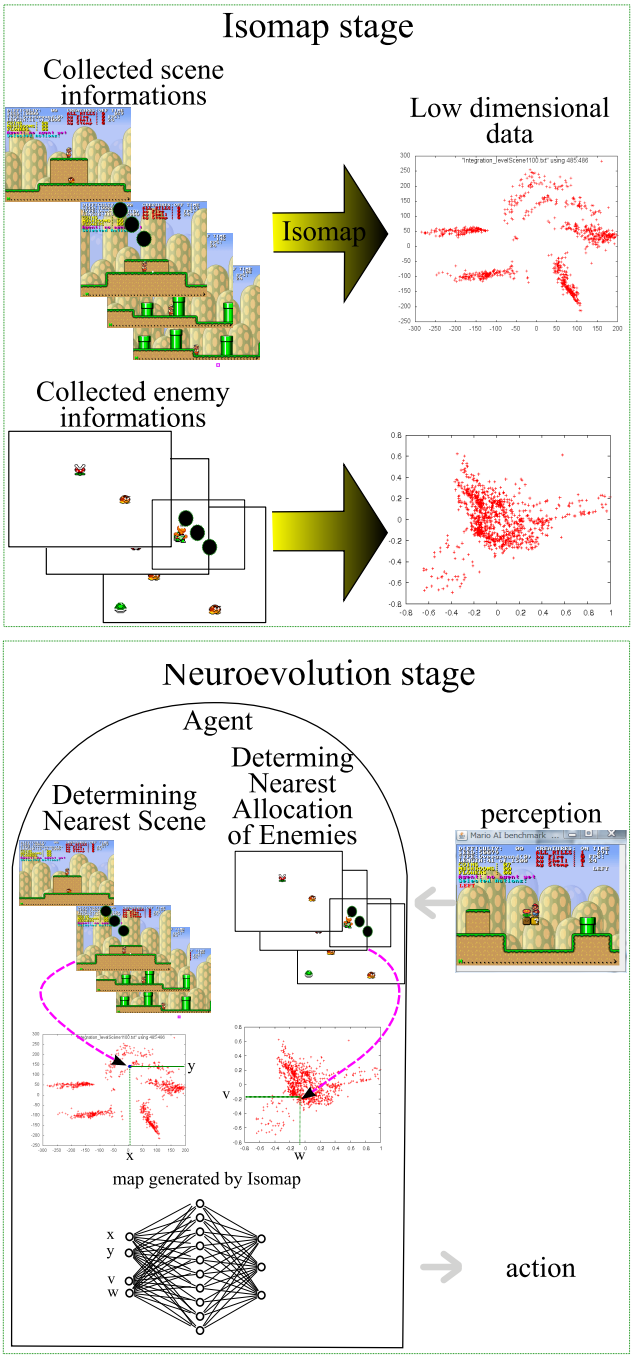

Fig. 4. A depiction of the procedure of the proposed method

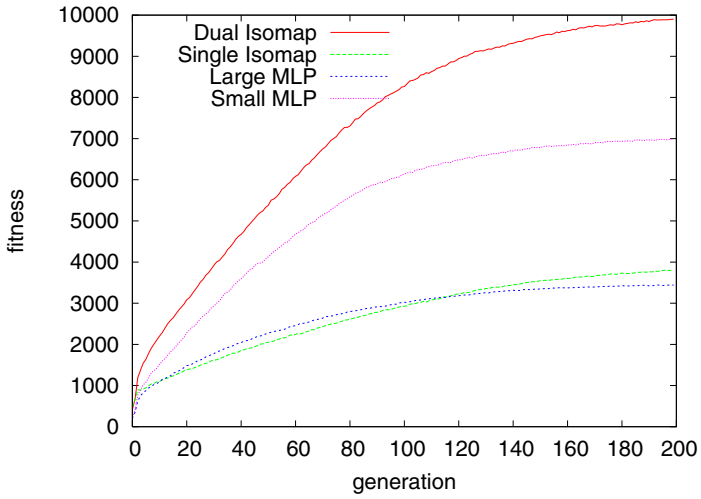

Fig. 5. Experimental results: Comparison with NNs: raw input NN (small and large), and NN with low-dimensional data of only scene information

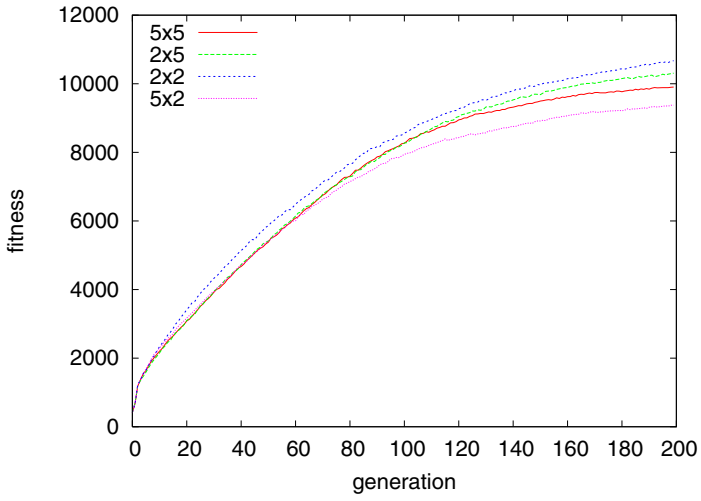

Fig. 6. Experimental results: the effect of the dimension of low-dimensional data. 5x2 in the legend stands for that the dimensions of the low-dimensional data of scene and enemy information are 5 and 2, respectively.

larger Neural Networks are set to be 21x10+10x3 = 240, and 101x10+10x3= 1040, respectively. Therefore, although the larger Neural Network can utilize rich input information, it seems to be stuck in local optima due to the broad search space. In the case of the proposed method, the number of weights is 13x10+10x3 = 160. 13 inputs of the proposed method are composed of 5-dimensional inputs by the maps of scene and enemy informations, two boolean values (MarioAbleToJump, and MarioOnGround), and one constant value for threshold. Hence, such less number of weight would contribute the performance improvements. The map of enemy information works well in comparison with the Single Isomap. The Single Isomap works well if the difficulty of levels is the easiest where enemies in a scene are at most two. This good performance of the proposed method is caused by the treatment of several enemies by using the map.

Secondly, the dimensions of the low dimensional data for scene and enemy informations are changed as depicted in Figure 6. The prior part and the posterior part of the legends in the graph stands for the dimension of the low dimensional data for scene and enemy, respectively. 5x2, for instance, means the result of the proposed method with 5-dimensional data for scene information, and 2-dimensional data for enemy information. 2x2 shows the best performance among them. 5-dimensional data of scene information did not work well.

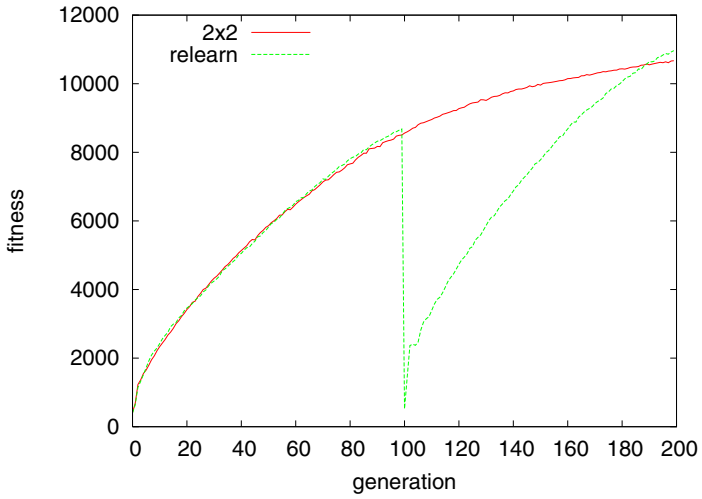

Fig. 7. Experimental results on restart strategy with invulnerable state

Finally, we examin the effectiveness of a restart strategy, where the state of Mario agents before the restart is invulnerable state. That is, in the first 100 generations, Mario agents are not killed by enemies but can fall in the holes. At the 101^{th} generation, all the individuals are randomly initilized. One elite until the 101^{th} generation is inserted into the population. In following 100 generations, usual Mario Environment is carried out. Figure 7 compares this restart setting with ordinal run as in the previous figures. Even if Mario agents is in invulnerable state, agents can stomp enemies so that they successfully learn how to stomp enemies without risks. The difference between lines before restart is not so large. The reason of this is that it is difficult to jump over holes.

V. Conclusions

In this paper, we introduced the proximity metrics in terms of enemies. By using this metrics, Isomap — a famous Manifold Learning Algorithm — was carried out in order to generate low-dimensional data for enemy informations. The generated low-dimensional data is used for input values of Neural Networks. That is, at every time step, transferred data by using map from raw inputs into the low-dimensional data are presented to Neural Networks. Experimental results showed the effectiveness of the proposed approach.

Acknowledgment

This work was partially supported by the Grant-in-Aid for Exploratory Research, the Grant-in-Aid for Scientific Research (B), and the Grant-in-Aid for Young Scientists (B) of MEXT, Japan (18656114, 21360191, and 21700254).

References

[1] Togelius, J., Lucas, S.M., Duc Thang, H., Garibaldi, J.M., Nakashima, T., Tan, C.H., Elhanany, I., Berant, S., Hingston, P., MacCallum, R.M., Haferlach, T., Gowrisankar, A., and Burrow, P. (2008) 'The 2007 IEEE CEC Simulated Car Racing Competition', *Genetic Programming and Evolvable Machines*, Available: http://dx.doi.org/10.1007/s10710-008-9063-0

[2] Loiacono, D., Togelius, J., Lanzi, P.L., Kinnaird-Heether, L., Lucas, S.M., Simmerson, M., Perez, D., Reynolds, R.G., and Saez, Y. (2008) 'The WCCI 2008 Simulated Car Racing Competition', *Proceedings of the IEEE Symposium on Computational Intelligence and Games*

[3] Lucas, S.M. (2005) 'Evolving a Neural Network Location Evaluator to Play Ms. Pac-Man', *IEEE Symposium on Computational Intelligence and Games*, pp.203–210.

[4] Lucas, S.M. and Kendall, G. (2006) 'Evolutionary Computation and Games', *IEEE Computational Intelligence Magazine*, Vol. 1, No. 1, pp.10–18.

[5] IES

[6] Tateyama, T., Kawata, S., Oguchi, T. (2006) 'A teaching method using a self-organizing map for reinforcement learning', *Artificial Life and Robotics*, **7(4)**, 193–197 (2006) Vol. 7, No. 4, pp.193–197.

[7] Handa, H., Ninomiya, A., Horiuchi, T., Konishi, T., Baba, M. (2001) 'Adaptive State Construction for Reinforcement Learning and its Application to Robot Navigation Problems', *Proceedings of the 2001 IEEE Systems, Man and Cybernetics Conference*, pp.1436–1441.

[8] Hiroyasu, T., Miki, M., Sano, M., Shimosaka, H., Tsutsui, S., Dongarra, J. (2003) 'Distributed Probabilistic Model-Building Genetic Algorithm', *Proceedings of the 2003 Genetic and Evolutionary Computation Conference (GECCO 03)*, pp.1015–1028.

[9] Najafi, M., Beigy, H. (2009) 'Using PCA to improve evolutionary cellular automata algorithms', *Proceedings of the 2009 Genetic and Evolutionary Computation Conference (GECCO 09)*, pp.1129–1130.

[10] Obayashi, S., Sasaki, D. (2003) 'Visualization and Data Mining of Pareto Solutions Using Self-Organizing Map', *Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization*, pp.796–809.

[11] Goldberg, D. E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.

[12] Nishizuka, K., Yairi, T., Machida, K. (2005) 'Simultaneous Localization and Mapping of Mobile Robot Using Nonlinear Manifold Learning', *Proceedings of the 36th International Symposium on Robotics*.

[13] Togelius, J., Karakaovskiy, S., Koutnik, J., and Schmidhuber, J. (2009) 'Super mario evolution', *Proceedings of IEEE Symposium on Computational Intelligence and Games (CIG)*

[14] Bojarski, S., and Congdon, C.B. (2010) 'REALM: A Rule-Based Evolutionary Computation Agent that Learns to Play Mario', *Proceedings of behalf of the organizing committee we would like to welcome you to the 2010 IEEE Conference on Computational Intelligence and Games (CIG 2010)*, pp.83–90.

[15] Tenenbaum, J.B., de Sliva, V., Lagford, J.C. (2000) 'A Global Geometric Framework for Nonlinear Dimensionality Reduction'. *Science*, Vol. 290, No. 22, pp.2319–2323.

[16] Huo, X., Ni, X.S., Smith, A.K. (2007) 'A survey of manifold-based learning methods', *Recent Advances in Data Mining of Enterprise Data*, Edited by Liao, T.W., and Triantaphyllou, E., Singapore: World Scientific.

[17] Roweis, S.T., Saul, L.K. (2000) 'Nonlinear Dimensionality Reduction by Locally Linear Embedding', *Science*, Vol. 290, No. 22, pp.2323–2326.

[18] Togelius, J., Karakaovskiy, S., and Baumgarten, R. (2010) 'The 2009 Mario AI Competition', *Proceedings of 2010 IEEE Congress on Evolutionary Computation (CEC)*, pp.4264–4271.

[19] Stanley, K.O., Miikkulainen, R. (2002) 'Evolving neural networks through augmenting topologies', *Evolutionary Computation*, Vol. 10, No. 2, pp.99–127.

[20] Shi, Y (2004) 'Particle Swarm Optimization', *IEEE coNNectionS*, Vol. 2, No.1, pp.8–13.

[21] Kennedy, J., and Eberhart R.C.(1995) 'Particle swarm optimization', *Proceedings of IEEE International Conference on Neural Networks*, pp.1942–1948.

[22] Eberhart, R.C, and Shi, Y. (2001) 'Particle swarm optimization: developments, applications and resources', *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, Vol.1, pp.81–86.

[23] Clerc, M. (1999) 'The swarm and the queen: towards a deterministic and adaptive particle swarm optimization,' *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, Vol.3, pp.1951–1957.