Dimension Reduction by Manifold Learning for Evolutionary Learning with Redundant Sensory Inputs

Hisashi Handa, *Member, IEEE*, and Hiroshi Kawakami

***Abstract*— The optimization of the number and the alignment of sensors is quite important task for designing intelligent agents/robotics. Even though we could use excellent learning algorithms, it will not work well if the alignment of sensors is wrong or the number of sensors is not enough. In addition, if a large number of sensors are available, it will cause the delay of learning. In this paper, we propose the use of Manifold Learning for Evolutionary Learning with redundant sensory inputs in order to avoid the difficulty of designing the allocation of sensors. The proposed method is composed of two stages: The first stage is to generate a mapping from higher dimensional sensory inputs to lower dimensional space, by using Manifold Learning. The second stage is using Evolutionary Learning to learn control scheme. The input data for Evolutionary Learning is generated by translating sensory inputs into lower dimensional data by using the mapping.**

I. Introduction

The alignment of sensors of evolutionary learning agents is one of the most important problems to design effective intelligent agents/robotics. Learning mechanisms in such agents are often unable to function well if insufficient information is given to agent, i.e., wrong alignment of sensors, or a small number of sensors. Meanwhile, agents with redundant sensory inputs takes considerable amount of learning time since learning algorithms must cope with high dimensional input data.

In this paper, Isomap, one of Manifold Learning Algorithms, is used to reduce the number of dimensionality of sensory inputs [1], [2]. By using the reduced inputs, agents decide their actions and learn policies to achieve a given task. An important feature of the Manifold Learning Algorithms is to preserve local topological relationship among data. Figure 1, for instance, depicts the S-Shaped data, which is often used to explain the effectiveness of the Manifold Learning. The left graph in this figure denotes original data in three dimensional space, which is sampled from two dimensional manifold. Note that colors of points have no special meanings. They are just for ease of understandings. The right 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.

Agents with a large number of sensors can capture obstacles with higher resolution. As mentioned above, it is quite difficult to learn policies from such high dimensional space. In this paper, instead of the pursuit of the optimal number and alignment of sensors of agents, translated data by using Manifold Learning is used as the input of agents. We assume that the distribution of high dimensional sensory inputs is sparsely sampled from a manifold. Sensors are used to measure a physical quantity so that several sensors are equipped to a robot. Such sensors are correlated with each other. Therefore, the sensory inputs are often distributed on a certain distribution which represents the nature of robot's environment.

In this paper, we propose a two-stage learning method for mobile robots: The first stage is to learn the mapping from high dimensional sensory inputs to low dimensional data. The high dimensional sensory inputs are collected by the elopement in another environment of tasks where there are various kinds of obstacles. The Manifold learning is used to generate the low dimensional data. The second state is to learn the policy of robots by using Evolutionary Algorithms. At every time step, the robots perceive the high dimensional sensory inputs as the same as in the first stage. Then, the low dimensional data associated with perceived the high dimensional data is given to an individual in the Evolutionary Algorithms. This paper elucidates the effectiveness of the dimension reduction in the case of Evolutionary Learning.

II. Manifold Learning

The first generation of Manifold Learning algorithms, i.e., Locally Linear Embedding and Isomap, is proposed in 2000 [1], [2], [3]. These have attracted much attention especially in image processing community since these can embed the relationship among a large number of images into two dimensional space naturally. Hence, several subsequent algorithms have been proposed such as Laplacian Eigenmaps, Hessian Eigenmaps, and so on. In this paper, in order to investigate the effectiveness of the information processing on Manifolds, we employ a basic Manifold Learning Method, i.e., Isomap.

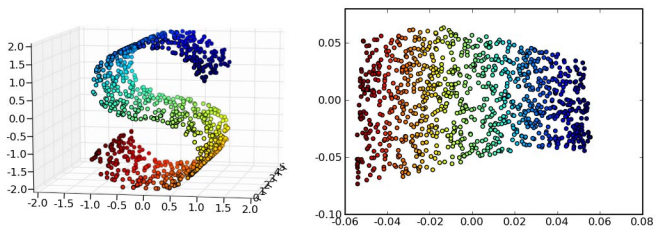

Fig. 1. S-Shaped data (LEFT) and the typical result by Manifold Learning (RIGHT)

Hisashi Handa is with the Graduate School of Natural Science and Technology, Okayama University, Okayama, 700-8530, Japan (phone: +81-86-251-8250; fax: +81-86-251-8256; email: handa@sdc.it.okayama-u.ac.jp).

Hiroshi Kawakami is with the Graduate School of Informatics, Kyoto University, Kyoto, 606-8530, Japan.

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

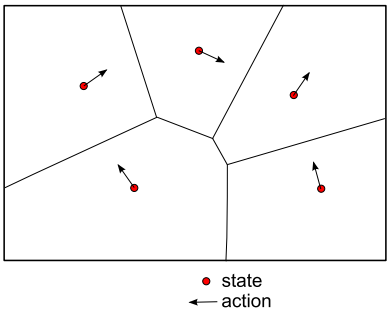

Fig. 2. Example of an Individual: Instances in perceptual input space

A. Isomap

Isomap, proposed by Tenenbaum *et al.* is one of the most famous Manifold Learning Algorithms [1]. In the Isomap, the geodesic distance on Manifolds is used instead of the Euclidean distance. As mentioned in section I, the Isomap can extract low dimensional data effectively due to the geodesic 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., Euclidean distance between the input data x_i and x_j.
2) For all the pair s 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 and by using the the shortest path distance $d_G(i,j)$.

III. Instance-Based Policy Learning

The instance based policy learning proposed by Miyamae is an evolutionary approach for solving reinforcement learning problems [4]. It is composed of several vectors, called instances. Each instance consists of a state part and an action part. For a given perceptual input at each time step, the nearest instance is activated as nearest neighbor method. The action of the activated instance is taken. Figure 2 depicts an example of instances in perceptual input space, where the dimensions of states and actions are 2 and 1, respectively. The position of circles and the orientation of arrows denote the state part and the action part of instances, respectively. As delineated in the figure, the perceptual input space is segmented to several subspaces as in Voronoi diagrams. Each subspace is associated with one of instance. That is, each of instance activates for perceptual inputs in a corresponding subspace. The arrows in the figure illustrate actions for corresponding instances.

Figure 3 describes the genotype for the Instance Based Policy learning. s_{ij} and a_{ik} denote the j^{th} element of state vector and the k^{th} element of action vector of i^{th} instance.

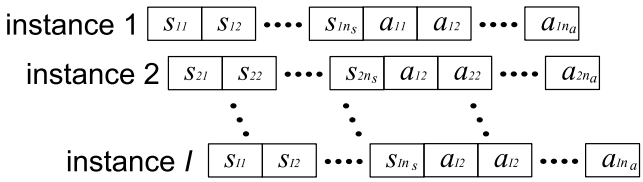

Fig. 3. Representation of Individuals in Instance Based Policy learning

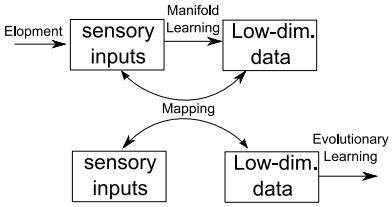

Fig. 4. Diagram of the proposed method

I indicates the number of instances, which is predefined. n_s and n_a represents the number of states and actions, respectively. All the variables s_{ij} and a_{ik} are represented by a real value. Hence, any Evolutionary Algorithms for continuous function optimization problems can be used. This paper utilizes CMA-ES while the original paper of the IBP learning method uses the Real-Coded GA proposed by their research group for evolution [4], [5]. The reason of the utilization is due to the availability of the source code. We believe there is no significant difference between the CMA-ES and the Real-Coded GA since we do not have to find out the optimal policies with high degree of precision as in ordinal function optimization problems.

IV. Proposed Method

A two-stage learning method for mobile robots is proposed in this paper as depicted in Figure 4. The first stage is to constitute a mapping from sensory inputs $\boldsymbol{x}$ to low dimensional data $\boldsymbol{y}$. The second stage is Evolutionary Learning to achieve a given task. In this stage, at every time step t, sensory inputs $\boldsymbol{x}_t$ is translated to corresponding input $\boldsymbol{y}_t$ by using the mapping. Hence, the inputs for the learner is $\boldsymbol{y}_t$.

A. Constitution of Mapping

At first, data collection is carried: A robot moves in a given environment around. In this paper, we set up another environment for this elopement, where there are variety of obstacles. After a large number of sensory inputs are gathered, data with no activated sensors are eliminated. Moreover, a predefined number of data is randomly chosen from the eliminated data set.

The dimension reduction method is carried out for the chosen data. This paper examines not only Isomap but also Kernel PCA algorithms for this purpose [6]. The chosen data $\boldsymbol{x}_i$ and the reduced data $\boldsymbol{y}_i$ are associated, where $i = 1, \ldots, n_d$, and n_d indicated the number of the chosen data.

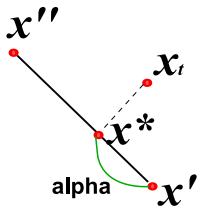

Fig. 5. Translation of sensory inputs

B. Translation of Sensory Inputs in Evolutionary Learning Phase

As mention above, at every time step t, sensory inputs $\boldsymbol{x}_t$ should be translated: Firstly, the nearest and the second nearest point $\boldsymbol{x}', \boldsymbol{x}''$ from the chosen data for $\boldsymbol{x}_t$ is found out. Secondly, the current sensory inputs $\boldsymbol{x}_t$ is projected to the line defined by two points $\boldsymbol{x}', \boldsymbol{x}''$. The projected point $\boldsymbol{x}^*$ is regarded as the relative position α on the line as delineated in Figure 5:

$$\alpha = \frac{\boldsymbol{x}^* - \boldsymbol{x}'}{\boldsymbol{x}'' - \boldsymbol{x}'}.$$

The inputs $\boldsymbol{y}_t$ for individuals are defined as follows:

$$\boldsymbol{y}_t = \alpha(\boldsymbol{y}'' - \boldsymbol{y}') + \boldsymbol{y}',$$

where $\boldsymbol{y}'$ and $\boldsymbol{y}''$ are points in reduced space, which are associated with $\boldsymbol{x}'$ and $\boldsymbol{x}''$, respectively.

In the case that the dimension of sensory inputs is high and the number of chosen data is large, it takes much time for finding out the nearest and the second nearest data $\boldsymbol{x}', \boldsymbol{x}''$. Locality-Sensitive Hashing (LSH) is used for finding such nearest points effectively [7].

V. Experiments

A. Configuration of Robots

We employ Simbad, a Java 3d robot simulator, for constructing simulated environment [8]. The mobile robot used in this paper is described as follows: The radius of the robot is 0.3 meters. 20 time steps per second are simulated. The robot has a large number of sonar sensors. We examined 72, 24, 12, and 6 sonar sensors for the proposed method and the Instance Based Policy Learning without dimension reduction method. In addition, 4 and 3 sonar sensors are examined for the IBP without dimension reduction method. The allocation ways of these sensors are the same: The first sensor is set to be in the front of the robot. Other remaining sensor is equiangularly allocated, i.e., at every 5 degree for 72 sonar sensors. The range of sensors is 1.5 meters. The robot goes forward with 0.5 meters per second if there is no activated sensor. Otherwise the translation and the rotational velocity of the robot is set to be 0.2 meters per second and $(a - 0.5) \times \pi/2$ meters per second, respectively, where a denotes the action of agent. The action a in this paper continuously varies from 0 to 1.

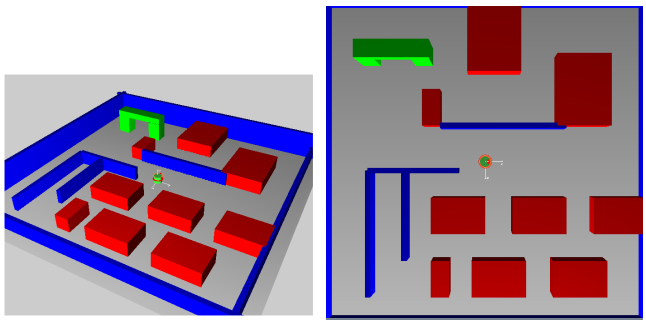

Fig. 6. Simulated environment for collecting a variety of sensory inputs

B. Dimension Reduction

Figure 6 delineated the simulated environment for collecting variety of sensory inputs. The size of the field is 18 meters $times$ 18 meters. Various size of walls and blocks are stored. A robot with 72 sonar sensors moves in this field around for sufficient time. As mentioned in the previous section, data for dimension reduction is randomly chosen. The number of chosen data n_d is set to be 3000. From this 3000 data for 72 sonar sensors, we generate other kinds of dataset by neglecting some sensor values, i.e., data data for 24, 12, and 6 sonar sensors.

Several mappings are generated by using Isomap or Kernel PCA. As mentioned above, we now have 4 kinds of dataset. For each dataset, dimension reduction method is carried out. The dimensions of reduced space are set to be 2, 5, and 10. In the case of the dataset for 6 sonar sensors, we only apply the 2 and 5 dimensions of reduced space. Figure 7 shows the distributions of data in reduced space by Isomap (UPPER) and by Kernel PCA (LOWER). These are result of chosen data for 72 sonar sensors and of reducing to 2 dimensional space.

C. Experimental Settings

Table I summarizes the methods examined in this paper. As mentioned previously, for chosen data with 6 sonar sensors, the dimension reduction to "10 dimensional reduced space" is not carried out. Three simulated environments are examined as depicted in Figure 8. In these depictions, a robot is located on the initial position. The goal area is located at the red line in the left side of these figures. 500 seconds (equivalent to 10,000 steps) are allowed to use for a single examination. The episode will be terminated if the robot reaches the goal, the robot bumps obstacles, or 500 seconds are exceeded.

The evaluation of a single examination is calculated as follows: The following function e is applied if a robot reaches to the goal.

$$e = 0.1 + 1.0/(\text{No. steps})$$

The second term in the right side of this equation is a very small number in comparison with the first term, i.e., 0.1. Therefore, the evaluation for success is almost 0.1 but it is greater if the robot could reach to goal faster. Otherwise, the

TABLE I
SUMMARY OF EXAMINED METHODS

No. sonar sensors	Dim. of Reduced Space	Dim. Reduction	Evolutionary Learning
72, 24, 12, or 6	10, 5, or 2	Isomap	IBP
		Kernel PCA	
72, 24, 12, 6, or 4	n/a	n/a	

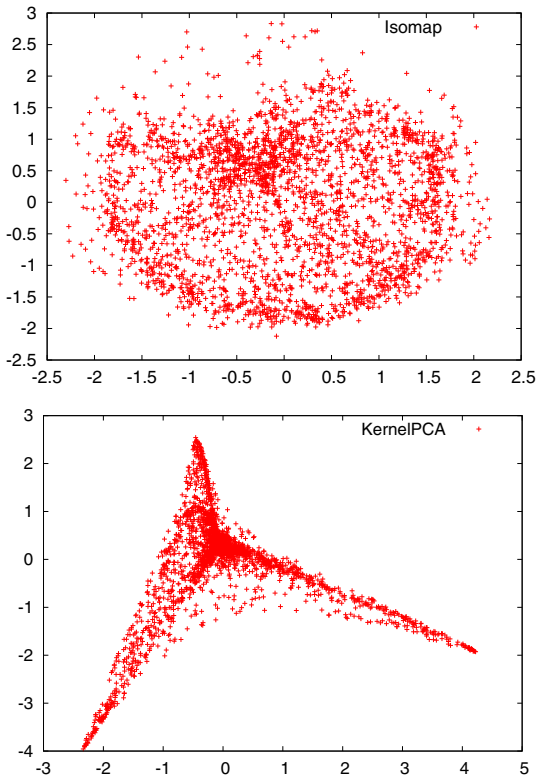

Fig. 7. Reduced spaces by Isomap (UPPER) and Kernel PCA (LOWER)

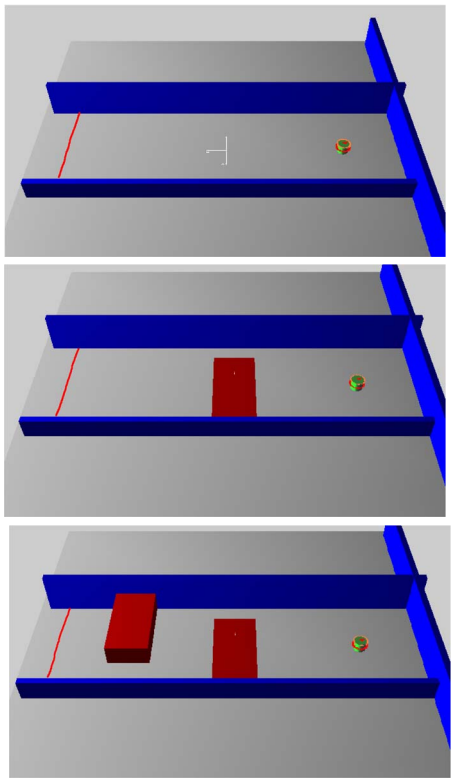

Fig. 8. Simulated Environments

evaluation is as follows:

$$e = -1.0/(\text{No. steps}) \times (\text{distance to the goal})$$

This evaluation is a very small negative number. The evaluation is worse if the robot bumps promptly or the robot could not get up close to the goal. For a single fitness evaluation, five examinations are carried out. The fitness function of individuals is calculated by the sum of five evaluations.

The number of instances in the Instance Based Policy optimization is set to be 5. n_s is 2, 5, or 10, which is the same as the dimensions of reduced space. n_a is 1, i.e., action a in the previous subsection. The length of individuals is $(n_s + n_a) *$ (the number of instances), i.e., 15, 30, or 55.

D. Experimental Results

Figure 9 shows the experimental results for the simplest simulated environment, i.e., the upper picture in Figure 8. The top graph is the result of IBP without dimension reduction. The graphs on the left side is the result of the proposed method, namely, IBP with dimension-reduced inputs by Isomap. The graphs on the right side is the result of IBP with dimension-reduction by Kernel PCA. The dimension of reduced space is 2 (the second row); 5 (the third row); and 10 (the bottom row). All the plotted lines are averaged values over 20 runs. The x and y axes indicate the number of fitness evaluations and the fitness value, respectively. For the success and fail examinations, the evaluation value is almost 0.1 and 0, respectively. Therefore the averaged fitness is almost same as (0.1 $\times$ the averaged number of examinations) per single fitness evaluation. The legends of plotted lines means that "IBP.j" indicates the results are by IBP without dimension reduction, which have j sensors, and "Isomap.$j.k$" and "KernelPCA.$j.k$" indicate results by IBP with corresponding dimension reduction method from j sensors to k dimension. In the simplest simulated environment, robots should learn they must avoid walls. All the algorithms perform well. The

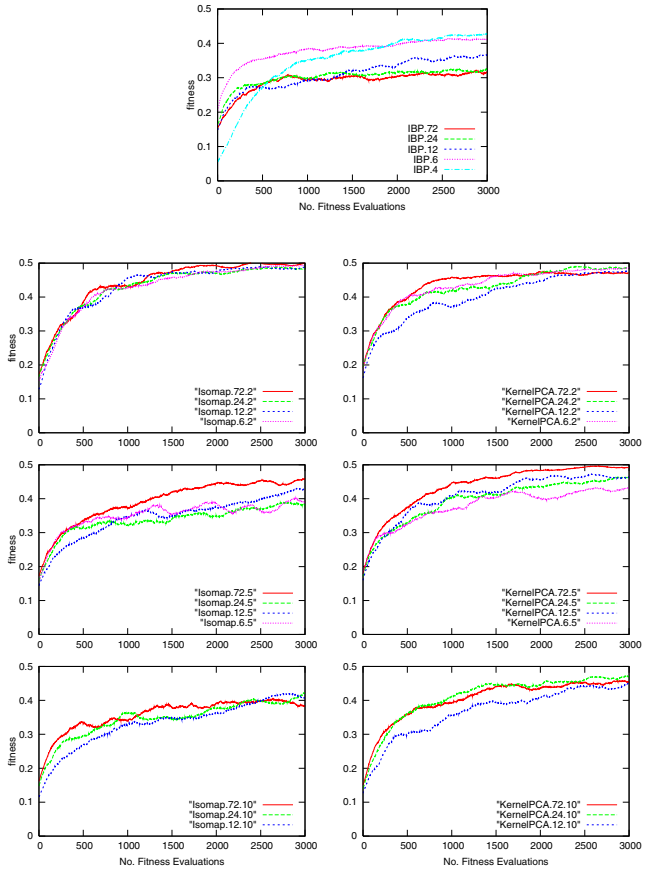

Fig. 9. Experimental results: Simulated environments with no obstacles (UPPER IBP only; RIGHT IBP with Isomap; LEFT IBP with Kernel PCA)

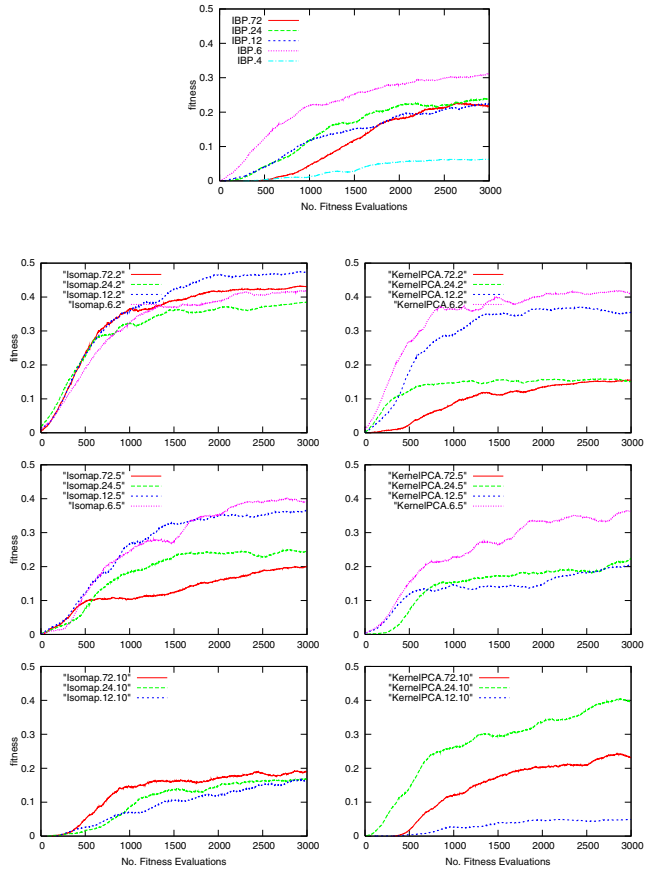

Fig. 10. Experimental results: Simulated environments with one obstacle (the allocation of graphs are the same as Figure 9

performances are slightly decreasing if the number of states, i.e., j in IBP, and k in Isomap and Kernel PCA.

Figure 10 shows the experimental results for the simulated environment with one obstacle, i.e., the middle picture in Figure 8. Isomap which reduces 2-dimensional space, and Kernel PCA(6.2, 12.2) work well. IBP with a larger number of sensors decrease their performance.

Figure 11 illustrates the results for two obstacles. Isomap (6.2, and 12.2) keep their performance high. In the case of higher reduced dimensions, the degree of performance deterioration of Isomap is worse than one of KernelPCA, except for KernelPCA.72.*.

VI. Related works

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 [9], [10], [11], [12], [13]. In the case of Evolutionary Learning, there is few research. Although it would be caused by our poor skill of Google, we can guess some reasons of this: One of main stream of applying Evolutionary Learning to robotics is of Learning Classifier Systems [14]. 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. Another evolutionary approach is use of Neural Networks, i.e., NeuroEvolution [15]. In this case, they would rely on the information processing capability of Neural Networks for non-linear phenomena. In robotics, Manifold Learning have attracted much attention for generating Maps [16]. Our research can be regarded as an extension of this study to Evolutionary Learning.

VII. Conclusions

In this paper, we proposed the use of Manifold Learning for Evolutionary Learning with redundant sensory inputs in order to avoid the difficulty of designing the allocation of sensors. The proposed method is composed of two stages: The first stage is to generate a mapping from higher dimensional sensory inputs to lower dimensional space, by using Manifold Learning. This paper employs Isomap for this purpose. The second stage is using Evolutionary Learning to learn control scheme. The input data for Evolutionary Learning is generated by translating sensory inputs into lower dimensional data by using the mapping. Three algorithms, i.e., Instance-Based Policy learning algorithm without dimension reduction, with Isomap, and with Kernel PCA,

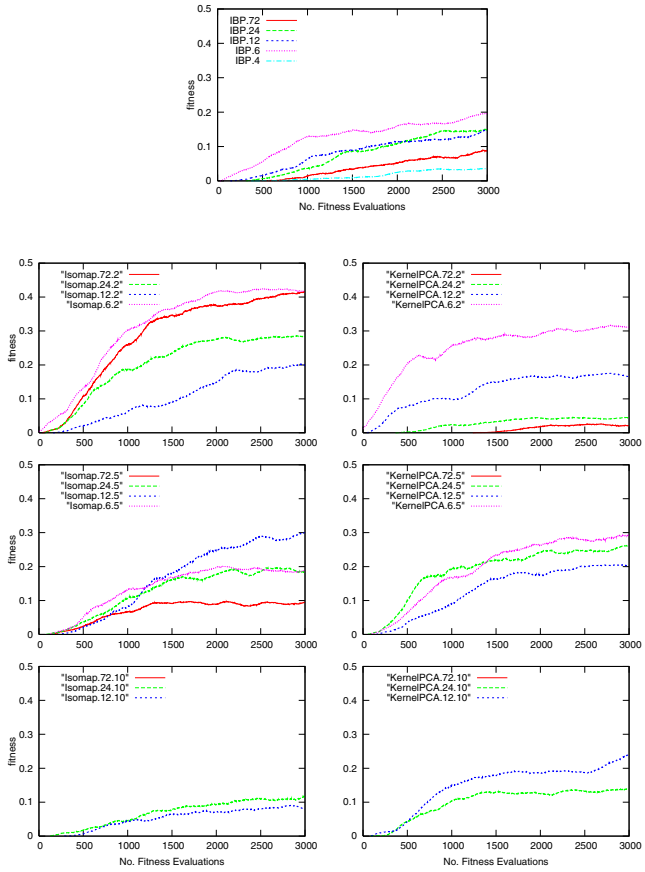

Fig. 11. Experimental results: Simulated environments with two obstacles (the allocation of graphs are the same as Figure 9

are examined on the navigation task of mobile robots. The experimental results show the effectiveness of the proposed method. That is, we do not have to consider the allocation of sensors anymore.

Future works are described as follows: We should compare with Evolutionary Learning with optimal number and allocation of sensory inputs. It is good to know how our approach is economic. The proposed method is two-staged algorithm, that is, batch-process is adopted. It would be better to apply on-line version of Manifold Learning for practical application. In this case, during evolution, the meanings of input value could be changed by Manifold Learning. Some isomorphism mechanisms should be devised. We may be able to incorporate the geodesic distance into Evolutionary Learning, instead of the use of Manifold Learning. In this case, we need to take account into the curse of dimensionality.

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] J. B. Tenenbaum, V. de Sliva, and J. C. Lagford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, pp. 2319–2323, 2000.

[2] X. Huo, X. S. Ni, and A. K. Smith, "A survey of manifold-based learning methods," in Recent Advances in Data Mining of Enterprise Data, Edited by T. W. Liao and E. Triantaphyllou, Singapore: World Scientific, 2007.

[3] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, no. 22, pp. 2323–2326, 2000.

[4] A. Miyamae, J. Sakuma, I. Ono, S. Kobayashi, "Instance-based Policy Learning by Real-coded Genetic Algorithms and Its Application to Control of Nonholonomic Systems," *Transactions of the Japanese Society for Artificial Intelligence*, vol. 24, no. 1, pp. 104–115, 2009

[5] N. Hansen and A. Ostermeier. "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pp. 312–317, 1996.

[6] S. Mika, B. Scholkopf, A. Smola, K-R. Muller, M. Scholz, and G Ratsch, "Kernel PCA and De-Noising in Feature Spaces" *Advances in Neural Information Processing Systems 11*, pp. 536–542, 1999.

[7] A. Andoni and P. Indyk, "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions," *Communications of the ACM*, vol. 51, no. 1, pp. 117–122, 2008.

[8] L. Hugues, and N. Bredeche, "Simbad : an Autonomous Robot Simulation Package for Education and Research," *Proceedings of Simulation of Adaptive Behavior (SAB 2006)*, pp. 831–842, 2006.

[9] T. Tateyama, S. Kawata and T. Oguchi, "A teaching method using a self-organizing map for reinforcement learning," *Artificial Life and Robotics*, vol. 7, no. 4, pp. 193–197, 2006.

[10] H. Handa, A. Ninomiya, T. Horiuchi, T. Konishi, M. Baba, "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, 2001.

[11] T. Hiroyasu, M. Miki, M. Sano, H. Shimosaka, S. Tsutsui, and J. Dongarra, "Distributed Probabilistic Model-Building Genetic Algorithm," *Proc. 2003 Genetic and Evolutionary Computation Conference*, pp. 1015–1028, 2003.

[12] M. Najafi and H. Beigy, "Using PCA to improve evolutionary cellular automata algorithms," *Proc. 2009 Genetic and Evolutionary Computation Conference*, pp. 1129–1130, 2009.

[13] S. Obayashi and D. Sasaki, "Visualization and Data Mining of Pareto Solutions Using Self-Organizing Map," *Proc. Second International Conference on Evolutionary Multi-Criterion Optimization*, pp. 796–809, 2003.

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

[15] R. Miikkulainen, and K. Stanley, "Evolving Neural Networks Through Augmenting Topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[16] K. Nishizuka, T. Yairi, and K. Machida, "Simultaneous Localization and Mapping of Mobile Robot Using Nonlinear Manifold Learning," *Proc.36th International Symposium on Robotics*, 2005.