2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Skill Decomposition by Self-Categorizing Stimulus-Response Units

Hsien-I Lin and C. S. George Lee*
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-2035
{sofin, csglee}@purdue.edu

Abstract— **Endowing robots with the ability of skill learning enables them to be versatile and skillful in performing various tasks. This paper proposes a skill-decomposition framework, which differs from previous work in its capability of decomposing a skill by self-categorizing it into significant stimulus-response units (SRU). The proposed skill-decomposition framework can be realized by stages with a 5-layer neuro-fuzzy network with supervised learning, resolution control and reinforcement learning, to enable robots to identify a sufficient number of significant SRUs for accomplishing a given task. Computer simulations and experiments with a Pioneer DX-3 mobile robot were conducted to validate the self-categorization capability of the proposed skill-decomposition framework in learning and identifying significant SRUs from task examples.**

Index Terms— **Stimulus-response unit, skill decomposition, neuro-fuzzy network, resolution control, reinforcement learning.**

I. Introduction

Current humanoid robots have become more human-like in appearance and mechanism, however they are still not as skillful as humans because they lack the ability to categorize and memorize skills and utilize them to learn new skills. Skill learning is referred to as a process of acquiring skills to achieve a given task, and early robotics research on skill learning focused on manufacturing tasks such as assembly [1], cutting [2], and deburring [3], etc. A variety of skill representations and learning algorithms were proposed to acquire skills. Among these skill representations, they are dichotomized into non-primitive-based and primitive-based representations.

For non-primitive-based representations, they are further divided into local-approximation and global-approximation methods. For local-approximation methods, Albus [4] et al. proposed the Cerebellar Model Arithmetic Computer (CMAC) to acquire robot skills. CMAC is a table-look-up method that reproduces the relation between sensor inputs and system-command outputs. Since CMAC had achieved some success in acquiring skills, researchers turned their interests to local-approximation methods for acquiring skills. Radial-basis-function network (RBFN) is another approach that exhibits locality. Baroglio et al. [5] integrated a symbolic interpretation and RBFN to demonstrate that robots exhibited satisfactory performance in a "peg-in-a-hole" task. Although local-approximation methods have shown their success in skill representation, locality impedes the output performance in high-dimensional tasks.

To improve local-approximation methods in skill representation, three prevalent global-approximation methods – multi-layer neural networks, fuzzy logics, and Hidden Markov models – were engaged to acquire robot skills. Neural networks are usually trained by a backpropagation algorithm without specific skill models; for example, Nechyba and Xu [6] proposed a neural-network-based method to extract strategies of skills from an expert and provide them to an apprentice. For fuzzy logic, skill-learning methods are implemented with domain knowledge. Wasik and Safiotti [7] proposed a fuzzy-rule-based control system to learn robot manipulation. They demonstrated that pick-and-place tasks could be realized by a set of behaviors arbitrated by fuzzy rules. Hovland [1] considered that human actions might possess inherent stochastic property and employed Hidden Markov models to acquire human skills.

For primitive-based skill representations, many robot skill-learning systems adopted motion primitives [8], [9] or behavior-based systems [10]–[12] to perform various tasks. For motion primitives, they serve as basic units to perform specific tasks. For example, Speeter [8] defined a set of motion primitives of a Utah/MIT Dextrous Hand such as open, pinch, rotate, swing, etc. to perform manipulation tasks. For behavior-based systems, skills can be acquired by constructing a network of behaviors. Behaviors are usually encoded by reactive rules [10] or mathematical formalism [11]. Although motion-primitive-based or behavior-based approaches achieved some success in skill learning, they required much storage and lacked generality of primitives. In addition, these approaches cannot autonomously adjust the behaviors of primitives in a dynamic environment. They are often employed as a means of implementing a skill-learning system. Unfortunately, they are manually-designed, which is time-consuming by examining the domain knowledge of tasks in a robot system.

Although it is easy to dynamically adjust parameters of non-primitive-based methods, it is unclear and not easy to change the behavior of learned skills by these parameter adjustments. Modularity and reusability of learned skills are also poor for learning new skills by non-primitive-

This work was supported in part by the National Science Foundation under Grant IIS-0427260. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

*This material is based upon work supported by (while serving at) the National Science Foundation.

based methods. On the other hand, primitive-based methods have good modularity and reusability in learning new skills, but their primitives are usually manually-designed and may not be adjustable to various situations of a task. To provide a platform for integrating non-primitive-based and primitive-based methods, this paper proposes a self-categorizing, skill-decomposition framework, which acquires and self-categorizes a skill into a sufficient number of stimulus-response units for accomplishing a given task. A stimulus-response unit (SRU) is defined as a mapping from the domain of perceptual stimuli to the domain of action responses. The proposed skill-decomposition framework can be realized by stages with a 5-layer neuro-fuzzy network with supervised learning, resolution control, and reinforcement learning, and is discussed in more detail in the next section.

II. Proposed Skill-Decomposition Framework

Figure 1 shows the proposed skill-decomposition framework, which can be conveniently realized in three stages. In the first stage, a five-layer, neuro-fuzzy network acquires a skill and categorizes it into initial SRUs that capture the behavioral patterns of the skill as demonstrated by a human. Supervised learning is performed to tune the parameters of the neuro-fuzzy network. Since some of these initial SRUs may be unnecessary due to extraneous actions demonstrated by a human, resolution control is proposed to prune unnecessary SRUs and generate a sufficient number of significant SRUs in the second stage. In the third stage, reinforcement learning is utilized to validate the sufficiency of the significant SRUs for accomplishing the given task. If not, their behaviors are autonomously adjusted in the reinforcement-learning stage until the task is accomplished within an acceptable number of trials.

A. Neuro-fuzzy-based Skill Representation with Supervised Learning

Since a skill is considered as a stimulus-response mapping from the domain of stimuli to the domain of responses, a skill can be represented by a set of SRUs. Thus we propose to represent a skill as a fuzzy-rule-based system and realize it by a five-layer, neuro-fuzzy network (see Fig. 2) [13]. Each SRU is regarded as a fuzzy rule in the neuro-fuzzy network. After a neuro-fuzzy network is trained by task examples via supervised learning, the structure and parameters of the neuro-fuzzy network are learned, and the SRUs represented by fuzzy rules are self-categorized from the neuro-fuzzy network. There are merits of representing

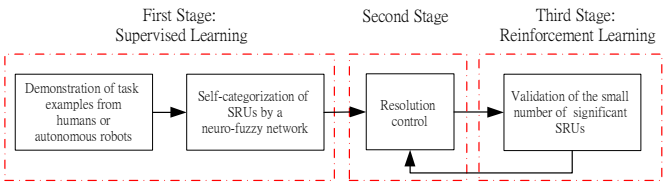

Fig. 1. The proposed skill-decomposition framework.

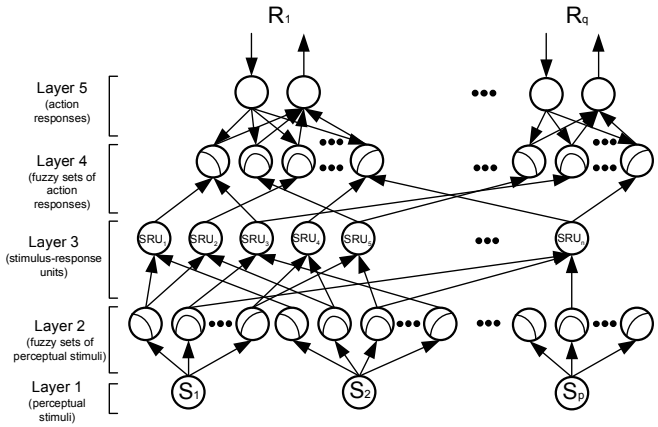

Fig. 2. Skill representation is realized by a neuro-fuzzy network (S: perceptual stimulus; R: action response; SRU: stimulus-response unit).

a skill by a neuro-fuzzy network. First, a neuro-fuzzy-based skill representation provides low-level, connectionist-learning capability and high-level, fuzzy IF-THEN rule thinking. Thus, skills as represented by SRUs can be easily learned and adjusted by using a neuro-fuzzy-based supervised learning from task examples. Second, the connectionist structure of a neuro-fuzzy-based skill representation provides a mechanism for different types of learning (e.g., supervised and reinforcement learning) because connections in a neuro-fuzzy network can propagate and memorize signals. Thus, the benefits of supervised and reinforcement learning can be appropriately combined and realized in a neuro-fuzzy-based skill representation. Third, after a skill has been learned, its fuzzy rules can be extracted and expressed explicitly from SRUs of a neuro-fuzzy network.

B. Resolution Control

After initial SRUs have been categorized by supervised learning, resolution control is employed to prune unnecessary SRUs, resulting in a small number of significant SRUs. The idea of resolution control is illustrated in Figs. 3(a), (b), (c), and (d). Figures 3(a) and (b) show four SRUs A, B, C, and D, and each of them has two perceptual stimuli, S_1 and S_2, and two corresponding action responses, R_1 and R_2. In Fig. 3(a), SRUs A and B are close by measuring the distance in the perceptual space, and so are SRUs C and D in Fig. 3(b). By resolution control, SRUs A and B, and C and D are combined into SRUs A' and C' by merging their respective fuzzy sets in terms of statistically averaging their membership functions. From Fig. 3(c), A' (in dash line) covers a larger domain in the perceptual space, but A' loses some detailed perceptual information that has been described by A and B; so does C'. Thus, the resolution of perceptual stimuli of A' is lower than the original resolution provided by A and B, and so is the resolution of SRU C'. Meanwhile, extraneous actions generated from unnecessary SRUs are averaged out with other significant SRUs by the process of resolution control.

In our proposed framework, we construct a resolution binary tree (RBT) in order to generate a small number of significant SRUs. The concept of RBT is to establish

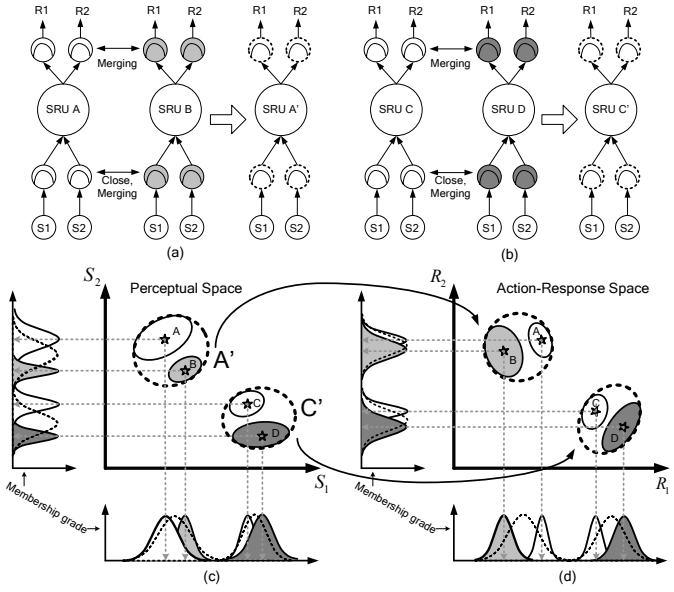

Fig. 3. Concept of resolution reduction of SRUs. (a) and (b) Six SRUs A, B, C, D, A', and C', each of which has two perceptual stimuli, S_1 and S_2 and two action responses, R_1 and R_2. (c) Merging membership functions of perceptual stimuli of SRUs A and B, and C and D. (d) Merging membership functions of action responses of SRUs A and B, and C and D.

a lookup table that describes the means and variances of the membership functions of the fuzzy sets belonging to perceptual stimuli and action responses for each SRU in the small number of significant SRUs. The advantage of this RBT is that the number of SRUs quickly decreases by half. The procedure of building a RBT is described by the RBT algorithm described below.

RBT Algorithm: Given the number of initial SRUs from a learned neuro-fuzzy network, the RBT algorithm clusters them into two clusters with an equal number of SRUs by their locations in the perceptual-stimuli space. If the number of initial SRUs is odd, then one cluster has one more SRU than the other. Then, the algorithm recursively classifies each cluster into two sub-clusters until the number of SRUs in a sub-cluster is one. The clustering algorithm is implemented by a K-means method by measuring the Euclidean distance (distance between the means of the fuzzy sets belonging to the same perceptual stimulus of two SRUs), and it ends up with generating a binary tree where each leaf node represents a SRU. To generate a new SRU for each node (except for leaf nodes) in the binary tree, the algorithm merges the SRUs from its child nodes. The merging procedure is repeated from the bottom to the top layer of the binary tree.

T1. [*Determination of two initial centers.*] Randomly choose SRU_i and SRU_j from SRU_1 to SRU_n as $cluster(1)_{center}$ and $cluster(2)_{center}$, respectively. Also, $|cluster(1)| \leftarrow 0$ and $|cluster(2)| \leftarrow 0$, where $|\cdot|$ is the count of number of SRUs in the cluster.

T2. [*Cluster indication for SRUs.*] $c_{SRU_k} \leftarrow argmin_{w=\{1,2\}} \|SRU_k - cluster(w)_{center}\|$ and $|cluster(c_{SRU_k})| \leftarrow |cluster(c_{SRU_k})| + 1$ for $1 \leq k \leq n$, where c_{SRU_k} is the cluster of SRU_k and $\|\cdot\|$ is the Euclidean distance from SRU_i to SRU_j and is defined as $\sqrt{\sum_{\forall stimulus}(Mean^{SRU_i}_{Stimulus} - Mean^{SRU_j}_{Stimulus})^2}$.

T3. [*Center update.*] Calculate $cluster(1)_{center}$ and $cluster(2)_{center}$ by averaging the means of stimuli of the SRUs in $cluster(1)$ and $cluster(2)$.

T4. [*Error calculation of cluster.*] $E \leftarrow \sum_{w=\{1,2\}} \sum \|SRU_k - cluster(w)_{center}\|$ $\forall SRU_k \in cluster(w)$.

T5. [*Check error convergence.*] IF $|E_{current} - E_{previous}| \leq \tau$, where τ is a design threshold, THEN continues, ELSE go to Step T2.

T6. [*Size difference of cluster.*] $\Delta N \leftarrow |cluster(1)| - |cluster(2)|$.

T7. [*Cluster re-indication for SRUs.*] IF $\Delta N \geq 0$, THEN assign ΔN number of SRU from $cluster(1)$ to $cluster(2)$, which is closest (the shortest Euclidean distance) to any $SRU \in cluster(2)$, ELSE assign ΔN number of SRU from $cluster(2)$ to $cluster(1)$, which is closest to any $SRU \in cluster(1)$.

T8. [*Repeat clustering.*] Designate $cluster(1)$ and $cluster(2)$ as parent nodes in the binary tree. Repeat Steps T1 to T7 for each parent node, resulting in two child nodes until leave nodes are created where $|cluster(w)| = 1$ or 0 for $w = 1, 2$.

T9. [*Initialization of generating a small number of significant SRUs.*] Start at the bottom layer of the tree.

T10. [*Merging of child nodes.*] IF either of the two child nodes belonging to the same parent node is dummy ($|cluster(w)| = 0$), THEN assign the SRU in the non-dummy node as the new SRU in the parent node ELSE merge the two SRUs from the child nodes into the new SRU in the parent node.

T11. [*Repeat of merging.*] Repeat Step T10 until the top layer of the tree is reached.

END RBT Algorithm.

C. Reinforcement Learning

After the supervised learning, the structure and parameters of the neuro-fuzzy network (see Fig. 2) have been learned, and the resolution control generates a small number of significant SRUs. In the third stage, reinforcement learning is introduced to test and validate whether these significant SRUs are sufficient for accomplishing the task. Thus, the reinforcement-learning stage has two main functions: providing a process for validating the suitability of small number of significant SRUs, and providing a learning mechanism for adjusting the behavioral patterns of these SRUs to accomplish the task. In the reinforcement-learning stage, a skill is represented by significant SRUs, and an adaptive-heuristic-critic algorithm is adopted to adjust the behaviors of SRUs to achieve the goal of given task. Figure 4 shows a neuro-fuzzy network with reinforcement learning. The neuro-fuzzy network organizes SRUs by using two major subnets, critic and action subnets.

The purpose of the critic and action subnets is to obtain appropriate behaviors of SRUs for accomplishing a given task. SRUs in the action subnet provide well-structured,

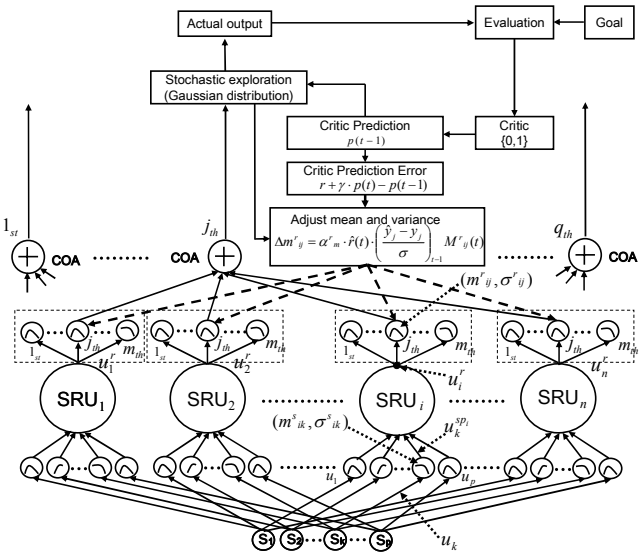

Fig. 4. A neuro-fuzzy network in the reinforcement-learning stage.

multi-step actions rather than single-step actions in a conventional reinforcement approach since a fuzzy rule can be used to generate actions for many states in the task space. On the other hand, the critic subnet provides the parameter learning for tuning behaviors of SRUs because a critic is an indication showing the goal achievement for the task, and it strongly drives and shapes the learning process and the results of reinforcement learning. With well-structured actions and parameter learning, the skill-learning time in the reinforcement-learning stage is dramatically reduced. For the critic subnet, it consists of critic, critic prediction, and critic-prediction error. The critic is a binary signal providing success or failure for the outcome of the performance. The critic prediction, $p(t)$, is to predict the critic signal in order to choose a better action from the action subnet at time step $(t-1)$. In other words, the predicted critic can help the action subnet to perform a more efficient random search since reinforcement learning is an exploitation-and-exploration process. The critic-prediction error is used to calculate the error between the predicted and actual critics, and update the parameters of critic prediction in the critic subnet and SRUs in the action subnet.

For the action subnet, it consists of SRUs, stochastic exploration, and mean-and-variance adjustments of fuzzy sets. SRUs function as initial action templates and are identified from task examples in the supervised-learning stage. The outputs of SRUs are wired together to produce the network's outputs. Thus, the network's outputs are generated based on exploiting SRUs. In addition, stochastic exploration is added at the network's output to assist the reinforcement-learning stage in exploring a near optimal solution. Meanwhile, a critic-prediction error is fed back to adjust the behaviors of SRUs to generate a more correct predicted critic through changing the means and variances of the membership functions of fuzzy sets. These changes in membership functions are performed by backpropagation of the critic signal until the task is achieved. In our mobile-robot experimental examples, we set a hundred trials as the upper limit of the number of trials to judge whether a robot achieves the given task or not. This upper limit can be adjusted – depending on how quickly we expect a robot to accomplish a task. Although a small upper limit can be used to quickly judge whether a robot succeeds or fails a task, it may cause the robot to easily fail the task. Thus, an adequate upper limit of the number of trials is necessary. The updating rules and detailed calculations of the adaptive-heuristic-critic algorithm are referred to [14], [15].

III. Computer Simulations and Experimental Work

Computer simulations using the player/stage mobile robot control software and two experiments on an ActivMedia Pioneer P3-DX mobile robot were conducted to validate the performance of the proposed skill-decomposition framework.

A. Experiment 1: Hallway-Passing Skill

In this example, we implemented a simple rule for the robot to pass a hallway – when the robot approaches a wall, it will turn toward the other side to avoid bumping into the wall. If a neuro-fuzzy network with supervised learning is utilized to learn the action patterns of the autonomous robot, it will result in a zigzag fashion of passing the hallway because the neuro-fuzzy network will learn the trajectory faithfully. However, passing a hallway in a zigzag fashion is not the skill that we would like the robot to learn. We would like the proposed three-stage skill decomposition to learn the hallway-passing skill utilizing SRUs and discover a sufficient number of SRUs to pass the hallway without moving in a zigzag fashion.

The first stage, a 5-layer, neuro-fuzzy network with supervised learning, will capture the action patterns from task examples and categorize them into initial SRUs. Figure 5(a) shows a task example of passing the hallway with the stated simple rule. The training data were collected by a SICK LMS-200 laser ranger equipped on the Pioneer P3-DX mobile robot. The hallway in our simulation is a hallway of the ground floor of EE building at Purdue University. The training data for the neuro-fuzzy network comprise of four sensory inputs: minimum distance from the right-hand wall, minimum distance from the left-hand wall, the centermost distance from obstacles, and its current orientation, and two actuator outputs: turning angle and speed. The sampling rate was 10Hz. We set the number of membership functions to 5 for each perceptual stimulus and 10 for each action response to generate 124 initial SRUs. Figures 5(b) and (c) show the training results from the neuro-fuzzy network.

With the initial 124 SRUs, resolution control constructed a resolution binary tree. At the bottom layer of the tree, there are 128 nodes (SRUs) because $2^6 < 124 < 2^7$ and 4 nodes are "dummy nodes." After resolution control, Figures 6(a)-(e) show the simulation results of different resolutions of SRUs when the angle of initial robot orientation was 30 degrees. In Fig. 6(a), when the resolution is 128 (highest),

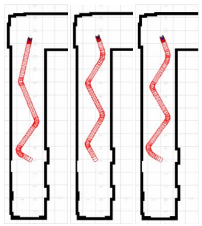

Fig. 5. Skill of passing a hallway. (a) Task example. (b) and (c) Training results from the neuro-fuzzy network with an initial robot orientation of 30 degrees and 60 degrees, respectively, with respect to the upright.

the result was similar to Fig. 5(b). Other resolutions of SRUs are shown in Figs. 6(b), (c), (d), and (e). When the number of SRUs was reduced to less than 128, the mobile robot could pass the hallway with fewer zigzag actions. It came out with a more straight-line hallway passing skill to achieve the goal. However, Fig. 6(e) shows that when the number of SRUs was 8, the robot could not pass the hallway in the first trial. Also, Figs. 6(g)-(j) show that when the number of SRUs was less than 128 and the initial angle of robot orientation was 60 degrees, the robot still could not pass the hallway in the first trial. With these failures, the reinforcement learning was used to adjust the behaviors from these small numbers of SRUs.

The adjustment process tuned the behaviors of the hallway-passing skill based on SRUs. Figure 4 shows how the critic r adjusts the means and variances of the fuzzy sets belonging to perceptual stimuli and action responses. The critic is given 0 at each time step when the robot does not fail the task, or -1 when it fails the task. Figures 7(a)-(b) and (e)-(f) show the reinforcement learning that took 2 trials to complete the task when the number of SRUs was 8, and the initial angle of robot orientation was 30 degrees. When the initial angle of robot orientation was 60 degrees, Figs. 7(c)-(d) and (g)-(h) show the reinforcement learning that took 3 trials to complete the task. In one hundred of continuous tests with $\alpha_m^r = 0.3$, $\alpha_\sigma^r = 0.3$, $\alpha_m^s = 0.3$, $\alpha_\sigma^s = 0.3$, $\omega = 0.1$, $\gamma = 0.95$, $\beta = 0.1$, and $\lambda = 0.8$, it took 8.21 and 12.12 trials on the average for 8 SRUs in a test to finish the task when the initial angle of robot orientation was 30 and 60 degrees, respectively. Other testing results for -30, -45, -60 and 45 degrees of robot orientation are shown in Fig. 8.

In this example, we set the upper limit of 100 trials in

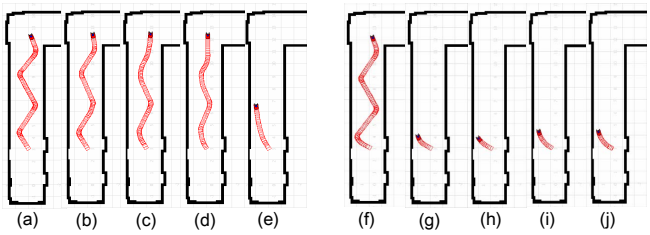

Fig. 6. Skill of passing a hallway with different numbers of SRUs when the initial angle of robot orientation is 30 degrees: (a) 128 SRUs; (b) 64; (c) 32; (d) 16; (e) 8. Similarly, when the initial angle of robot orientation is 60 degrees: (f) 128 SRUs; (g) 64; (h) 32; (i) 16; (j) 8.

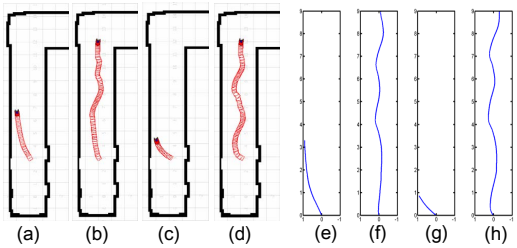

Fig. 7. The adjustment of the behaviors of 8 SRUs. When the initial angle of robot orientation is 30 degrees, simulation results: (a) Trial 1 before the reinforcement learning, (b) Trial 2 after the reinforcement learning; experimental results: (e) Trial 1, (f) Trail 2. Similarly, when the initial angle of robot orientation is 60 degrees, simulation results: (c) Trial 1, (d) Trial 3; experimental results: (g) Trial 1, (h) Trial 3.

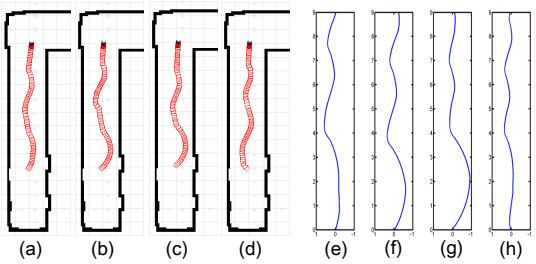

Fig. 8. (a)-(d) and (e)-(h) are simulation and experimental results, respectively, with 8 SRUs when the initial angle of robot orientation is: -30 degrees ((a)&(e)), -45 degrees ((b)&(f)), -60 degrees ((c)&(g)), and 45 degrees ((d)&(h)).

the reinforcement learning for distinguishing whether the resolution of SRUs is successful or not in completing the task. If the number of trials is under 100 and the task is completed, then the resolution of SRUs succeeds the task, otherwise, it fails the task. As for using 4 or 2 SRUs, both could not pass the hallway within the limit of a hundred of trials. Thus, 4 or 2 SRUs are not sufficient for accomplishing the task. From the results of this example, having 8 SRUs is sufficient and appropriate for the robot to learn the skill of passing the hallway. Figure 9 shows the mobile robot was passing a hallway in EE building.

Fig. 9. A P3-DX mobile robot is passing a hallway in EE building at Purdue University.

Figure 10 shows the number of SRUs activated at a specific position with a 10-degree increment between -90 and 90 degrees, where the x-axis represents the normalized position (from -1 to 1: from right to left) of the robot in the hallway. In Fig. 10, when the number of SRUs is 128, there are more SRUs activated near the walls than the middle of the hallway. Although the high resolution of SRUs occurs near the walls, it induces unnecessary actions of the hallway-passing skill in a zigzag fashion. After resolution control,

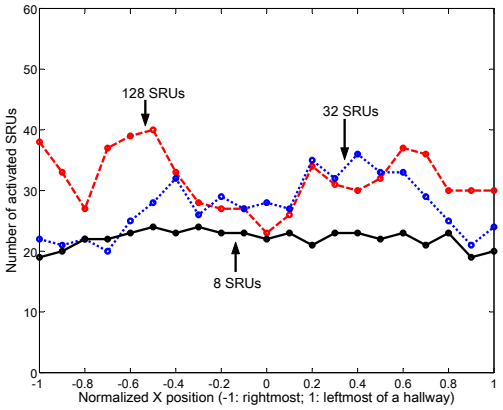

Fig. 10. Number of activated SRUs by summing up forward orientations (dashed line: 128; dotted line: 32; solid line: 8 SRUs).

128 SRUs are reduced to 32 and 8 SRUs. Figure 10 also shows the number of activated SRUs with 32 and 8 SRUs. Apparently, the activated number of SRUs near the wall drops dramatically when 128 SRUs are reduced to 32 or 8 SRUs. This result indicates that resolution control prunes unnecessary SRUs that induce extraneous actions moving near the walls. When the number of SRUs is pruned to 8, Fig. 10 shows its uniform-like distribution of activated number of SRUs for accomplishing the task.

B. Experiment 2: Skills of Traveling around an Ellipse and a Circle

In the second experiment, we asked the robot to learn two different skills, traveling around an ellipse and a circle. We manually commanded the robot to perform these two skills, and then the robot was asked to self-categorize sufficient numbers of SRUs for traveling both of them. The training data shown in Figs. 11(a) and (b) were obtained by recording the sensor inputs and actuator outputs of the robot when the robot was manually controlled. After the neuro-fuzzy-based supervised learning, the initial numbers of SRUs were 192 and 137 for traveling around the ellipse and the circle, respectively. Then, resolution control and reinforcement learning identified 16 and 8 as the numbers of significant SRUs for traveling around the ellipse and the circle, respectively. These two skills were demonstrated in Figs. 11(c) and (d) using their respective numbers of significant SRUs. In addition, these two numbers of significant SRUs were tested for different sizes of ellipse and circle. Figures 11 (e)-(f) and (g)-(h) show their simulation and experimental results of traveling around a smaller ellipse and circle, respectively.

IV. Conclusions

In this paper, we have presented a self-categorizing, skill-decomposition framework, where a 5-layer neuro-fuzzy network with supervised learning, resolution control and reinforcement learning were used in stages to acquire and decompose a skill into a sufficient number of SRUs for accomplishing a given task. The salient feature of the proposed skill-decomposition framework is its capability of learning a skill by self-categorizing it into significant SRUs without human intervention. The synthesis of learned skills into new skills is currently being studied and will be reported in a future paper. Computer simulations and experiments on a Pioneer P3-DX mobile robot have validated the self-categorization capability of the proposed skill-decomposition framework.

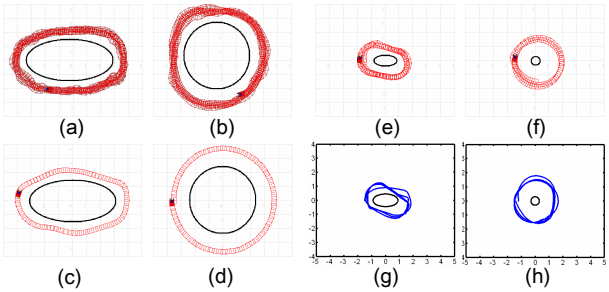

Fig. 11. (a) and (b): Training data of traveling around an ellipse and a circle, respectively. (c) Traveling around an ellipse by utilizing 16 sufficient SRUs. (d) Traveling around a circle by utilizing 8 sufficient SRUs. (e)-(f) and (g)-(h): Simulation and experimental results of traveling around an ellipse and a circle, respectively. (e) and (g): Another small ellipse by utilizing 16 sufficient SRUs. (f) and (h): Another small circle by utilizing 8 sufficient SRUs.

References

[1] G. Hovland, P. Sikka, and B. McCarragher, "Skill acquisition from human demonstration using a hidden Markov model," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Apr 1996, pp. 2706–2711.

[2] T. Shibata, T. Abe, K. Tanie, and M. Nose, "Motion planning of a redundant manipulator based on criteria of skilled operators," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, vol. 4, Oct 1995, pp. 3730–3735.

[3] H. Asada and S. Liu, "Transfer of human skills to neural net robot controllers," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Apr 1991, pp. 2442–2448.

[4] J. Albus, "A new approach to manipulator control: The cerebellar model articulation controller (CMAC)," *Trans. of ASME J. Dynamic Syst. Meas., and Contr.*, vol. 63, no. 3, pp. 220–227, Sep 1975.

[5] C. Baroglio, G. Attilio, M. Kaiser, M. Nuttin, and R. Piola, "Learning controllers for industrial robots," *Mach. Learning*, vol. 23, pp. 221–249, 1996.

[6] M. Nechyba and Y. Xu, "Human skill transfer: neural networks as learners and teachers," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 3, Aug 1995, pp. 314–319.

[7] Z. Wasik and A. Safiotti, "A fuzzy behavior-based control system for manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 2, Sep 2002, pp. 1596–1601.

[8] T. Speeter, "Primitive based control of the Utah/MIT dextrous hand," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, Apr 1991, pp. 866–877.

[9] U. Thomas, B. Finkemeyer, T. Kroger, and F. Wahl, "Error-tolerant execution of complex robot tasks based on skill primitives," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Sep 2003, pp. 3069–3075.

[10] R. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robot. Autom.*, vol. 2, no. 1, pp. 14–23, Mar 1986.

[11] R. Arkin, "Motor schema based navigation for a mobile robot: An approach to programming by behavior," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 4, Mar 1987, pp. 264–271.

[12] M. Matarić, "Behavior-based control: Main properties and implications," in *Proc. IEEE Int. Conf. Robot. Autom., Workshop on Architectures for Intelligent Control Systems*, May 1992, pp. 46–54.

[13] C. T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, no. 12, pp. 1320–1336, Dec 1991.

[14] A. Barto, R. Sutton, and C. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vol. 13, no. 5, pp. 834–846, Oct 1983.

[15] C. T. Lin and C. S. G. Lee, *Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems*. Upper Saddle River, NJ: Prentice-Hall, 1996.