Neural Comput & Applic (2007) 16: 155–166
DOI 10.1007/s00521-006-0056-8

ORIGINAL ARTICLE

Chandimal Jayawardena · Keigo Watanabe
Kiyotaka Izumi

Controlling a robot manipulator with fuzzy voice commands using a probabilistic neural network

Received: 8 August 2005 / Accepted: 19 April 2006 / Published online: 6 July 2006

Abstract Natural language commands are generated by intelligent human beings. As a result, they contain a lot of information. Therefore, if it is possible to learn from such commands and reuse that knowledge, it will be a very efficient process. In this paper, learning from such information rich voice commands for controlling a robot is studied. First, new concepts of fuzzy coach-player system and sub-coach are proposed for controlling robots with natural language commands. Then, the characteristics of the subjective human decision making process are discussed and a Probabilistic Neural Network (PNN) based learning method is proposed to learn from such commands and to reuse the acquired knowledge. Finally, the proposed concept is demonstrated and confirmed with experiments conducted using a PA-10 redundant manipulator.

Keywords Coach-player system · Sub-coach · Natural language commands · Subjective decisions · PNN

1 Introduction

Robots found their first real-world application on the factory floor. Still, heavy industry is the environment in which robotics plays its most important role. However, working robots are gradually spreading, gradually improving, and gradually moving into new areas. If the dreams of researches come true, in future, robots will assist the elderly and disabled people into and out of wheelchairs and beds, be conversant in several languages, watch over babies, and provide a sympathetic ear to the lonely [1, 2].

C. Jayawardena (✉) · K. Watanabe · K. Izumi
Department of Advanced Systems Control Engineering,
Graduate School of Science and Engineering,
Saga University, Saga, Japan
E-mail: chandimal@ieee.org
Tel.: +81-952-288602
Fax: +81-952-288587

Although, initially, the importance of robots was found mainly in heavy industries, isolated from people, now a new important dimension has been added: that is, the human–robot interaction. The area of human–robot interaction has been developed into such an extent, even socially interactive robots have received the attention of researchers. Socially interactive robots are capable of showing human like behavior when dealing with another human, i.e., communicating as peers using natural languages, gestures, etc. [3].

As the human–robot relationship is becoming more important, the studies on human–robot communication have been given a high priority. The importance of voice communication with robots can be found in the areas like nursing and aiding elderly people, helping disabled people, helping people in complex tasks such as surgery and implementing space restricted systems where the usage of other input-output devices is not feasible.

Among recent related work, we can identify two lines of research both of which are equally important in achieving true human-like behavior. This is a result of having two viewpoints for the same problem.

One line of research concentrates on embedding robots with more human-like cognitive capabilities. For example, Oates et al. [4] presented an unsupervised learning method that allowed a robotic agent to identify and represent qualitatively different outcomes of actions. They used human experience to evaluate the method. Roy [5] presented a computational model that was able to learn words from multisensory data. In more recent interesting work presented in Roy et al. [6], they proposed a set of representations and procedures that enabled a robotic manipulator to maintain a "mental model" of its physical environment by coupling active vision to physical simulation with the view of creating an interactive robot, which was able to engage in cooperative task with human. Ballard and Chen [7] and Chen and Ballard [8] presented a multimodal interface that was able to learn words from human users in an unsupervised manner in which the users performed everyday tasks while providing natural language descriptions of their tasks.

This line of research is very important; however, due to extremely demanding technical and theoretical requirements of such systems, still they have a long way to go in order to be applied in practical domains.

The other line of research concentrates on controlling ordinary robots by human-friendly means. By *ordinary* we mean the robots that are controlled by conventional methods and are already being utilized for real work. For example, Lin and Kan [9] proposed an adaptive fuzzy command acquisition method for controlling machines using natural language commands such as "*move forward at a very high speed*." In Pulasinghe et al. [10], similar commands were used to control a mobile robot handling out-of-vocabulary words. Chatterjee et al. [11] and Pulasinghe et al. [12] discussed how robot manipulators could be controlled with fuzzy voice commands to perform assembling tasks.

The advantage of this line of research is that it enables us to develop intelligent human interfaces for existing robotic systems.

The work presented in this paper is related to the second line of research discussed above in the sense that it concentrates on controlling ordinary robotic systems by human friendly means rather than developing a robot with human like cognitive capabilities.

In natural language communication, encountering words and phrases with fuzzy implications is inevitable. Therefore, any system that accepts true natural language commands should be able to understand their fuzzy meanings. On the other hand, natural language commands are inherently information rich because they are generated by experienced and intelligent human beings. They are very useful in machine control because they can fine-tune the performance of a machine. For example, a command like "*move slowly*" may contain much information regarding the nature of terrain, the distance to obstacles, etc. Therefore, learning from such commands and re-using that knowledge effectively may be quite useful.

This paper proposes a method of learning from information rich fuzzy voice commands for controlling a robot.

First of all, two new concepts, fuzzy coach-player system and sub-coach, are introduced. The analogy between the fuzzy coach-player system concept and the real-world relationship between a coach and a player is discussed. The sub-coach concept is proposed to eliminate the limitations of the fuzzy coach-player system. The possibility of learning from natural language commands is then discussed. Inherent subjective nature of natural language commands is discussed and a mathematical model which enables learning from such commands is proposed. For the interpretation of fuzzy and non-fuzzy components of natural language commands, a method of interpretation of fuzzy voice commands is explained by using simple fuzzy reasoning. Moreover, implementation details of the sub-coach are discussed, in which a modified version of the conventional Probabilistic Neural Network (PNN) architecture is used in the implementation of learning capability of the sub-coach. Finally, the effectiveness of the present approach is demonstrated using some experimental results obtained from a system implemented for controlling a redundant manipulator.

The organization of this paper is as follows: Fuzzy coach-player systems are introduced in Sect. 2. Learning from fuzzy voice command is described in Sect. 3. Section 4 discusses the interpretation of natural language commands. Implementation of the sub-coach is discussed in Sect. 5. Section 6 presents the experimental results and Sect. 7 summarizes what is presented in this paper and presents conclusions.

2 What are fuzzy coach-player systems?

2.1 Coach player system

Consider the process of controlling a robot to achieve a complex task using voice commands. In order to do this cooperative task successfully, some information needs to be exchanged between the robot and the human. Flow of information from human to robot is via verbal commands. Flow of the same from robot to human is via visual observation of robot by human.

The human user may make the robot complete a certain task in several steps by issuing a series of commands generated by observing the robot's behavior at each step. This is similar to the relationship between a coach and a player in a certain sport or a game. Therefore, this type of a system can be called a coach-player system; in particular, it is called a fuzzy coach-player system when fuzzy commands are used. This concept is illustrated in Fig. 1.

There are three important features in a coach-player system.

1. Command interpretation and execution by the player: Player interprets the fuzzy user command according to the current context and performs some actions.

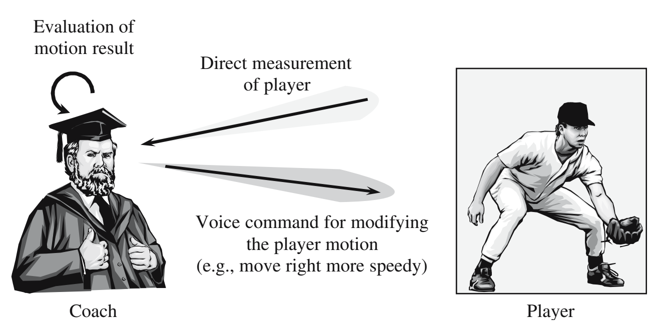

Fig. 1 Concept of fuzzy coach-player system

The result will be a change in its state or performance.
2. Evaluation of the player by the coach: Coach observes the change in state or performance of the player and evaluates it subjectively. This subjective evaluation depends on the coach's knowledge, experience, attitude, etc. Depending on the evaluation, the coach decides whether to issue another improvement command or not.
3. Improvement of the player: As a result of obeying to coach's commands, the player will continuously improve its performance toward coach's intended direction. This will continue until the coach is satisfied.

For example, consider a system where a human is guiding a mobile robot to move from one point to another. A typical series of commands would be,

- "move forward slowly"
- "move little more forward"
- "turn to right"
- "move far"
- "stop," etc.

The user may continue to command like this until the robot reaches the destination.

Although this kind of a coach-player system is useful, there is a disadvantage also: i.e., the need to issue the same command over and over. By eliminating this, it is possible to improve the proposed system further. A concept of a sub-coach is introduced as the solution.

2.2 Sub-coach

Sub-coach is a software agent which stands in between the user (coach) and the robot (player). It can learn from fuzzy voice commands issued by the user and can use that knowledge to control the robot without getting help from the user. However, the sub-coach can consult the coach in situations where it does not have sufficient knowledge to handle a particular situation.

The concept of sub-coach for learning was first proposed in [13]. There the concept was demonstrated with a crisp decision making sub-coach. The same concept was further amplified by incorporating both crisp and fuzzy decisions in [14] and [15].

In a simple coach-player system, the user directly issues commands to the robot. Once a sub-coach has been introduced to this type of a system, initially, it will be just an observer. That is, it observes commands issued by the user and the actions performed by the robot in response to those commands. Thus, gradually the sub-coach can build a knowledge which is sufficient to issue commands to the robot, to perform activities which are similar to what the robot did during the learning period, consulting the human user only when required. The sub-coach concept is illustrated in Fig. 2. Controller interprets fuzzy voice commands and issues the control commands to the robot in the required format. The sub-coach learns from both user commands and the status of the robot.

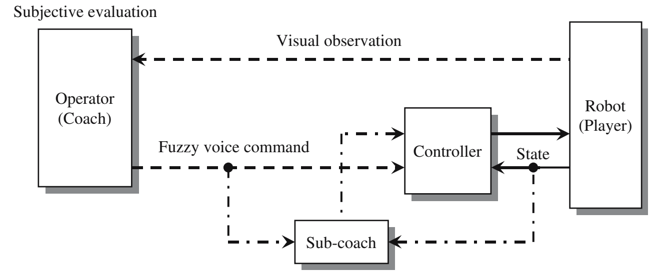

Fig. 2 Introduction of a sub-coach into a coach-player system

3 Learning from fuzzy voice commands

When controlling a robot with voice commands, the command of the user depends on the state of the robot world. Here, the robot world includes the robot itself, the working environment and the final objective to be achieved. The user evaluates the world state subjectively using his knowledge and experience, and issues the next command which he thinks the most appropriate. For example, when controlling a mobile robot to navigate through obstacles, if the user thinks that the robot might clash with an obstacle ahead if it continues to travel at the current velocity, he might say "*robot, slow down*." In response, the mobile robot will reduce it's speed. Consequently, the world state will change; thus avoiding collision.

Therefore, the process of controlling a robot using a series of voice commands can be seen as changing the robot world state repetitively until the required target is achieved.

3.1 Robot world state

Robot world state is defined using two kinds of parameters. One is, the kind of parameters which defines the state of the robot itself: for example, velocity, position, etc. The other is, the kind of parameters which indicates the closeness to the final objective: for example, distance to the target point to be reached, the depth of a hole drilled, etc.

Assume, during the learning phase, that the human user controls the robot to complete a set of jobs of similar nature. Each job is completed in a number of steps. In each step, the user has to issue a voice command depending on the current robot world state.

Let S be the complete set of all possible world states and S_i be a general element of the set. Then it follows that,

$$S_i = \{x_1, x_2, \ldots, x_p, y_1, y_2, \ldots, y_r\} \quad (1)$$

Here, x_1,x_2,...,x_p are the parameters that define the state of the robot itself, where p is the number of such parameters. y_1,y_2,...,y_r are the parameters which indicate the closeness to the final objective and r is the number of such parameters. $(p+r)$ is the total number of parameters required to define a world state. Thus, a world state is a $(p+r)$ dimensional entity and it is a member of a $(p+r)$ dimensional state-space. All these parameters are scalar quantities. Whenever a vector is involved, its components are used as different parameters. We will come to more concrete definitions once we come to the implementation.

3.2 Learning by the sub-coach

Let C be the complete set of all valid commands and C_j be a general element of the set. Assume that the command issued by the user in response to the world state S_i is C_j. Then, we have

$$f : S \rightarrow C \quad (2)$$

Here, f is a subjective function which depends on the knowledge, experience, attitude, etc. of the user. For example, C_i can be something like "*go very little*."

Since the robot world state may depend on the values of various continuous parameters, S would be continuous. Thus, S may contain infinite number of points. On the contrary, due to the limitations of any feasible system, the number of valid commands is limited. Therefore, we can assume that C is discrete and finite. Thus, f is a serjective function as shown in Fig. 3 [16].

The objective of learning by the sub-coach during training is to learn the subjective function f so that in a later case, it can find the correct command corresponding to a world state not encountered during the training. However, since C contains only a finite number of elements, this problem is reduced to a pattern classification problem where the number of classes is equal to the number of valid commands. Thus, if the sub-coach can classify an incoming pattern correctly, it can make correct decisions.

Theoretically, any element in S can be mapped to any element in C. Since the elements of S are comprised of an infinite number of continuous values, if the abrupt decision changes between very close states tend to be frequent, the above classification will fail. However, the argument for decision making based on classification is further supported by the inherent subjective nature of human decisions.

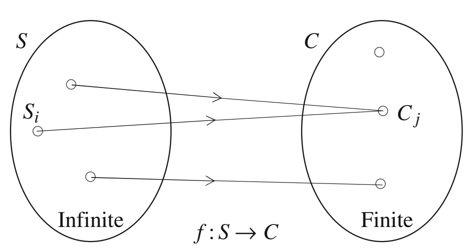

Fig. 3 Relationship between robot world states and commands

3.3 Subjective decision making

The most significant feature in the above explained decision making process is originated from the fact that the human user commands are fuzzy in their very nature. That means, although we define a state of the robot by various measurable parameters, the human user understands them when he makes a decision only using his own senses. Therefore, the decisions made by the user are not objective; rather they are subjective decisions.

Let us consider an example illustrated in Fig. 4. Assume that the user wants to guide a robot manipulator to move its tip from *Source* to *Target*. Also, assume that the user can move the manipulator tip either up/down (y axis) or left/right (x axis) using verbal commands. As the first step, the user might say "move right" or "move down". In this case, the exact coordinate positions are *Source* = (450.80, −132.55) and *Target* = (597.94, −291.93). But for his decision, the user does not use these very accurate information. Instead, he might think "what is the best way to move from the source area to the target area." Thus, the user may take the same decision to move from another point in the source area to another point in the target area.

This concept is not valid only for moving between points. In this example, the subjective decision of the user depends only on the source position and the target position. Therefore, a state can be defined using x,y coordinates of *Source* and *Target*; i.e., they are s_x, s_y, t_x and t_y. Then, according to Eq. (1), a state can be defined as $\{s_x, s_y, t_x, t_y\}$, whose dimension is 4. Thus, in this case, a state is a member of a four-dimensional state-space. According to the above explanation, for two sufficiently

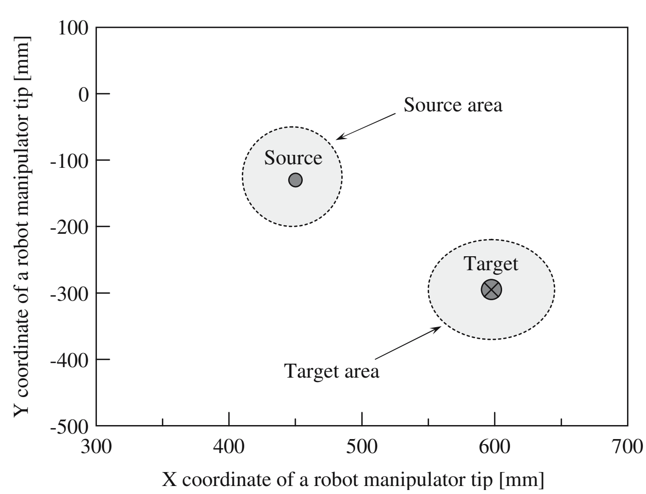

Fig. 4 Position information perception by the user

closer members in the state-space, the user may take similar decisions. Thus, it should be valid for any state-space as far as the user decisions are subjective.

Therefore, although S can contain an infinite number of continuous values, frequency of abrupt decision changes among sufficiently closer states is much smaller than the theoretical maximum. Thus, the classification problem becomes less complex.

4 Command interpretation

If a human user commands a robot manipulator with voice commands to reach a certain point, it has to be performed in a step-by-step fashion. At each step, there are two decisions to be made. They are,

1. Direction to move and
2. Distance to move.

After making these decisions, the user has to issue a voice command which includes both the direction to move and the distance to move. Out of these two components in the command, the direction command component is a non-fuzzy decision. The possible decisions are *left, right, forward, backward, up* and *down*. By looking at the present tip position of the robot, the target and the placement of obstacles, the user can subjectively decide the best direction to move.

On the other hand, the distance command component is a fuzzy command. That is because, when natural language commands are used to instruct distances, a command such as "*move little*" is more convenient than that containing numerical values. Therefore, interpretation of these fuzzy motion commands is one of the tasks to be performed.

The set of allowed commands is shown in Table 1. Any combination of a direction command component and a distance command component can be used as a command. For example, "*go very little right*" would be a valid command.

4.1 Fuzzy motion commands

All natural language commands are fuzzy in their very nature. Their meanings are subjective and context dependent. For example, what is meant by "*move little*" by a human is not a fixed value.

In this implementation, it has been assumed that the actual amount to be traversed in response to a distance command depends on the distance traversed immediately before that. This assumption is based on the observation of natural human tendency.

For example, a human who just traveled 10 km may consider another 1 km as a short distance, while another one who just traveled 100 m may consider the same 1 km as a long distance. This kind of an approach has been adopted in [12]. The similarity between the two approaches comes from the fact that, in both systems the actual response to the previous command is used as an input when interpreting the present command. However, in the system discussed in this paper, simple fuzzy reasoning is used while in [12], a fuzzy neural network has been used.

In the process of interpreting the meanings of fuzzy distance commands, following twelve rules are used for fuzzy reasoning [17, 18]:

R^1 : If a is 'very little' and l is L then h is VVS
R^2 : If a is 'very little' and l is M then h is VS
R^3 : If a is 'very little' and l is H then h is S
R^4 : If a is 'little' and l is L then h is S
R^5 : If a is 'little' and l is M then h is B
R^6 : If a is 'little' and l is H then h is VB
R^7 : If a is 'medium' and l is L then h is VB
R^8 : If a is 'medium' and l is M then h is VVB
R^9 : If a is 'medium' and l is H then h is F
R^{10} : If a is 'far' and l is L then h is F
R^{11} : If a is 'far' and l is M then h is VF
R^{12} : If a is 'far' and l is H then h is VVF

where a is distance command character variable, l is previous distance, and d is new distance. Fuzzy labels for the previous distance and the new distance are defined by,

VVS : Very Very Small
VS : Very Small
S : Small
B : Big
VB : Very Big
VVB : Very Very Big
F : Far
VF : Very Far
VVF : Very Very Far

Table 1 Fuzzy commands used by the human user

Direction command component (D_i)	Distance command component (d_i)
go up	
go down	very little
go right	little
go left	medium
go forward	far
go backward	

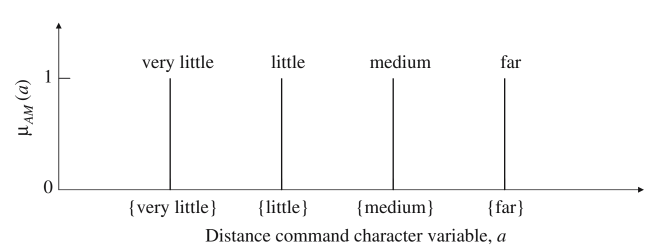

Fig. 5 Membership functions for action modification

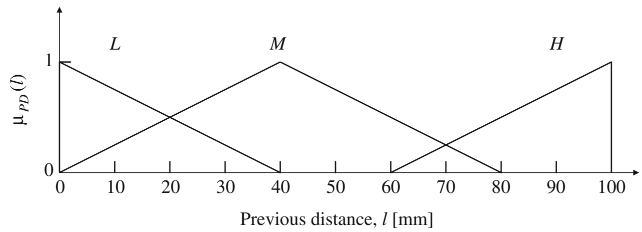

Fig. 6 Membership functions for previous distance

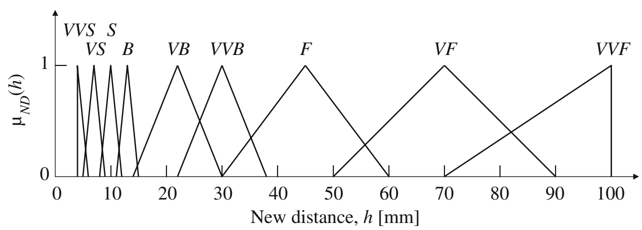

Fig. 7 Membership functions for new distance

L : Low
M : Medium
H : High

The distance command character variable a has a discrete support set and its membership value is either 1 or 0, i.e., a has singleton membership functions. The membership functions for a and l are shown in Figs. 5 and 6, respectively. Consequent part gives the new distance, d and the membership functions are shown in Fig. 7.

The firing strength of the ith rule, α_i is computed as,

$$\alpha_i = \mu_{AM_i}(a) \cdot \mu_{PD_i}(l) \tag{3}$$

Here, "·" is the algebraic product. Using Larsen's product operation rule as the fuzzy implication function, the ith rule leads to the decision,

$$\mu_{ND'_i}(h) = \alpha_i \cdot \mu_{ND_i}(h) \tag{4}$$

Consequently, the membership function $\mu_{ND'}$ of the inferred consequence is given by,

$$\mu_{ND'}(h) = \mu_{ND'_1}(h) \vee \ldots \vee \mu_{ND'_{12}}(h) \tag{5}$$

$$\mu_{ND'}(h) = \alpha_1 \cdot \mu_{ND_1}(h) \vee \ldots \vee \alpha_{12} \cdot \mu_{ND_{12}}(h) \tag{6}$$

To obtain the crisp output value, a defuzzification strategy is required. Using the well-known Center-of-Area method, the crisp output value of the new distance, h_0 is obtained as follows:

$$h_0 = \frac{\int_{-\infty}^{+\infty} \sum_{i=1}^{12} \alpha_i h \mu_{ND_i}(h) dh}{\int_{-\infty}^{+\infty} \sum_{i=1}^{12} \alpha_i \mu_{ND_i}(h) dh} \tag{7}$$

After the crisp value of the distance to be traversed, h_0 is decided, it can be directly used to control the tip position of the PA-10 manipulator using its tip position deviation control mode [19]. Initially, there is no distance traveled in response to the previous command. Therefore, the initial input value was decided according to the workspace of the manipulator.

5 Implementation of sub-coach

5.1 Gaining knowledge

From the discussion in Sect. 3.1, the robot-world state for the motions in three dimensional space is defined as,

$$S_i = \{\boldsymbol{s}_i^T, \boldsymbol{t}_i^T\} \tag{8}$$

where $\boldsymbol{s}_i$ is the current position vector of the robot, $\boldsymbol{t}_i$ is the final target position vector, and

$$\boldsymbol{s}_i^T = [s_{x_i}, s_{y_i}, s_{z_i}] \tag{9}$$

$$\boldsymbol{t}_i^T = [t_{x_i}, t_{y_i}, t_{z_i}] \tag{10}$$

where $(s_{x_i}, s_{y_i}, s_{z_i})$ and $(t_{x_i}, t_{y_i}, t_{z_i})$ are the x,y,z coordinates of the current position and of the final target, respectively. Command C_i is defined as

$$C_i = \{D_i, d_i\} \tag{11}$$

where D_i is the direction command component and d_i is the distance command component.

Possible values of D_i and d_i are shown in Table 1. As explained above, to interpret the fuzzy distance commands, the actual distance traveled in response to the previous command is used. Let the actual distance traveled in response to C_i be l_i. Possible fuzzy values of l_i are *low, medium* and *high*.

As explained in Sect. 3.2, gaining knowledge by the sub-coach means, learning the subjective function f. For this purpose, a PNN is used.

5.2 Decision making

Decision making by the sub-coach was realized using a PNN. The PNN architecture used in this paper is a modified version of the conventional PNN architecture. The PNN was first proposed in [20]. Because of ease of training and a sound statistical foundation in Bayesian estimation theory, PNN has become an effective tool for solving many classification problems [21–25].

One of the principal advantages of the PNN approach is that it is very much faster than the well-known back propagation approach, for problems in which the incremental adaptation time of back propagation is a significant fraction of the total computation time [20, 26, 27].

According to the explanation given in Sect. 3.3, the decision making process of the sub-coach is essentially a pattern classification problem. The input for the decision making algorithm is the robot world state. If we consider the direction decision making as an example, each state is associated with a direction decision. Since the number

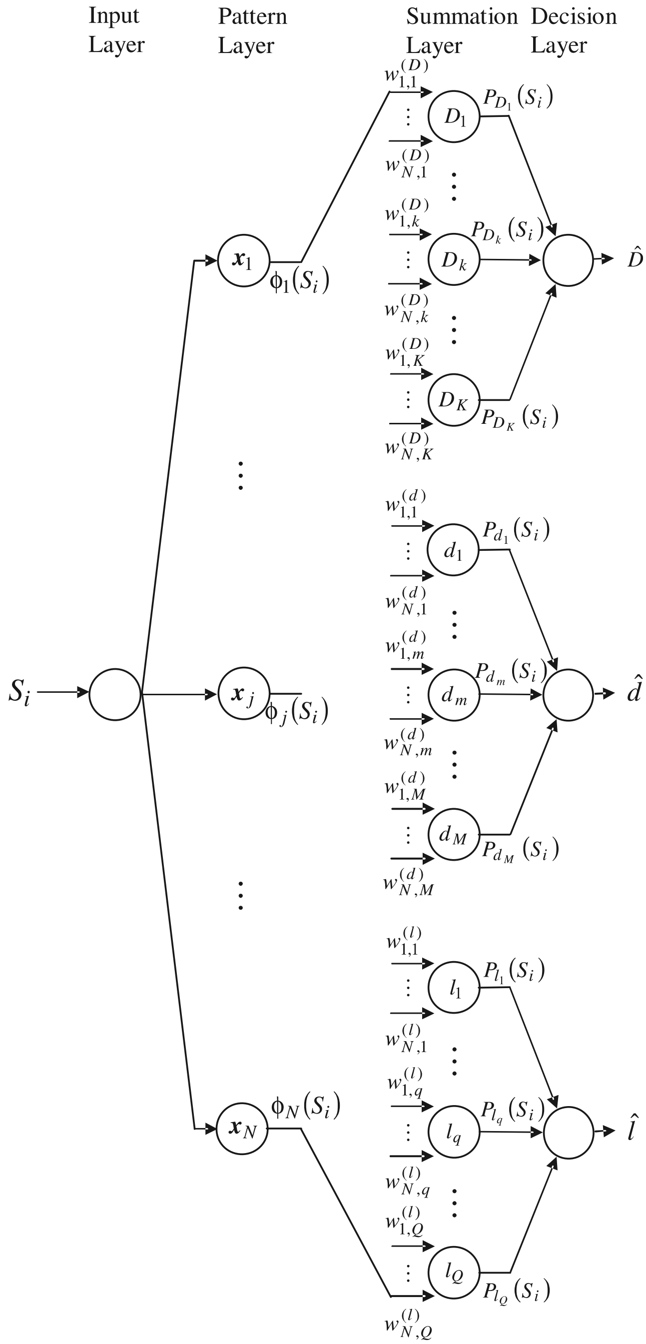

Fig. 8 Probabilistic neural network architecture (PNN)

of direction decisions is finite, selecting the most suitable decision is the same as categorizing the input state into the correct category. Deciding the most suitable distance and the most possible previous distance is performed in the same manner.

The PNN architecture used is shown in Fig. 8. The summation layer and the decision layer are composed of three parallel segments since this network is used to find three different values in parallel. That is, finding the most appropriate direction command, (D_i), the most appropriate distance command, (d_i), and the most appropriate actual distance associated with d_i, (l_i) is equivalent to three pattern classifications. These three segments can be called as segment D, segment d, and segment l.

In the figure,

- N : number of neurons in the pattern layer, or number of learned states (number of entries in the knowledge base);
- K : number of neurons in the segment D of the summation layer, or number of possible direction decisions;
- M : number of neurons in the segment d of the summation layer, or number of possible distance commands; and
- Q : number of neurons in the segment l of the summation layer, or number of possible distances traveled in response to the previous command.

Assume that S_i is the input received by the PNN. The input neurons are merely distribution units that supply the same input value to all the pattern neurons.

Each neuron in the pattern layer corresponds to a previously learned state. For example, the weight vector $\boldsymbol{x}_j$ associated with the jth neuron of the pattern layer is composed of the jth state of the training data set. Each neuron in the pattern layer forms the dot product of the input pattern vector S_i with its weight vector $\boldsymbol{x}_j$ and then performs a nonlinear operation on the dot product. Thus, the output of the jth neuron is given by,

$$\phi_j(S_i) = \exp\left\{\frac{-(S_i - \boldsymbol{x}_j)^T(S_i - \boldsymbol{x}_j)}{2\sigma^2}\right\} \tag{12}$$

where σ is a smoothing parameter.

Here, we should observe that the number of neurons in the pattern layer is equal to the number of training samples. As the training data set becomes larger, the network size may grow proportionally. Thus, one of the outstanding issues associated with PNN is determining network size. Various research have been carried out on reducing the number of neurons in the pattern layer [21, 24, 28]. However, in this paper, this issue has not been addressed because the number of training samples is of a manageable size, i.e., 55 in experiment 1 and 60 in experiment 2. On the other hand, it is a topic for a separate research which is out of the scope of the present work.

Weights that connect the pattern layer and the summation layer are defined as follows:

$$w_{j,k}^{(D)} = \begin{cases} 1 & \text{if } D_j = D_k \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where $k = 1,2,\ldots,K$,

$$w_{j,m}^{(d)} = \begin{cases} 1 & \text{if } d_j = d_m \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

where $m = 1,2,\ldots,M$, and

$$w_{j,q}^{(l)} = \begin{cases} 1 & \text{if } l_j = l_q \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

where $q = 1,2,...,Q$.

Each neuron in the pattern layer connects to each neuron in each segment of the summation layer. For example, $w_{j,k}^{(D)}$ is the weight that connects the jth neuron of the pattern layer to the kth neuron in the segment D of the summation layer. If the direction decision D_j associated with the state j is equal to D_k, then $w_{j,k}^{(D)}$ is 1; otherwise, it is 0.

Neurons in the summation layer compute the maximum likelihood of D_i, d_i, and l_i associated with the state S_i being equal to D_k, d_m, and l_q. It is given by,

$$P_{D_k}(S_i) = \frac{\sum_{j=1}^{N} \phi_j(S_i) w_{j,k}^{(D)}}{\sum_{j=1}^{N} w_{j,k}^{(D)}} \quad (16)$$

$$P_{d_m}(S_i) = \frac{\sum_{j=1}^{N} \phi_j(S_i) w_{j,m}^{(d)}}{\sum_{j=1}^{N} w_{j,m}^{(d)}} \quad (17)$$

$$P_{l_q}(S_i) = \frac{\sum_{j=1}^{N} \phi_j(S_i) w_{j,q}^{(l)}}{\sum_{j=1}^{N} w_{j,q}^{(l)}} \quad (18)$$

The decision layer classifies the state S_i based on the output of all neurons in the summation layer by using

$$\hat{D} = D_k \ \text{ if } \ P_{D_k}(S_i) = \max\{P_{D_1}(S_i), \ldots, P_{D_K}(S_i)\} \quad (19)$$

$$\hat{d} = d_m \ \text{ if } \ P_{d_m}(S_i) = \max\{P_{d_1}(S_i), \ldots, P_{d_M}(S_i)\} \quad (20)$$

$$\hat{l} = l_q \ \text{ if } \ P_{l_q}(S_i) = \max\{P_{l_1}(S_i), \ldots, P_{l_Q}(S_i)\} \quad (21)$$

where $\hat{D}$, $\hat{d}$, and $\hat{l}$ denote the most probable direction command, the most probable distance command, and the most probable distance traveled in response to the previous command respectively.

Here, $\hat{l}$ needs further explanation. The most probable distance command $\hat{d}$ is decided using the distance commands associated with previous states in a small neighborhood of the current state. However, the meanings of these fuzzy commands are context dependent; i.e., they depend on the corresponding previous distances traveled. Consequently, the command estimated based on past data is valid only for a certain context. $\hat{l}$ is the context, i.e. the immediate previous distance for which the command $\hat{d}$ is valid.

To deduce the correct distance command (d_i) from these values, the algorithm shown in Fig. 9 is used. It can be explained as below.

Assume that $\hat{d}$ and $\hat{l}$ are *medium* and L respectively. As explained above, these are the distance command and the actual distance traveled in response to the previous command corresponding to a small neighborhood. In other words, for the neighborhood, the distance command had been "*medium*" in the context in which the distance traveled in response to the previous command was L. User had issued that command after observing that the actual distance traveled in response to the previous command was *low*. Assume that, after interpreting this command, the robot had traveled 25 mm.

For the current state, the sub-coach also needs to issue a similar command. Whatever the distance command, its interpreted crisp value should be less than 25 mm because beyond that point, the sub-coach doesn't know whether there are any obstacles or not. On the other hand, it should command the robot to travel the maximum possible distance to ensure the highest efficiency. Thus, if l_i is *low*, then the sub-coach can issue "*medium*" as the next command. However, if l_i is *medium* or *high*, then it has to issue "*little*," because otherwise, the interpreted crisp distance will be more than 25 mm.

IF $\hat{d} = \textit{very little}$ THEN $d_i = \textit{very little}$
ELSE IF $\hat{d} = \textit{little}$ THEN
  IF $\hat{l} = L$ THEN
    IF $l_i = L$ THEN $d_i = \textit{little}$
    ELSE $d_i = \textit{very little}$
  ELSE IF $\hat{l} = M$ THEN
    IF $l_i = L \textit{ or } M$ THEN $d_i = \textit{little}$
    ELSE $d_i = \textit{very little}$
  ELSE IF $\hat{l} = H$ THEN $d_i = \textit{little}$
ELSE IF $\hat{d} = \textit{medium}$ THEN
  IF $\hat{l} = L$ THEN
    IF $l_i = L$ THEN $d_i = \textit{medium}$
    ELSE $l_i = \textit{little}$
  ELSE IF $\hat{l} = M$
    IF $l_i = L \textit{ or } M$ THEN $d_i = \textit{medium}$
    ELSE $d_i = \textit{little}$
  ELSE IF $\hat{l} = H$ THEN $d_i = \textit{medium}$
ELSE IF $\hat{d} = \textit{far}$
  IF $\hat{l} = L$ THEN
    IF $l_i = L$ THEN $d_i = \textit{far}$
    ELSE $d_i = \textit{medium}$
  ELSE IF $\hat{l} = M$
    IF $l_i = L \textit{ or } M$ THEN $d_i = \textit{far}$
    ELSE $d_i = \textit{medium}$
  ELSE IF $\hat{l} = H$ THEN $d_i = \textit{far}$

Fig. 9 Algorithm to deduce d_i from $\hat{d}$

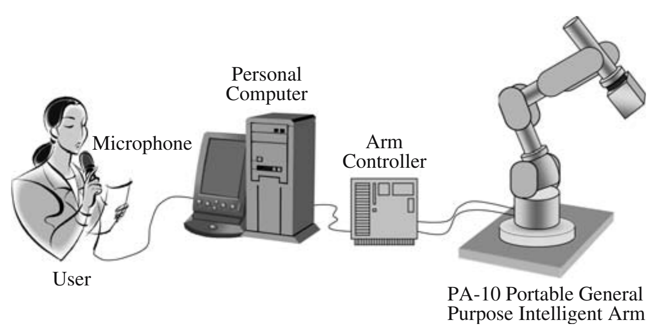

Fig. 10 The experimental setup

As explained above, the direction decisions made by the sub-coach are non-fuzzy. They are, *up, down, right, left, forward,* and *backward*. Assuming that the conditions which influence the direction decision are the same for all the members in a small neighborhood, the sub-coach can use $\hat{D}$ as the actual direction command (D_i) suitable for the current state.

6 Experiments

Two experiments were conducted to demonstrate the proposed concept. In the first experiment, the objective was to move the tip or the end effector of a manipulator in 2D space from any point to a target point on a table avoiding obstacles. In the second experiment, the objective was to perform a set of pick and place operations which involve the motions in 3D space.

For both experiments, the experimental setup is shown in Fig. 10. It consists of a microphone, a personal computer, a PA-10 portable general purpose intelligent arm and the arm controller. The speech recognition software, the sub-coach program and the operational control program of PA-10 are hosted in the personal computer whose operating system is Windows XP. The speech recognition is performed using IBM Via Voice commercial software.

The flowchart shown in Fig. 11 shows the operation of the sub-coach, where S_i is the current state for which a decision is required and C_i is the suitable command corresponding to the state S_i.

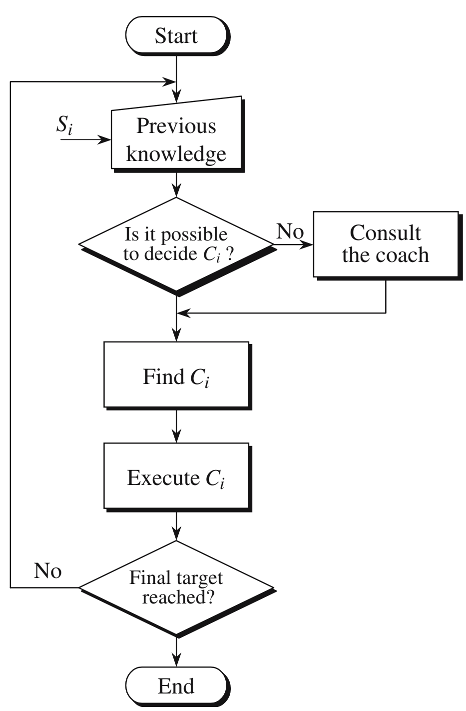

Fig. 11 Command generation of the sub-coach using the knowledge base

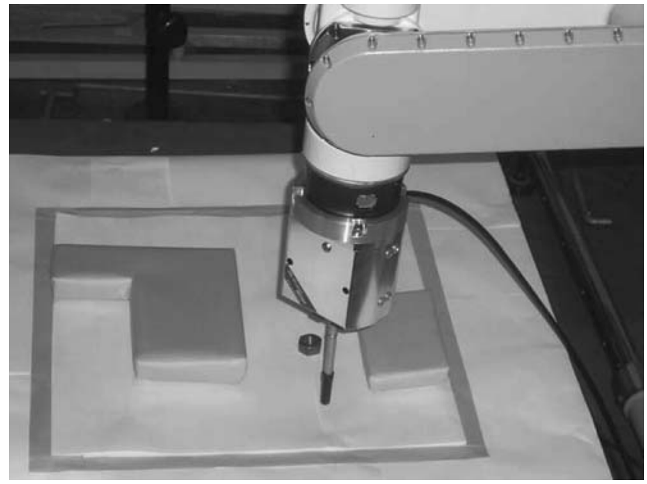

Fig. 12 View of the experimental setup in experiment 1

6.1 Experiment 1: 2D motion

A view of the experimental setup is shown in Fig. 12. In this experiment, first the training data were collected and the PNN was trained. Then, the trained PNN was used by the sub-coach for decision making.

Four training movements were set so as to cover different areas of the working space of the robot. They are shown in Fig. 13, where at each point marked with a circle, the user has taken a direction decision and a distance decision. S_i, C_i and l_i at those positions were used as the training data samples. A portion of the training data is shown in Table 2. Note here that altogether 55 samples were used for the training.

Once the training has been completed, some test movements have been performed with the sub-coach controlling the robot. Three of such test movements are shown in Fig. 14.

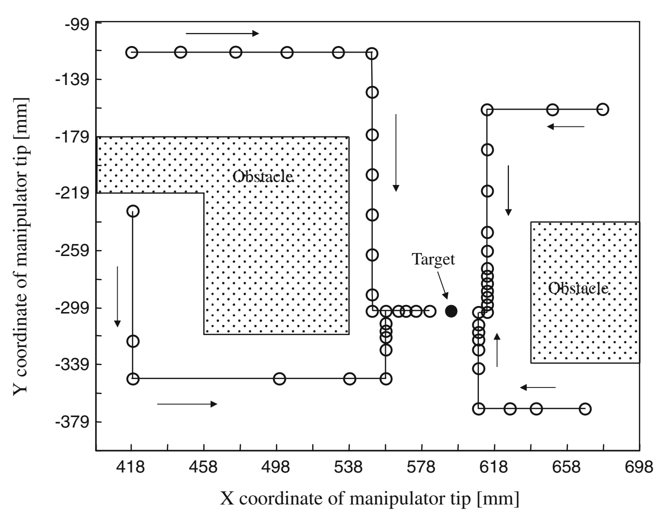

Fig. 13 Training movements performed by the human user (coach) in experiment 1

Table 2 Training data for PNN in experiment 1

i	State (S_i)						C_i		l_i
	s_{x_i}	s_{y_i}	s_{z_i}	t_{x_i}	t_{y_i}	t_{z_i}	D_i	d_i	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
5	531.70	−120.12	552.34	597.97	−292.00	552.30	Fo	L	M
6	545.36	−120.41	552.83	597.97	−292.00	552.30	R	M	H
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
13	550.63	−298.29	552.63	597.97	−292.00	552.30	Fo	M	Lo
14	569.13	−302.47	552.66	597.97	−292.00	552.30	Fo	L	M
15	417.99	−230.91	552.27	597.96	−290.96	552.31	R	F	H
16	418.07	−322.07	552.32	597.96	−290.96	552.31	R	L	H
17	418.09	−342.08	552.40	597.96	−290.96	552.31	Fo	F	H
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
55	606.35	−305.18	552.76	597.95	−290.98	552.35	B	VL	Lo

C_i coach's Command, l_i actual distance traveled in response to C_i, *Fo* forward, *R* right, *B* backward, *L* little, *M* medium, *F* far, *VL* very Little, *H* high, *Lo* low

The broken lines indicate all the guided training movements performed by the human user. The knowledge base was built using these movements. Solid lines indicate the test movement. At places marked with circles, the sub-coach has taken a direction decision and a distance decision. It is observed that the sub-coach alone can guide the robot arm tip to come very closer to the final target. To reach the exact target point, more finer movements are necessary and the knowledge of the sub-coach is not sufficient for that. Therefore, such finer movements has to be performed with the consultation of the human user.

6.2 Experiment 2: 3D motion

A view of the experimental setup is shown in Fig. 15. In this experiment the training of the PNN was performed online. The objective was to pick the objects from an object table and place them in a bin located away from the object table.

Two object moving motions are shown in the Fig. 16. Here, the coordinates of the object with reference to the object table are assumed to be known in advance by some other means. For example, object coordinates on the table can be calculated using an image taken from a camera placed above the table. Therefore, in this experiment what is learned is the path from the table to the bin avoiding surrounding obstacles.

During the on-line training, objects placed in different places of the table were moved to the bin. At the points marked with circles in the Fig. 16, decisions were made. First, the sub-coach made the decision and if it is correct according to the human evaluation, it was executed; otherwise, the human user issued the correct command.

Figure 17 shows the total number of incorrect decisions made by the sub-coach vs. the total number of decisions made during 9 pick and place movements. Once the sub-coach made an incorrect decision, the knowledge base of the sub-coach is updated using the correct human user command. Thus, the number of incorrect decisions is equivalent to the number of training samples for the PNN.

It is seen that initially the number of errors is higher due to lack of knowledge. As the knowledge base grows, the growth of accumulated error is reduced and finally saturated. Observe that after about 60 training samples, the learning was converged for this particular task.

7 Conclusions

The learning of sub-coach has been discussed in the framework of fuzzy coach-player system by applying a probabilistic neural network. First, the importance of learning from information rich natural language commands was discussed and the new concepts of fuzzy coach-player system and sub-coach were introduced. Then the characteristics of subjective human decision making process were discussed and a mathematical model which could be used for subjective decision making was developed.

Two experiments were conducted using a PA-10 redundant manipulator. In the first experiment, the sub-coach was trained with training movements covering different areas of a table in 2D space where some obstacles were placed. A training movement was to move the arm-tip of the robot from a point located far away to a target point. In doing so, the user commanded the robot to move its tip little by little avoiding obstacles. At each step, the user took two decisions, i.e. direction to move and distance to move. From those decisions, the sub-coach built its knowledge base.

After the training, test movements were made. In the test movements, all the direction and distance decisions were performed by the sub-coach without any intervention of a human. It was observed that the sub-coach alone was able to guide the robot arm-tip to come very closer to the final target avoiding obstacles successfully.

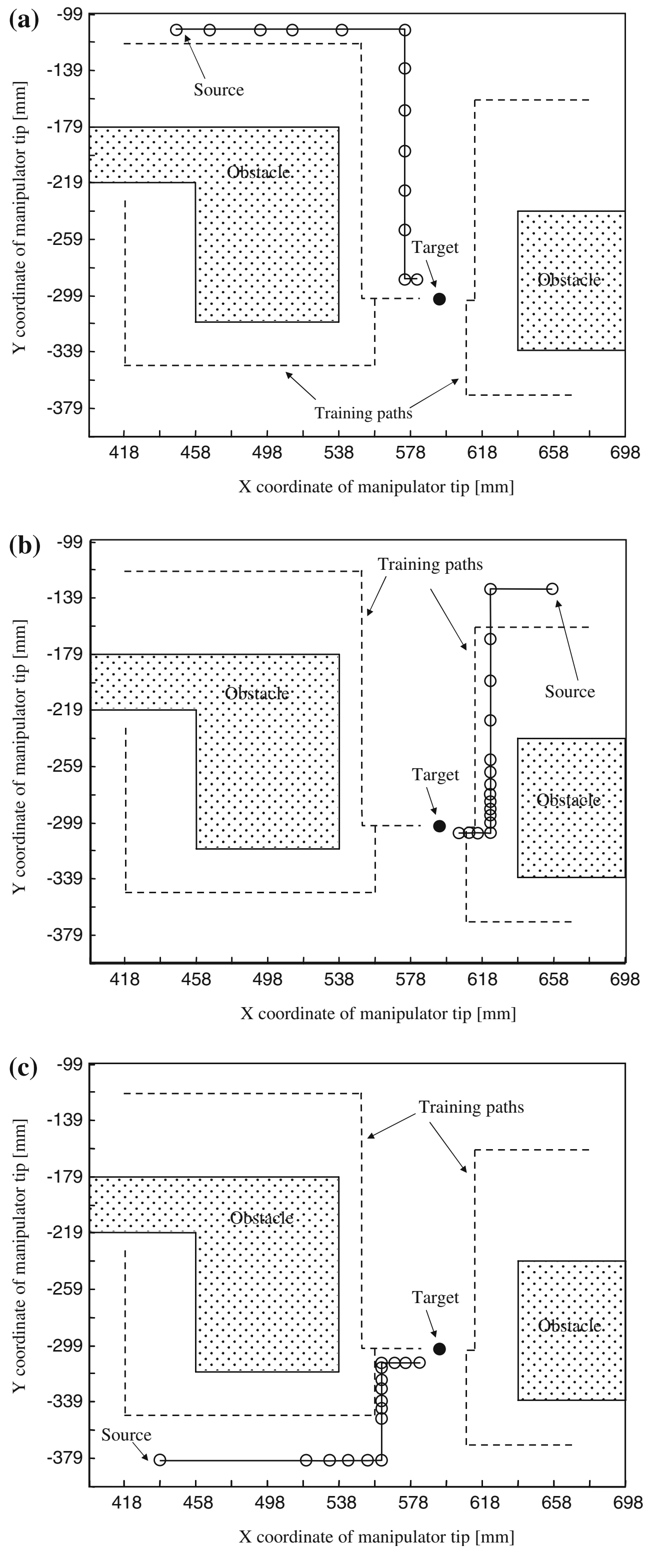

Fig. 14 Some test movements in experiment 1

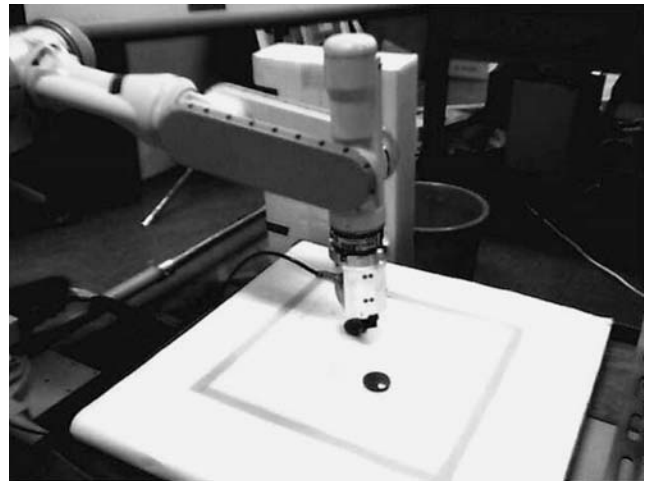

Fig. 15 View of the experimental setup in experiment 2

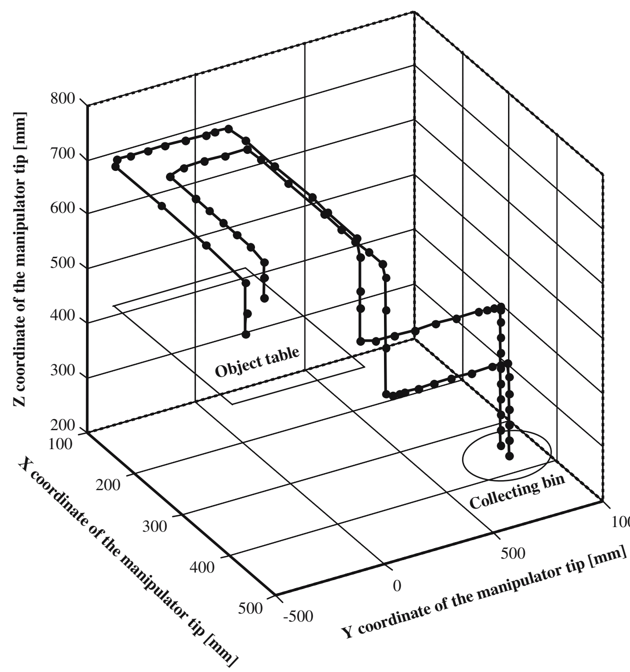

Fig. 16 Sample training movements in experiment 2

For finer movements, the human user intervention is necessary.

In the second experiment it was required to move the tip or the end effector of the manipulator in 3D space. The objective was to pick the objects from an object table and to place them in a bin located away from the object table. In doing so, the user commanded the robot to move its tip from a position in the table to the bin avoiding obstacles. At each step, the sub-coach took two decisions, i.e. direction to move and distance to move, and these decisions were evaluated by the human user. If the decision was correct, it was executed, whereas if it was incorrect, the correct decision was issued by the human; thus improving the knowledge of the sub-coach. It was observed that after about 60 training samples, the learning was converged for this particular task.

Thus, we can see that it is possible to hand over coarse tasks to the sub-coach while finer tasks are

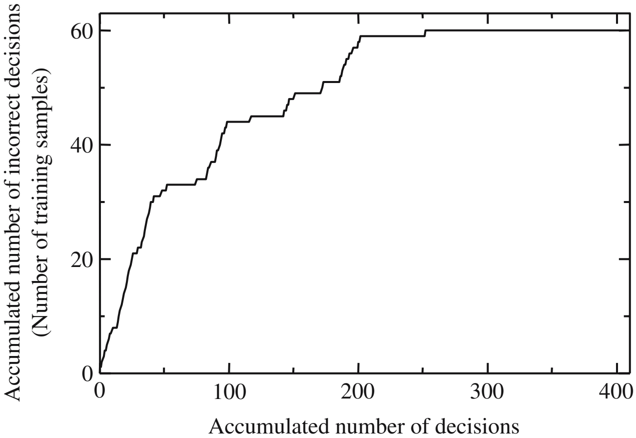

Fig. 17 Convergence of learning in experiment 2

performed by the human user, in a sophisticated environment. For example, in [11] and [12] the authors presented a natural language controlled robotic manipulator which can be used to perform an assembly task. However, since the learning was not incorporate in their work, user needed to issue similar commands repetitively during an assembling task. Moreover, when performing multiple assembling tasks of similar nature, similar command sequences were needed to be repeated. Using the method presented in this paper, this kind of redundancy can be avoided effectively reducing the burden of a controlling user. On the other hand, a user may control more than one robots at the same time, just monitoring and helping them as needed.

Although the proposed concept was illustrated with experiments conducted with a robotic manipulator, the same method may also be suitably applied for other robotic systems. The most important feature of the proposed method is that it utilizes the inherent fuzzy nature of spoken language commands to generate probable commands for unknown cases.

References

1. Menzel P, D'Aluisio F (2000) Robosapiens. The MIT Press, UK, London
2. Pineau J, Montemerlo M, Pollack M, Roy N, Thrun S (2003) Towards robotic assistants in nursing homes: challenges and results. Rob Auton Syst 42(3–4):271–281
3. Fong T, Nourbakhsh I, Dautenhahn K (2003) A survey of socially interactive robots. Rob Auton Syst 42:143–166
4. Oates T, Schmill MD, Cohen PR (2000) A method for clustering the experience of a mobile robot that accord with human judgement. In: Proceedings of the seventh national conference on artificial intelligence:846–851
5. Roy D (2003) Grounded spoken language acquisition: experiments in word learning. IEEE Trans on Multimedia 5(2):197–209
6. Roy D, Hsio K, Mavridis N (2004) Mental imagery for a conversational robot. IEEE Trans Syst Man Cybern B Cybern 34(3):1374–1383
7. Ballard DH, Chen Y (2003) A multimodal learning interface for word acquisition. Proc IEEE Int Conf Acoust Speech Signal Processing 5:784–787
8. Chen Y, Ballard DH (2004) A multimodal learning interface for grounding spoken language in sensory perceptions. ACM Trans Appl Perceptions 1(1):57–80
9. Lin CT, Kan MC (1998) Adaptive fuzzy command acquisition with reinforcement learning. IEEE Trans Fuzzy Syst 6(1):102–121
10. Pulasinghe K, Watanabe K, Izumi K, Kiguchi K (2004) A modular fuzzy-neuro controller driven by spoken language commands. IEEE Trans Syst Man Cybern B Cybern 34(1):293–302
11. Chatterjee A, Pulasinghe K, Watanabe K, Izumi K (2005) A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems. IEEE Trans Ind Electron 52(6):1478–1489
12. Pulasinghe K, Watanabe K, Izumi K, Kiguchi K (2003) Control of redundant manipulators by fuzzy linguistic commands. In: Proceedings of the SICE annual conference, Fukui, Japan:2819–2824
13. Jayawardena C, Watanabe K, Izumi K (2004) Knowledge acquisition by a sub-coach in a coach-player system for controlling a robot. In: Proceedings of the 4th international conference on advanced mechatronics, Hokkaido, Japan:601–606
14. Jayawardena C, Watanabe K, Izumi K (2004) Probabilistic neural network based learning from fuzzy voice commands for controlling a robot. In: Proceedings of the international conference on control, automation, and systems, Bangkok, Thailand:2011–2016
15. Jayawardena C, Watanabe K, Izumi K (2004) Intelligent subcoach construction in a fuzzy coach-player system for controlling a robot manipulator. In: Proceedings of the joint 2nd international conference on soft computing and intelligent systems and 5th international symposium on advanced intelligent systems (SCIS & ISIS 2004), Yokohama, Japan:CD-ROM
16. Råde L, Westergren B (1999) Mathematics Handbook for Science and Engineering. 4th edn. Springer, Berlin Heidelberg New York
17. Lee CC (1990) Fuzzy logic in control systems: fuzzy logic controller, Part I. IEEE Trans on Systems, Man and Cybernetics 20(2):404–418
18. Lee CC (1990) Fuzzy logic in control systems: fuzzy logic controller, Part II. IEEE Trans on Systems, Man and Cybernetics 20(2):419–435
19. Mitubishi Heavy Industries Ltd., PA-10 portable general purpose intelligent arm programming manual - rev.1
20. Specht DF (1990) Probabilistic neural networks. Neural Networks 3(1):109–118
21. Raghu PP, Yegnanarayana B (1998) Supervised texture classification using a probabilistic neural network and constraint satisfaction model. IEEE Trans on Neural Networks 9(3):516–522
22. Ganchev T, Fakotakis N, Kokkinakis G (2003) Impostor modeling techniques for speaker verification based on probabilistic neural networks. In: Proceedings of IASTED international conference on signal processing, pattern recognition, and applications, Rhodes, Greece:185–190
23. Ganchev T, Fakotakis N, Kokkinakis G (2003) Locally recurrent probabilistic neural networks for text independent speaker verification. Proc of the EuroSpeech 3:1673–1676
24. Romero RD, Touretzky DS, Thibadeau RH (1997) Optical chinese character recognition using probabilistic neural networks. Pattern Recognition 30(8):1279–1292
25. Musavi MT, Chan KH, Hummels DM, Kalantri K (1994) On the generalization ability of neural-network classifier. IEEE Trans on Pattern Analysis and Machine Intelligence 16(6):659–663
26. Duda RO, Hart PE, Stork DG (2004) Pattern classification. 2nd edn. Wiley, New York
27. Mao KZ, Tan KC, Ser W (2000) Probabilistic neural-network structure determination for pattern classification. IEEE Trans on Neural Networks 11(4):1009–1016
28. Burrascano P (1991) Learning vector quantization for the probabilistic neural network. IEEE Trans on Neural Networks 2(4):458–461