Probabilistic Neural Network Based Learning from Fuzzy Voice Commands for Controlling a Robot

Chandimal Jayawardena, Keigo Watanabe, and Kiyotaka Izumi

Department of Advanced Systems Control Engineering, Graduate School of Science and Engineering, Saga University, Japan
(Tel: +81-952-8602; Fax: +81-952-8587; Email: chandimal@ieee.org, {watanabe,izumi}@me.saga-u.ac.jp)

Abstract: Study of human-robot communication is one of the most important research areas. Among various communication media, any useful law we find in voice communication in human-human interactions, is significant in human-robot interactions too. Control strategy of most of such systems available at present is on/off control. These robots activate a function if particular word or phrase associated with that function can be recognized in the user utterance. Recently, there have been some researches on controlling robots using information rich fuzzy commands such as "go little slowly". However, in those works, although the voice command interpretation has been considered, learning from such commands has not been treated. In this paper, learning from such information rich voice commands for controlling a robot is studied. New concepts of the coach-player model and the sub-coach are proposed and such concepts are also demonstrated for a PA-10 redundant manipulator.

Keywords: fuzzy voice commands, coach-player model, sub-coach, probabilistic neural network (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 become successful, 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].

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 researches. Socially interactive robots are capable of showing human like behaviour when dealing with another human, i.e. communicating as peers using natural languages, gestures, etc.[2].

As the human-robot relationship is becoming more important, the studies on human-robot communication too has been given a high priority. Among various communication media, any useful law that we find in voice communication in human-human interactions, is significant in human-robot interactions too. 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.

Most of the available systems used for such applications are based on on/off control. These robots activate a function if a perticular word or a phrase associated with that function can be recognized in the user utterance. However, in natural language communication, encountering words and phrases with fuzzy implications is inevitable. On the other-hand, words with fuzzy implications are useful in machine control because they can fine tune the performance of the machine. For example, a command like "*move slowly*" will contain many information regarding the nature of the terrain, distance to obstacles, etc.[3].

Lin and Kan[4] proposed an adaptive fuzzy command aquisition method to control machines by natural language commands such as "*move forward at a very high speed*". In [5] robot control using such information rich voice commands has been studied. In a more practical approach adopted in [6], controlling a redundant manipulator by such commands to achieve a complex assembling task has been demonstrated.

However, in all those systems, although the interpretation of fuzzy voice commands has been considered well, the importance of learning from such commands has not been addressed. As explained above, this type of commands contains a lot of information because of the fact that they are generated by intelligent and experienced human beings. Therefore, if it is possible to learn from them and re-use that knowledge effectively, it will be a very efficient process. This paper proposes a learning method from such fuzzy voice commands for controlling a robot.

Consider the process of controlling a robot to achieve a complex task using voice commands. If the robot does not have any prior knowledge of the task and it cannot learn from the previous commands, the user has to issue a series of commands at each step such as:

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

This type of a system can be seen as a coach-player system, since a coach (or a human user) commands a robot (or a player) by observing its performance at each step. The robot acts accordingly and the user will stop commanding when the performance of the robot comes to a satisfactory level (the work at hand is completed). Although this kind of coach-player systems is very useful in many situations, their disadvantage is the need of issuing commands at each small step. This can be eliminated by introducing the concept of a

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

The concept of a sub-coach for learning has been first proposed in [7]. However, there, the demonstration of the concept was limited to crisp decision making by a sub-coach. In this paper, that concept is more amplified by introducing fuzzy decision making too.

2. Learning from Fuzzy Voice Commands

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. It observs commands issued by the user and the actions performed by the robot in respect to such 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 it did during the learning period, without consulting the human user. Of course, in certain situations, the sub-coach may consult the human user if it does not possess enough knowledge to take a decision according to current circumstances.

When controlling a robot with voice commands, the command of the user depends on the state of the robot at that perticular instance. User evaluates this 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*".

Therefore, the process of controlling a robot using a series of voice commands can be seen as changing the state of the robot repetitively until the required target is achieved. Current state of the robot is defined by all the parameters which may affect the users's next command. It may include position information, velocity information, etc. The exact definition of the current state depends on the application under consideration.

Say, during the learning phase, the human user issued commands with respect to N states of the robot. That means, the sub-coach had the opportunity to learn the response of the human with respect to N possible states of the robot. If the ith state is written as S_i, it is possible to define S_i as,

$$S_i = \{x_1, x_2, ..., x_p\} \tag{1}$$

Here, $x_1, x_2, ..., x_p$ are the state-parameters that define the state of the robot. p is the number of parameters required to define a state. Thus, a state is a p dimensional entity and it is a member of a p dimensional state-space. State-parameters can either be scalars or vectors depending on the application. If the command issued at the ith state is C_i, it can be defined as,

Table 1. Fuzzy commands used by the human user.

Action (D_i)	Action Modification (d_i)
go up	
go down	very little
go right	little
go left	medium
go forward	far
go backward	

$$C_i = f(S_i) \tag{2}$$

Here, $f(\cdot)$ is a subjective function which depends on the knowledge, experience, attitude, etc. of the user. If enough intelligence is built into the sub-coach, it can learn from $C_1, C_2, ..., C_N$ issued in response to the states $S_1, S_2, ..., S_N$. Thus, it can build a knowledge base containing learned states and corresponding commands and use them later for decision making.

3. Decision Making

In any job, if the current state of the robot is S_i, the human user or the coach uses $\{S_i$, *his knowledge, his experience, his attitude, etc.*$\}$ to decide the next command C_i. If the same state is presented to the sub-coach, it uses $\{S_i, the\ knowledge\ base\}$ to decide C_i.

The most significant feature in the decision making process of the sub-coach is originated from the fact that the human user commands are fuzzy in their very nature. That means, although we can 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 objevtive; rather they are subjective decisions. The sub-coach can utilize this property to generate decisions, which are similar to the human user commands.

As explained above, if the dimension of a state is p, then it is a member of a p dimensional state-space. Since the human user perceives a state using his own senses, there is a high correlation between the decisions he takes in response to two sufficiently closer members in the state space. Therefore, if the sub-coach can find a sufficiently closer previous state from the knowledge base, it can deduce a suitable command for the current state.

Thus, finding the previous command issued in repect to a sufficiently closer state is the major task in the decision making process.

4. System Overview

To prove the proposed concept, the following test model was used.

1. Objective:
   - Moving the tip of a robot manipulator from one point to another avoiding obstacles.
2. Decisions that are required to take at each step:
   - Direction to move.

Table 2. Knowledge base, where tip position deviations (relative to the tool frame) of PA-10 are given in milimeters.

i	State (S_i)						Coach's Command (C_i)		Actual distance travelled in response to 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	Forward	Little	Medium 1
6	545.36	−120.41	552.83	597.97	−292.00	552.30	Right	Medium	High
...	...	...	...	...	...	...	...	...	
...	...	...	...	...	...	...	...	...	
13	550.63	−298.29	552.63	597.97	−292.00	552.30	Forward	Medium	Low
14	569.13	−302.47	552.66	597.97	−292.00	552.30	Forward	Little	Medium 1
15	417.99	−230.91	552.27	597.96	−290.96	552.31	Right	Far	High
16	418.07	−322.07	552.32	597.96	−290.96	552.31	Right	Little	High
17	418.09	−342.08	552.40	597.96	−290.96	552.31	Forward	Far	High
...	...	...	...	...	...	...	...	...	...
55	606.35	−305.18	552.76	597.95	−290.98	552.35	Backward	Very Little	Low

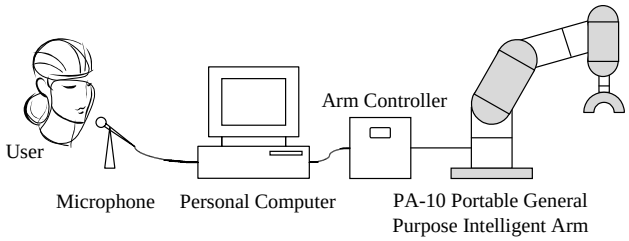


Fig. 1. The experimental setup.

- Distance to move.

3. Fuzzy commands used by the human user to command the robot based on his subjective evaluation of the above decisions:

- Any combination of an 'Action' term and an 'Action Modification' term given in Table 1.

Out of the two decisions required to take at a step, direction decision is a non-fuzzy decision. The possible decisions are *up, down, left, right, forward*, and *backward*. However, the distance dicision is a fuzzy decision. In [10] for the interpretation of fuzzy-commands, the response for the previous command has been used. In this paper, in addition to that, fuzzy partitioning of the input-space [11] is used for the fuzzy inference.

The experimental setup is shown in Fig. 1. 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 for PA-10 are hosted in the personal computer whose operating system is Windows XP. The speech recognition was performed using IBM Via Voice SDK.

5. Implementation

5.1. Building the knowledge base

State of the robot, S_i was defined as

$$S_i = \{\boldsymbol{s}_i^T, \boldsymbol{t}_i^T\} \tag{3}$$

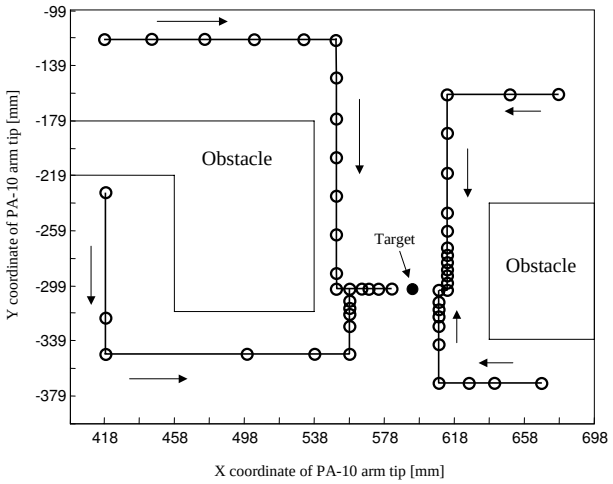


Fig. 2. Training movements performed by the hman user (coach).

where

$$\begin{aligned} \boldsymbol{s}_i^T &= \text{Current position vector of the robot} \\ \boldsymbol{t}_i^T &= \text{Final target position vector} \end{aligned}$$

and

$$\boldsymbol{s}_i^T = [s_{x_i}, s_{y_i}, s_{z_i}] \tag{4}$$

$$\boldsymbol{t}_i^T = [t_{x_i}, t_{y_i}, t_{z_i}] \tag{5}$$

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

$$C_i = \{D_i, d_i\} \tag{6}$$

where

$$\begin{aligned} D_i &= \text{Direction command} \\ d_i &= \text{Distance command} \end{aligned}$$

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 travelled is used. Let the actual distance travelled in response to C_i be l_i. Possible values of l_i are *low, medium 1, medium 2* and *high*.

Four training movements were set so as to cover different areas of the working space of the robot. They are shown in the Fig. 2. At each point marked with a circle, the user has taken a direction decision and a distance decision. Using the S_i, C_i and l_i at those positions, the sub-coach has built a knowledge base shown in Table 2. After the training, the knowledge base consisted of 55 entries.

5.2. Decision making

Decision making by the sub-coach was realized using a Probabilistic Neural Network (PNN). The PNN architecture used in this paper is a new modified version of the conventional PNN architecture. It is shown in Fig. 3. The difference in this architecture is, the summation layer and the decision layer are composed of three parallel segments. That is because, the same network is used to find three different values in parallel. They are, the direction command (D), the distance command (d), and the actual distance travelled in response to the previous command (l). These three segments can be called as segment D, segment d, and segment l.

In the figure,

$N =$ Number of learned states (number of entries in the knowledge base)
$K =$ Number of possible direction commands
$M =$ Number of possible distance commands
$Q =$ Number of possible distances travelled 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 pattern layer neuron corresponds to a previously learned state. For example, jth neuron in the pattern layer corresponds to the jth state in the knowledge base. Weight vector $\boldsymbol{x}_j$ associated with the jth neuron of the pattern layer is composed of the jth state in the knowledge base.

Each pattern neuron forms the dot product of the input pattern vector S_i with a weight vector $\boldsymbol{x}_j$, i.e., $z_{ij} = S_i^T \boldsymbol{x}_j$ and then performs a nonlinear operation on the dot product. When using $\exp\left[(z_{ij}-1)/\sigma^2\right]$ as the nonlinear operator, the output of the jth neuron is given by,

$$\phi_j(S_i) = \exp\left\{\frac{-(S_i-\boldsymbol{x}_j)^T(S_i-\boldsymbol{x}_j)}{2\sigma^2}\right\} \tag{7}$$

if $\|S_i\|^2 = \|\boldsymbol{x}_j\|^2 = 1$, where σ is a smoothing parameter [8] [9].

$\phi_j(S_i)$ is the activation level applied to the summation layer neurons by the jth pattern layer neuron due to the input S_i. Weights that connect the pattern layer and the summation layer are defined as follows:

$$w_{j,k}^{(D)} = \begin{cases} 1 & \text{if } D_j = D_k \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $k = 1, 2, ..., K$

$$w_{j,m}^{(d)} = \begin{cases} 1 & \text{if } d_j = d_m \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where $m = 1, 2, ..., M$

$$w_{j,q}^{(l)} = \begin{cases} 1 & \text{if } l_j = l_q \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

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

Each pattern layer neuron connects to the 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 (the jth learned state in the knowledge base) is equal to D_k, then $w_{j,k}^{(D)}$ is 1. Othewise, it is 0.

The summation layer neurons 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. They are found as follows:

$$P_{D_k}(S_i) = \frac{\sum_{j=1}^{N} \phi_j(S_i) w_{j,k}^{(D)}}{\sum_{j=1}^{N} w_{j,k}^{(D)}} \tag{11}$$

$$P_{d_m}(S_i) = \frac{\sum_{j=1}^{N} \phi_j(S_i) w_{j,m}^{(d)}}{\sum_{j=1}^{N} w_{j,m}^{(d)}} \tag{12}$$

$$P_{l_q}(S_i) = \frac{\sum_{j=1}^{N} \phi_j(S_i) w_{j,q}^{(l)}}{\sum_{j=1}^{N} w_{j,q}^{(l)}} \tag{13}$$

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

$$\hat{D} = D_k \text{ if } P_{D_k}(S_i) = \max\{P_{D_1}(S_i), \ldots, P_{D_K}(S_i)\} \tag{14}$$

$$\hat{d} = d_m \text{ if } P_{d_m}(S_i) = \max\{P_{d_1}(S_i), \ldots, P_{d_M}(S_i)\} \tag{15}$$

$$\hat{l} = l_q \text{ if } P_{l_q}(S_i) = \max\{P_{l_1}(S_i), \ldots, P_{l_Q}(S_i)\} \tag{16}$$

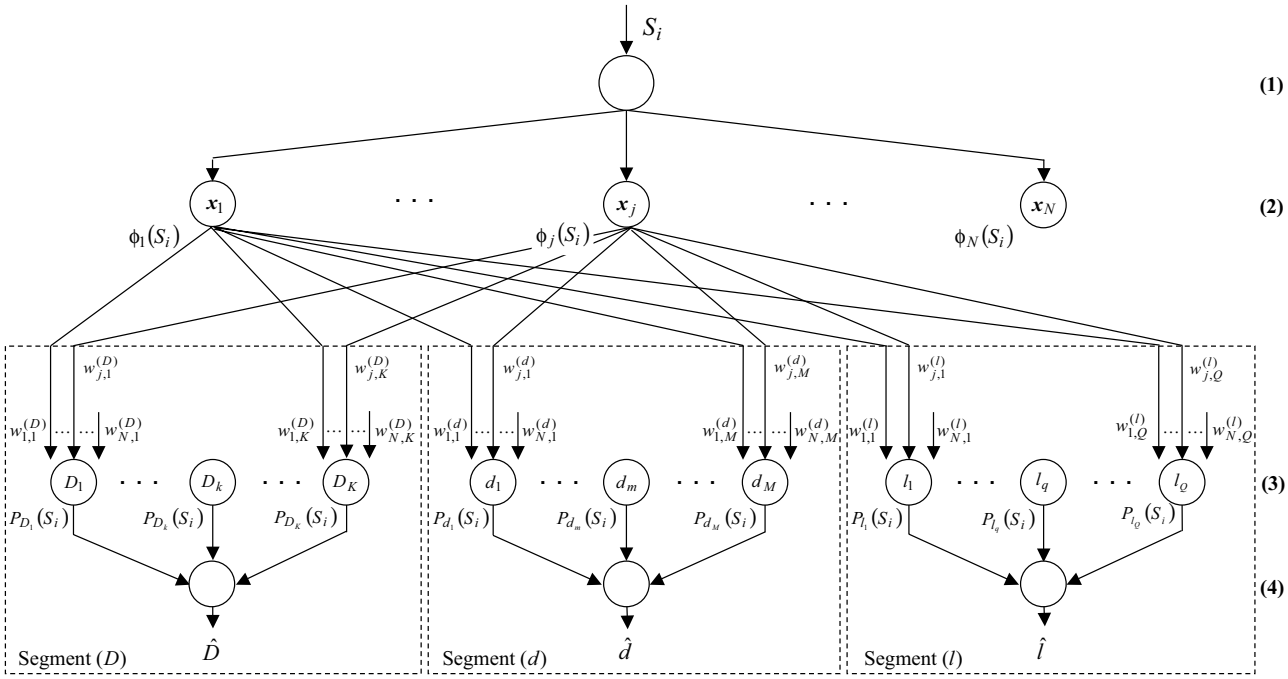
where $\hat{D}$, $\hat{d}$, and $\hat{l}$ denote the direction command, the distance command, and the actual distance travelled in response to the previous command associated with the nearest neighbour in the state space.

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 (e.g. distance to obstacles) are the same for all the members in a small neighbourhood, the sub-coach can use $\hat{D}$ as the actual direction command (D_i) suitable for the current state.

In contrast to the direction commands, the distance commands are fuzzy. They are, (e.g. *go very little, go , go little*, etc.). As explained earlier, the fuzzy command interpretation algorithm in this implementation uses the actual distance travelled in response to the previous command. What we obtain as $\hat{d}$ is the fuzzy distance associated with the nearest neighbour (e.g. *little*). However, the previous distance corresponding to the nearest neighbour and the current state cannot necessarily be equal. Therefore, $\hat{d}$ cannot be directly used for the current state.

To decide the correct distance command (d_i), the following algorithm is used.

IF $\hat{d} =$ *very little* THEN $d_i =$ *very little*



(1) Input Layer; (2) Pattern Layer; (3) Summation Layer; (4) Decision Layer

Fig. 3. Probabilistic neural network architecture (PNN).

ELSE IF $\hat{d} = little$ THEN
  IF $\hat{l} = low$ THEN
    IF $l_i = low$ THEN $d_i = little$
    ELSE $d_i = very\ little$
  ELSE IF $\hat{l} = medium1$ THEN
    IF $l_i = low\ or\ medium1$ THEN $d_i = little$
    ELSE $d_i = very\ little$
  ELSE IF $\hat{l} = medium2$ THEN
    IF $l_i = high$ THEN $d_i = very\ little$
    ELSE $d_i = little$
  ELSE IF $\hat{l} = high$ THEN $d_i = little$
ELSE IF $\hat{d} = medium$ THEN
  IF $\hat{l} = low$ THEN
    IF $l_i = low$ THEN $d_i = medium$
    ELSE $l_i = little$
  ELSE IF $\hat{l} = medium1$
    IF $l_i = low\ or\ medium1$ THEN $d_i = medium$
    ELSE $d_i = little$
  ELSE IF $\hat{l} = medium2$
    IF $l_i = high$ THEN $d_i = little$
    ELSE $d_i = medium$
  ELSE IF $\hat{l} = high$ THEN $d_i = medium$
ELSE IF $\hat{d} = far$
  IF $\hat{l} = low$ THEN
    IF $l_i = low$ THEN $d_i = far$
    ELSE $d_i = medium$
  ELSE IF $\hat{l} = medium1$
    IF $l_i = low\ or\ medium1$ THEN $d_i = far$
    ELSE $d_i = medium$
  ELSE IF $\hat{l} = medium2$
    IF $l_i = high$ THEN $d_i = medium$
    ELSE $d_i = far$
  ELSE IF $\hat{l} = high$ THEN $d_i = far$

The above algorithm can be explained as below.

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

For the current state also, the sub-coach needs to issue a similar command. For that, the only knowledge it has is the above fact. 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 too is *low*, then sub-coach can issue "*medium*" as the next command. However, if l_i is *medium 1, medium 2* or *high*, then it has to issue "*little*", because otherwise, the interpreted crisp distance will be more than 25 mm.

Once the sub-coach has been implemented and trained, test movements have been performed with the sub-coach controlling the robot. One of such test movements is shown in Fig. 4.

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

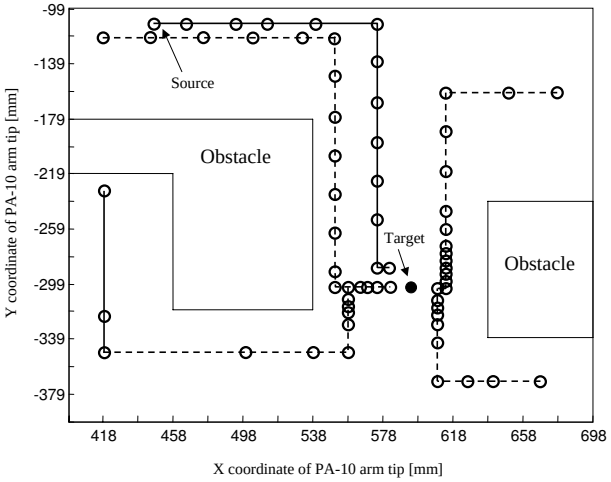


Fig. 4. A test movement.

a direction decision and a distance decision. It can be observed that the sub-coach alone could guide the robot arm tip to come very closer to the final target. For finer movements, the sub-coach can consult the human user.

Thus, we can see that it is possible to hand over coarse tasks to the sub-coach while finer tasks are performed by the human user, in a sophisticated environment. This can largely reduce the burden of a user who controls a robot with voice commands. On the other hand, a user can control more than one robots at the same time, just monitoring and helping them as needed.

6. Conclusion

The learning of sub-coach has been discussed in the framework of fuzzy coach-player model by applying a probabilistic neural network.

The sub-coach was trained with training movements covering different areas of the working space of the robot. The working space of the robot consisted of obstacles. 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 test movements, all the direction decisions and distance decisions were performed by the sub-coach without any intervention of a human. It was observed that the sub-coach alone could guide the robot arm tip to come very closer to the final target avoiding obstacles successfully. For finer movements, the sub-coach can consult the human user.

Thus, we can see that it is possible to hand over coarse tasks to the sub-coach while finer tasks are performed by the human user, in a sophisticated environment. This can largely reduce the burden of a user who controls a robot with voice commands. On the other hand, a user can control more than one robot at the same time, just monitoring and helping them as needed.

Using this model, it is possible to improve the usability of redundant manipulators by controlling them using fuzzy voice commands. Additionally, the same method may be suitably applied for other robotic systems too, though it was illustrated for a PA-10 redundant manipulator in this paper. The most important feature of the proposed method is that it utilizes the inherent fuzzy nature of spoken language commands to generate possible commands for unknown cases.

References

[1] P. Menzel and F. D'Aluisio, *Robosapiens*, The MIT Press, England, 2000.

[2] T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots," *Robotics and Autonomous Systems*, vol. 42, pp. 143–166, 2003.

[3] K. Pulasinghe, K. Watanabe, K. Kiguchi, and K. Izumi, "Voice controlled modular fuzzy neural controller with enhanced user autonomy," *Artificial Life and Robotics*, vol. 7, pp. 40–47, 2003.

[4] C. T. Lin and M. C. Kan, "Adaptive fuzzy command aquisition with reinforcement learning," *IEEE Trans. on Fuzzy Systems*, vol. 6, no. 1, pp. 102–121, Feb. 1998.

[5] K. Pulasinghe, K. Watanabe, K. Izumi, and K. Kiguchi, "A modular fuzzy-neuro controller driven by spoken language commands," *IEEE Trans. on Systems, Man, and Cybernetics-Part B:Cybernetics*, vol. 34, no. 1, pp. 293–302, 2004.

[6] K. Pulasinghe, K. Watanabe, K. Izumi, and K. Kiguchi, "Control of redundant manipulators by fuzzy linguistic commands," in *Proc. of the SICE Annual Conference 2003*, Fukui, Japan, Aug. 2003, pp. 2819–2824.

[7] C. Jayawardena, K. Watanabe, and K. Izumi, "Knowledge acquisition by a sub-coach in a coach-player system for controlling a robot," *The 4th International Conference on Advanced Mechatronics*, Hokkaido, Japan, Oct. 2004, unpublished.

[8] D. F. Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, no. 1, pp. 109–118, 1990.

[9] K. Z. Mao, K. C. Tan, and W. Ser, "Probabilistic neural-network structure determination for pattern classification," *IEEE Trans. on Neural Networks*, vol. 11, no. 4, pp. 1009–1016, July 2000.

[10] K. Pulasinghe, K. Watanabe, K. Kiguchi, and K. Izumi, "A noval modular neuro-fuzzy controller driven by natural language commands," in *Proc. of the 40th SICE Annual Conference*, 312C-4 (CD ROM), Nagoya, Japan, Jul. 2001.

[11] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. SMC-15, no. 1, pp. 116–132, 1985.