Artif Life Robotics (2003) 7:40–47
DOI 10.1007/s10015-003-0235-y

ORIGINAL ARTICLE

Koliya Pulasinghe · Keigo Watanabe · Kazuo Kiguchi
Kiyotaka Izumi

Voice-controlled modular fuzzy neural controller with enhanced user autonomy

Received and accepted: October 10, 2002

Abstract In this article, a fuzzy neural network (FNN)-based approach is presented to interpret imprecise natural language (NL) commands for controlling a machine. This system, (1) interprets fuzzy linguistic information in NL commands for machines, (2) introduces a methodology to implement the contextual meaning of NL commands, and (3) recognizes machine-sensitive words from the running utterances which consist of both in-vocabulary and out-of-vocabulary words. The system achieves these capabilities through a FNN, which is used to interpret fuzzy linguistic information, a hidden Markov model-based key-word spotting system, which is used to identify machine-sensitive words among unrestricted user utterances, and a possible framework to insert the contextual meaning of words into the knowledge base employed in the fuzzy reasoning process. The system is a complete system integration which converts imprecise NL command inputs into their corresponding output signals in order to control a machine. The performance of the system specifications is examined by navigating a mobile robot in real time by unconditional speech utterances.

Key words Fuzzy neural networks · Natural language commands · Key-word spotting · Contextual meaning of words

K. Pulasinghe (✉)
Faculty of Engineering Systems and Technology, Graduate School of Science and Engineering, Saga University, 1-Honjomachi, Saga 840-8502, Japan
Tel. +81-952-28-8602; Fax +81-952-28-8587
e-mail: koliya@ieee.org

K. Watanabe · K. Kiguchi · K. Izumi
Department of Advanced Systems Control Engineering, Saga University, Saga, Japan

This work was presented, in part, at the Seventh International Symposium on Artificial Life and Robotics, Oita, Japan, January 16–18, 2002

1 Introduction

Natural language (NL) is the most effective, efficient, and natural communication medium of human beings. This means an increasing demand for conversational interfaces (CIs) for the consumer electronics that are ubiquitous in everyday life. Among these, there is special interest in consumer electronics in the fields of the rehabilitation of handicapped persons (i.e., nursing),[1] working environments where both hands are busy (e.g., helpers), and toys for toddlers and older children (i.e., entertainment).[2,3] Moreover, speech signals contain not only the lexical content, but also the speaker's gender, emotions, and personality. This encourages the use of speech in NL-based systems (NLBSs) with different objectives. From the user's point of view, everyone is more or less equally skillful in the use of their mother tongue, although they have different skills in their professions, e.g., engineering, medicine, or politics. Therefore, machines driven by NL are free from technical complexities, and user training is very easy or not required. The downside is that NL-based interfaces are not persuasive in mission-critical control equipment which needs high precision, because poor signal quality, recognition delay, and possible missrecognition may badly affect the devices and their environment.

In the early days of voice-controlled machines, machine functions were activated by comparing the input user utterance with a stored template, as in the voice-controlled wheelchair developed by Mazo et al.[4] Each command was restricted to a few words, and had an associated function.[4,5] To control the machine, the user had to speak in-vocabulary words, i.e., a set of words which had been selected to activate the machine functions. Since robots have limited functionality, a fairly small vocabulary was enough to handle the command and control dialogues in human–robot interactions.[1] CI-based research grew rapidly after reliable and accurate speech recognizers were developed based on the hidden Markov model (HMM).[6] In addition, these methods needed to identify key words in running utterances in order to enhance speech-based machine control. This HMM-

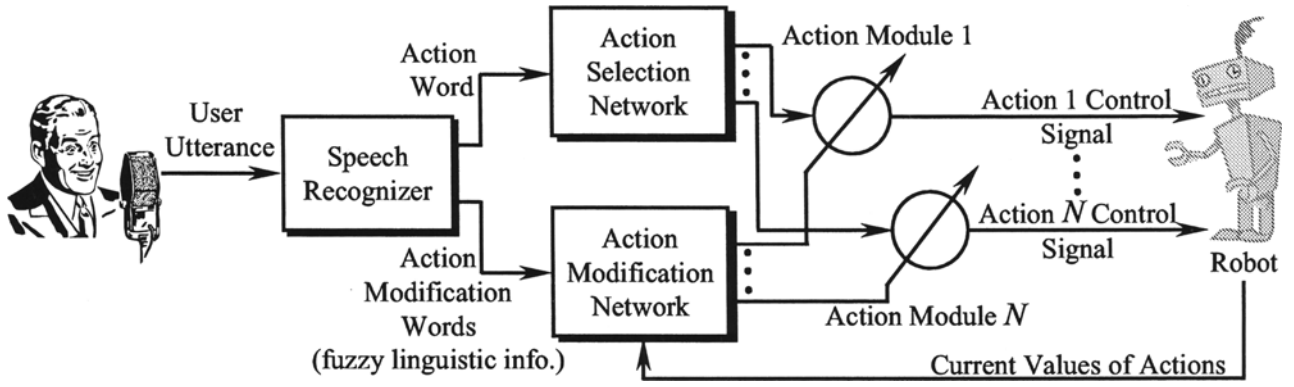


Fig. 1. The system architecture

based key-word spotting approach was first proposed by Rose and Paul[7] to solve the problems in the automated call routing systems in the telephone industry. Their discovery was followed by several improvements to the system, and the HMM-based approach was established as the most efficient method of key-word spotting.[8,9] Key-word spotting gave considerable freedom to the user when controlling machines using unrestricted user utterances.[10,11] NLBS has received much attention recently with the advent of pet robots for entertainment, e.g., AIBO and PaPeRo.[2,3]

Almost all the present day controllers (including the systems mentioned above) based on NL commands carry out the explicit semantic action requested by the user, but in natural conversations, the occurrence of words having fuzzy implications is inevitable. Words with fuzzy meanings are very important in machine control because they tune the performance of the robot's function. For example, "*move slowly*" is not only an activation command in robot navigation, but also emphasizes how the robot should navigate in a particular terrain. Current NLBSs are insensitive to these fuzzy implications. In addition, contextual meaning, i.e., the meaning of the voice command according to the current state of the machine, has a greater importance in voice-controlled systems. We present a FNN-based approach to controlling machines with imprecise NL commands and a concern for the contextual meaning of the commands. Moreover, this system is able to identify machine-sensitive words from the unrestricted user utterances.

In the rest of this article, a brief system overview is given in Sect. 2. The major components, i.e., the speech recognizer (SR), the action selection network (ASN), and the action modification network (AMN), with their characteristics, are described in Sects. 3, 4, and 5, respectively. A method of representing the contextual meaning of words is explained in Sect. 6. Finally, we discuss the experimental results in Sect. 7, and our conclusions and future direction are given in Sect. 8.

2 System overview

As illustrated in Fig. 1, the system captures the user's utterances by means of a microphone. The captured user utterance (speech signal) is processed in the SR in order to recognize the machine-sensitive words (pseudosentence) in the utterance, and to pass this pseudosentence to a perceptron-based artificial neural network (ANN) to decode it into an action word (the verbs) and action modification words (the adverbs). The nature of the action word (e.g., *move*) is definite or precise, but the action modification words (e.g., *very fast*) are not definite or precise.[10] Therefore, the system manipulates them in a different manner. The action word is fed into a perceptron-based ASN to initiate the prospective action. Action modification words together with the current machine status, are fed into the AMN, which modifies the operating behavior of the action initiated by the ASN. Each action is implemented as a separate module. These action modules emit activation signals for each particular function of the robot. The ASN initiates one of these actions at a time.

3 Speech recognizer

The SR is developed using a HMM toolkit, which is an integrated suite of software designed to build and manipulate continuous-density HMMs in order to building automatic speech recognition systems.[12] The hidden Markov model toolkit used in this design was developed at the Speech, Vision and Robotics Group of the Engineering Department, Cambridge University. The SR consists of two parts, as shown in Fig. 2. The key-word spotting module consists two parallel networks, as shown in Fig. 3. The outline of the function carried out by the key-word spotter is explained in the sample dialogue below.

User Robot, Can you **go very fast**
Robot You ask me to **go very fast**
User Yes/No

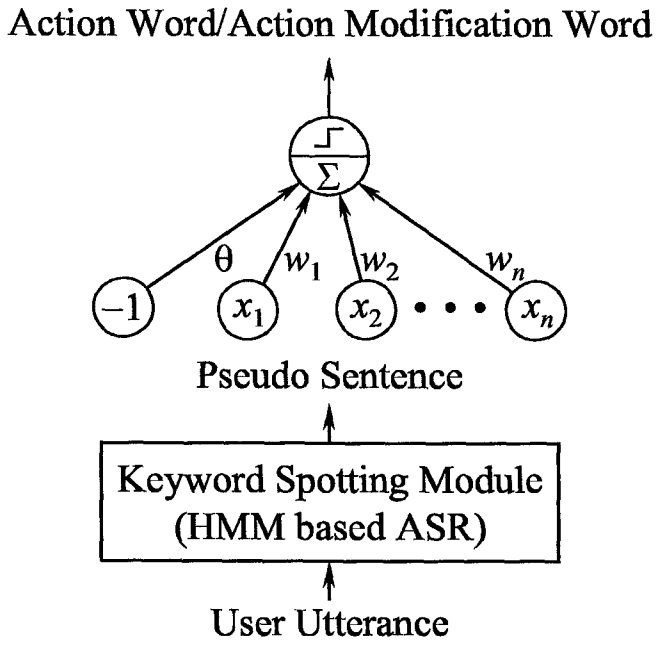


Fig. 2. The speech recognizer

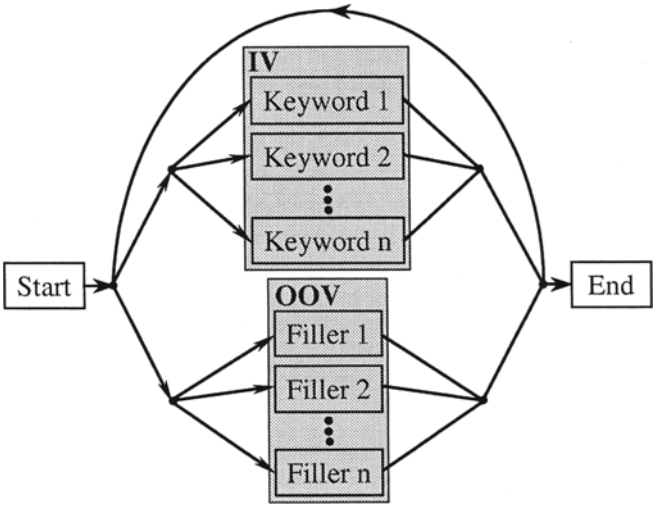


Fig. 3. The architecture of the key word spotter

The key-word spotter filters the machine-sensitive words "*go very fast*" from the complete user utterance "*Robot, Can you go very fast.*" This is achieved by implementing a filler network to identify the out-of-vocabulary (OOV) words as an alternative network to the baseline speech recognition (in-vocabulary or IV) network, as in Fig. 3. The pseudosentence extracted by the key-word spotter is transfered to the perceptron-based ANN for identification and classification. Here, we do not need any multilayered networks or radial basis function networks that perform a higher-order nonlinear mapping.[13] Each perceptron is trained to separate a particular word from the rest. If the word exists in the user command, the output of the perceptron is set to one. Otherwise, it is set to zero.

The learning and performing behavior of the perceptrons is described below using the notations depicted in Fig. 2, where w_i is the weight from the i-th input, and a threshold (or bias) at the output node, θ, is assigned to the input -1. Each word taken from the pseudosentence is presented to the ANN as its input. In the learning phase, the initial weights $w_i(0)$, $(1 \leq i \leq n)$ are set to small random values. At first, inputs $x_1, x_2, \ldots, x_n$ are presented to the network to calculate the outputs $y_i(t)$ as follows:

$$y_i(t) = f_h\left(\sum_{i=1}^{n} w_i(t)x_i(t) - \theta\right) \tag{1}$$

where

$$f_h(\alpha) = \begin{cases} +1 & \text{if } 0 \leq \alpha \\ -1 & \text{otherwise} \end{cases}$$

Then the training pairs, i.e., the continuously valued inputs $x_1, x_2, \ldots, x_n$, and the desired outputs $d_i(t)$ are presented to the ANN to update the weights according to the learning rule derived as in the literature,[13] such that

$$w_i(t+1) = w_i(t) + \eta\left[d_i(t) - y_i(t)\right]x_i(t) \tag{2}$$

with

$$d_i(t) = \begin{cases} +1 & \text{if input is the desired word} \\ -1 & \text{otherwise} \end{cases}$$

where η is a small positive learning rate, and $d_i(t)$ is the desired output for the current input. The weights are unchanged if the ANN makes the correct decision. After this learning phase, the ANN is employed in the SR to identify the presence of a particular word in the pseudosentence.

4 Action selection network (ASN)

The function of the ASN employed in the proposed system is to initiate the desired action recognized at the SR. Action words recognized at the SR are fed into the ASN, as illustrated in Fig. 4, to initiate the desired action. Owing to the existence of synonyms in NL, there may be different words which ignite the same machine function at this stage. Therefore, the number of action words in the vocabulary may be greater than the number of machine functions. For this reason, the ANN shown in Fig. 4 represents M number of action words, which ignite N number of machine actions, where $M \geq N$, and w_i^j denotes the weight from the i-th input word to the j-th output action. Thus, the ASN generates a binary output and ignites the desired output module to trigger the desired action. As an example, if the user wants to move the mobile robot in a forward direction, then they can use one of the words from the group "go," "run," and "move" to trigger the moving module by suppressing the other modules, and vice versa. The ASN is also an ANN consisting of N perceptrons, where each perceptron is

trained by using the same methodology as explained for the ANN at the SR.

5 Action modification network (AMN)

The inherent properties of FNN controllers, i.e., their ability to manipulate imprecise data, naturally persuade us to select one as the controller when designing a machine control system driven by NL.[14,15] The system proposed here uses the linguistic rules of fuzzy algorithms to understand the user's wishes, which come in the form of adverbs/fuzzy predicates (FPs) of the NL. The architecture of the proposed AMN is illustrated in Fig. 5. According to fuzzy-reasoning terminology, layers A–C represent the **antecedent** part of the FNN, whereas layers E–F represent the **consequence** part. As shown in Fig. 5, the domain of discourse of action 1 is described by the fuzzy variable A^1 with q linguistic values, and that of action N is described by the fuzzy variable A^N with t linguistic values. Similarly, the FP associated with action 1, FP^1, is composed of r FPs, and that of FP^N is composed of s FPs. Thus, each action is unique in the sense of its domain of discourse and the FPs associated with that particular action. It is assumed that each node of the same layer has a similar function, as described below. Here, we denote the output of the i-th node in layer X as $O_{X,i}$.

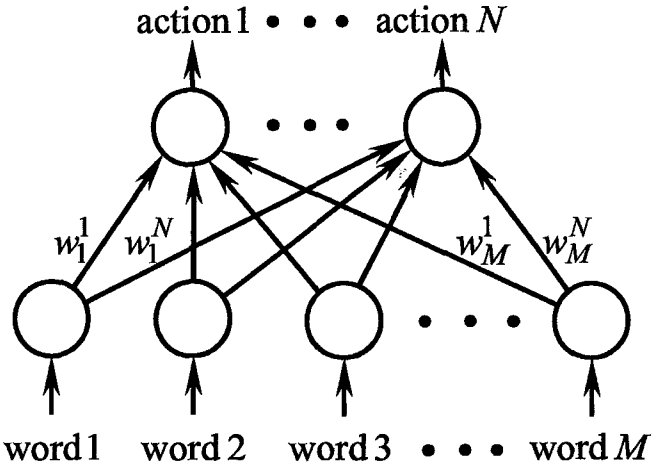
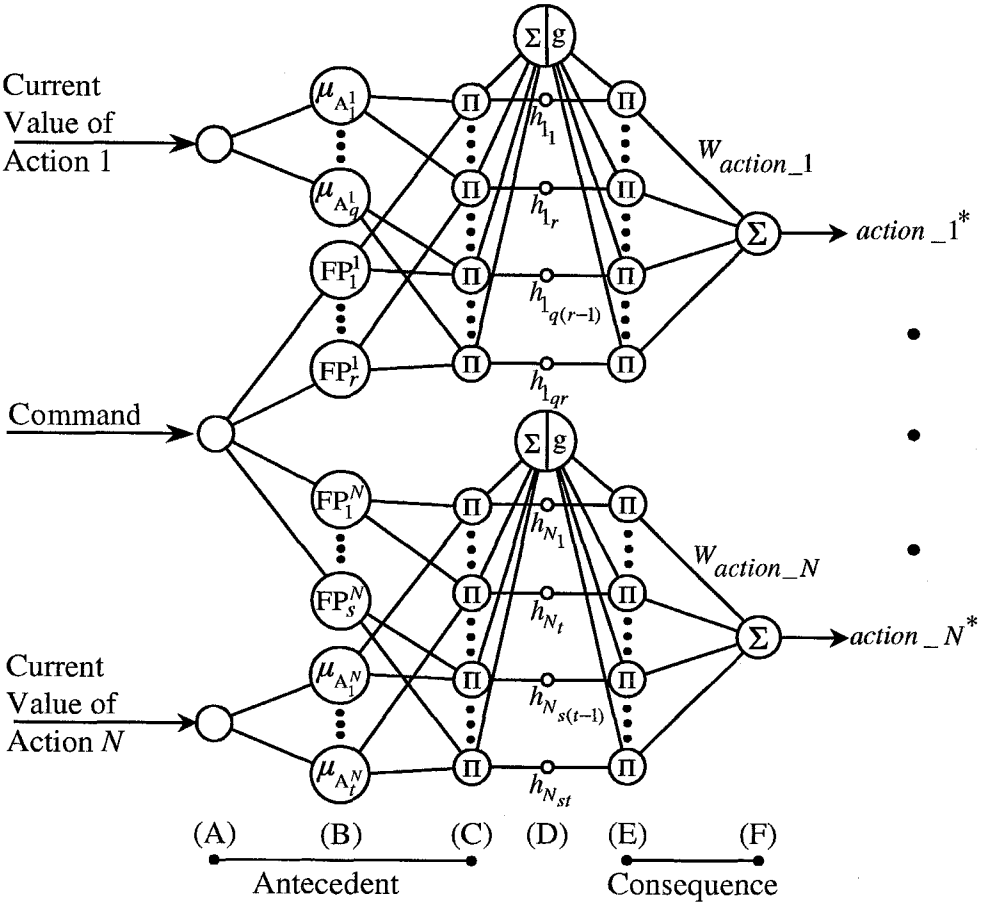


Fig. 4. The action selection network

5.1 Layer A

Layer A consists of two types of nodes: one is for command nodes to represent the availability of FPs in the pseudosentence, and the other is for the different actions of the machine. Each action is labeled as current value of action k, where $k = 1, \ldots, N$. It is assumed that the current

Fig. 5. Action modification network



value of action k, i.e., the crisp input to the k-th action node, is x_k. No computation is carried out at this layer. This layer takes the current values of all machine actions and the FPs of the pseudosentence simply acts as a distribution layer for the current operating state of the machine and user requests.

5.2 Layer B

Layer B acts as the fuzzification layer of the FNN. In this layer, the output of a node connected to the current value of action k acquires the fuzzy membership value of the universe of discourse.

Suppose the input to any p-th node, where $p = 1, \ldots, P$, of the current value of the k-th action is x_k, and A_p^k is the p-th *linguistic value* of the *linguistic variable* A^k associated with action k. Then the output of the p-th node is given by

$$O_{B,p} = \mu_{A_p^k}(x_k) \tag{3}$$

Similarly, the outputs of the FP nodes connected to the command node assign their value to 1 (like a *fuzzy singleton*), depending on their existence in the pseudosentence. Suppose that FP_q^k denotes the q-th FP, where $q = 1, \ldots, Q$, associated with the k-th action fired by the pseudosentence. Then the output of the q-th node connected to the FPs of the k-th action can be expressed as

$$O_{B,q} = FP_q^k = \begin{cases} 1 & \text{if FP exists in the command} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

5.3 Layer C

In this layer, the nodes labeled II compute the T-norm of the antecedent part. Although there are several methods to compute the T-norm, the algebraic product of the incoming signals, denoted by "*" is applied in the proposed system. The output of any node t, where $t = 1, \ldots, T$, in this layer is described by the equation

$$O_{C,t} = h_{kt} = \mu_{A_p^k}(x_k) * FP_q^k \tag{5}$$

where h_{kt} represents the firing strength of the t-th rule of the k-th action. Note that there are T rules associated with this action, where $T = P \times Q$.

5.4 Layer D

The first node of layer D at each FNN, which has symbols Σ and g, generates the output through the function

$$g(x) = \frac{1}{x} \tag{6}$$

with a linear summed input. Then the output of the first node of action K is given by

$$O_{D,1} = \frac{1}{\sum_{t=1}^{T} h_{kt}} \tag{7}$$

Other nodes simply carry forward the outputs of previous nodes to the next layer.

5.5 Layer E

Each node labeled II in this layer multiplies the value carried forward by previous node with the output of the first node at layer D. Then the output of any j-th node of this layer can be given by the equation

$$O_{E,j} = \frac{h_{kj}}{\sum_{t=1}^{T} h_{kt}} \tag{8}$$

5.6 Layer F

Layer F is the defuzzification layer of the FNN. The node labeled Σ in this layer calculates the overall output, i.e., the quantified performance value for the k-th desired action, as given below

$$O_{F,k} = \text{action\_}k^* = \frac{\sum_{j=1}^{T} h_{kj} w_{kj}}{\sum_{i=1}^{T} h_{ki}} \tag{9}$$

where w_{kj} denotes a constant value in the consequence part of the j-th rule for the k-th action. The overall output is the weighted mean of w_{kj} with respect to the weight h_{kj}, i.e., the firing strength of rule j.

5.7 Training the AMN

The connection weights are trained by applying the back-propagation algorithm. The adaptation process is illustrated in Fig. 6.

As shown in Fig. 6, the error is calculated by comparing the output of the expert knowledge with that of FNN for the same input data, x. The adaptation of the j-th weight of the k-th action, w_{kj}, at the l-th time-step is given by the equation

$$w_{kj}(l+1) = w_{kj}(l) + \gamma[y_d - y_a] \frac{h_{kj}}{\sum_{i=1}^{T} h_{ki}} \tag{10}$$

where γ represents a small positive learning rate, and y_d and y_a represent the desired output and actual output, respectively, for the k-th action selected for the training.

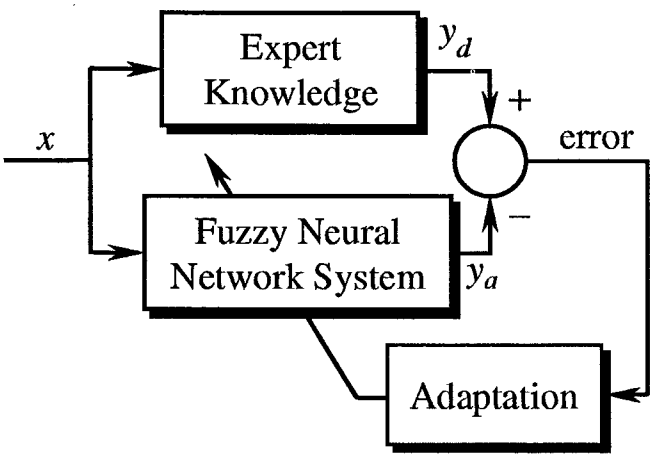


Fig. 6. Scheme for the adaptation of AMN

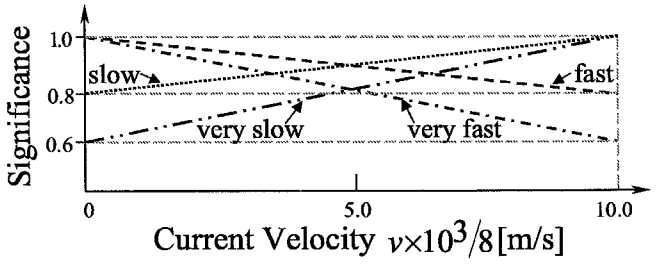


Fig. 7. Contextual meaning of fuzzy predicates of the command "move"

6 Contextual meaning of words

The mathematical implementation of the contextual meaning (significance) of words is very difficult, because it largely depends on the user's state of mind and the situation. However, it must be taken into consideration because machine control is strongly affected by the user's commands and the current situation. As an example, when a machine reaches its maximum speed, the user may command it to increase its speed. At that time, the machine's reaction should be completely different from the situation where the user asks the machine to increase its velocity while it is running at a very low speed. Figure 7 illustrates the method we introduced to absorb the contextual meaning of words. According to Fig. 7, the FP "very fast" diminishes its significance or meaningfulness when the machine arrives at its maximum speed. At that time, the user's request to "go very fast" has no meaning, i.e., its contextual value is very low. This implementation is included in the knowledge base of the fuzzy-reasoning process.

7 Experimental results

The above concepts have been applied to navigate a mobile robot, Khepera, in real time. The experiments were implemented piecewise to give a clearer picture. The output of the key-word-spotting module in the SR for user utterances is given below

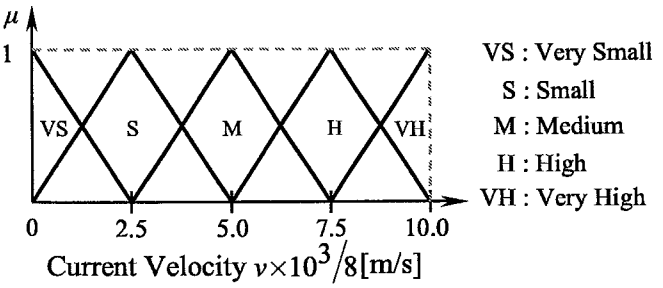


Fig. 8. Membership functions of the current velocity

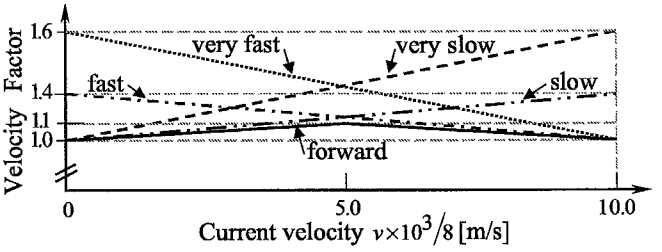


Fig. 9. Factors for the desired velocity

User	robot, go very fast
Recognizer	FILLER GO VERY FAST
User	please turn left
Recognizer	FILLER TURN LEFT
User	can you turn right
Recognizer	FILLER FILLER TURN RIGHT

The above "FILLER" occurrences are then filtered out for further processing. The alternative network with one filler module has been trained to identify the words "can, I, please, robot, you, want" from the key-words "backward, fast, forward, go, left, move, right, slow, stop, to, turn, very."

The proposed system comprises two action modules, called a turning module and a moving module. The words connected with the turning function, i.e., "turn left," "turn right," and "turn backward," are very precise, although the words connected with the moving function, i.e., "very fast" and "very slow," are imprecise. The AMN was designed to interpret these imprecise words for the moving module. The membership functions for the fuzzy variable "velocity" are shown in Fig. 8. The input space was limited to a maximum speed of 10 pulses/0.01 s (0.08 m/s) because the area of the working environment was 0.9 m × 0.9 m. The FNN was trained to achieve expert knowledge, i.e., the desired velocity in our example, which is ascertained by multiplying the current velocity by the velocity factor at a speed derived from the graph shown in Fig. 9, e.g.,

$$\textit{Desired velocity} = \textit{Velocity factor} \times \textit{Current velocity} \quad (11)$$

The significance of the words was taken into consideration in the design process of the velocity factors.

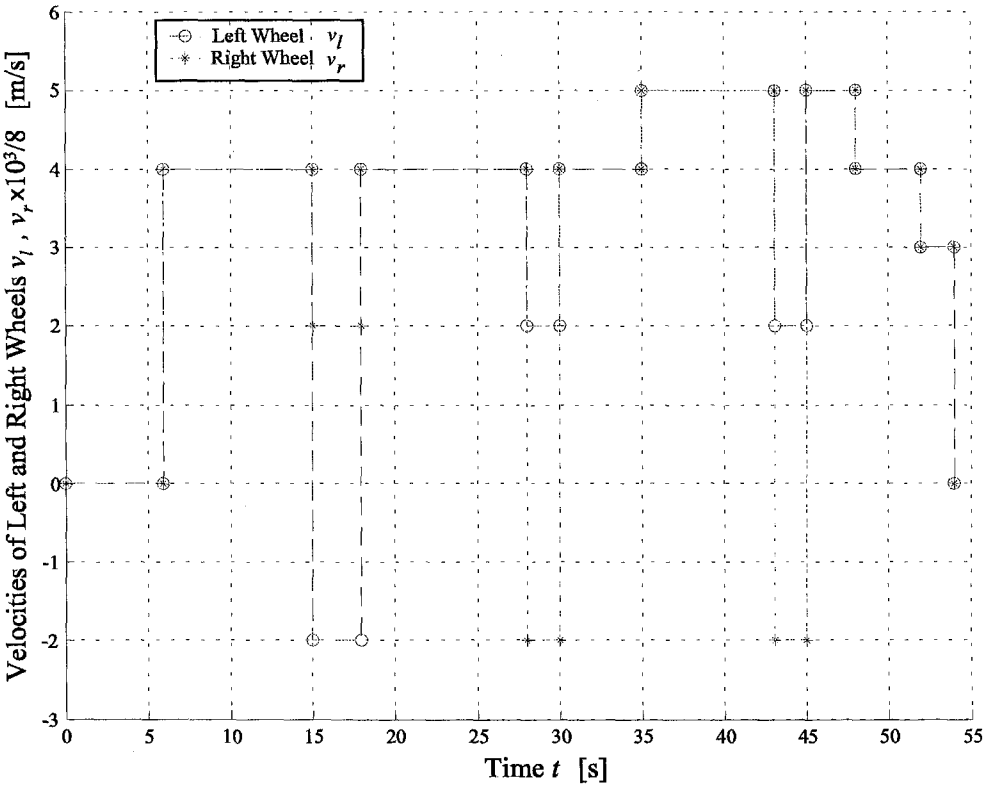


Fig. 10. Velocity profile of the two wheels of Khepera

The training process was terminated when the error reached 0.01, and the weights were initialized to small random in the range $-0.5 \leq w_{kj} \leq 0.5$. The learning rate, γ was assigned to 0.1, as in the ANNs. The velocity profile shown in Fig. 10 illustrates the velocities of the left and right wheels of its 54-s navigation for the commands "robot, go very fast," "please turn left," "can you turn right," "go very fast," "please turn right," "robot go slow," "go slow," and "stop." The velocity values evaluated at the FNN are rounded off to the nearest integer value because velocity commands to Khepera should be in integer format, as shown in Fig. 10.

8 Conclusions and future directions

Our system has been shown to be capable of handling fuzzy linguistic information in the user's commands by ignoring redundant words, which makes an environment which is conducive to natural and flexible conversation, and is sensitive to contextual meaning in the natural language.

The proposed system is static, which means it does not adapt to new words. Since machines are fundamentally limited to the services they can perform, we can make the in-vocabulary words include all the words used to describe those services. Therefore, the designer has to collect a large corpus of user expressions to create the in-vocabulary words used for the semantic actions of the machine functions and the fuzzy predicates attached to those semantic actions. Careful investigations should be carried out to select the key-words, because the use of words is affected by gender, age, and even social background.

Human–human interaction is not comparable with the present system. Humans update their vocabulary dynamically, learn meanings which are both general and contextual, integrate conversations with gesture, etc. These are the future directions of this research work.

References

1. Roy N, Baltus G, Fox D, et al. (2000) Towards personal service robots for the elderly. Proceedings of the Workshop on Interactive Robots and the Entertainment 2000 (WIRE). Pittsburgh, USA
2. Sony Electronics, USA. AIBO Homepage [online]. Available http://www.us.aibo.com/ers_210/index.php
3. NEC Corporation, Japan. "Personal Robot PaPeRo [online]. Available http://www.incx.nec.co.jp/robot/PaPeRo/english/p_index.html
4. Mazo M, Rodrìguez FJ, Làzaro JL, et al. (1995) Electronic control of a wheelchair guided by voice commands. Control Eng Pract 3:665–674
5. Komiya K, Morita K, Kagekawa K, et al. (2000) Guidance of a wheelchair by voice. Proceedings of IECON 2000. Nagoya, Japan, pp 102–107
6. Rabin LR, Juang BH (1986) An introducton to hidden Markov models. IEEE Trans Acoustic Speech, and Signal Processing 3(1): Jan, 1986, pp 4–16
7. Rose RC, Paul DB (1990) A hidden Markov model-based keyword recognition system. Proceedings of IEEE ICASSP '90. Albuquerque, USA, pp 129–132
8. Jeanrenaud P, Ng K, Siu M, et al. (1993) Phonetic-based word spotter: various configurations and application to event spotting. Proceedings of EUROSPEECH '93. Berlin, Germany, pp 1057–1060

9. Bazzi I, Glass JR (2000) Modeling out-of-vocabulary words for robust speech recognition. Proceedings of ICSLP 2000, Beijing. China
10. Pulasinghe K, Watanabe K, Kiguchi K, et al. (2001) Modular fuzzy neural controller driven by voice commands. Proceedings of ICCAS 2001. Cheju, Korea, pp 194–197
11. Sugisaka M, Fan X (2001) Control of a welfare life robot guided by voice commands. Proceedings of ICCAS 2001. Cheju, Korea, pp 390–393
12. Young SJ (1993) The HTK hidden Markov model toolkit: design and philosophy. Tech Rep TR.153, Department of Engineering, Cambridge University, Cambridge
13. Haykin S (1994) Neural networks: a comprehensive foundation. Macmillan, New York
14. Lin CT, Kan MC (1998) Adaptive fuzzy command acquisition with reinforcement learning. IEEE Trans Fuzzy Systems 6(1):Feb, 1998, pp 102–121
15. Jang JSR, Sun CT (1995) Neuro-fuzzy modeling and control. Proceedings of IEEE 83(3):Mar, 1995, pp 378–406