ELSEVIER

Robotics and Autonomous Systems 37 (2001) 25–51

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Perception and remembrance of the environment during real-time navigation of a mobile robot

K. Madhava Krishna, Prem K. Kalra*

Department of Electrical Engineering, Indian Institute of Technology, Kanpur 208016, India

Received 28 August 2000; received in revised form 20 June 2001

Communicated by T.C. Henderson

Abstract

This paper deals with the advantages of incorporating cognition and remembrance capabilities in a sensor-based real-time navigation algorithm. The specific features of the algorithm apart from real-time collision avoidance include spatial comprehension of the local scenario of the robot, remembrance and recollection of such comprehended scenarios and temporal correlation of similar scenarios witnessed during different instants of navigation. These features enhance the robot's performance by providing for a memory-based reasoning whereby the robot's forthcoming decisions are also affected by its previous experiences during the navigation apart from the current range inputs. The environment of the robot is modeled by classifying temporal sequences of spatial sensory patterns. A fuzzy classification scheme coupled to Kohonen's self-organizing map and fuzzy ART network determines this classification. A detailed comparison of the present method with other recent approaches in the specific case of local minimum detection and avoidance is also presented. As for escaping the local minimum barrier is concerned this paper divulges a new system of rules that lead to shorter paths than the other methods. The method has been tested in concave, maze-like, unstructured and altered environments and its efficacy established. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Memory; Mobile robot; Real-time navigation; Local minimum; Self-organizing map (SOM); Spatio-temporal reasoning; Fuzzy ART

1. Introduction

Although there exist numerous approaches that deal with real-time navigation of a mobile robot using range sensors such as fuzzy logic approaches [1–3], heuristics [4], wall following [5,6], force field [7,19] and neural network methods [8–10] most of them do not discuss strategies for incorporating memory or remembrance properties. A navigating robot must not only avoid obstacles based on the range input but also comprehend the nature of its environment, remember over time such comprehended scenarios, recollect them and associate in time perceptions of environment that resemble each other. Such requirements demand spatial and temporal reasoning capabilities. Considering the environment of the robot as an experience of a sequence of sensor patterns; this paper discusses a method that reduces to order such experiences by a classification scheme to achieve the necessary spatial and temporal reasoning properties.

The classification scheme consists of a fuzzy rule base that maps the instantaneous range inputs into a set of classes coupled to a SOM [11] and fuzzy ART network [12] that learns a sequence of such classes ordered in time. The SOM models the experience of spatio-temporal patterns in terms of landmarks learned offline while ART imparts plasticity by learn-

* Corresponding author.
E-mail addresses: kkrishna@iitk.ac.in (K. Madhava Krishna), kalra@iitk.ac.in (P.K. Kalra).

0921-8890/01/$ – see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0921-8890(01)00150-6

ing and storing new patterns during real-time that are not among the ones modeled by the SOM. Spatial reasoning is provided by the weight vectors of the neuron that are codified representations of the local space of the robot. Storing the lattice positions of the weight vectors during a traversal imparts memory, temporal ordering of the lattice position facilitates recollection while temporal association occurs through recall when perception of a similar environment maps to the same neuron.

Such spatio-temporal reasoning embellishes the performance of a mobile robot through more appropriate paths, detecting local minimum situations and scene recollection properties. This paper compares in detail the performance of recent approaches proposed to circumvent the local minimum problem [4,13–16]. Certain limitations are found in the virtual target approach [15,16] when the robot reverts into the same limit cycle which it tries to overcome for a class of environments and a modification scheme is suggested using a subgoal stack to overcome this limitation. The paper also reasons that the proposed method for overcoming the local minimum gives shorter paths than the other algorithms from the instant the local minimum is detected. The convergence of the algorithm in reaching the target from the instant of local minimum detection is ascertained.

The paper is organized as follows. Section 2 discusses the classification scheme for incorporating cognition and remembrance properties. Section 3 discusses through graphical simulations the advantages of using this classification scheme in maze-like, changed and irregular environments. Section 4 does the comparative study of the recent algorithms proposed in the literature for overcoming the local minimum barrier and Section 5 winds up the paper with concluding remarks.

2. Cognition and remembrance for the navigation algorithm

To possess cognition features the robot must become aware of the kind of environment it observes while navigating to a certain degree. In vision-based systems a single snapshot can capture the necessary information required for understanding the nature of the environment present in the neighborhood. On the contrary sensor-based systems require a sequence of sensor samples of the environment before an understanding of the environment can be attempted. For example to understand its passage through a narrow corridor the robot needs a minimum number of sensor samples for reaching such an opinion. At the instant when it just enters the corridor (Fig. 1(a)), it can only understand the presence of an obstacle on its left and right at *near* distances from the sensor readings (near is a typical fuzzy variable used in sensor-based navigation systems [1–3]). However over a few samples (Fig. 1(b)) the robot can reach a fair conclusion about its passage through a corridor. If the corridor leads to an opening on the left (Fig. 1(c)) the robot should get aware of it. If it leads to a blind end (Fig. 1(d)) it must arrive at that conclusion just as a human would, through the samples it had seen so far. Hence the robot experiences a corridor though the temporal sequence of sensor patterns and reducing to order such experiences by an appropriate classification scheme the robot can become cognizant of its passage through a corridor leading to a blind end.

2.1. Fuzzy classifier

In the above example the robot's experience of the corridor is represented through the vector $[u_s(t), u_s(t+1), \ldots, u_s(t+n-1)]$ where $u_s(t)$ denotes the sensor sample at the instant it entered the corridor and $u_s(t+n-1)$ is the sample of the environment when it reached the dead end. Any sample of the environment is given by $u_s = [u_0, u_1, \ldots, u_6]$ where each u_i, $i = \{0, 1, \ldots, 6\}$, is the range reading of an ultrasonic sensor during that sample. Seven such sensors are placed in the form of an arc on the circumference of the robot subtending an angle of 90° at the center of the robot. Two issues become prominent here.

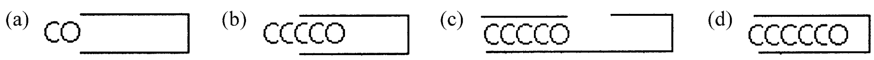


Fig. 1. (a) Robot enters a corridor; (b) after a few instants; (c) corridor with a slit in its left; (d) meets a dead end.

1. The space complexity involved in storing sequences of sensor samples where each sample is a seven-dimensional vector of range readings.
2. How many '*n*', of such samples shall represent a landmark. In other words is there a way to fix the upper bound on *n*.

To reduce the space complexity and to facilitate fast learning of sample sequences by the SOM and ART a fuzzy classifier is used that maps each sample u_s to a particular class. The SOM and the ART can be trained on a sequence of one-dimensional classes rather than a sequence of seven-dimensional sensor samples. Initially the seven sensors are clubbed into three groups namely left, center and right. The left group consists of sensor 4 down to 6, center group represented by sensor 3 and right group by sensors 0 to 2. The minimum reading of the sensors in each group is considered as the reading of that group. The fuzzy classifier classifies the readings of such groups into one of the nine classes as shown in Table 1. The mathematical formulations of the classification scheme are not considered here for brevity. Evidently there can be other ways of classifying the range reading to classes. The reason to have specified the range readings in terms of far, near and very near is essentially due to the same kind of partitioning employed in the fuzzification part of the inference scheme for collision avoidance. Thus the fuzzified range readings become an end product for further inferencing leading to collision avoidance as well as spatio-temporal classification responsible for imparting cognition and remembrance features. Increasing the number of classes, it is felt, may not affect in a significant manner the performance of the algorithm since most of the typical indoor landmarks get represented through a sequence of these nine classes (Table 1). There can also be other ways of grouping the seven sensors into the three groups. The suggestion by one of the referees to have the center group reading as the maximum range of sensors 2, 3 and 4 is found useful in filtering variations in range readings on certain occasions.

Table 1
Spatial classification of the range readings by fuzzy rule base

Right sensor	Center sensor	Left sensor	Class
Very near	Very near	Very near	0
Near	Near	Near	1
Near	Near	Far	2
Near	Far	Near	3
Near	Far	Far	4
Far	Near	Near	5
Far	Near	Far	6
Far	Far	Near	7
Far	Far	Far	8

The second issue is regarding the upper bound on the number of classes (since each sample is reduced to a class) that represents a landmark. Through practical experience it has been surmised that the change in the sequence of the classes holds the key to identify typical landmarks more than the sequence of classes itself. The change in the class sequence physically represents what can be called as an 'observation event'. The detection of the observation event is crucial for the robot to make an adequately intelligent decision. This will be further elaborated in Section 3. In the corridor example (Fig. 1(d)) the observation event occurs when the sequence of classes changes from 3 to 1 to 0 represented as [3, 1, 0]. The change represents the termination of the corridor into a blind end, which occurs due to the change in sensor readings when the robot nears the blind end. Similarly the corridor in Fig. 1(c) is identified through the sequence of changes [3, 4, 8]. Based on this observation event the robot becomes aware of its traversal through a narrow corridor opening to freespace on its left. The point to be noted is that no two successive classes in the sequence of classes will be the same. Since it is the changes in the sequence that is of primary concern than the number of classes in the sequence it is significant that all such dead ended corridors can be depicted by [3, 1, 0] irrespective of the size of the corridor. Through experience it has been observed that the typical landmarks (Fig. 2) encountered in an indoor environment can be represented through not more than two changes in sequence. The upper bound on the number of classes is thus fixed at 3 irrespective of the nature of the landmark. It is noteworthy that a landmark experienced through a very long sequence of spatial sensor patterns can be cognized through a sequence of only three classes leading to considerable reduction in memory. A sequence of three classes can be learned and identified easily by SOM and ART in reasonable time whereas an increase in their number

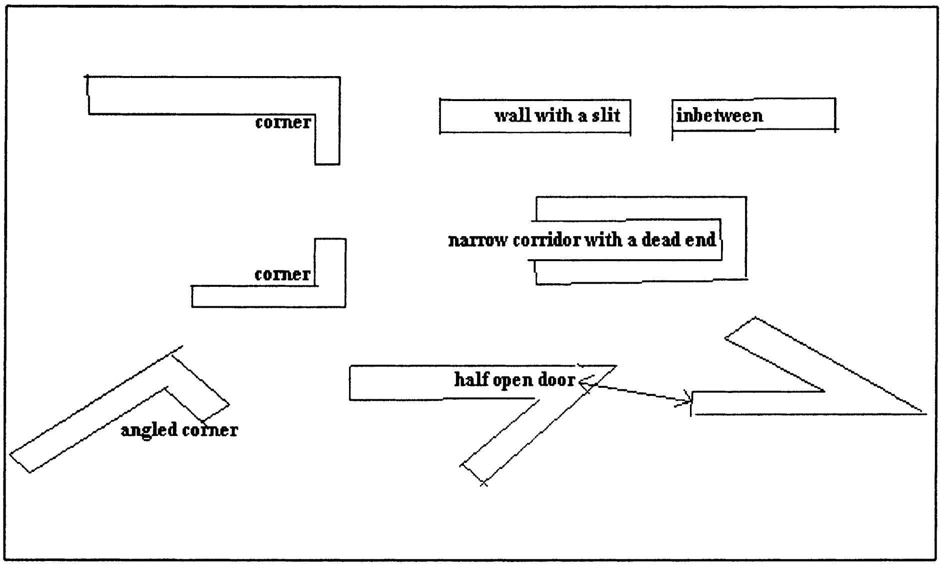


Fig. 2. Some typical landmarks.

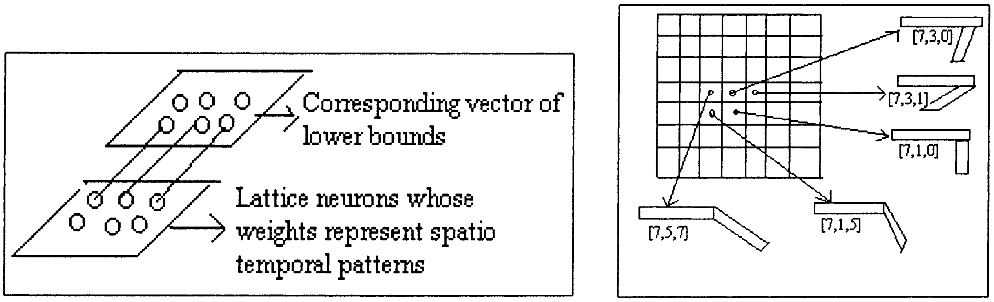
requires a large number of neurons to capture the information and an expensive training time.

Though the length of each class in a sequence need not be known for landmark recognition the minimum number of times a particular class must occur in a sequence to prevent misidentification of stray or noisy sequences to a landmark must be known. In case of the dead ended corridor the first class '3' must occur at the least four times, the second class twice and the third once. This minimum bound on the number of times each of the three classes in a sequence needs to occur is another three-dimensional sequence called the vector of lower bounds.

To summarize a landmark is modeled as an experience of spatio-temporal patterns. Each sample of such a pattern consists of range readings obtained by the seven sensors. For reducing complexity of storage, representation and learning a long sequence of such samples each sample is mapped to a class by the fuzzy classifier. The number of such classes in a temporal sequence is fixed at 3 with no two consecutive classes being identical. The minimum number of times each class must occur in a sequence to filter noisy samples is termed the lower bound for that class.

2.2. Classification by SOM and fuzzy ART

The SOM is trained for the typical landmarks that are expected to occur in a general environment. The robot navigates across various landmarks such as those shown in Fig. 2 using a fuzzy algorithm. The range samples encountered during the traversal are classified by the fuzzy classifier, grouped to form the temporal sequence of classes as described in Section 2.1 and stored. The SOM is a lattice of 7×7 neurons that are initialized to normalized random values. A sequence of classes is picked up in random from the stored sequences of classes and presented to the network. The winner is determined and the winner and the other neurons are updated according to the standard update rule with a Gaussian neighborhood function [17]. The number of iterations should be of two orders more than the total number of neurons in the lattice [17] for the SOM to capture the essential topology of the input space. Once the SOM is trained for various landmarks the robot navigates across a particular landmark with reduced sizes. The number of occurrences of each class in the sequence of the smallest landmark the SOM is able to recall is stored as a neuron whose weights represent the lower bounds. The neuron is stored in the position corresponding to



(a) (b)

Fig. 3. (a) Lattice neurons with their vector of lower bounds; (b) mapping of the landmarks by SOM through its weight vectors.

the lattice position of the winning neuron of the SOM (Fig. 3(a)). Fig. 3(b) shows the weight vectors and their corresponding landmarks after learning. Similar landmarks get mapped to neighboring positions in the lattice.

After such training the robot navigates in a real-time workspace. On encountering a sequence of patterns that represents a particular landmark the neuron in the lattice is triggered and the robot becomes aware or cognizant of the presence of such a landmark in its vicinity. Thus SOM imparts spatial cognition features to the robot.

However all the experiences of spatio-temporal patterns cannot be modeled through the landmarks learnt offline. Under some situations this can result in the robot not getting aware of its trapped condition in a local minimum discussed in Section 3. A provision for online learning and classification can hurdle over such local minimum barriers. Fuzzy ART with complement coding scheme serves as a good alternative. The ART network can dynamically add new patterns to its knowledge base and can afford to forget those patterns that do not occur frequently. As the robot navigates the sensor samples are classified by the fuzzy classifier and the temporal order of classes extracted. The temporal sequence is the input to the ART network, which either maps the sequence with an earlier one or adds a new pattern to the existing set of patterns. These decisions are governed by an appropriate choice of the vigilance parameter [12].

The classification scheme discussed in this section attempts to impart spatio-temporal reasoning abilities to the robot by reducing to order the robot's experience of the environment through the sequences of sensor samples. The efficacy and various concomitant utilities of such a classification scheme is presented for static environments in Section 3.

3. Simulation

The mobile robot navigates its environment based on certain fuzzy rules that operate on the sensory input space of the robot. The input space, U, of the mobile robot can be represented as

$$u = [u_{\mathrm{s}}, u_{\mathrm{df}}]^{\mathrm{T}}, \quad u \in U,$$

where u_{s} represents the sensory input space of seven sensors. Input u_{df} represents the angular difference between the mobile robot's instantaneous direction vector and the vector joining the robot's center to the target, also called the difference angle.

The fuzzy rule base acts on the input space, U through two primary modules, the goal reaching module and the obstacle avoidance module to output a motion command to the robot. The goal reaching module imparts a command that orients the robot towards the target and the obstacle avoidance module turns the robot in such a way to prevent collisions. Both these modules guide the robot to its destination avoiding the obstacles. The purely fuzzy algorithm performs in a densely cluttered environment (Fig. 4) but fails in environments that require spatial and temporal reasoning capabilities. In the simulations the robot has been modeled as a circle of radius 5 pixels that rotates about its center and is aware of only its start and final positions. Each of the range sensors transmits a conical beam of width 10°. The least

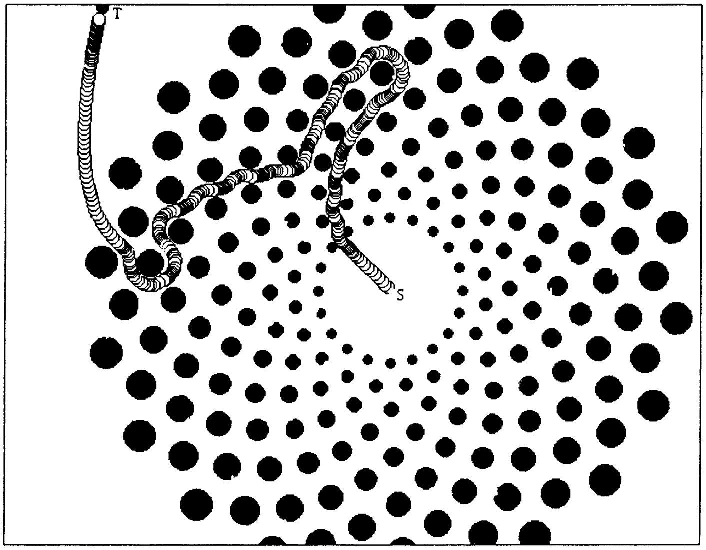


Fig. 4. Navigation in a densely cluttered environment.

distance obtained within the cone becomes the range input to the algorithm due to that sensor. The main issue in sensor modeling here is the correspondence between the range measure and the actual position of the object. A typical ultrasonic sensor returns a radial measure *d* of the distance to the nearest object within its cone, but the angular location of the object is not known. Hence it is not possible to pinpoint the exact location of the sensed portion of the object on a world map. Moreover there are the associated problems of specular reflection and sensor misreading that further makes it difficult to locate the object. Reliable ways of overcoming these issues to a reasonable extent involves mainly the certainty grid of Elfes [20] and its derivative histogram grid due to Borenstein and Koren [7,19]. These methods project the range readings onto a map divided into cells. The certainty values of the cells are updated with repetitive sampling of the object and high certainty values are obtained in cells close to the actual location of the object. In the histogram grid method the cell corresponding to the reading *d* is assumed to lie on the sensor's acoustic axis. The location of the object on a map is required to be known to derive the attractive and repulsive forces that drive the robot, which are calculated based on these locations.

However fuzzy logic approaches typical of reactive navigation operate directly on sonar readings. Exact or even reasonable estimates of the positions of an object on a map are not required. Based on the range readings and the target location the fuzzy rules guide the robot towards the target. Inaccuracies in sensor readings can be minimized without the need for representing the readings on a map. The two stage process of filtering the moderate and major variations in range readings discussed in Section 3.7 seem suitable for tackling most of these variations. The subsequent parts of this section portray the advantages of incorporating spatial and temporal reasoning properties in the real-time navigation algorithm. These form the novel aspects of this work.

3.1. Recent memory through spatial reasoning

Intelligent decisions are a natural consequence of possessing reasoning and cognition properties. It is an integral part of human navigation. A purely fuzzy-based algorithm comprising of only the two primary modules is unable to functionally replicate such reasoning as far as navigational aspects are concerned. By incorporating a SOM some of these drawbacks are alleviated. Figs. 5 and 6 show the navigation of a robot across a right-angled corner. Fig. 5 is the path obtained through the fuzzy algorithm without the classification scheme incorporated. Along the side

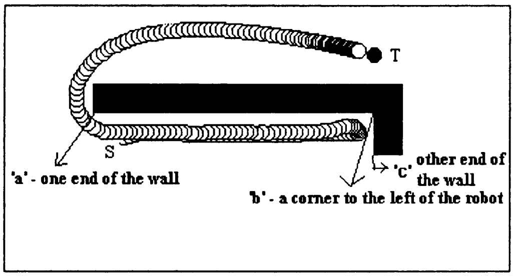


Fig. 5. Path traversed by the robot without spatial reasoning. S and T are the start and target locations, respectively.

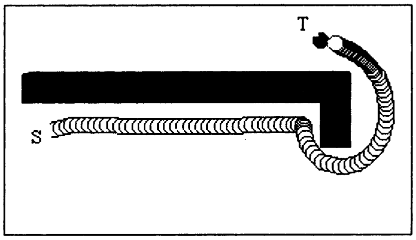


Fig. 6. Path tracked through spatial understanding.

'ab' of the wall the target attracting module tends to turn the robot to its left while the obstacle avoidance module turns the robot to its right. The action spaces of both the modules offset each other and the robot follows the wall till it arrives at the corner 'b'. At the corner the readings of the left and right group of sensors become comparable and the obstacle avoidance module does not give a prominent turning behavior on either directions. The turning propensity of the robot however is to its left due to the presence of the target on the left. On turning left the robot meets the same wall again resulting in a path where the robot travels the length 'ab' of the wall twice. The robot seems to have forgotten that it had been seeing the wall on its left all this while only to turn left biasing its decision based entirely on the most recent range readings. In other words the robot lacks an ability to reason about its environment.

Fig. 6 is the path tracked when the SOM is embedded in the navigation algorithm. As the robot approaches the corner 'b' an observation event occurs through a change in the sequence classes from 7 to 1 to 0. This triggers the neuron in the SOM lattice with the weight vector [7, 1, 0], which is essentially the coded representation of the right-angled corner experienced by the robot. Upon such cognition of the environment appropriate decisions by the algorithm lead to a shorter path (Fig. 6). This is termed as the recent memory divulged by the SOM by which the robot becomes cognizant of its latest experience of the environment. Such spatial cognition or reasoning ability imparted to the algorithm is a specific consequence of the SOM and as discussed in Section 3.3.1 is a provision that the ART is not capable of providing.

3.2. Recollection of cognized scenarios by temporal ordering

One of the typical features of human navigation is their ability to recollect their experiences during a traversal. This involves spatial cognition at an instant followed by memorizing of such cognized scenarios to facilitate recollection or recall later. Such features can possibly have diverse advantages. A few of those possibilities are explored in this section when such capabilities are transferred to the algorithm.

In our algorithm storing the lattice positions of the winning neurons imparts memory while recording the temporal order of the lattice positions of the winning neurons enables recollection. Table 2 gives in temporal order the winning weight vectors of the spatio-temporal patterns and their corresponding lower bounds for the path tracked by the robot in Fig. 6. The fourth column of the same table shows

Table 2
Temporal order of the winning weight vectors, the lower bounds and lattice positions during a navigation

Temporal order of occurrence in navigation	Weight vector of spatio-temporal pattern	Corresponding vector of lower bounds	Lattice positionin a 7×7 lattice
1	(8, 7, 1)	(1, 4, 2)	(6, 5)
2	(7, 1, 0)	(2, 1, 4)	(6, 2)
3	(1, 0, 8)	(1, 4, 1)	(1, 7)

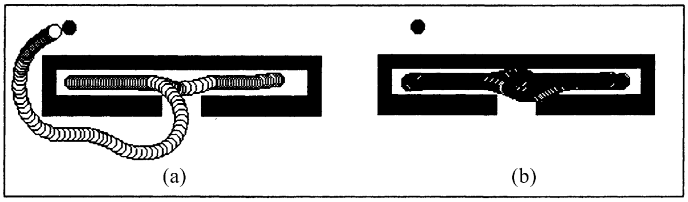
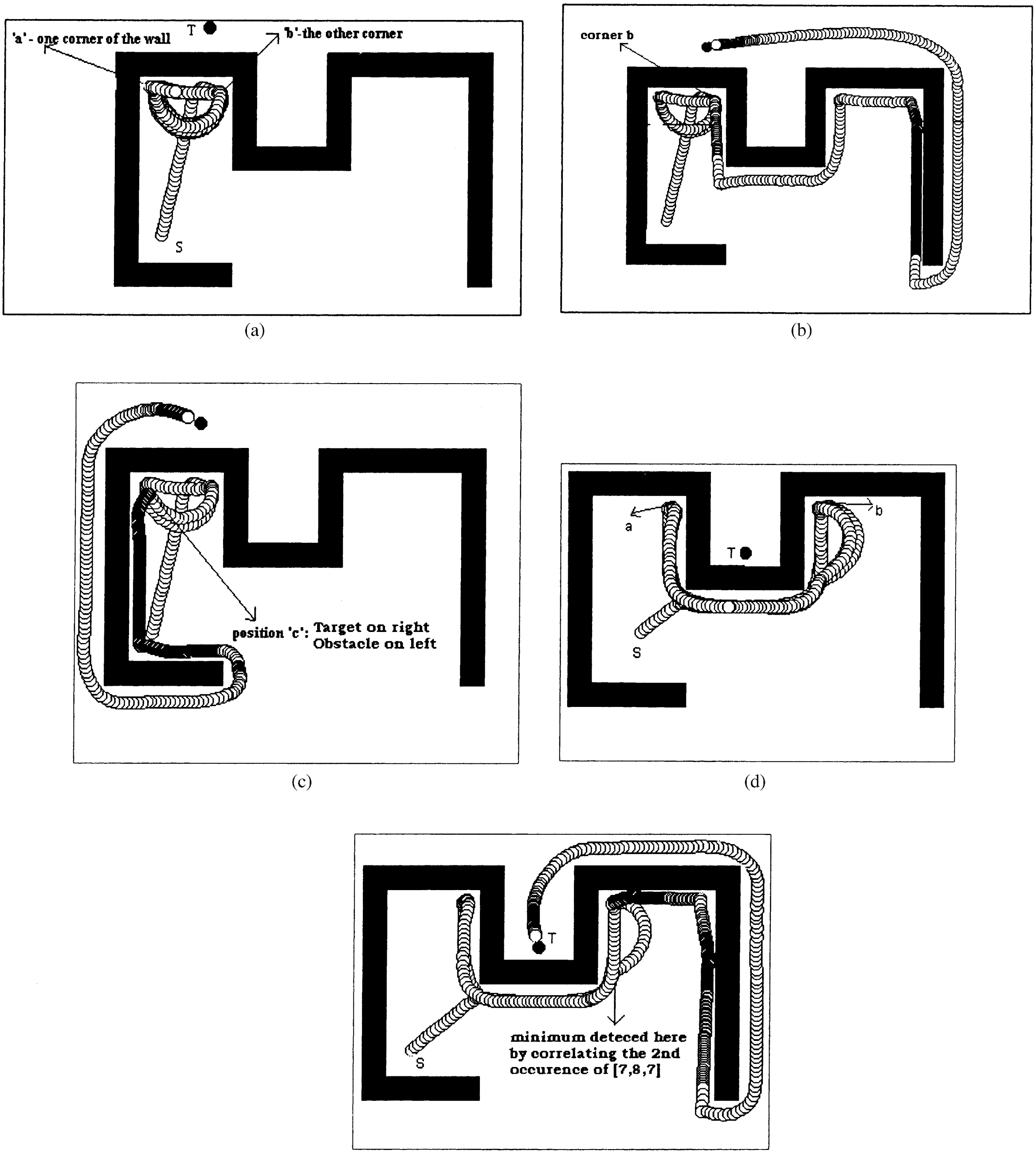


Fig. 7. (a) Path resulting by reclaiming cognition of the dead end; (b) path in the absence of recollection property.

the lattice positions of the winning neurons of a 7×7 lattice. Based on this temporal ordering the robot can visualize the history of its navigation by retrieving the sequence of landmarks. This reordering captures functionally to some extent the way humans can recount their journey as "*I first started moving towards the target until I met a wall. I turned such that the wall came on my left. It was long and ended in a corner bending towards the right. I took a turn at that corner and the wall still continued on my left till I reached the target*". Such recollection can have significant advantages towards more sensible decisions. Fig. 7(a) shows the path tracked by the robot to reach a target on the other side of the corridor with a narrow slit. The robot begins from S and moves towards the target. When it reaches the slit the target attracting and obstacle repulsing modules act in such a way the robot does not come out of the slit but proceeds further along the corridor. When it reaches the dead end on the other side it retraces its path to reach the slit again. Now spatial cognition followed by memorizing and retrieval leads to the following reasoning, "*I started moving through the target. The passage was through a narrow corridor. I passed across a slit and reached the other end of the corridor. I turned around and moved in a direction opposite to my earlier direction when I came across the slit. Because I know that I am on my way back having met a dead end I take a turn at the slit and plan my path again*". Here the robot makes an intelligent guess to get out of the corridor through the slit by reclaiming cognition of its encounter of a dead end earlier. Functionally this reasoning is simulated by suitable commands to the robot when the temporal order of the winning lattice positions stored in a queue matches a traversal that leads to a blind end followed by an encounter of a slit. In the absence of such spatio-temporal reasoning the robot gets into a local minimum loop shown in Fig. 7(b) due to the nature of the fuzzy algorithm. Detection of local minimum by temporal correlation is described elaborately in Section 3.3. *Care should be taken to note that in this instance the robot actually* **did not detect** *a local minimum* by correlating similar experiences. For on the return journey the robot experiences a slit on its right, which is different from the experience of a slit on its left during the onward journey. Rather the decision was motivated by recollecting the cognition of the blind end on the side closest to the target.

3.3. Local minimum detection by temporal association of perceptions

An inherent consequence of recollection is temporal correlation or temporal association. Temporal recollection involves recalling the scenarios cognized in past from memory. Recognizing the similarity between a currently perceived scenario with a one perceived earlier by recollecting from memory is termed as correlation or association. This is realized in the algorithm when a neuron that won previously during a traversal wins again. By temporally associating perceptions of a similar environment the algorithm detects a local minimum or a limit cycle path [14–16]. Fig. 8(a) shows an example of a local minimum loop between the two corners 'a' and 'b' which occurs due to the contradicting nature of the two primary modules in a concave or maze-like environments. Though there exists algorithms [4,13–16] that surmount the local minimum the uniqueness of the present approach lies in its recognition of its trapped condition by correlating similar experiences of the environment. In Fig. 8(b) when the robot cognizes the corner at 'b' for the second time it associates in time the cognition of a similar corner previously and finding its spatial coordinates at both the instants to be roughly the same



(a) (b) (c) (d) (e)

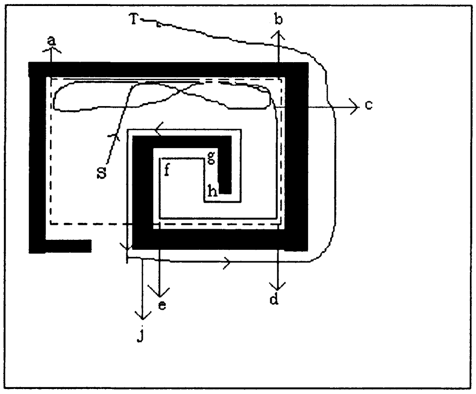
Fig. 8. (a) A local minimum path; (b) temporally associating the two perceptions of the corner 'b' detects the local minimum; (c) fuzzy ART detects local minima ([7, 8, 5]) earlier than SOM; (d) another infinite loop. Robot oscillating between 'a' and 'b', (e) robot guided out of the oscillations.

understands its trapped state. Suitable rules are executed to steer the robot from the trap. The sequence of steps that lead to recovery from local minimum are described later in this section. It would be argued in Section 4 that the paths traced by the robot through these recovery rules are shorter than those traced in other algorithms. Before discussing the nature of the recovery rules it is to be mentioned the algorithm maintains a queue that records the lattice position of the winning neuron and the coordinates of the robot at the instant the robot cognizes a landmark. The queue is allocated and maintained dynamically. At any instant it stores a maximum of 20 coordinates and lattice positions in temporal order. Whenever a landmark is cognized the algorithm checks for a previous occurrence of the landmark through the lattice positions stored in the queue. Occurrence of the same lattice position in the queue indicates the experience of a similar landmark. If the corresponding coordinates at those lattice positions match an encounter with the same landmark is implied and local minimum is detected. A similar landmark is discerned from the same landmark by comparing the coordinates of the robot at both the instants from the queue.

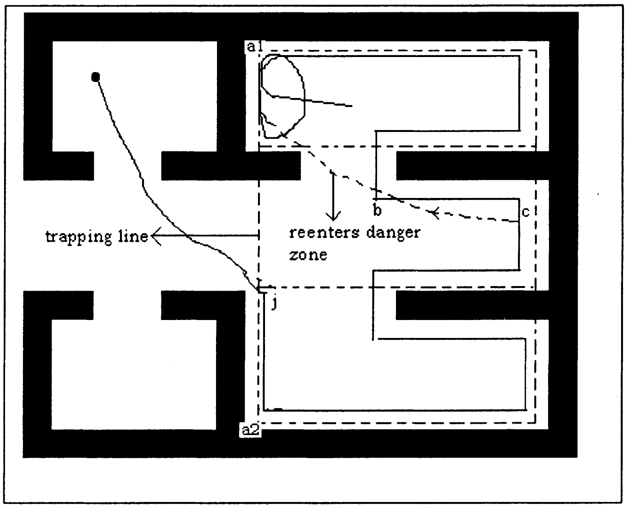
3.3.1. The recovery rules

The steps are explained with the help of Fig. 9(a) and 9(b), which can be considered a very general and a difficult case for recovery from local minimum. The path the robot would trace is only sketched and not simulated in these cases to highlight certain important parts of the figures. The algorithm also records the coordinates of the robot at those instances a corner or a sharp bend in the obstacle was cognized between the two similar perceptions that lead to the detection of the minimum. In Fig. 9(a) between two perceptions of corner at ‘b’ the robot cognizes corner ‘a’. The coordinates of the robot at ‘a’ and ‘b’ are noted. The following rules are then executed. An analysis of the rules is given subsequently.

1. The orientation of the target and the nearest obstacle with respect to the robot when the minimum is detected are recorded. (At ‘b’ the target and obstacle are on the left of the robot.)
   1.1. If the target and obstacle are on the same side of the robot as in Fig. 9(a) fuzzy rules guide the robot until the target and obstacle fall on either sides of the robot (position ‘c’ in Fig. 9). Algorithm branches to step 2.



(a)



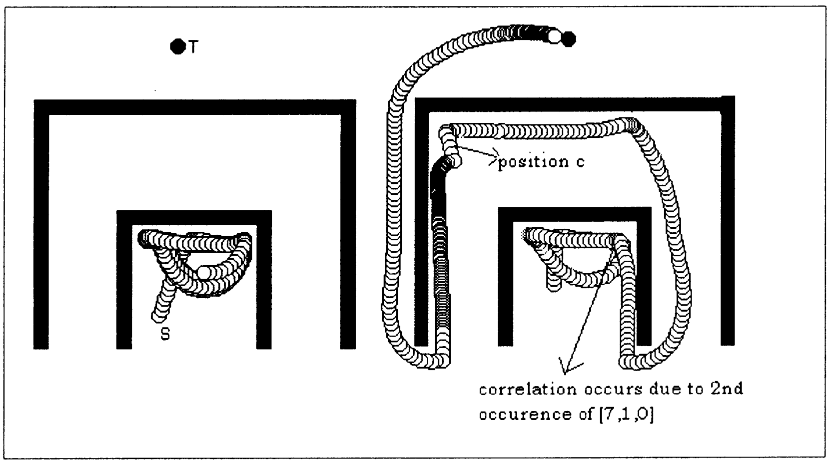
(b)

Fig. 9. (a) Sketch showing the path tracked by the robot for analysis of recovery rules. The robot leaves wall following at position ‘j’ in its path shown by an arrow. (b) If obstacle following were not continued form ‘c’ the robot shall re-enter the danger zone through the dashed curve shown. The robot crosses the TL ‘a1 a2’ at ‘j’.

   1.2. If the target and obstacle are on either sides the robot rotates such that the target and obstacle swap sides. At position ‘c’ in Fig. 8(c), target is on the right and obstacle on the left when minimum is detected, the robot turns such that target comes on left and obstacle on the right. Algorithm branches to step 2.
2. The robot tracks the silhouette of the obstacle until it reaches a location outside the bounding rectangle. At this location the robot evaluates if the target and the obstacle lie on its same side. If they are the robot slips into the rule base mode else it continues contour following until the target and the obstacle appear on its same side. This happens at position ‘j’ in Fig. 9(a); the bounding rectangle is shown in dashed lines. The construction of the rectangle will be dealt subsequently.
3. From this instant (position ‘j’ in Fig. 9(a)) fuzzy rules guide the robot until:
   3.1. Target is reached. Process ends.
   3.2. An obstacle appears on the other side of the target (position ‘c’ in Fig. 9(b)), step 4 is executed.
4. The robot follows the obstacle till it reaches a position such that condition given in step 2 is executed. In this way steps 2–4 are executed until:
   4.1. Target is reached. Process ends.
   4.2. The robot comes on the other half-plane of the trapping line, that side of the trapping line which contains the target. In Fig. 9(b) the robot crosses the trapping line ‘a1 a2’ at position j to come over to that half-plane of the trapping line which contains the target. The algorithm branches to step 5. (The trapping line concept is dealt when the rules are analyzed.)
5. Once the robot crosses the trapping line the robot breaks free of the obstacle which it tracked and operates under fuzzy rules until:
   5.1. Target is reached. Process ends.
   5.2. While avoiding a new obstacle that is encountered the target and obstacle appear on either sides of the robot (position ‘c’ in Fig. 10(b) of the double walled obstacle). Algorithm branches to 2.

3.3.2. A retrospection of the recovery rules

Rule 1.1: The positions of the target and obstacle play an important role while robot tries to escape from the clutches of the local minimum. They are important considerations regarding when the robot can leave wall following while still under partial influence of the minimum. If the robot departs from wall following when the target and obstacle are on either sides it can rush to the target only to meet the same wall it tried to escape by following it. Hence the algorithm strictly observes wall following while the target and obstacle are on either sides and the robot is under the partial influence of the local minimum. The robot lies within



(a) (b)

Fig. 10. (a) Double walled obstacle; (b) correlation occurs due to second occurrence of [7, 1, 0].

the *partial influence of the local minimum* when it is on that half plane of the trapping line that does not contain the target. The robot is under the *complete influence of local minimum or danger zone* when it lies inside the bounding rectangle and simply follows the wall without any considerations regarding the positions of the target and obstacle relative to it.

Rule 1.2: This rule has no big impact on the algorithm excepting it can lead to shorter traversals when local minimum is detected earlier by fuzzy ART instead of SOM discussed in Section 3.3.3.

Rule 2: The bounding rectangle (BR) delineates the zone within which the robot is under the complete influence of the local minimum. As mentioned earlier a queue maintains the spatial positions of the robot whenever a landmark was cognized. From this queue the coordinates of the robot when it cognized a corner or an angled bend between the first cognition of the landmark whose second cognition lead to the discovery of the minimum and the cognition of an opening while wall following is noted. In Fig. 9(a) these are the coordinates of the positions of the robot when the landmarks ‘a’, ‘b’, ‘d’, ‘e’, ‘f’, ‘g’, and ‘h’ were recognized. Here ‘a’ was cognized between the two perceptions of ‘b’ and ‘h’ was recognized when the opening was detected. From these positions the maximum and minimum coordinates are computed which form the corners of the BR. Normally the robot comes out of the BR when it perceives an opening during the course of tracing the contour and turns around it. In some case such as in Fig. 9(a) the opening ‘h’ lies well inside the BR and the robot comes out of BR at ‘j’. Coming out of BR is a significant landmark since it signifies the end of the domain of that configuration of obstacles responsible for the minimum. Nonetheless wall following is not completely renounced for reasons mentioned below.

Rule 4: Here the robot comes out of the danger zone but is still within the partial influence of the minimum. Hence the constraint the robot switches to wall following when the target and obstacle come on either sides. For if it would continue to act through the fuzzy rules it can under certain conditions due to the target’s influence re-enter the same danger zone that it had escaped and relapse into the same local minimum path. This can continue ad-infinitum as shown in dashed lines in Fig. 9(b). In Fig. 9(b) at ‘b’ the robot comes out of the danger zone, target and obstacle lie on the same side and is guided by fuzzy rule base. At ‘c’ obstacle and target come on either sides. If wall following is not executed the robot re-enters the same danger zone and regresses into an infinite loop. It is also to be noted that the size of the rectangle keeps increasing and rule 2 operates such that the robot does not re-enter an area that it had already escaped.

The trapping line indicates the end of the influence of the configuration of obstacles that lead to the local minimum. Beyond this line the robot can navigate freely under the fuzzy rules without fear of relapse into the same loop it avoided.

Rule 5: Having crossed the trapping line the robot is fit to encounter a new local minimum. Nonetheless to prevent longer traversals a fresh local minimum path is avoided through the rule 5.1 which automatically slips to wall following when obstacle and target appear on either sides and once again the sequence of rules 2–5 are executed. This occurs at ‘c’ in Fig. 10(b) of the double walled obstacle.

Figs. 8, 10 and 11 depict the consistency of the algorithm in detecting local minimum type situations and the efficacy of the recovery rules in rescuing the robot from such traps. The infinite loop is shown in Figs. 8(a), 8(d), 10(a) and 11(a) where only the fuzzy algorithm is implemented. The guidance out of the loop when the local minimum is detected by the SOM is shown in Figs. 8(b), 8(e), 10(b) and 11(b). These figures also denote the instants during the robot’s journey when temporal correlation occurs through the weight vectors representative of the similar spatial scenarios that resulted in the correlation.

Fig. 11(a) shows a simulation where the robot is unable to come through an opening in the maze at location ‘c’ due to conflicting behaviors of the two modules. Fig. 11(b) illustrates circumvention of the trapped situation before temporal correlation occurs. This is due to the recent memory capacity of the SOM, which understands through the weight vectors the robot’s unduly long traversal through the maze and pulls it out through the hole in the maze (Fig. 11(b)).

3.3.3. *The role of fuzzy ART*

The robot understands its environment through the cognition properties imparted by SOM. In certain cases the robot can encounter an environment which

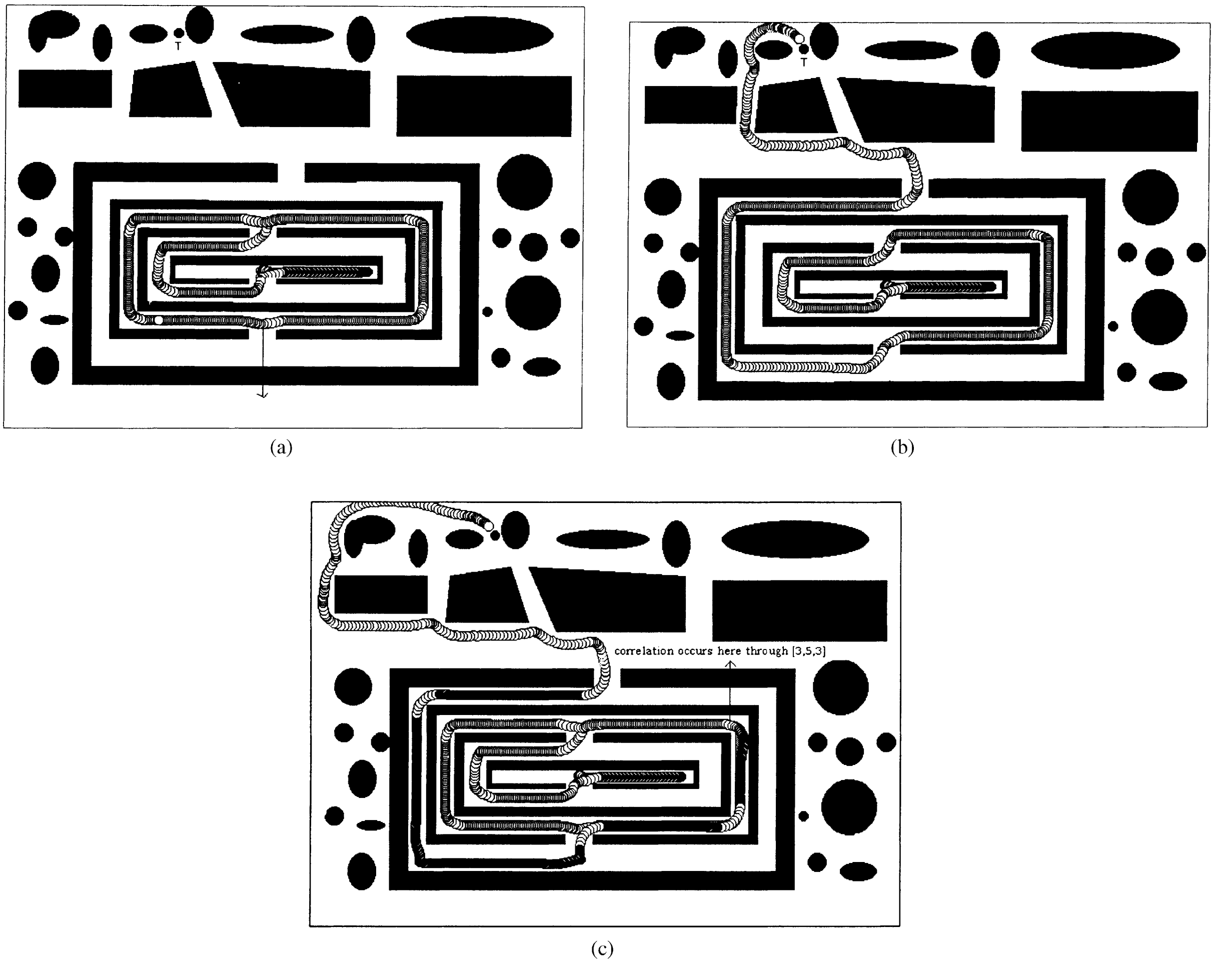
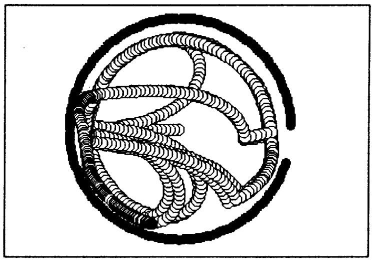
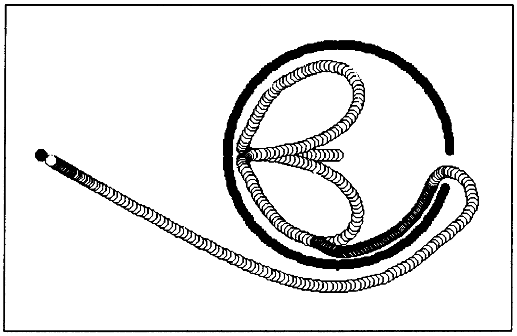


Fig. 11. (a) Robot unable to pull itself out because of conflicting target attracting and obstacle avoidance behaviors. (b) Upon cognizing the slit through [3, 4, 8] the robot pulls out of the maze. (c) Without SOM, fuzzy ART pulls the robot out of the maze. Absence of SOM is seen as robot completes a full traversal between the second and third loops from the center.

it is unable to map to a neuron in the SOM lattice. This can lead to a local minimum like situation shown in Fig. 12(a) that goes undetected. The ART network however is plastic to new situations through online learning and classification. Upon occurrence of a pattern that is not among the ones which can be perceived by SOM, the ART learns it, stores and recalls when the robot experiences a similar scene again. The ART network detects the local minimum not detected by SOM in Fig. 12(b). The ART can also detect local minimum ahead of SOM in certain cases as in Fig. 8(c), which at times can lead to a shorter path. However it must be noted while the ART can provide for memory, recollection and correlation it cannot impart spatial reasoning which is a privilege of the SOM alone. Fig. 11(c) is an example when the ART network alone is present and SOM is absent. Here absence of spatial reasoning leads to the robot completing a full traversal between the second and the third obstacle loops from the center. However due to temporal association property of ART local minimum is detected during the robots second visit between the obstacle loops and the recovery rules rescue the robot.



(a)



(b)

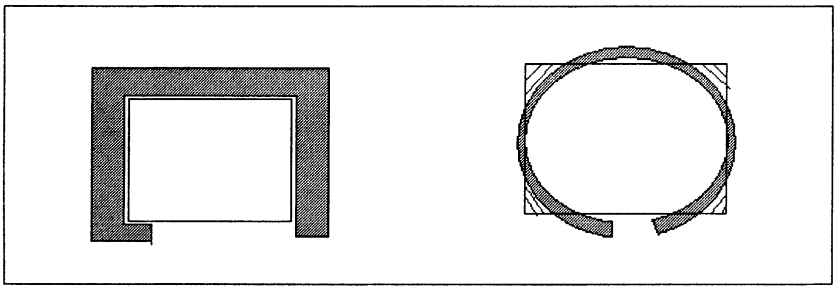
Fig. 12. (a) An environment where the SOM is unable to detect the local minima. (b) Fuzzy ART pulls the robot out of the local minima.

3.4. Convergence analysis of the recovery rules

The convergence of the algorithm is discussed based on the recovery rules that are activated once the local minimum is detected. In other words the concern here is to establish on a more formal basis the fidelity of the rules derived in Section 3.3.1 and analyzed in Section 3.3.2.

Theorem 1. *If a local minimum is detected and the target is reachable from the point of detection, the algorithm will leave the configuration of obstacles that caused the minimum after a finite length path.*

Proof. The robot leaves the obstacle once it gets out of the BR. The BR is constructed based on the locations visited by the robot where it experienced a sharp turning behavior (generally bends and corners). Since these locations lie inside that configuration of obstacles the BR is finite in size. Moreover the BR lies completely within the configuration of obstacles if they form a rectangular shape (Fig. 13(a)). Hence while following the contour the robot shall eventually reach a location that lies outside the BR. Once outside the BR the fuzzy rules and obstacle following work in tandem to navigate the robot past the trapping line. This way the robot overcomes a configuration of obstacles that caused the minimum. If the configuration does not constitute a rectangle some parts of the BR can project outside the configuration (Fig. 13(b)). However if the obstacles were of finite width these projections would be minimized. Even if the obstacles were of negligible thickness it is evident that contour following would eventually take the robot to a location outside the BR. In the extreme case the robot during the course of obstacle following would ultimately cross the trapping line (TL) which forms one of the sides of the BR. This is because *there exists at least one point on the TL that lies within or on the configuration of obstacles* for it is constructed based on positions that lie within the configuration. The robot while wall following would eventually come on the other side of this point. □



(a) (b)

Fig. 13. (a) For a configuration of objects that are rectangular the BR lies completely inside the configuration. (b) Parts of BR that project outside the configuration is shown shaded.

It is to be noted that to make sure that the leaving point does not lie inside the configuration of obstacles the robot continues wall following for a small distance after it detects its location outside of the BR. Once outside the configuration of obstacles that caused the minimum the recovery rules 3 and 4 guarantee that the robot does not re-enter the same local minimum it avoided. It has been shown before [13] that a leaving point that lies outside the configuration of obstacles along with the absence of a regress to the same local minimum path guarantees attainment of the target. Based on the satisfaction of the above conditions the second theorem can be stated, the proof for which can be found in the earlier approaches [4,13].

Theorem 2. *The algorithm finds the target if it is reachable from the position of cognizing the minimum.*

If the target is unreachable the algorithm can be terminated once a loop inscribing the configuration of obstacles is completed. This is the routine procedure adopted across all approaches. Hence it can be stated as a corollary:

Corollary. *The algorithm always terminates after a finite length path.*

Hence in principle the algorithm is capable of reaching the target if a local minimum is detected. In practice the algorithm's performance depends on a reasonable construction of the BR. While the construction of BR appears trivial in a structured environment a doubt may still linger regarding its construction in not so structured environments. The subsequent section deals with the performance of the algorithm in more unstructured environments.

3.5. Performance in less structured environments

Firstly it is to be clarified that the algorithm has been developed from the point of view of indoor real-time navigation. In *indoor* environments typical objects encountered as well as rooms, corridors have regular geometric shapes and are well structured. Secondly the local minimum situation occurs at bends or corners due to abrupt change in the turning behavior of the robot. The configuration of obstacles that actuate an abrupt turning behavior in a robot can be detected even if the environment is unstructured. Essentially this involves learning the temporal sequence of sensor patterns that indicate an abrupt turning behavior of the robot and identify such sequences during navigation. While in structured environments abrupt turning behavior occurs at right-angled corners and bends such as the meeting of two walls in unstructured environments it occurs when the bends do not form a perfect right angle and may possess a curved or irregular shape. For example the BR for an irregular environment is shown sketched in Fig. 14. The probable path of the robot is shown sketched with arrows indicating the direction of navigation. The obstacle is shown mirrored on one side. Fig. 15 portrays the simulation results in three environments that are not exactly structured. The consistency in detecting the turning behavior from sensor patterns is seen in the algorithm's ability to attain the target in these environments. The BR is also shown in the figures. In Fig. 15(a) irregularity can be seen in the bends that are not exactly at right angles and somewhat resemble the curved bends of the structure in Fig. 14. Figs. 15(b) and 15(c) portray the performance in the presence of some convex and circular objects along with the configuration of obstacles that constitute the minimum. When other objects lie in proximity to bends or corners the detection of turning behavior is more difficult to perceive. However the algorithm seems competent to counter such situations.

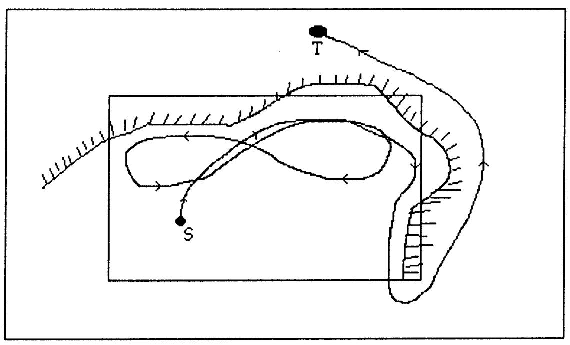


Fig. 14. BR for an unstructured environment. Obstacle contour is shown mirrored on one side. The path tracked by the object as well as the rectangle is shown in solid lines.

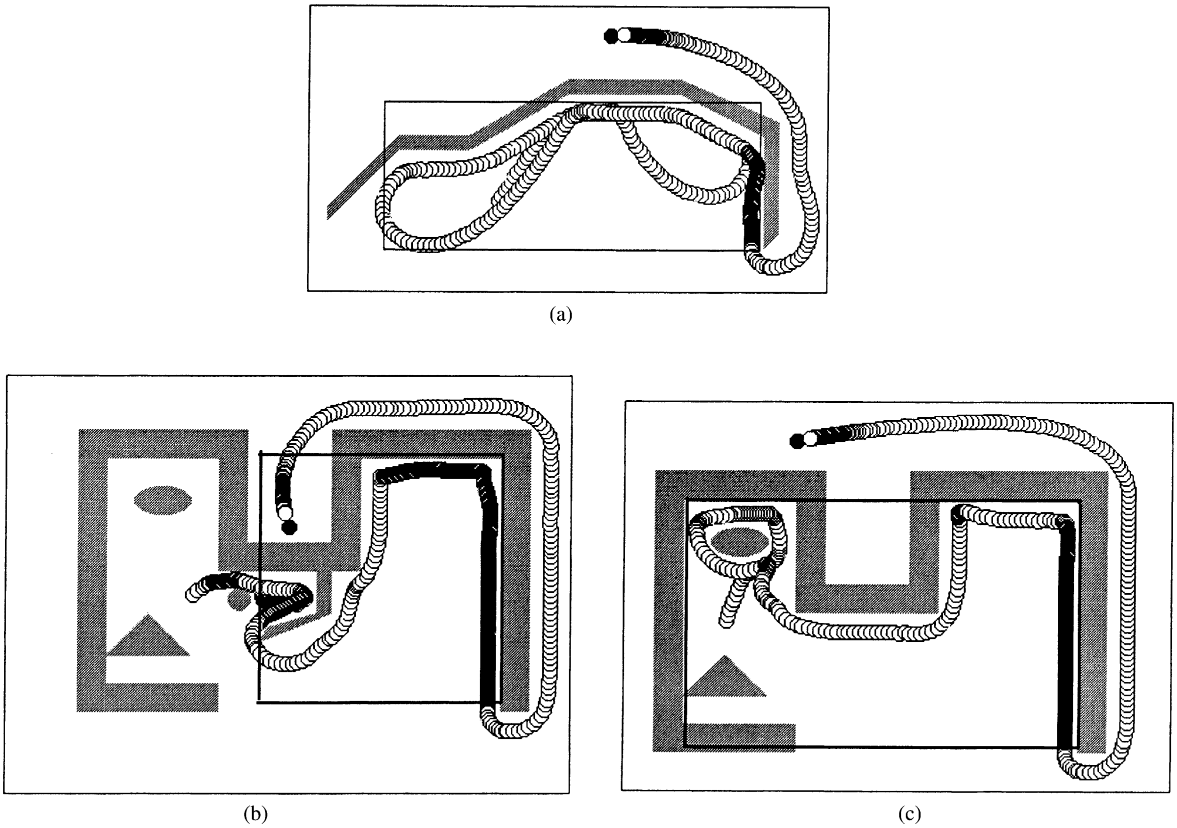
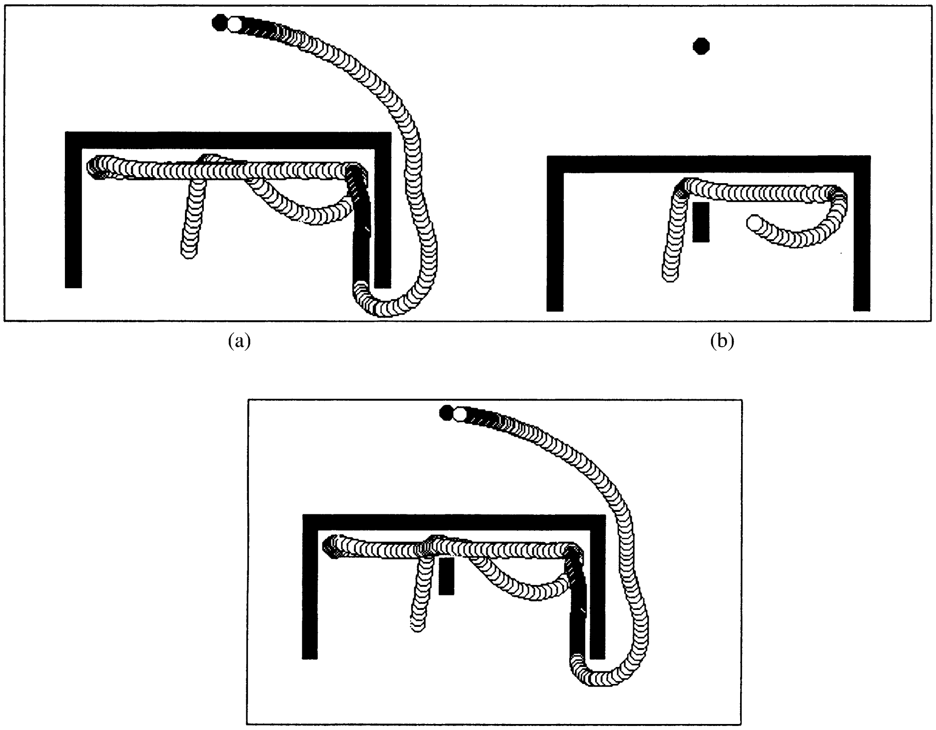


Fig. 15. (a) A configuration of obstacles whose bends are somewhat similar to those in Fig. 14. The BR is also shown. (b) The presence of convex objects near the configuration that is responsible for turning behavior makes it unstructured. (c) Presence of elliptical object near the bends lends to some irregularity.

3.6. Performance in modified environments

Since the algorithm has elements of environment understanding and memorizing the understood contents a question regarding its performance when the objects of the environment are rearranged or altered can arise. Specifically does a modified environment invalidate the contents of the memory in such a way that the robot's navigational performance completely collapses. The answer to this question depends on the kind of modifications performed (insertion of new objects, removal of existing objects and rearranging objects) and the position in the environment where these modifications occurred. Here we briefly analyze the situation when modification in the form of insertion of a new object to the environment occurs. Essentially the concern here is capability of the robot in reaching its destination despite the modifications. The following generalizations regarding the performance can be made when modification in the form of insertion occurs.

1. An insertion of an object or objects at locations that are not proximal to critical positions does not affect the performance of the algorithm in any tangible manner. By critical position we mean those positions that are instrumental in the detection of local minimum like situation or positions that are responsible for an abrupt turning behavior of the robot. Fig. 16(a) shows the path of the robot in a certain environment under unmodified conditions. Fig. 16(b) is the instant of perturbing the

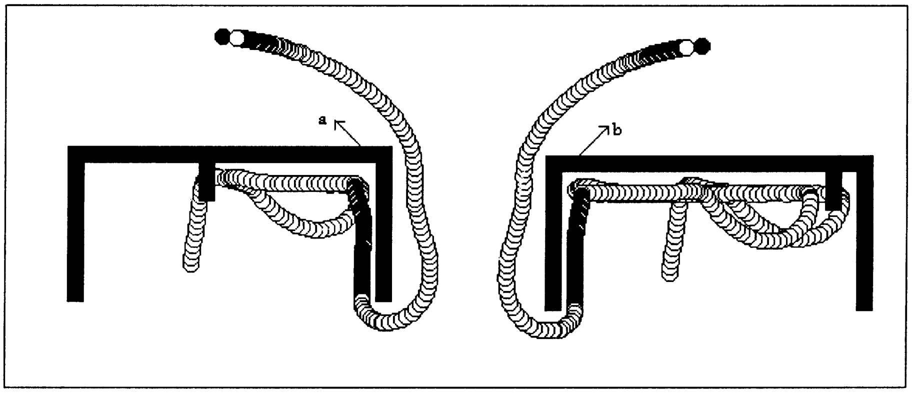


(a) (b) (c)

Fig. 16. (a) The original obstacle configuration. (b) Instant of introducing a new object. (c) The path traced in the modified environment has not tangible difference from the path traced in the original environment of ‘a’.

environment by inserting an object. Fig. 16(c) is the path obtained after this modification occurs. Evidently there is no appreciable change in the nature of the paths in Figs. 16(a) and 16(c).

2. Insertion of an object or objects in such a way that it forms a critical position by itself or insertion near a configuration that constitutes a critical position can alter the path lengths significantly. The paths can get elongated or compressed depending on the timing and position of insertion. Fig. 17(a) shows an insertion that reduces the path length considerably when compared with Fig. 16(a) as local minimum gets detected earlier. Fig. 17(b) is an example where the path increases noticeably due to delayed detection of minimum. Here temporal correlation of the corners at the other end leads to the detection. In Fig. 17(a) the bar was inserted after robot’s first visit of the corner at ‘a’ while in Fig. 17(b) the bar gets inserted during the robot’s cognition of the corner at ‘b’ after it had already been to ‘a’ once.
3. However insertion and removal of an object in an alternating fashion between any two consecutive visits of the robot can trap it permanently as it fails to experience the previous environment at the same location. For example alternating insertion and removal of objects at locations marked by arrows in Fig. 18(a) can trap the robot forever. Tapping the facilities of environment recollection properties through certain heuristics can overcome such situations. A simple heuristic of the following form can come in handy: “*If the robot cognizes a configuration of objects that entails an abrupt turning behavior three times without the recovery rules being invoked then enter local minimum avoidance mode (LMA)*”. The rule can be seen at work in Fig. 18(b). Here the middle bar was inserted after the robot had reached the corner ‘b’ after visiting ‘a’.



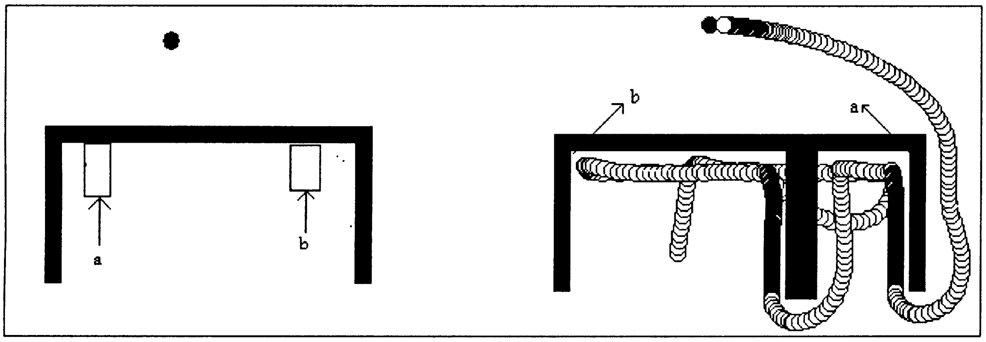
(a) (b)

Fig. 17. (a) An insertion that leads to a formation of a new landmark. A shorter path results as the modification was introduced during the robot's first visit to corner 'a'. (b) A modification when the robot cognized the corner 'b' results in a substantially longer path.

When the robot cognizes the middle bar at 'c' during its return from 'b', recalling previous experiences of the corners at 'a' and 'b' and the fact that the recovery rules have not been invoked branches the algorithm into LMA mode. As a matter of fact the above heuristic also reduces the length of path in Fig. 17(b) to its original form of Fig. 16(a).

Thus modifying the features of the environment does not have an adverse effect on the algorithm's primary objective of reaching the goal. It is always possible to come up with a scheme of insertion and removal of objects in an alternating or cyclic fashion that can confuse the robot despite the heuristics. However barring such intentional and planned modifications a random singular or stray modification in general would not prevent the robot from attaining its goal. This section further indicates the enhancements to real-time navigation due to spatio-temporal reasoning.

Generally algorithms that use maps are considered robust to such changes in environment. Any change in the environment gets reflected on the map and the paths are recomputed according to these changes. Since objects are represented in terms of cells with occupancy values image understanding from these cells require reconstruction techniques and hence difficult and computationally intensive. In the current method internal representations are not in the



(a) (b)

Fig. 18. (a) Alternate insertion and removal of objects at the two locations indicated by the two locations indicated by the arrow can trap the robot forever. (b) Robot comes out of the configuration of obstacles by recollecting previous experiences of the corner at 'a' and 'b'.

form of an assortment of cells but rather in terms of landmarks. For example the robot's internal representation takes the following form, "*there's a corner near spatial location 'a', a dead end near location 'b', a bend near 'c' and a corridor near (270, 200)*". These internal representations are stored in a codified form in a queue data structure as described in Section 3.3. Each element of the queue contains the lattice position of the winning neuron and the coordinates of the robot at that location. Modifications to the environment can get appended to the queue without deleting the original contents. Thus the queue structure is some kind of an implicit map. However the map info has not been used in the traditional sense of planning the paths, rather they are used as aids that help in memory-based reasoning for the robot.

3.7. Sensor related problems

As is well known ultrasonic sensors come with their own baggage of problems and a navigation algorithm must also consider effects of reducing these. To overcome the noise inherently associated with range readings a two tier filtering scheme is adopted in this present approach. At the first stage the fuzzy classification scheme acts as a vector quantizer and hence has some inherent noise filtering capacities. Moderate variations in readings get smoothed as they get mapped to the same class. For example when the robot approaches the corner shown in Figs. 5 and 6 the readings of the left sensor tend to grow more than the actual distance due to multiple reflections. However the classification scheme classifies both the actual and elongated readings into the same class. The fuzzy classifier serves to filter most of the moderate variations in the range readings.

At the second stage the attempt is to counter drastic variations in sensor readings such as outliers or any spurious signals. Some heuristic filtering measures are adopted for this purpose. For example a sequence of classes 77777177 encountered by the robot during its sojourn gets recast as 77777777 for such abrupt changes can occur due to (i) rotational variations or (ii) stray noise/outliers. In other words the pattern sequence 7–1–7 cannot be due to any configuration of static objects and can be safely filtered away as noise or outlier. Similar measures are adopted to detect noise within a stream of data. They are somewhat akin to Borenstien's popular VFF [7] or VFH method [19], which ascribes a certainty measure to each reading. So as the robot sees a class 7 in a repetition an implicit increase in certainty of that reading occurs and the robot is assured of its traversal along a wall on its left. Indeed these are the main considerations in fixing the vector of lower bounds described in Section 3.2.

Finally it is to be mentioned that many such sources of error can be surmounted through laser range finders and the present method is applicable to both sonar as well as lasers.

A real-time implementation of the current algorithm is presently beyond the scope of our lab considering its primary research thrusts and objectives. However the first author as a graduate student at a different lab could navigate a robot in real-time through the same configuration of sensors as mentioned here. The details of these real-time runs can be found in the paper [9] cited in this paper. The same sonar model was applied in simulations. For real-time purposes the reliability of range data was improved through a median-filtering scheme. Here each sensor samples the environment five times in rapid succession. The median of the reading obtained in these five samples is considered the most reliable reading. Median filtering is commonly used as a robust statistical tool for filtering noise and outliers [21]. This way a circular array of seven sensors with a maximum range of 1 m can be sequentially fired in 205 ms. The robot is assumed stationary during those 205 ms for computational convenience. However this assumption has negligible impact on the navigational performance of the robot. Median filtering has some philosophical resemblance to the certainty approach in that the most reliable reading (reading with high certainty) is extracted through this method. In simulations median filtering is not very consequential and hence not adopted.

4. Comparison with other methods

In this section the methods [4,13–16] proposed recently in the literature for overcoming the local minimum problem are analyzed and compared with the present approach.

4.1. Huang and Lee's method

This method [4] is an extension of Lumelsky's Bug algorithm [18] where the wall following scheme used in Lumelsky's approach for overcoming the local minimum trap is selectively used. The algorithm operates in two modes the H mode and T mode. H mode operates when the obstacles are convex and the T (track) mode comes into play when concave obstacles are encountered. The switch from H mode to T mode occurs when the difference in successive turns made by the robot exceeds 120°. The algorithm switches back to H mode at a point 'b' during its traversal which is collinear with points 'a', 'c' and is between 'a' and 'c', where 'a' is the location at which the robot switched initially to T mode and 'c' is the target position. The following handicaps can be seen:

1. The detection of local minimum is done empirically by comparing differences between successive rotations. This leads to occasional misidentification of situations as a local minimum. Fig. 19(a) shows a misclassification of the environment as a local minimum and the robot begins to track the obstacle contour though it is not trapped leading to long traversals. Fig. 19(b) is the path obtained by

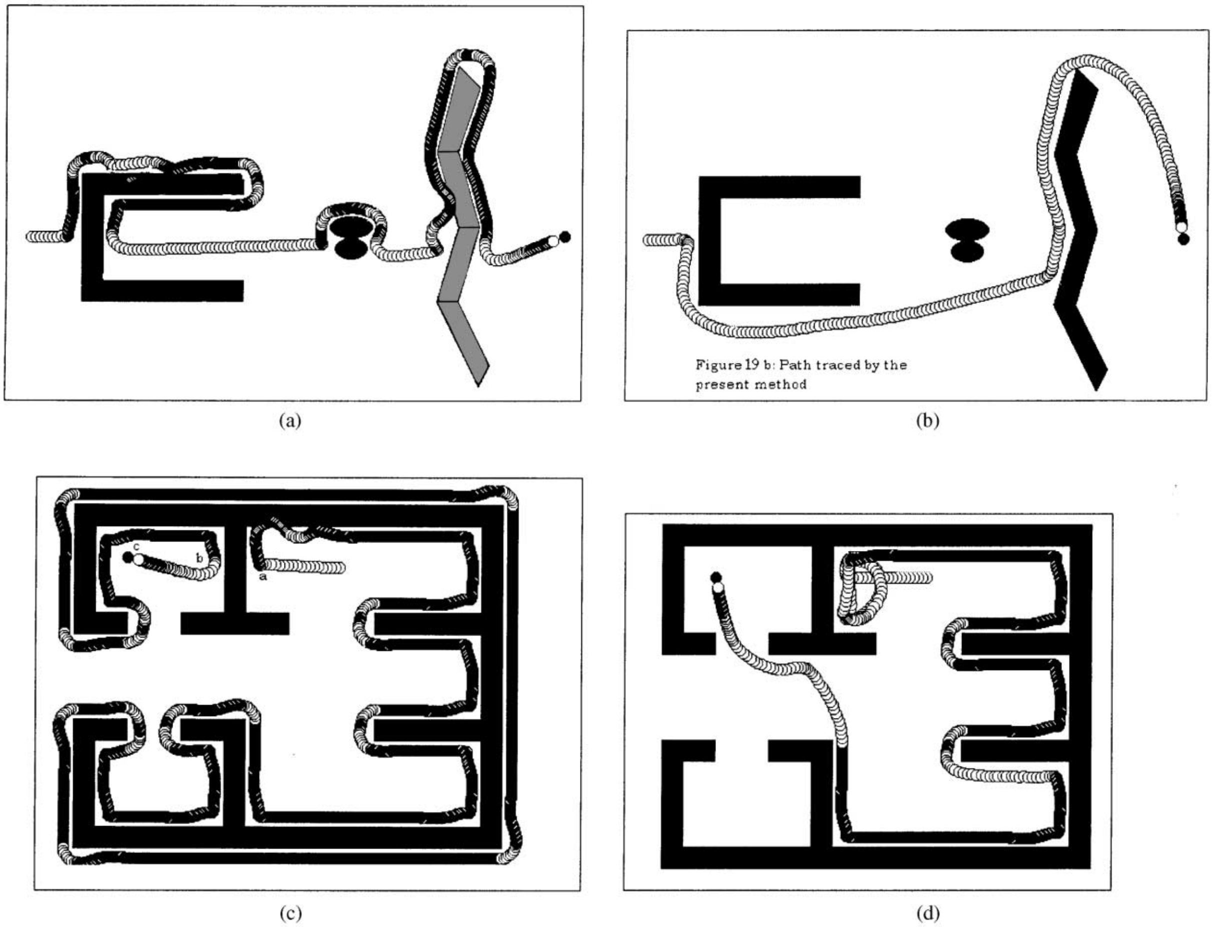


Fig. 19. (a) An example form a previous paper. Heuristic determination results in misidentification of local minima. (b) Path traced by the current method. (c) According to Huang and Lee's method the robot leaves wall following when 'a', 'b', and 'c' are collinear and 'b' lies between 'a' and 'c'. (d) A much shorter path results from the present method.

the current method. The current method does not recognize the configuration of obstacles to represent a local minimum since it does not experience the same scenario twice in its traversal. The path obtained is shorter.

2. The leaving condition is too conservative which also leads to long paths. Fig. 19(c) is the path obtained using Huang and Lee's approach while Fig. 19(d) is the path obtained by current approach using the recovery rules described in Sections 3.3.1 and 3.3.2. These sections discussed the recovery rules using Fig. 9(b), which is the sketched counterpart of the actual simulated graph of Fig. 19(d).

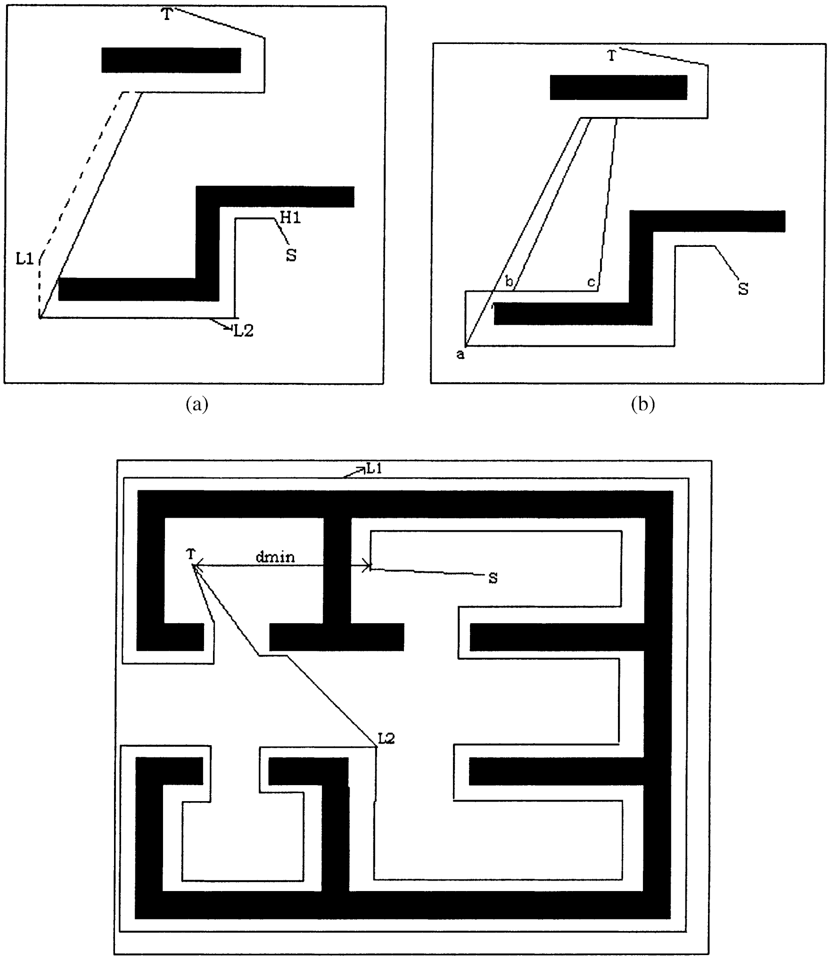
4.2. Kamon and Rivlin's method

Kamon and Rivlin [13] proposed a method whose main contribution was a new leaving condition that generated paths with much reduced lengths when compared with the Bug algorithms. Their algorithm was called Distbug. There is no specific scheme for determining the robot's trapped condition. The algorithm automatically slips into wall following upon detecting the first obstacle. The robot leaves the obstacle boundary when either the target becomes visible or according to the inequality $d(X,T) - F \leqslant d_{\min}(T) - \mathit{Step}$, where $d(X, T)$ is the distance from the current location, X, of the robot to the target, T, $d_{\min}(T)$ is the minimal distance to the target since the last instantiation of wall following, F is the distance in freespace from X and *Step* is a predefined constant. If no obstacles are detected F is set to R, the maximum range of the sensors. The implication of the above inequality is that the robot escapes from wall following:

1. when the target becomes visible or
2. when the range readings in the direction of the target indicate that the robot can proceed towards the target without meeting an obstacle at a location farther from the target than the current obstacle whose contour the robot follows.

This condition also guarantees that the robot would not revert into the same area bounded by those obstacles which it had just escaped from. The leaving condition is dependent on the maximum sensor range R and the *Step* size. When global information of the environment is absent *Step* is determined heuristically. Increasing value of *Step* increases path lengths while reducing *Step* results in robot leaving the wall following mode earlier. However choosing too small a step size can result in the robot performing many path cycles before it eventually avoids the obstacle. To prevent this the robot must follow the boundary upon hitting a obstacle in a direction, which takes it closer to the target. This is naturally achieved in our proposed algorithm by the target attracting module. In the discussion to follow the *Step* size is taken as negligible or zero.

The claim is that the present leaving criteria results in the robot leaving the obstacle following mode earlier than the condition proposed in the Distbug algorithm. In all those figures relevant to the remaining part of this Section 4.2 the robot's path is represented by sketches and not by actual simulation graphs. Fig. 20(a) shows the path traced by the robot using the Distbug algorithm in dashed line. The robot switches to wall following at H1 and leaves wall following at L1 when the inequality holds. In the same figure the current algorithm switches from wall following to the fuzzy rule base mode at position 'L2' when the robot comes out of the BR and obstacle and target are on the same side of the robot. The paths leading to the detection of local minimum is not shown since we are primarily concerned about the leaving criterion and not the detection criterion. It is clear from this figure the leaving criteria of the present algorithm can result in shorter paths. However increasing R in Distbug leads to earlier departure from obstacle following as Fig. 20(b) indicates where the algorithm starts moving towards the target from position 'a' and matches the performance of the present algorithm. Increasing R any further shall not decrease the path length anymore since the inequality would not get satisfied. Decreasing R on the other hand results in Distbug churning out longer and longer paths, the robot leaves wall following at positions 'b', 'c' on decreasing R in the same figure. On decreasing R further the performance of Distbug reduces to the performance of the standard Bug algorithm. Fig. 20(c) shows the path traced by Distbug in solid line with the leaving position marked at L1 when R is set to the maximum sensor range used in the simulations of our present work. Increasing R to more than twice the maximum value used in the current algorithm results in the Distbug leaving at L2 and the performance matches the present algorithm (Fig. 20(c)). It is to be noted that L1 is the first instant



(a) (b) (c)

Fig. 20. (a) Path planned by Distbug shown in dashed line and the current path in solid line. Overlapping parts of both the paths is also shown in solid line. (b) Paths with varying *R*. Robot departs from wall following at 'a', 'b' and 'c' as *R* decreases resulting in longer paths. (c) For varying *R* Distbug has its leave points at L1 and L2.

when the robot exits from wall following. But the presence of the wall immediately on its left forces it to continue with wall following till the target is sighted. Increasing *R* further has no effect for the inequality $d(X,T) - F \leqslant d_{\min}(T) - Step$ is not satisfied. In other words the algorithm does not see an obstacle closer to target than, $d_{\min}$ (marked in the same figure), however high the maximal sensor range be. It can be summarized the best performance of Distbug on increasing *R* matches the current algorithm and on decreasing *R* performance of Distbug reduces to that of the Bug algorithms. The fuzzy nature of target attracting module in the current algorithm, which turns the robot towards its target in smooth increments can in some cases veil this advantage.

4.3. Virtual target approach

The virtual target approach [15,16] is the most recent work on local minimum avoidance as on date when this paper is being written. It presents a very elegant scheme, as the algorithm does not change the

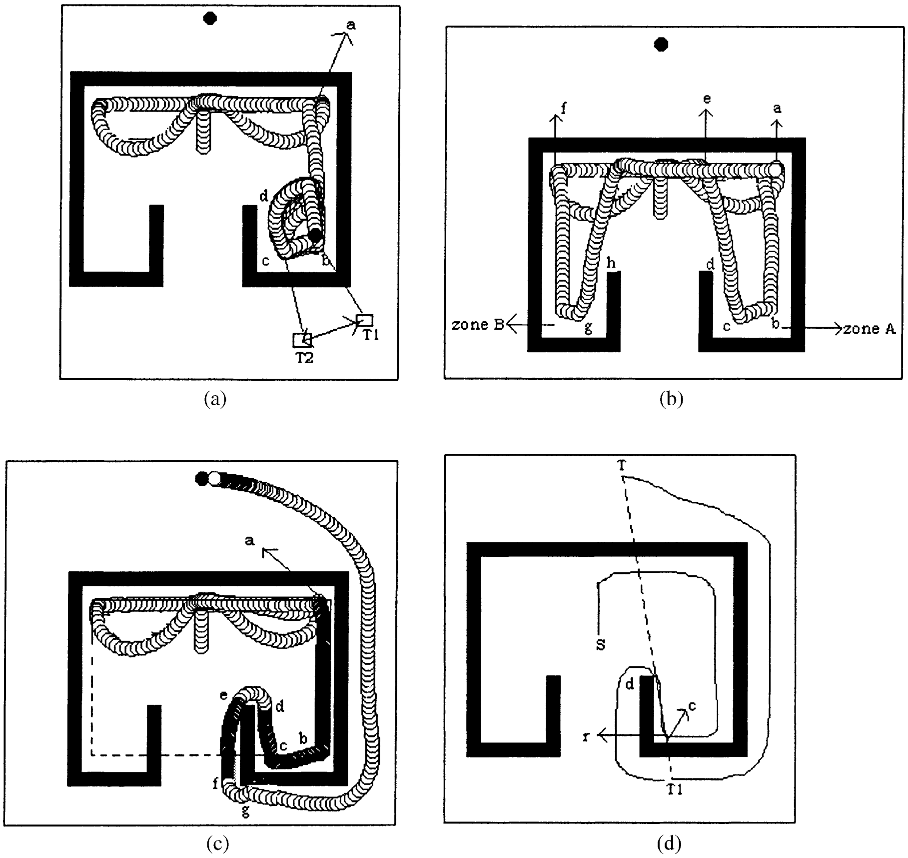


Fig. 21. (a) The failure of the virtual target approach. (b) Oscillation between the zones A and B. (c) The current method overcomes the local minimum. (d) Modified path traced by the virtual target approach on using a subgoal stack.

control structure of the fuzzy algorithm. In other words the algorithm does not switch to a wall following module upon detection of a local minimum. The local minimum is detected when an abrupt change in the target orientation with respect to the robot occurs, i.e. when the target position changes from left to right or vice versa over an instant. Upon detection of the local minimum the robot continues to navigate with the existing fuzzy rule base using a new virtual target orientation as a modified input to the rule base. The target is switched back to its original position when an opening in the obstacle is detected. Upon scrutiny however the algorithm fails to transcend the local minimum trap in some situations. Fig. 21(a) shows a simulation where the virtual target strategy fails. Failure occurs in those situations where the robot encounters a local minimum when operating in the virtual target mode. In Fig. 21(a) target is switched at point ‘a’ where an abrupt change in the target orientation occurs. The virtual target orientation is given by $tv = -(\pi - tr)$, where tv and tr are the virtual and real target orientations, respectively. From position ‘a’ the robot navigates using the virtual target till it reaches the end of the obstacle at position ‘d’. However the robot meets another local minimum situation between points ‘b’ and ‘c’ due to the influence of the virtual target. The orientation of the virtual target with respect to the robot while it navigates from ‘b’ to ‘c’ wavers between the rays passing through the two boxes ‘T1’ and ‘T2’. Target attracting and obstacle repulsing tendency conflict each other and the robot falls again into the local minimum trap from which it is unable to escape. A possible means to get out of this trap is to navigate the robot under the influence of a new virtual target and switch back to the real target when the end is detected. The orientation of the second virtual target is chosen with respect to the first virtual target in the same manner and the orientation shall almost coincide with the real target. Hence the robot can be considered to be operating under the influence of the real target from point ‘c’ itself instead of point ‘d’. Upon reaching ‘d’ a formal switch can be made to the real target. The problem is still not resolved as the robot wanders into oscillations between the two symmetric zones of the obstacle marked ‘A’ and ‘B’ shown in Fig. 21(b). The explanation for this phenomenon is as follows. At ‘d’ even if the robot would rotate around the bend the real target attracts the robot and the robot rushes to meet the same wall again at ‘e’. The robot traverses now towards the left side of the wall and at ‘f’ switches to a virtual target. At ‘g’ it switches to a new virtual target till an opening is detected at ‘h’ where a switch is made to the real target. At ‘h’ under the influence of the real target the robot reaches the wall yet again and the cycle continues forever with the robot oscillating back and forth between zones A and B. Fig. 21(c) shows the path traced by the robot using the present algorithm for the same workspace. In Fig. 21(c) the robot departs from wall following at ‘g’ which lies outside the BR shown in dashed line and where the target and obstacle lie on the same side of the robot. Due to the reasons cited above the virtual target approach cannot surmount the barrier given in Figs. 19(c) and 19(d) for the same starting and target locations.

4.3.1. Modifications to the virtual target approach

A modification to the virtual target approach is suggested to overcome its vulnerabilities. The proposal is to maintain a stack containing the target orientations that have been switched. The top most element of the stack is the target position that has been switched most recently. Upon reaching the end of an obstacle the top most target is popped from the stack and the robot visits it. On visiting this target the robot attempts to visit the next target in the stack. Thus the robot visits each of the subgoals in the order in which they are popped out of the stack and eventually reaches the original target which is the last to be popped out. The stack contains the intermediate subgoals the robot should visit and hence is termed as the subgoal stack. Using this strategy the robot can overcome the situation discussed in Figs. 21(a) and 21(b). Fig. 21(d) shows the sketch of the path the robot would adopt using the modified approach. At point ‘a’ the position of the real target, ‘T’, is pushed on to the stack. At ‘c’ the position of the virtual target ‘T1’ is pushed on to the stack. When the robot reaches the opening at ‘d’ switch is now made to the virtual target ‘T1’ instead of the real target, which lead to the oscillations shown in Fig. 15(b). From ‘d’ the robot navigates under the influence of ‘T1’. On reaching ‘T1’ switch is made to the original target ‘T’, which the robot eventually reaches. There is still a catch in this reasoning. When the robot reaches ‘c’ and detects an abrupt change in the orientation of the virtual target it pushes the position of the virtual target, T1, onto the stack. The position of ‘T1’ is unknown to the algo-

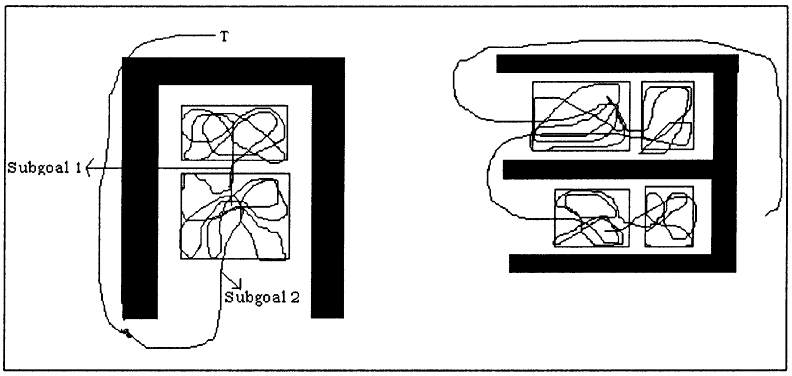


Fig. 22. In the virtual obstacle approach the robot falls into local minimum many times before it reaches the target.

rithm for it is only a virtual target. Only its orientation is known. The algorithm actually pushes on to the stack the approximate position of the location 'c' and its own location at that instant 'r'. Location 'c' can be found through sensor readings. The robot then navigates from 'd' under the influence of the virtual target positioned at 'c' till it reaches a position 'T1' which is approximately collinear with the line from 'r' to 'c' such that 'c' lies between 'r' and 'T1'. Employing a subgoal stack appears to be an appropriate modification to eliminate the limitations of the virtual target approach.

Table 3
Comparison of the recent approaches that tackle the local minimum problem

Method	Detecting criterion	Leaving criterion	Comments
Huang and Lee [4]	Difference in rotation of the robot between successive instants	When points a, b and c are collinear and b is between a and c. a is the hit point, b the leaving point and c the target	• Empiric detection leads to erroneous classification of local minima. • Leaving criterion is highly conservative. Paths longer than other methods that employ wall following.
Distbug [13]	No explicit criterion	Target is visible or the nearest obstacle in the direction towards the target is closer to the target than the current obstacle whose contour the robot traces	• A highly efficient leaving criteria leads to more optimal paths compared to standard bug algorithms. • Leaving criterion is dependent on maximal sensor range. Increasing sensor range implies earlier egress from wall following.
Virtual target [15,16]	Abrupt change in robot's turning tendency due to a change in target orientation	An opening in the obstacle is detected	• The algorithm does not change the structure of the fuzzy control system. • Regresses into the same infinite loop it tries to avoid and unsuitable for environments where local minimum occurs when the robot navigates under the influence of a virtual target. • Modification proposed in this paper which corrects the above problem.
Virtual obstacle [14]	Robot visits twice the same location with the same orientation	When the subgoal created is reached	• Very large memory requirements. • As the length of the concave obstacle like a corridor increases the path length increases manifolds due to creation of many virtual obstacles. Gives longest traversals among all the methods considered.
Current approach	Robot recognizes previous experience of a similar environment at the same position	When the robot reaches a location outside the BR where the target and obstacle are on the same side of the robot	• Employs spatial and temporal reasoning for detection. • Leaving condition less conservative then the other methods that employ wall following.

4.4. The virtual obstacle approach

Pin and Bender [14] proposed a virtual obstacle approach for the local minimum problem. Local minimum is detected when the robot visits approximately the same position twice with the same orientation. When it occurs the minimum and maximum coordinates of the positions that occurred between the two visits is taken to form the two diagonal vertices of a rectangular obstacle. The remaining two vertices get fixed automatically. The space surrounding the obstacle is divided into eight sectors and the sector opposite to the current goal is determined. The subgoal is placed at the mid-point of that sector and the robot navigates to the subgoal and reaches it. Navigation then continues with the original goal and considering distances from the virtual obstacles as well as real obstacles. The approach has the following disadvantages.

1. Considerable memory is involved in storing all the coordinate positions and their orientations at those positions between the two visits. Memory is also involved in storing the coordinates of the virtual obstacle every time they are formed.
2. The robot gets into a number of local minimum paths till it eventually reaches the target. Each such local minimum path consists of a number of oscillations till the minimum is detected. Every local minimum involves creation of a new virtual obstacle and a subgoal and increases the path many folds. The final path traversed is extremely long and cumbersome. Other approaches do not suffer from this problem of having to endure many local minimum paths before reaching the target. Fig. 22 shows the nature of the path computed by the virtual obstacle approach and the corresponding increase in path lengths as the number of local minima the robot encounters increases.
3. In certain cases as shown in their paper [14] the robot can enter a local minimum when the subgoal is located in unreachable locations and a new virtual obstacle has to be created. This too plays a part in making the path unwieldy.

4.5. Summary

When should the robot depart from wall following is an interesting and a possible area of research for autonomous robots prevailing over the local minimum barrier in the absence of prior knowledge of the environment. Leaving too early can result in the robot regressing into the same local minimum path it tries to escape and this can go on forever. Leaving too late can lead to lengthy traversals. The algorithm provides a new leaving criterion that gives shorter paths than those algorithms that tackle the local minimum. The paths are shorter than the algorithms that employ and that do not employ wall following. The performance of both the approaches that do not employ wall following, the virtual target and the virtual obstacle approaches is less satisfactory than those employing it. The virtual target approach regresses into infinite loops in some cases and the paths of virtual obstacle are unwieldy.

As far as detection of local minimum is concerned the present method employs spatial understanding and temporal association properties imparted by the classification scheme. The claim is that it is more consistent with human intuition involving memory and recollection features.

The comparisons are summarized in a tabular form in Table 3.

5. Conclusions

A method for imparting spatial reasoning, memory, recollection and temporal correlation features for a mobile robot has been proposed. A classification scheme consisting of a fuzzy classifier coupled to a SOM and fuzzy ART network has been adopted for this purpose. The performance of the algorithm is embellished through more acceptable paths due to scene understanding, memorizing and recall of such learned scenarios and detection of local minimum traps by temporally associating similar perceptions of the environment. The paper also describes a new criterion for departing from wall following that results in reduced paths compared to other methods that tackle the local minimum and at the same time avoids relapsing into the same local minimum path due to the earlier departure from wall following. A comparative study of the recent methods proposed reveal that those methods that do not employ wall following to escape the minimum either relapse into the same infinite loop in certain situations or furnish very long paths. The proposed method seems suitable for concave, maze-like and altered environments.

Acknowledgements

The authors would like to thank the anonymous referees for their comments and the co-editor T.C. Henderson for having to bear the queries from us regarding the manuscripts status.

References

[1] F.G. Pin, Y. Watanabe, Navigation of mobile robots using a fuzzy behaviorist approach and custom designed fuzzy inferencing boards, Robotica 12 (6) (1994) 491–503.
[2] P.S. Lee, L.L. Wang, Collision avoidance by fuzzy logic for AGV navigation, Journal of Robotic Systems 11 (8) (1994) 743–760.
[3] H.R. Beom, H.S. Cho, A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning, IEEE Transactions on Systems, Man and Cybernetics 25 (3) (1995) 464–477.
[4] H.P. Huang, P.C. Lee, A real-time algorithm for obstacle avoidance of autonomous mobile robots, Robotica 10 (1992) 217–227.
[5] J.L. Crowley, Dynamic world modeling for an intelligent mobile robot, in: Proceedings of the IEEE Seventh International Conference on Pattern Recognition, Montreal, Quebec, 1984, pp. 207–210.
[6] V. Lumelsky, T. Skewis, Incorporating range sensing in the robot navigation function, IEEE Transactions on Systems, Man and Cybernetics 30 (1988) 1058–1069.
[7] J. Borenstein, Y. Koren, The vector field histogram — fast obstacle avoidance for mobile robots, IEEE Transactions on Robotics and Automation 7 (3) (1991).
[8] A. Dubrawski, J.L. Crowley, Learning locomotive reflexes: A self supervised neural system for a mobile robot, Robotics and Autonomous Systems 12 (1994) 133–142.
[9] I.J. Nagrath, L. Behera, K. Madhava Krishna, K. Deepak Rajasekhar, Real time navigation of a mobile robot using Kohonen's topology conserving neural networks, in: Proceedings of the International Conference on Advanced Robotics, Monterey, CA, July 1997, pp. 459–464.
[10] G.H. Hamzei, et al., Becoming incrementally reactive: Online learning of an evolving decision tree array for robot navigation, Robotica 17 (1999) 325–334.
[11] T. Kohonen, Self-organizing map, Proceedings of the IEEE 78 (9) (1990) 1460–1480.
[12] G.A. Carpenter, S. Grossberg, D.B. Rosen, Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system, Neural Networks 4 (1991) 759–771.
[13] I. Kamon, E. Rivlin, Sensory-based motion planning with global proofs, IEEE Transactions on Robotics and Automation 13 (6) (1997) 814–822.
[14] F.G. Pin, S.R. Bender, Adding memory processing behavior to the fuzzy behaviorist approach: Resolving limit cycle problems in mobile robot navigation, Intelligent Automation and Soft Computing 5 (1) (1999) 31–41.
[15] W.L. Xu, S.K. Tso, Sensor-based fuzzy reactive navigation for a mobile robot through local target switching, IEEE Transactions on Systems, Man and Cybernetics 29 (3) (1999).
[16] W.L. Xu, A virtual target approach for resolving the limit cycle problem in navigation of a fuzzy behavior-based mobile robot, Robotics and Autonomous Systems 30 (4) (2000) 315–324.
[17] M.H. Hasoun, Fundamentals of Artificial Neural Networks, MIT Press, Cambridge, MA, 1995, p. 114.
[18] V.J. Lumelsky, A.A. Stepanov, Path planning strategies for a point mobile automaton moving amidst obstacles of arbitrary shape, Algorithmica 2 (1987) 403–430.
[19] J. Borenstein, Y. Koren, Vector field histogram — fast obstacle avoidance for mobile robots, IEEE Transactions on Robotics and Automation 21 (3) (1991).
[20] A. Elfes, A sonar-based mapping and navigation system, Technical Report, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 1985, pp. 25–30.
[21] P. Rousseeuw, A. Leroy, Robust Regression and Outlier Detection, Wiley, New York, 1981.

K. Madhava Krishna was born in Madras, India, in 1974 and obtained his undergraduate and postgraduate degree from the Birla Institute of Technology and Science, Pilani, in Electronics Engineering. He is currently working towards his Ph.D. degree in the Indian Institute of Technology at Kanpur. His primary research interests are in the areas of mobile robotics, soft computing and intelligent systems. He is also keenly interested in consciousness studies, philosophy of mind, mind–brain problem and philosophy of quantum physics and AI.

Prem K. Kalra was born in 1957 in Agra, India. He obtained his undergraduate degree from Dayalbagh Educational Institute and post graduate degree from Indian Institute of Technology at Kanpur. His doctoral degree was from the University of Manitoba. His research interests are in neural networks specifically in the area of new neuron models, fuzzy logic and AI.