Landmark-Based Navigation for a Mobile Robot

Carl Owen and Ulrich Nehmzow
University of Manchester
Department of Computer Science
Manchester M13 9PL
United Kingdom
owenc,ulrich@cs.man.ac.uk

Abstract

In this paper we present a landmark based navigation mechanism for a mobile robot. The system uses a self-organising mechanism to map the environment as the robot is led around that environment by an operator. Detected landmarks, and their relative position towards each other, are recorded in a map that can subsequently be used to plan and execute paths from the robot's current location to a 'goal' location specified by the user.
The main motivation for the research described in this paper is to develop a mobile robot navigation system that is robust (through the use of perceptual landmarks), and allows the robot to plan *arbitrary* paths within its known environment. The system presented here achieves these objectives.

1. Introduction

Research in mobile robotics has produced two main approaches to modeling environments: metric and topological. In the first scheme, an accurate metric map of the robot's environment is either constructed by the robot or supplied by a human designer (Kampman and Schmidt, 1991; Knieriemen and von Puttkamer, 1991). In the latter approach the environment is modeled as a graph containing nodes representing distinct locations, pathways between locations are denoted by arcs connecting the appropriate nodes (Kurz, 1996; Yamauchi and Beer, 1996; Zimmer, 1996; Mataric, 1992).

A metric map has the advantage of being a simple and natural representation for human users. However, due to the amount of detail contained in such a representation, these maps are time consuming to construct, require large amounts of memory, and are often over specified for the task of general navigation.

Conversely, the topological approach gives a compact representation since only distinctive places within the environment are encoded. In addition, this type of map is well suited for use with the various path planning algorithms that have been developed within the field of artificial intelligence (e.g. A^*, Best First Search). One of the main problems with this method is *perceptual aliasing* i.e. distinct locations within the environment appearing identical to the robot's sensors.

Various approaches to the problem of perceptual aliasing have been utilized within the topological mapping paradigm. One method is to effectively increase the robot's perceptual resolution by adding and combining additional information from differing sensor modalities (sensor fusion). In (Kortencamp and Weymouth, 1994), for example, vision is used to augment the sonar data of the robot. However, this type of approach cannot be guaranteed to disambiguate all situations, and is more useful as a tool for reducing, rather than eliminating, perceptual ambiguity.

Other systems use positional information to disambiguate perceptually similar but physically distinct locations (Kurz, 1996; Zimmer, 1996; Mataric, 1992). However, since positional information based on the robot's internal odometry is subject to drift effects, some means of correction is required if the robot is to map anything other than small scale environments. In (Kurz, 1996), for example, an extended Kalman-filter is used for drift compensation.

The system described in this paper constructs a topological map of the environment based on a process of self-organisation of the robot's sensory data. The approach of self-organisation to landmark detection was chosen for two reasons. Firstly, interpreting the world using the robot's relatively impoverished sensors is difficult for the human designer. Thus, user defined landmarks tend to be rather simplistic. The type of environment that can be categorised using this method is therefore restricted. Secondly, since the clustering techniques used in self-organisation enable a generalisation over perceptions, this approach gives a robust, noise tolerant, method of landmark detection.

Global positioning information does not form part of the representation of the system described here, thus obviating the need for drift compensation. An exploration strategy is used to resolve perceptual ambiguity.

The paper is structured as follows: Section 2. deals

[0] Proc. Simulation of Adaptive Behaviour '98, MIT Press 1998.

with the navigation mechanism. In this section the implementation details of the system are discussed. Section 3. sets out the results of several experiments conducted within the robotics laboratory of Manchester University. This section also details a performance metric that was used to measure the robot's navigational ability. Section 4. draws conclusions and discusses further work.

2. The navigation system

The navigation mechanism consists of two main processes:

1. Mapbuilding - construction of a topological vector map.
2. Map interpretation - path planning and execution.

These two processes are performed consecutively, i.e. the mapbuilding process is fully completed before map interpretation begins.

The robot used for our experiments was a Nomad 200 mobile robot (see figure 1). This robot is equipped with sixteen ultrasonic range finding sensors (range up to 6.5 m), sixteen infrared (IR) sensors (range up to 60 cm), twenty tactile sensors, a compass and a monochrome CCD camera. In the experiments described here, only the sonar sensors and compass were used.

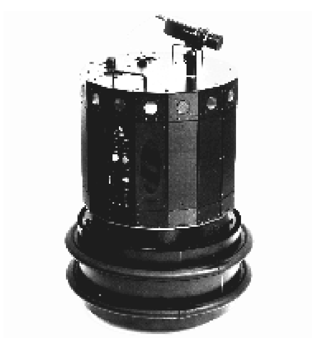


Figure 1 The Nomad 200 mobile robot.

2.1 Mapbuilding

The concept of the vector map is depicted in figure 2.

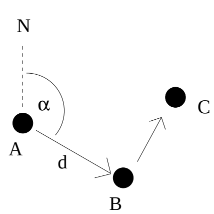


Figure 2 The vector map. Three points 'A', 'B' and 'C' represent particular perceptual landmarks within the environment, the arcs connecting the nodes record the compass direction, α, and distance, d, between these landmarks.

By employing a vector mapping of this type we restrict the use of odometry to the measurement required between locations, thus limiting the accumulation of odometry error.

In previous work, a Self-Organising Feature Map (SOFM) (Kohonen, 1988) has been used to map the robot's environment. Using this method, mapbuilding is achieved by performing an unsupervised clustering of the sensory data of the robot as it moves through the environment (Nehmzow and Smithers, 1991; Owen and Nehmzow, 1996). When the network has settled, regions on its surface represent 'perceptual landmarks' within the robot's environment.

However, the requirement for a settlement period is relevant to the issue of 'lifelong learning'. If the environment were to change after training then a further period of settlement would be required before vector links could be added. In addition, the size of the network needs to be pre-defined before use. Determination of optimal network size is difficult since no indication of the number of landmarks that the robot will perceive can be gained by simply observing the environment in which the robot will operate. Choosing too small a network would result in saturation, and a highly unstable network. On the other hand, the larger the network the more expensive the algorithm becomes at lookup time.

Other network methods that have been used in the context of landmark detection include the Restricted Coulomb Energy (RCE) classifier (Reilly et al., 1982), used, for example, by Kurz (Kurz, 1996), and the Adaptive Resonance Theory (ART) classifier (Carpenter and Grossberg, 1987), used, for instance, by Duckett and Nehmzow (Duckett and Nehmzow, 1997). Neither of the latter methods require a settlement period to achieve a stable clustering of the input space (thus allowing immediate placement of vector links), and both are able to grow to accommodate new classifications.

Since, as discussed, pre-determination of network size is difficult, and lifelong learning is seen as a future development for this system, it was decided that one of the growing networks would be used as the basis of the mapping mechanism. The RCE model was chosen in preference to the ART classifier since the latter requires a large number of parameters to be set before use, the values of which can only be determined experimentally.

Although the SOFM is better able to generalise on noisy inputs, smoothing of the input vector can reduce noise (see below), and in practice the RCE network was found to produce a stable clustering of the robot's perceptual space.

The clustering mechanism The RCE-Classifier uses a method of classification based on self-organisation. Each class is represented by a *representation vector* (R-vector). Training the RCE-Classifier involves determining the R-vectors.

When a pattern is presented to the classifier, the input is compared to each of the existing R-vectors, using some form of similarity measure (e.g. dot product) in order to determine a winner (i.e. the R-vector of highest similarity). If the similarity between the input pattern and the winning R-vector is within a predetermined threshold then the input pattern belongs to the class of this winning R-vector. If the similarity is outside the threshold then the input pattern becomes a new R-vector. Thus the boundaries of classes are determined by the nearest neighbour law. Figure 3 shows an example for a two-dimensional input vector.

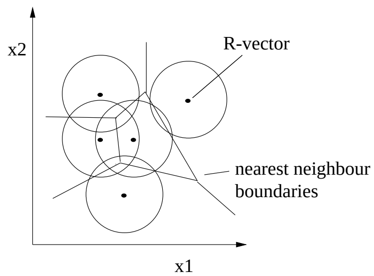


Figure 3 RCE-Classifier, two-dimensional example. Each 'dot' in the diagram represents an R-vector. The circles surrounding each R-vector denote the 'threshold area' within which an input pattern must fall in order to belong to the corresponding R-vector's class. In the case of patterns falling within more than one threshold area, the nearest neighbour law applies.

The input vector The input vector in this application consists of the sixteen readings of the robot's sonar sensors. This input vector is normalised and the dot product of the input and R-vector is used as the measure of similarity for the RCE-classifier. The threshold for adding new R-vectors was fixed at 0.9.

In using the RCE-classifier to process the input vector, perceptual features within the environment are determined autonomously by the robot; each class generated by this method can be viewed as a 'perceptual landmark' within the environment.

Since landmarks in this scheme are determined by the sensory patterns obtained by the robot, perception will be affected by the direction in which the sensors are facing. This means that a location within the environment could be perceived differently when approached from different directions. In order to overcome this problem the robot's on board compass is used to align the turret (which contains the sensors) to compass north at all times. Thus, the input vector generated by the system will depend on position alone, regardless of steering orientation.

Since sonar sensor signals are prone to noise, some form of preprocessing is desirable. In these experiments a smoothing procedure is applied by delaying any large change in the input data by a pre-defined number of time steps, with the reading in question simply taking on its old value for the duration of this period. If the new data value still holds on completion of the allotted interval, it is then allowed to pass through to the classifier as part of the input vector. The purpose of this mechanism was to remove "spikes" from the input data. Both the amount of change defined as large, θ, and the number of time steps to skip, γ, were determined experimentally by observing data time series from several locations within the computer science building.

Building the vector map The vector map is built by the robot as it is led around the environment by the user. Movement of the robot is restricted to forward or turn (these commands cannot be carried out simultaneously), and the robot travels at a constant speed of 4 inches/sec.

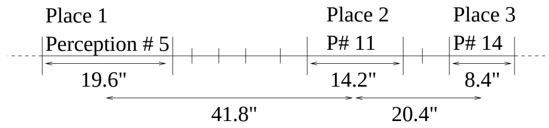
As the robot moves forward it continuously takes in all 16 sonar readings and processes them using the RCE-classifier, with preprocessing as detailed above, to generate the 'perceptual landmarks'. Each time the perception changes the robot takes note of the distance travelled since the start of the last perception. As some of these landmarks can occupy very small areas, and as such can be easily missed on subsequent visits to the same location, only landmarks that persist for a pre-determined minimum distance are considered for the purpose of navigation. In the results detailed here, only landmarks that are visible over a travelling distance of over 4 inches are added to the vector map.

Each place node of the vector map consists of a perception number and a list of the places connected to that node. For each connected place, details of compass direction, distance between the centres of the nodes, and size of the place node's influence are stored.

Each time a place node is allocated it is added to the list of connections for the previous node. In addition, this previous node's details are added to the current node's list (i.e. each link is recorded bi-directionally). Figure 4 shows an example of three connected place nodes and the corresponding vector map.

Storing the size of the place node's influence area allows the robot to travel to the centre of this node once found. Note that this size can be multiply defined for one particular place node, this is necessary since the size of a node's influence will depend on the direction from which it is approached (see figure 5). The distance measurement gives an upper bound on distance to travel in searching for the connected place.

Perceptual aliasing The system, as proposed so far, takes no account of perceptual aliasing (identical sensory perceptions at different locations). As such, distinct locations within the environment that have similar perceptual signatures may excite the same place node. One solution to this problem would be to allow the robot to explore the environment on finding a location whose perception has previously been encountered. Exploration in




```
Place 1
Perception number: 5
Number of links: 2
    Link 0:
        Place: 0
        Direction: 31
        Distance: 338
        Size: 224
    Link 1:
        Place: 2
        Direction: 211
        Distance: 418
        Size: 142
```

```
Place 2
Perception number: 11
Number of links: 2
    Link 0:
        Place: 1
        Direction: 29
        Distance: 418
        Size: 196
    Link 1:
        Place: 3
        Direction: 209
        Distance: 204
        Size: 84
```

```
Place 3
Perception number: 14
Number of links: 2
    Link 0:
        Place: 2
        Direction: 26
        Distance: 204
        Size: 142
    Link 1:
        Place: 4
        Direction: 207
        Distance: 150
        Size: 144
```

Figure 4 Three connected place nodes and the corresponding vector map. Start and end points for each perception (as given by the RCE-classifier) are denoted by vertical lines along the path. Here only three 'landmarks' (labelled *5*, *11* and *14*) are persistent enough to warrant place node generation, these landmarks have been given the corresponding place labels, *1*, *2* and *3* in the vector map. Distance and size in the vector map are measured in tenths of inches.

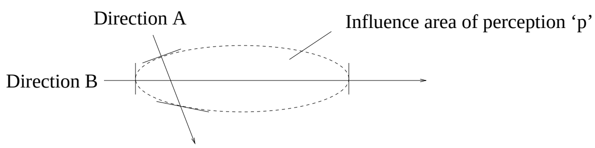


Figure 5 An example of multiply defined place size. The dotted ellipse indicates the influence area of perception 'p'. The arrows indicate two different directions from which this perception was encountered, the 'size' of 'p' measured for direction A will be smaller than that for B.

this instance would involve following the links given for each place node that shares the current perception. If, for a given place node, the correct perception is encountered along each of the links, then the system assumes that this place has been re-encountered and adds new links accordingly. Otherwise, a new place node is created. An exception to this behaviour is made when the robot moves from a previously encountered place node along a link already contained in that node's link list; if the place found along this link is as expected then no exploration is performed (i.e. the robot uses prediction when traversing previously followed paths). We call this strategy *incremental mapbuilding*, and this is the mechanism used in the experiments described here.

2.2 Map interpretation

The object of the map interpretation phase is to plan a route from the current location to an arbitrary, externally specified goal location using the vector map. The basis of the planning mechanism is the 'best-first search' algorithm, which is used in this instance to determine the *shortest* known path between the current location and the goal location.

Path following Once a path to the goal has been generated the robot's path following task can begin. For this task the robot takes each node in turn and attempts to find the next node along the path by taking the trajectory indicated on that path (the goal and starting location being provided by the user).

If the robot fails to find a given node then it reverses to the centre of the node on the path that was last identified. From here a new path is calculated with this node as the start position.

The goal location is only assumed to have been found when it appears in the right context according to the planned path (i.e. if the robot happens to encounter a place with the same perception as the goal before it appears on the planned path, then it is assumed *not* to be the goal location).

3. Experimental results

In this section we present the results of testing carried out with the system in a simple environment set up within the robotics laboratory at Manchester. Figure 6 shows the environment used in the experiments, the route along which the robot was guided through the environment (chosen arbitrarily), and the corresponding map generated by the system.

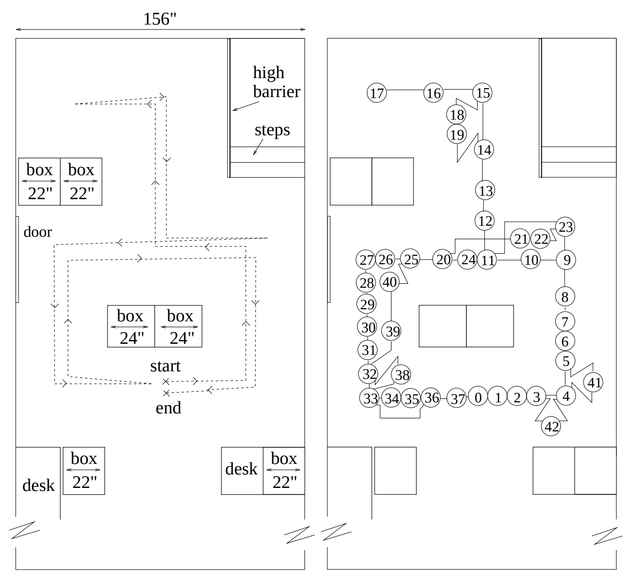


Figure 6 The left diagram shows the environment used in the experiments, the dotted line depicts the path along which the robot was guided. The corresponding map generated by the system is shown on the right (each circled number indicates a place node on the map).

For this experiment four different goal and start positions were chosen from the environment so as to cover the area traversed by the robot during mapbuilding. The results for each of these trials are given below. In each diagram the shortest route from start to goal is shown, along with any alternative routes taken by the robot. Each of the four trials was completed ten times by the robot.

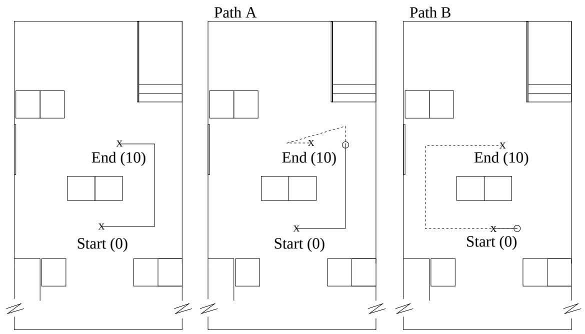


Figure 7 Trial 1. The figure on the left shows the shortest path from start location to goal location. In 8 out of 10 attempts the robot traversed this path successfully. The other two figures show two alternative routes taken by the robot due to failure to locate place nodes along the shortest path. The point at which failure occurred is indicated by a circle, the dotted line denotes the new path.

Trial 1 (figure 7): In 8 out of the 10 attempts made by the robot, the shortest route was traversed successfully. For the remaining two attempts, the robot failed to locate one of the place nodes on each occasion, thus initiating a new path planning phase. In both cases the robot was able to follow the new path successfully to the goal location (see figure 7).

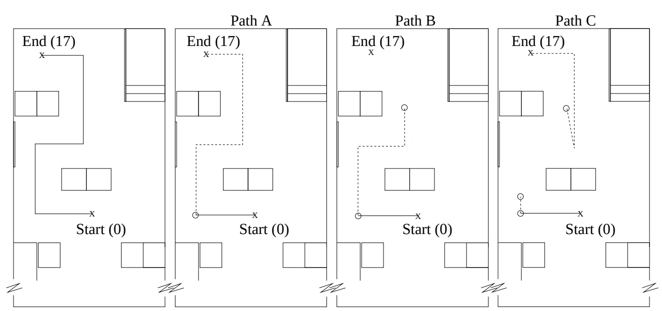


Figure 8 Trial 2. The figure on the left shows the shortest path from start location to goal location. In 6 out of 10 attempts the robot traversed this path successfully. In two of the remaining cases only one new path needed to be generated (see path A). Path A looks identical to the shortest route due to the fact that re-planning was possible via alternative nodes along this route. Of the remaining two attempts, one required two new paths to be generated (path B), whilst the other required three (path C). Each new path is indicated by a dotted line with a circle at the start point.

Trial 2 (figure 8): In 6 out of the 10 attempts for this trial, the shortest route was traversed successfully. Of the remaining four attempts, two required one new path generating phase in order for the robot to locate the goal, one required two and the other required three (see figure 8).

Trial 3 (figure 9): In this trial, for all 10 attempts the robot successfully traversed the shortest path.

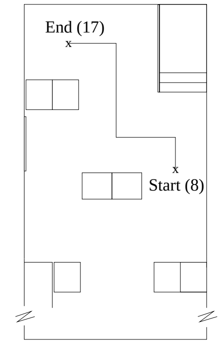


Figure 9 Trial 3. This figure indicates the shortest path from start to goal for this trial. In all 10 attempts this path was traversed successfully.

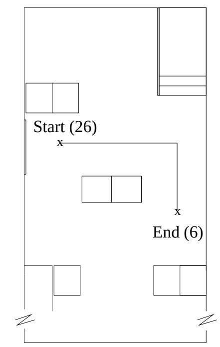


Figure 10 Trial 4. This figure indicates the shortest path from start to goal for this trial. In all 10 attempts this path was traversed successfully.

Trial 4 (figure 10): As in trial 3, for all 10 attempts the robot successfully traversed the shortest path.

As can be seen from trials 1 and 2, the robot is not always able to find a place at the location given by the map. This phenomenon can be particularly prominent at locations near to junctions or cluttered areas where even small deviations in the robot's position and/or orientation can substantially alter the robot's perception. However, the effects can be alleviated somewhat by extending the training period, thus giving opportunity for acquiring alternative perceptions for locations where perception is problematic. By taking the robot repeatedly to the same physical location, path planning is facilitated using alternative perceptions associated with that location.

3.1 A performance metric

In order to gain some quantitative measure of the robot's performance in its navigation task, we defined a metric, *distance efficiency*. To obtain this metric, the length of a particular run made by the robot is measured as a percentage of the shortest possible distance between the start and goal locations. The results for the distance efficiency metric are shown in figure 11.

From this graph we can see that trials 1 and 2 were problematic for the robot. As discussed in section 3., this

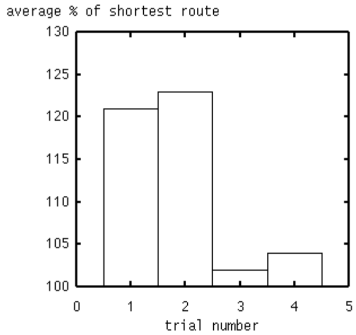


Figure 11 Distance efficiency. For each trial the total shown is an average over 10 traversals from start to goal location.

was due to the robot's inability to locate certain nodes along the generated path, thus resulting in new path planning and route traversal phases.

A particularly poor performance in trial 1 is due to the fact that, on one occasion, an alternative route along the shortest path was not available, thus requiring a lengthy detour in order to reach the goal (see path B, figure 7). In trial 2, replanning was required in four out of the ten attempts, with certain nodes proving consistently difficult to detect (see figure 8).

4. Conclusions and further work

In this paper we have presented a navigation system for a mobile robot, and tested it experimentally on a Nomad 200 mobile robot. The system uses a self-organising mechanism to build a representation of the environment based on landmark recognition. The representation takes the form of a vector map whereby recognised locations within the environment are linked together by arcs denoting direction and distance between them. Exploration is used both to disambiguate perceptually similar locations, and to recognise places that have been previously visited. Global positional information is not required. A metric was introduced (Distance Efficiency) and used to measure the robot's navigational behaviour.

In order for a robot navigation system to be useful it must operate in real world situations (Owen and Nehmzow, 1997). Much of the work to date in the field of mobile robot navigation has been conducted in small scale laboratory environments - the question arises of how these systems will scale up when faced with more complex environments covering much larger distances. Large scale experiments with the navigation mechanism described here are currently underway at Manchester University.

Our results show that the robot is not always able to find a place at the location indicated by the map. Rather than re-plan an alternative route immediately on failure to find a place node, a better approach would be to make a more concentrated effort to locate the lost node. To do this, one could simply re-attempt to locate the node along the trajectory on which failure occurred, or perform a more systematic search strategy. In doing this, lengthy detours such as the one depicted in figure 7 (path B) might be avoided. In addition, this behaviour gives the robot the opportunity to reach the goal where no alternative routes are available.

Although not tested in these experiments, one can assume that the system has some ability to cope with *dynamic environments*. If, for example, a path becomes blocked then the perception near the blocked position will change, thus forcing the robot to plan an alternative route (the perception will not be as expected). This property needs to be tested experimentally. In addition, some way to store information regarding such environmental change would be useful.

With the present system, mapbuilding and navigation are carried out as two separate phases. For dynamic environments this approach is unsuitable since changes to the robot's surroundings will introduce errors into the map. A more appropriate strategy would be to intersperse mapbuilding with map interpretation (lifelong learning). With this approach the robot would continue to map new information as it attempts to follow a path.

References

aisb97 (1997). *AISB–97 Workshop on Spatial Reasoning in Mobile Robots and Animals*, Technical Report Series UMCS-97-4-1, ISSN 1361-6161, http://www.cs.man.ac.uk/csonly/cstechrep/titles97.html. University of Manchester.

Carpenter, G. and Grossberg, S. (1987). Art2: Self-organisation of stable category recognition codes for analog input patterns. *Applied Optics*, 26(23):4919–4930.

Duckett, T. and Nehmzow, U. (1997). Experiments in evidence-based localisation for a mobile robot. In (aisb97, 1997).

Kampman, P. and Schmidt, G. (1991). Indoor navigation of mobile robots by use of learned maps. In (Schmidt, 1991).

Knieriemen, T. and von Puttkamer, E. (1991). Real-time control in an autonomous mobile robot. In (Schmidt, 1991).

Kohonen, T. (1988). *Self Organisation and Associative Memory*. Springer Verlag, Berlin, Heidelberg.

Kortencamp, D. and Weymouth, T. (1994). Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proceedings of the AAAI*.

Kurz, A. (1996). Constructing maps for mobile robot navigation based on ultrasonic range data. *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, 26(2):233–242.

Mataric, M. J. (1992). Integration of representation into goal-driven behaviour-based robots. *IEEE Transactions on Robotics and Automation*, 8(3).

Nehmzow, U. and Smithers, T. (1991). Mapbuilding using self-organising networks. In Arcady Meyer, J. and Wilson, S., editors, *From Animals to Animats*, Proceedings of SAB90 Paris, pages 152–159. MIT Press, Cambridge Mass. and London, England.

Owen, C. and Nehmzow, U. (1996). Route learning in mobile robots through self-organisation. In *Euromicro workshop on advanced mobile robots*. IEEE Computer Society, ISBN 0-8186-7695-7.

Owen, C. and Nehmzow, U. (1997). Middle scale robot navigation - a case study. In (aisb97, 1997).

Reilly, D. L., Cooper, L. N., and Erlbaum, C. (1982). A neural model for category learning. *Biological Cybernetics 45*, pages 35–41.

Schmidt, G., editor (1991). *Information Processing in Autonomous Mobile Robots*. Springer Verlag, Berlin, Heidelberg, New York.

Yamauchi, B. and Beer, R. (1996). Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, 26(3):496–505.

Zimmer, U. R. (1996). Robust world modelling and navigation in a real world. *Neurocomputing*, 13(2–4):247–260.