Exploration of Unknown Environments Using a Compass, Topological Map and Neural Network

Tom Duckett and Ulrich Nehmzow
Department of Computer Science
University of Manchester
Manchester M13 9PL
United Kingdom
duckettt@cs.man.ac.uk, ulrich@cs.man.ac.uk

Abstract

This paper addresses the problem of autonomous exploration and mapping of unknown environments by a mobile robot. A map-based exploration system is presented, in which a topological map of the environment is acquired incrementally by the robot, using an artificial neural network to detect new areas of unexplored territory. Using this approach, no manual intervention in the map acquisition process is required, and all computation is carried out in real-time on board the robot. Experiments are presented in which a Nomad 200 robot successfully mapped and navigated complex, real world environments containing transient changes such as moving people.

1 Introduction

In recent years, there has been a great deal of research on the topic of mobile robot navigation, and a number of successful navigation systems have been produced. Many systems either rely on pre-installed maps [14], or use passive mechanisms to build maps while the robot is manually steered around the environment by a human operator [8, 6]. In other systems, the sensor-motor data required for map building is first collected by the robot under manual control, then an off-line learning algorithm is used to find the best map to fit the data [13].

While all of the methods described above have their merits, manual intervention is costly and prone to human error. Similarly, reactive behaviours such as wall-following, though often very robust, cannot be guaranteed to build complete maps in large, complex environments. We therefore believe that the most flexible approach is for the robot to acquire its own maps through a process of autonomous, map-based exploration. In other words, the robot should be able to identify regions of unexplored territory, navigate to the identified areas using its own map, and incrementally update this map at the same time.

A version of the latter strategy was used here, in which the robot continuously tries to expand the territory which has already been charted. The basic idea is that the robot travels to the edge of the existing map, and then uses its range-finder sensors to detect new regions of unexplored territory. The new territory is added to the map, then the robot attempts to travel to the next unexplored edge of the map. The process is repeated until the robot has covered the entire environment.

The robot uses a graph-based representation of its environment, in which the nodes correspond to contiguous regions known as *places* and the links to possible transitions between places. A topological rather than metric representation was used because metric maps require large amounts of computation and also depend on precise position information for map learning. These requirements are particularly hard to satisfy in larger environments, especially if fully autonomous operation is required.

The approach differs from previous work in that it does not require high precision sensing or depend upon simplifying assumptions about particular environments, and has been tested in populated, real world environments. An artificial neural network is used to detect areas of unexplored territory, fusing together information from the robot's sonar and infrared sensors. All of the data required for training the network is collected by the robot itself, avoiding the need for

[0]Published at 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation, Monterey CA.

the system designer to determine the training signal. The complete system requires only minimal computational resources due to the compactness of its representations, thereby eliminating the need for off-line processing and increasing the autonomy of the robot.

In this paper, we assume that the robot has the ability to determine its own location in the topological map; full details of the self-localisation mechanism used in these experiments can be found in [3].

1.1 Related Work

Yamauchi [18] developed a technique called "frontier-based" exploration, in which a global occupancy grid [11] was used to represent the environment. Image segmentation techniques were used to extract regions in the grid between charted and unknown territory known as "frontiers". Exploration was then directed towards the frontiers. A disadvantage of this approach is that it depends critically upon accurate laser sensors and precisely corrected odometry, because exact position information is needed to update grid-based maps.

Thrun [15] also developed a map building system using a global occupancy grid. An artificial neural network was trained to translate neighbouring groups of sonar sensor readings into occupancy values in the grid. Exploration was then directed towards areas of high uncertainty in the acquired map. The required training examples were obtained using a simulator, though the trained neural networks were shown to work well on the real robot.

Edlinger and Weiss [5] developed a map building system in which the robot's map consisted of a set of laser range-finder scans and the topological relations between the scans. The system attempted to detect obstacle-free segments in the scans known as "passages", that is, regions of open space which are wide enough for the robot to move into. The detected passages were added to a stack of unexplored locations, which were visited in turn until the whole environment had been covered by the robot.

2 Exploration Strategy

The robot builds a topological map, which is augmented with metric information concerning the distance and angles between connected places. The map contains two different types of places (figure 1):

- **Predicted.** Places presumed to exist but not yet visited by the robot.

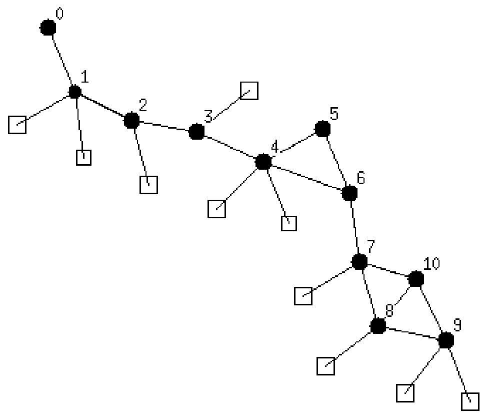

Figure 1: Example of Topological Map Building. Places predicted by the neural network but not yet visited by the robot are shown by squares. Places visited by the robot are shown by filled circles.

- **Confirmed.** Places actually visited by the robot.

Exploration consists of continuously trying to expand the territory already charted by the robot, using a neural network to add new 'predicted' places to the map. Subsequent movement by the robot is used to verify whether the 'predicted' places actually exist or not.

From its initial location, the robot adds the first set of 'predicted' places to the map, and then attempts to navigate to the nearest 'predicted' place. If the robot is able to move to a physically distinct new location in the environment without encountering any obstacles, the 'predicted' place is replaced by a 'confirmed' place, otherwise it is deleted. Whenever another 'confirmed' place is added to the map, the neural network is used again to predict further new places. This process is repeated until all 'predicted' places in the map have either been visited by the robot or deleted.

In order to implement this exploration strategy, the following mechanisms were required:

1. **Location Recognition.** We assume that the robot has the ability to locate itself within the map. The self-localisation algorithm described in [3] was used here; this algorithm is able to determine the most likely place occupied by the robot, and also the most likely displacement of the robot within each of the possible places.

2. **Open Space Detection and Compass Sense.** In order to add the new 'predicted' places to the

map, the robot requires the ability to determine its orientation (see section 3). In addition, some mechanism is required to add the new 'predicted' places to the map, i.e., to detect areas of unexplored territory in a particular direction. Individual range-finder readings are not well suited for this purpose because of problems such as occlusions caused by moving people, sensor noise, cross-talk and specular reflections. Instead, an artificial neural network was trained to learn the concept of "open space", combining noisy information obtained from many sensor readings (see section 5).

3. **Way Finding.** Dijkstra's algorithm was used for path planning. The robot's heading was controlled by taking into account the robot's current location in the map, the compass sense and the shortest path to the goal location. A pre-trained behaviour for moving forwards while avoiding obstacles was used for low level sensor-motor control [12].

4. **Local Dead Reckoning.** In order to determine whether a new 'confirmed' place should be added to the map, a local dead reckoning strategy was used (see section 4). If the robot managed to travel by a pre-specified distance threshold ($1m$) from the nearest stored place in the map without encountering any obstacles, then a new 'confirmed' place was added to the map.

5. **Consistency Maintenance.** Dead reckoning cannot be used for global position estimation during map building, due to the accumulated drift errors caused by wheel slippage. Therefore, some other mechanism was required to assign globally consistent coordinates to the places in the robot's map, using only the local metric relations between the places (see section 6).

The system was implemented on a Nomad 200 robot equipped with sonar, infrared and odometry sensors and a flux-gate compass (figure 2).

3 Compass Sense

A separate behaviour was used to rotate the robot's turret at small speeds in the direction of 'North', as indicated by the flux-gate compass. The effect of this behaviour was to smooth out local fluctuations in the magnetic field of the robot's environment. Using the compass in this way gave the robot a single view of each location, i.e., the appearance of locations to the robot depended on the robot's position alone, not its orientation.

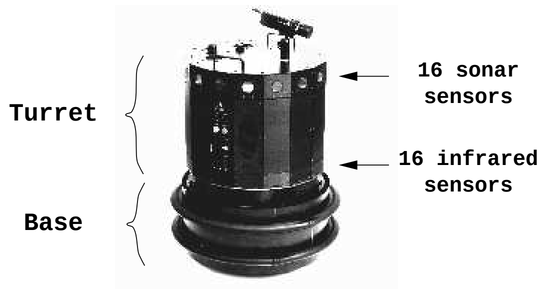

Figure 2: The Nomad 200 Mobile Robot *FortyTwo*. The sonar and infrared sensors are mounted on the turret, which can rotate independently relative to the base of the robot.

While this method is robust in dealing with minor variations in the magnetic field, severe compass errors caused by ferrous building materials could pose a problem in some environments. A more reliable compass sense could be obtained by integrating perceptual information from the robot's exteroceptive and proprioceptive sensors, as in the self-orientation system described by Li *et al* [9], or by using correlation with a vision-based map of the ceiling as in Thrun *et al* [16].

4 Dead Reckoning

Instead of using the robot's rotational wheel encoders for the on-line dead reckoning, we used the relative angular displacement of the turret from the base of the robot. This had the effect of removing the accumulated angular drift affecting the robot's raw odometry (figure 3), because the turret was anchored to 'North' by the compass sense. Using compass-based odometry leaves a translational drift error of approximately 2-5% of distance travelled.

5 Open Space Detection

A fully connected, feedforward neural network with 6 inputs, 3 hidden units and 1 output was trained to associate the sensory input in a given direction with the robot's ability to move by a pre-specified distance ($1m$) in that direction. The output of this network was the probability of open space in the given direction (figure 4).

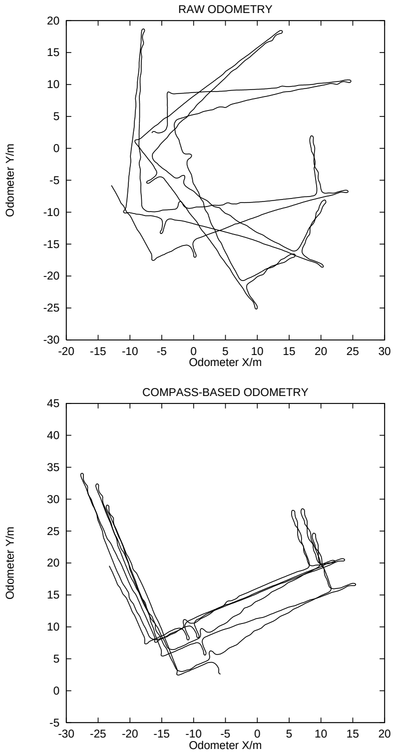

Figure 3: Top: Raw Odometry. Bottom: Compass-Based Odometry. The accumulated rotational drift in the robot's raw odometry was removed on-line using the compass sense.

5.1 Data Collection

The sensing strategy used by the robot consisted of rotating its turret to obtain a detailed scan, consisting of 144 sonar and 144 infrared readings at 2.5 degree intervals. For data collection, a scan was first taken, then the robot attempted to move as far as possible in an arbitrary direction until an obstacle was detected, recording both the sensor readings from the scan and the distance travelled. The data was collected in several different areas of the computer building at Manchester.

5.2 Pre-Processing

The recorded sensor readings were first processed to take into account the heading of the robot. A subset of 58 of the readings, centred around the direction of travel, was used as input to the classification mechanism. The following functions were then applied:

Median Filter. The robot's raw sensor readings rarely give accurate range measurements; the values may be too high, e.g., due to specular reflections, or too low, e.g., due to cross-talk or occlusions caused by moving people. To reduce these effects, groups of 5 or 6 adjacent sensor readings were combined to produce a single reading by taking their median value. This resulted in five sonar and five infrared inputs to the next pre-processing stage.

Symmetry Filter. This function was used to exploit the bilateral symmetry inherent in this classification task. For example, the left-most sonar reading was combined with the right-most sonar reading by taking the minimum of the two values (i.e., the nearest of the two obstacles detected). The middle-left and middle-right readings were similarly combined. This resulted in three sonar and three infrared inputs to the neural network, since the centre readings were not affected by this operation.

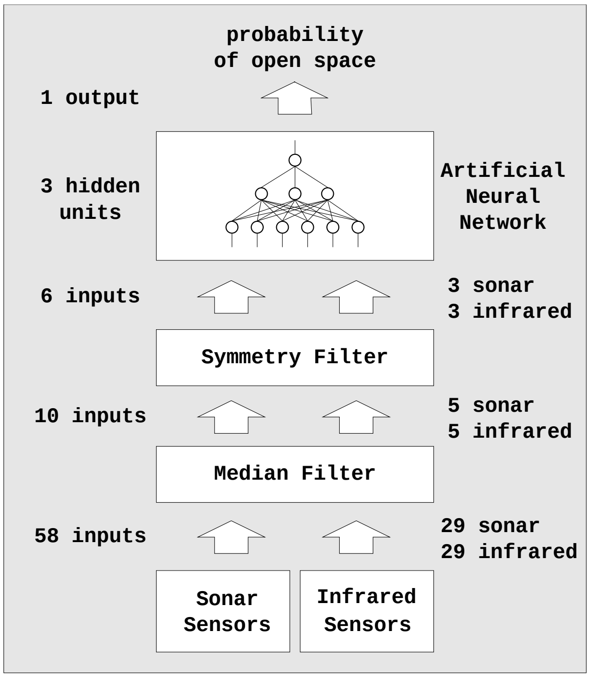

Figure 4: Architecture for Open Space Detection. Two pre-processing functions were applied to the sensory input, then an artificial neural network was used to detect the presence or absence of open space in a given direction.

5.3 Training and Testing

A key issue was that of misclassification errors. Though the performance of the network used here was very good (see results), any classification mechanism is bound to make some errors. These errors will either be 'false positives', where the robot predicts open space when the space is actually occupied, or 'false negatives', where the robot predicts occupied space when the space is actually open. In the exploration strategy presented, 'false positives' are not a major problem, because subsequent movement by the robot is used to verify whether the predicted places actually exist. However, 'false negatives' would pose a problem, as we do not want the robot to miss any areas of unexplored territory.

The solution adopted here was to bias the classifier mechanism into over-estimating the likelihood of open space in a given direction, thereby producing more 'false positives' but fewer 'false negatives' (none in the experiments presented here). The network was trained to output the probability of open space by using the cross-entropy error function instead of the sum-of-squares error normally used in neural network training [1]. During testing, a bias value (0.15) was added to the output of the network in order to produce the desired over-estimates. An input pattern was thus classified as "open space" if the output was greater than a threshold of 0.5, and "occupied space" otherwise.

6 Map Learning

6.1 Local Metric Relations

Whenever the robot moved between two distinct places i and j for the first time, a new topological connection was recorded in the map. In addition, the distance d_{ij} and heading θ_{ij} of the robot between the two places was recorded. The links between places were constrained to be bi-directional, that is, $d_{ij} = d_{ji}$ and $\theta_{ij} = \theta_{ji} + \pi$. These measurements were obtained using local dead reckoning and matching of local range-finder scans constructed from the robot's sonar readings [3] (see also [17, 10]).

During subsequent traversals of an existing link in the map, the measurements associated with the link were adapted using the following rules taken from Yamauchi and Beer [19]:

$$\begin{aligned} d'_{ij} &= \lambda d_{obs} + (1-\lambda) d_{ij}, \\ \theta'_{ij} &= \tan^{-1}\left(\frac{\lambda \sin\theta_{obs} + (1-\lambda)\sin\theta_{ij}}{\lambda\cos\theta_{obs} + (1-\lambda)\cos\theta_{ij}}\right), \end{aligned}$$

where the vector (d_{obs}, θ_{obs}) refers to the observed displacement, i.e., distance and heading, of the robot between the two places, and the link adaptation rate, $\lambda = 0.5$ in these experiments.

6.2 Consistency Maintenance

The problem addressed here was how to assign globally consistent coordinate values to the places in the robot's map. Each place in the map was represented by a Cartesian coordinate (x_i, y_i). A relaxation algorithm was used to find an optimal set of coordinates to fit the observed measurements, using only the local metric relations between the places.

In this approach, each link in the map can be modelled as a spring which connects two adjacent places i and j. The spring reaches minimum energy when the relative displacement between the coordinates of i and j is equal to the vector $(d_{ij},\ \theta_{ij})$ measured by the robot [10, 7]. Thus, global consistency is maintained in the map by minimising the following energy function:

$$E = \sum_i \sum_j{}' \left(x_i - x_j + d_{ij}\cos\theta_{ij}\right)^2 + \left(y_i - y_j + d_{ij}\sin\theta_{ij}\right)^2,$$

where $\sum_j'$ refers to the sum over the neighbours of a given node. There are a number of different algorithms which can be applied to solve this particular optimisation problem, including Gaussian elimination, stiffness methods and expectation-maximisation [4, 10, 7, 13].

7 Results

The neural network was trained using 276 examples to perform the open space classification task, resulting in a test error of 7.6%, A validation error of 4.0% was observed during the testing of the complete exploration system (this was lower than the test error because the data used for training and testing contained a higher proportion of "difficult" examples, such as junctions and corners).

The map-based exploration strategy was tested successfully in a number of untreated, real world environments at our computer building, which contained transient changes such as moving people, doors opening and closing, *etc.* An example map acquired by the robot in a corridor environment of size $34m \times 33m$ is shown in figure 5.

To assess the quality of the maps obtained, we considered the robot's ability to navigate using its own self-acquired map. Firstly, we considered the robot's

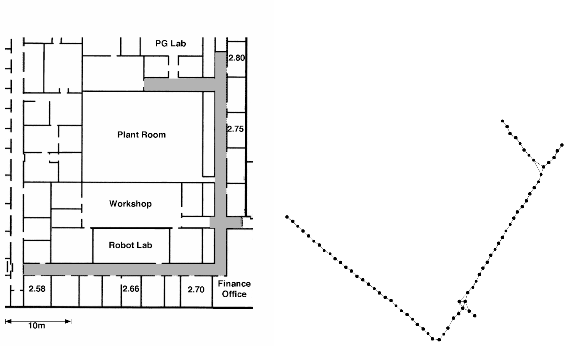

Figure 5: Left: floor plan of a corridor environment. Right: the corresponding map acquired by the robot.

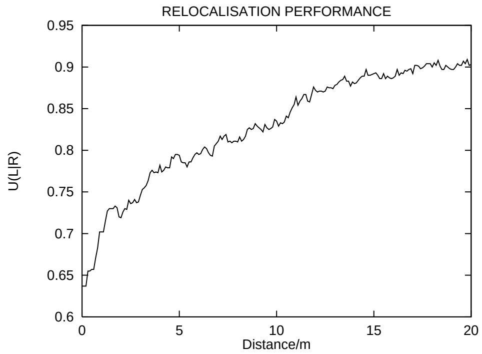

Figure 6: Relocalisation performance $U(L \mid R)$ under global uncertainty in the corridor environment.

ability to relocalise under global uncertainty, i.e., to recover its position after becoming lost. To assess localisation performance, the *Uncertainty Coefficient* $U(L \mid R)$ was measured against the distance travelled by the robot from an unknown starting position using wall-following (see figure 6). This statistic measures the extent to which the robot's response, R (the location estimates produced by self-localisation) predicts the robot's true location, L, as

$$
\begin{aligned}
U(L \mid R) &\equiv \frac{H(L) - H(L \mid R)}{H(L)}, \\
H(L) &= -\sum_j p_{\bullet j} \ln p_{\bullet j}, \\
H(L \mid R) &= -\sum_{i,j} p_{ij} ln \frac{p_{ij}}{p_{i\bullet}},
\end{aligned}
$$

where $p_{\bullet j} = \sum_i p_{ij}$, $p_{i\bullet} = \sum_j p_{ij}$, and p_{ij} refers to the probability of the robot's response being i when the robot's true location was j. Full details of the experimental procedure used to assess localisation performance can be found in [2].

Finally, the exploration system was validated through its integration into a complete navigation system [3]. The robot had to perform a delivery task, finding 100 different routes through the corridor environment in figure 5. The success rate was 100% when starting from a known location and 92% when starting from a completely unknown location, indicating the effectiveness of our approach.

8 Discussion

In this paper, we presented a complete map-based exploration system for a mobile robot. The basic mechanisms included a compass, a topological map augmented with metric information and a neural network trained to detect areas of open space, combined with our previous work on self-localisation using landmarks [2, 3]. Using this approach, real world environments of several hundred m^2 were mapped independently by a Nomad 200 robot without requiring off-line processing or human intervention in the exploration process.

Future work will need to examine the problem of self-orientation in more detail to improve the reliability of the compass sense (section 3). Another fundamental problem for any navigating robot is to build consistent maps in very large environments containing loops. So far, mobile robots have only been successful in "closing the loop" by using accurate range-finder sensing and precisely corrected odometry [15]. This approach will inevitably fail once the size of the environment is increased beyond the limits of the robot's mechanisms for position correction.

References

[1] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, ISBN 0-19-853864-2, 1995.

[2] T. Duckett and U. Nehmzow. Mobile robot self-localisation and measurement of performance in middle scale environments. *J. Robotics and Autonomous Systems*, 24(1–2), 1998.

[3] T. Duckett and U. Nehmzow. Knowing your place in real world environments. In *EUROBOT '99, 3rd European Workshop on Advanced Mobile Robots.* IEEE Computer Society Press, 1999.

[4] H.F. Durrant-Whyte. *Integration, Coordination and Control of Multisensor Robot Systems.* Kluwer Academic Publishers, Boston MA, 1988.

[5] T. Edlinger and G. Weiss. Exploration, navigation and self-localisation in an autonomous mobile robot. In *Autonomous Mobile Systems*, 1995.

[6] S. Engelson. *Passive Map Learning and Visual Place Recognition.* PhD thesis, Dept. of Computer Science, Yale University, 1994.

[7] M. Golfarelli, D. Maio, and S. Rizzi. Elastic correction of dead-reckoning errors in map building. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 905–911, 1998.

[8] D. Kortenkamp and T. Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *AAAI-94*, pages 979–984, 1994.

[9] G. Li, B. Svensson, and A. Lansner. Self-orienting with on-line learning of environmental features. In *AISB '97 Workshop on Spatial Reasoning in Animals and Robots*, 1997.

[10] F. Lu and E.E. Milios. Globally consistent range scan alignment for environment mapping. *J. Autonomous Robots*, 4:333–349, 1997.

[11] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 116–121, St. Louis, Missouri, 1985. IEEE Computer Society Press.

[12] U. Nehmzow. Autonomous acquisition of sensor-motor couplings in robots. Technical Report ISSN 1361-6161, UMCS-94-11-1, Dept. of Computer Science, Manchester University, 1994.

[13] H. Shatkay. *Learning Models for Robot Navigation.* PhD thesis, Dept. of Computer Science, Brown University, 1998.

[14] A. Stevens, M. Stevens, and H. Durrant-Whyte. OXNAV : Reliable autonomous navigation. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 2607–2612, 1995.

[15] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99, 1998.

[16] S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Daellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA : A second-generation museum tour-guide robot. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1999.

[17] G. Weiss and E. von Puttkamer. A map based on laser-scans without geometric interpretation. In *Proc. Intelligent Autonomous Systems 4 (IAS-4)*, pages 403–407, 1995.

[18] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proc. IEEE Int. Symposium on Computational Intelligence in Robotics and Automation*, 1997.

[19] B. Yamauchi and R. Beer. Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man and Cybernetics Part B*, 26(3), 1996.