Knowing Your Place in the Real World

Tom Duckett and Ulrich Nehmzow
Department of Computer Science,
University of Manchester,
Oxford Road,
Manchester M13 9PL,
England.
t.duckett@cs.man.ac.uk, u.nehmzow@cs.man.ac.uk

Abstract

This paper addresses the scalability of existing techniques for mobile robot navigation to work in large, unstructured environments. In particular, we are interested in the problem of relocalisation after the robot has been placed at some arbitrary location (unknown to the robot) in a previously explored environment. In the first instance, we assess the problems found when a system previously shown to work in a small-scale environment was transferred to a middle-scale environment.

Through analysis of the results, we are able to show that successful performance in the middle-scale environment requires a global compass sense. We provide a suitable quantitative measure for assessing localisation quality in the larger environment, and are able to quantify *post hoc* the performance which would be obtained given an appropriate compass mechanism. In addition, we use the same quality metric to evaluate various different approaches for pre-processing the robot's sensory information.

1 Introduction

Many researchers in the field of mobile robot navigation have now begun to evaluate their work on real robots rather than in simulation. So far, however, this has been mainly in specially constructed laboratory environments, and few have taken working systems out into larger, unstructured environments to assess their scalability. Furthermore, no generally accepted quantitative measures of performance currently exist in the field for assessing the behaviour of the robot within the laboratory [11], let alone "real world" environments. This work therefore represents a case study in attempting to quantify robot performance outside of the laboratory, and shows clearly that small-scale and middle-scale navigation present different sets of problems for mobile robotics researchers.

In previous research [2] [3], we addressed the problem of combining *exteroceptive* sensory information (from the robot's range-finding sensors) with *proprioceptive* sensory information (from the robot's wheel encoders) to enable the robot to establish its location from an unknown starting position. We showed that neither exteroceptive or proprioceptive sensory information alone was sufficient for the robot to relocalise without requiring *a priori* knowledge of its approximate location. We therefore developed a system which combined both range-finding sensor readings and *relative* use of odometry, enabling relocalisation even in environments containing no single globally distinctive place. However, this system was tested only in a laboratory environment and the question of scalability was not addressed at the time.

In the following experiments, we present a method for evaluating the performance of a mobile robot relocalisation system in middle-scale environments[1]. This addresses both the question of how to measure independently the true position of the robot, and how to assess the correlation over time between the response of the robot and the location recorded by the observer. In the subsequent analysis, we quantify the quality of localisation possible using the algorithm presented here together with a global compass sense. In addition, we use the same quality metric to compare various different approaches found in recent robotics literature for pre-processing the robot's sensory input during mapbuilding and relocalisation.

2 System Architecture

The mapbuilding and localisation system evaluated here is composed of a hierarchy of modules (see figure 2), executed concurrently on a Nomad 200 robot (see figure 1). During both mapbuilding and relocalisation, the robot explores its environment using a reactive behaviour previously acquired using instinct rules [10]. In the experiments presented here, only wall-following was considered, although the system was shown previously to function independently of any particular exploration strat-

[1] Navigating in environments of several hundred m^2 in size.

Figure 1: *The Manchester 'FortyTwo'.* The robot's 16 sonar and 16 infrared sensors are spaced at equal intervals around the turret. In these experiments, the turret was kept at a constant orientation to provide the robot with a "canonical view" of places, i.e., independent from the direction of travel. Exploration was carried out using the translational and rotational motors located in the base of the robot.

egy [2].

The robot begins by building a map of its environment, using a self-organising classifier mechanism to cluster its range-finding sensor readings. (The neural network classifier ART2 [1] was used in initial experiments; various other mechanisms are compared in section 4.) We then move the robot to some arbitrary position in the environment, its sensors being disabled during that move. The robot then begins to actively explore its new surroundings, and attempts to relocalise on the basis of evidence accumulated from both relative odometry and its current sensor data. Using this sensory evidence, the robot is able to choose between the competing place memories contained in the map. A winning location "hypothesis" emerges once a perceptually unique path has been traced through the map.

2.1 Map Building

The map constructed by the robot consists of a set of place memories, each consisting of a sensor prototype (e.g., ART2 classifier category) associated with an (x, y) coordinate obtained by dead-reckoning (see figure 3). This is similar to the mapbuilding mechanism described by Kurz [4], where the robot's (x, y) coordinates are averaged as the robot moves through a particular "situation area", i.e., an area of Cartesian space associated with a particular classifier category. A new place is entered into

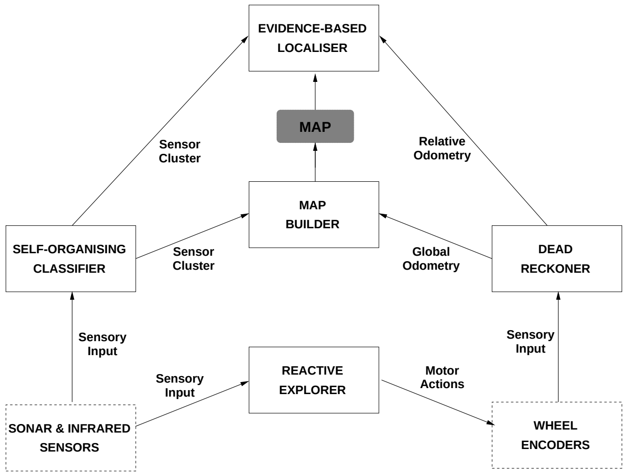

Figure 2: *System Architecture.* The solid-edged boxes denote behavioural modules, and arrows the dependencies between different parts of the system. The shaded box denotes the representation used for the map, and the dashed boxes show hardware components.

the map whenever the robot detects a significant change in its sensor data (e.g., whenever the currently perceived ART2 category changes) or the robot has travelled by more than a prespecified distance threshold D from the nearest stored point in the map[2].

2.2 Relocalisation

The relocalisation system works by comparing old and new information at each iteration, or in other words, by considering the *changes* in the moving robot's perceptions over time. In this system, the robot maintains a set of competing place hypotheses (based on the place memories stored in the map), each being associated with a confidence level reflecting the robot's certainty in that particular place being the robot's true location. The hypothesis which emerges with the single highest confidence level (see below) is regarded as the winner, and taken to be the robot's actual location. The algorithm used for performing relocalisation in these experiments is given in figure 4, and may be explained as follows[3]. (The exact implementation details depend on the particular experiment being carried out - see section 4.)

The first stage of the localisation process is to determine the place memory cell that most closely resembles the current sensory perception. Because of perceptual aliasing[4], typically a number of place memory cells

[2] In these experiments, $D = 0.50$ m.

[3] In these experiments, $D = 0.50$ m, $GAIN = 2.0$, $DECAY = 0.7$, $MIN = 0.5$, $T = 1.00$ m. A value of $T = 2D$ was chosen, because this corresponds to the maximum distance theoretically possible between stored place memories after mapbuilding has been completed.

[4] This refers to the situation where two or more places are per-

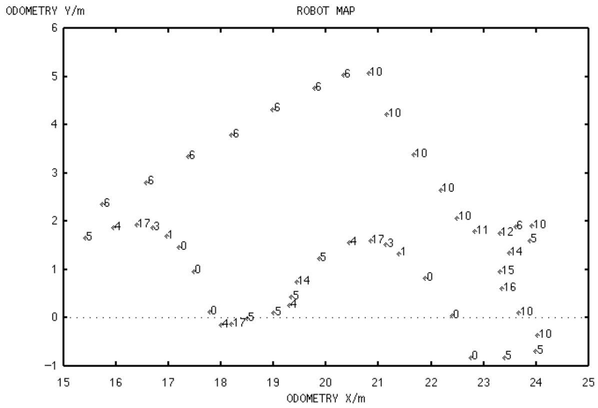

Figure 3: *Part of an Example Map Created By 'FortyTwo'.* The stored place memories follow the path of the wall-following robot around the area shown in figure 5 next to the lift. The numbers shown correspond to the categories assigned by a self-organising classifier. Perceptual aliasing occurs whenever two or more points share the same classifier category.

emerge as equally likely candidates for the robot's current position. The robot then performs an exploratory movement of an arbitrary nature, whose *relative* displacement in Cartesian space is observed by the localisation system. The current perception, previous perception and displacement are then used to narrow down the list of candidate cells for the robot's current position, providing positive or negative reinforcement of the confidence level associated with each of these cells.

3 Middle-Scale Experiments

The experimental data used for these experiments was collected while the robot explored an enclosed area located on the second floor of Manchester University's Computer Science building, as shown in figure 5. A reactive wall-following strategy was used with an average speed of 0.223 m/s. The total distance travelled by the robot over a full lap of the environment was approximately 118 metres, in an average time of 8 minutes 47 seconds.

3.1 Data Recording Techniques

The main problem addressed here was how to find some independent means of recording the "true" location of the robot over time for later comparison with the corresponding responses from the localisation system. As in similar experiments conducted by ethologists (e.g., see [12]), a grid consisting of 24 contiguous zones was marked on the floor of the robot's environment. Each time the robot moved into a new zone, the current time and zone

ceptually similar enough to be confused by the robot.

0. *Initialisation.* Formulate set of hypotheses, $H = \{h_0, \ldots, h_n\}$, consisting of all stored place memories which match the currently perceived sensor data. Initialise confidence levels: $\forall h_i \in H$, assign $conf(h_i) = 1$

1. Wait until current sensor data changes significantly (e.g., new classifier category) or robot has moved by distance T.

2. For each hypothesis h_i add relative change in odometry $(\triangle x, \triangle y)$ to the coordinates of h_i, (x_{h_i}, y_{h_i}).

3. Generate second set of candidate hypotheses, $H' = \{h'_0, \ldots, h'_N\}$.

4. For each h_i, find the nearest h'_j, recording the distance between them as d_{h_i}.

5. Accumulate evidence, given distance threshold T, minimum confidence level MIN, gain factor $GAIN > 1$ and decay factor $0 < DECAY < 1$:
   $\forall h_i \in H$

   if $d_{h_i} < T$ then
      combine $(x_{h'_j}, y_{h'_j})$ from the matching h'_j to adjust position estimate (x_{h_i}, y_{h_i})
      let $conf(h_i) = conf(h_i) \times GAIN$
   else
      let $conf(h_i) = conf(h_i) \times DECAY$
      if $conf(h_i) < MIN$ then delete h_i.

6. Delete any duplicates in H, preserving the one with the highest confidence level.

7. Add all remaining h'_j from H' which are not already contained in H to H, assigning $conf(h'_j) = 1$.

8. Repeat from step 1.

Figure 4: *Relocalisation Algorithm.*

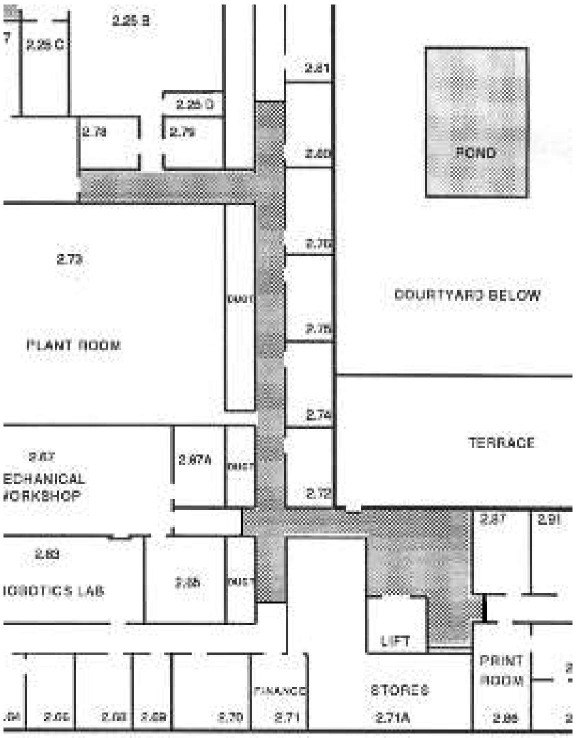

Figure 5: *Middle-Scale Environment.* The central shaded area shows the enclosure explored by the robot in these experiments (approximate size 25 $m \times 20$ m).

occupied by the robot were manually logged by a human observer.

At the same time, a "trace-replay" mechanism (as in Lee [6]) was used to record all of the sensory information available to the robot into a datafile at one second intervals. In order to synchronise the recording mechanisms of the robot and observer, the robot's voice synthesizer was used to continuously speak aloud the current time. Another program was later used to cross-reference the robot- and human-collected data in a common datastore. The collected data could then be played back to assess the performance of different mapbuilding and localisation systems as required.

NB. The use of a trace-replay mechanism can only be justified in the special case of a purely reactive exploration strategy, because the sensor information will be the same whether it is processed on-line or in later recall of the collected data. If, however, the exploration strategy were dependent in any way on the mapbuilding and localisation behaviours being studied, then a trace-replay mechanism would break the closed loop between perception and action, and therefore violate the independence of the human observer in the experiment.

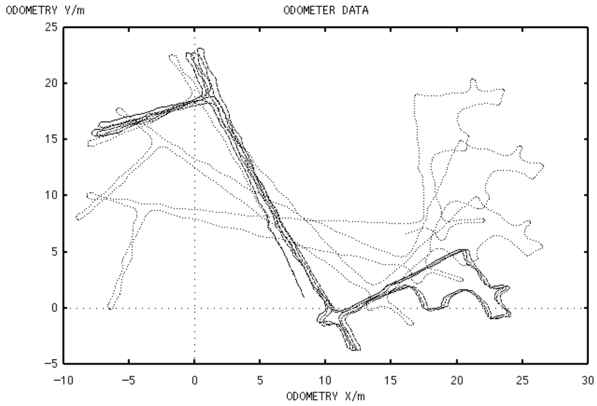

Figure 6: *Corrected Odometry Data.* The dotted line shows the uncorrected odometry data, the solid line after the simulated compass sense had been applied to remove angular drift.

3.2 The Need for a Compass Sense

3.2.1 Odometer Drift

In the current system, mapbuilding is considered to have been completed when the mapbuilder detects that it has returned to its initial location, i.e., when the wall-following behaviour has carried out a full circuit of the environment. The criteria used for deciding whether the robot is currently located within a place already stored in the map are (i) that the robot's internal position estimate from dead-reckoning is within a certain distance of the stored (x, y) coordinates, and (ii) that the currently perceived sensor data matches the prototype for that particular place. Using the raw odometry data from the trace-replay mechanism (see figure 6), the first criterion failed because the accumulated angular drift error in the robot's odometry was too big, and the robot therefore failed to produce a globally consistent map of its environment.

The solution taken here was to simulate the effects of a global "compass sense" by retrospectively applying a uniform rotational correction to the raw odometer data. A hill-climbing strategy was used to find the best angular adjustment, attempting to maximise the correlation between the odometer data from successive laps of the environment. The evaluation function used was the least mean squares statistic, calculated by averaging the distance between corresponding data points in each pair of consecutive laps. Hence, the "best fit" shown in figure 6 was found by applying a uniform correction of 99.13^o over 4 laps, or approximately $0.047^o/s$.

After this adjustment had been applied, the mapbuilder was able to construct a map successfully from the data collected during one of the recorded laps of the environment. The remaining data was then used to test the relocalisation system.

3.2.2 Robot Drift

In previous work (e.g., [10]), the rotational and turret motors of the Nomad 200 robot have been controlled in unison, so that the sensors always point in the same direction relative to the heading of the robot. Here, however, the rotational motor was controlled independently, and the turret left stationary, i.e., the robot explored its environment with its sensors always facing in the same global direction rather than the direction of travel. Thus, the appearance of locations to the robot depended on the robot's position alone, not its orientation, providing the robot with a "canonical view" of places.

In the small-scale experiments previously conducted [2], no problem was found with angular drift of the robot's turret against the rotational movement of the wheel motors. However, this factor had a considerable impact on the middle-scale experiments presented here. After four laps of the environment, it was observed that the turret had somehow managed to drift by a full quarter turn. This meant that after each lap, the robot's circular array of 16 infrared (or sonar) sensor readings had effectively been rotated by one place, so that each sensor now saw what its neighbour would have seen in the previous lap. In this situation, localisation can only fail, as current and stored sensor images can no longer be matched in a meaningful way.

Therefore, a similar approach to the one described above was adopted for retrospectively correcting the recorded sensor data. A uniform angular correction was applied to the robot's implied (i.e, initially zero) turret orientation. Whenever the accumulated correction was judged to have resulted in a rotational shift of the robot's sensor arrays (i.e., when the turret had drifted more than half-way into the next sensor "bin"), the robot's infrared and sonar sensor arrays were both subjected to a compensatory counter rotation. Using another least mean squares function to measure the correlation between the sensor readings taken in successive laps, the hill-climbing strategy yielded an optimal uniform correction of 90.00^o over 4 laps, or approximately $0.043^o/s$, to the sensor data.

It is significant that the angular correction required to the odometry data was different from the adjustment required to the turret orientation. This result confirmed that there are in fact two separate rotational drift effects occurring here; one affecting the robot's dead-reckoning mechanism, and another affecting the physical orientation of the robot itself. A future compass sense (now the subject of ongoing research) will need to be capable of compensating for both if the current system is to function fully in middle-scale environments. The following results were all obtained using the corrected data, and show that the robot would be able to localise provided that a real compass sense capable of fulfilling the above criteria can be found.

3.3 Statistical Analysis

In each of the experiments conducted, the particular mapbuilding system under investigation was first used to construct a map of the robot's environment. Each of the place memory cells created by the mapbuilder was then matched by its (x, y) coordinates to the nearest data point contained in the same lap data used for mapbuilding. Each cell could then cross-reference one of the zones recorded by the human observer. The zone label attached to each cell could then be regarded as the "response" of the robot whenever that particular cell emerged as the winner during the relocalisation process.

The relocalisation system was then tested on 186 separate datasets constructed from the remaining two laps worth of recorded data, starting at 5 second intervals along the route followed by the robot. Each dataset contained 120 items of recorded robot data, i.e., two minutes worth of continuous sensori-motor information produced by the wall-following robot. Thus, each experiment consisted of 186 trials, each resulting in a list of 120 output events $< t, r, l >$, where t is the time in seconds from the start of the trial, r the location predicted by the robot and l the location measured by the human observer.

Hence, we were able to capture two comparable distributions for each time step $t \in \{1, \ldots, 120\}$ from the nominal start of each experiment; one for all responses R by the robot, and the other for all locations L recorded by the observer at the same time. The following statistical treatment was used to find an entropy based measure of the *strength* of association between the two distributions at each time step t.

The first stage of the analysis was to combine all of the data from the 186 trials into 120 separate contingency tables, one for each time step t from the nominal start of the analysis. Each table contained 24 rows, one for each of the possible robot responses R, and 24 columns, one for each of the possible measured locations L. Each cell N_{ij} was therefore set to contain a count of the number of events when the robot predicted the value of R as i and the human measured the value of L as j. The total number of events N in each table was always 186. The row and column totals were then calculated for each table using the following equations.

$$N_{i\bullet} = \sum_j N_{ij}$$

$$N_{\bullet j} = \sum_i N_{ij}$$

The next set of equations were used to evaluate the entropy of the variables under consideration; in particular, we are interested in measuring the uncertainty in predicting the value of L, the measured location, given the value of R, the robot's response.

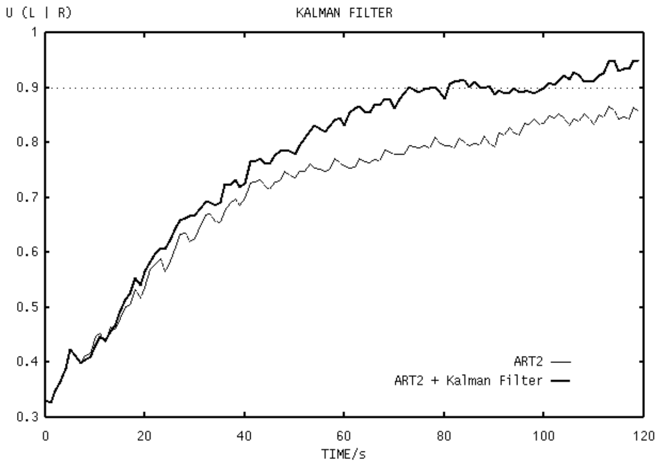

Figure 7: *Applying the Kalman Filter.*

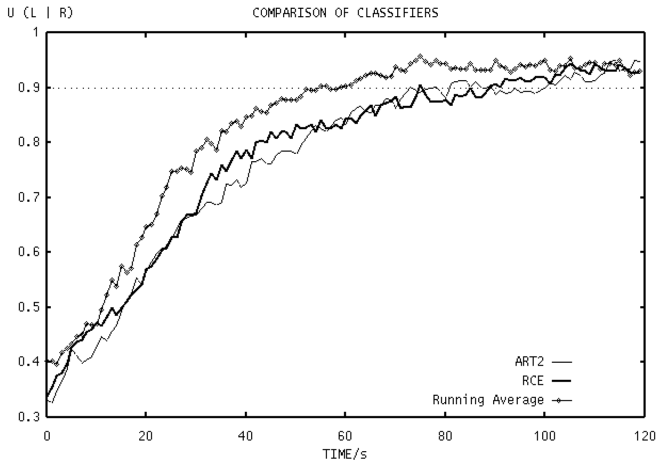

Figure 8: *Comparison of Classifier Mechanisms.*

$$H(L) = -\sum_j p_{\bullet j} \ln p_{\bullet j}$$

$$H(L \mid R) = -\sum_{i,j} p_{ij} ln \frac{p_{ij}}{p_{i\bullet}}$$

$$p_{ij} = \frac{N_{ij}}{N}, p_{i\bullet} = \frac{N_{i\bullet}}{N}, p_{\bullet j} = \frac{N_{\bullet j}}{N}$$

Finally, to measure the association or "dependency" of L on R, we use the following statistic known as the *uncertainty coefficient* of L given R.

$$U(L \mid R) \equiv \frac{H(L) - H(L \mid R)}{H(L)}$$

This measure has the added advantage of always lying between 0 and 1, where 0 indicates that the robot's response never predicts its true location, and 1 that the robot's response always predicts completely its true location.

4 Results

4.1 Applying the Kalman Filter

The first experiment involved running the most basic version of the localisation algorithm, using ART2 to classify the robot's range-finding sensor input. Here, the matching location hypotheses (see step 5 in figure 4) are combined simply by replacing the old coordinates (x_{h_i}, y_{h_i}) with the coordinates $(x_{h'_j}, y_{h'_j})$ taken directly from the place memory cell associated with the new candidate hypothesis. In this experiment (and all others apart from the last one in section 4.3), the robot was restricted to using its infrared sensors only to deliberately exacerbate the problem of perceptual aliasing.

By contrast, the next experiment involved merging the two sets of coordinates together using a Kalman filter. This is a familiar approach within mobile robotics, being used to combine position information from multiple sources of varying certainty [7]. This technique is often used in localisation systems which require *a priori* knowledge of the robot's position (e.g., Lee [6]), i.e, the robot starts off with a rough idea of its position, and the filter is then used to refine this position estimate on the basis of new perceptions.

Here, however, the technique is applied to the situation where the robot is "lost", starting off with no prior knowledge of its position. A separate Kalman filter is applied to each of the different location hypothesis, each filter creating its own individual estimate of position and associated confidence interval. The basic Kalman filter is computationally cheap (compared to the Extended Kalman Filter) and led to a significant increase in performance, as shown in figure 7. The Kalman filter was used in all of the subsequent experiments.

4.2 Which Classifier?

The next question to be addressed was whether the classifier mechanism used by the system was of critical importance. Initially, an ART2 network had been chosen because it offered the potential for "lifelong learning", allowing new prototypes to be added without losing previously acquired sensor information. (This is in contrast to a fixed size classifier mechanism such as the self-organising feature map, as used in [9]).

However, we read of a far simpler classifier mechanism with similar properties in Kurz [4], namely a simplified version of the Reduced Coulomb Energy network. Here, each prototype is simply a stored sensor image taken directly from the robot's range-finding sensor readings, and classification is decided by matching input patterns to the nearest stored prototype. All sensor inputs to the system are normalised, and the Euclidean distance is used for comparing current and stored sensor images. A new category is created only if an input pattern fails to lie within a fixed distance threshold of any of the existing prototypes[5].

To enable a fair comparison of the different classifiers, the clustering metric (i.e., the "magic number" which determines the sensitivity to perceptual detail) for each mechanism was carefully adjusted so that both produced a map containing a similar number of stored place memories[6]. The results are shown in figure 8; there is little or no difference in performance between these two classifiers. We therefore concluded that the RCE network probably represents a better choice for most mobile robotics applications on the basis of its much lower complexity.

The next question addressed was whether it is necessary to use a separate classification mechanism at all, instead of simply associating a separate stored sensor image with each of the place memories contained in the robot's map, as in the qualitative topological models of Zimmer [13].

The mapbuilder was modified so that each stored place contained its own individual prototype, unlike the ART2 and RCE classifiers, where one prototype could be associated with several different places in the map. As in the RCE network, all sensor inputs to the system were normalised, and the Euclidean distance was used for comparing sensor images. Each prototype was initialised from the immediate available sensor data, and then smoothed by computing a running average from all of the subsequent matching perceptions. A new place memory was created whenever the distance between the current sensor data and the nearest stored prototype exceeded a certain threshold, in addition to the usual criterion of distance travelled.

Similarly, a new version of the localiser was created using the same Euclidean distance threshold to determine whether the current sensor input matched any of the stored place memories, and also to decide when to trigger another iteration through the localisation algorithm (see step 1 of figure 4).

As in the previous experiment, the clustering metric was carefully selected to yield the same degree of perceptual resolution in the map[7]. The performance shown in figure 8 was significantly better than either the ART2 or RCE classifiers, although this was achieved at the cost of increased storage requirements and a computationally more expensive matching procedure.

[5]The adaptive learning component and the additional "AND" layer from the more general RCE network were omitted here, as in Kurz [4].

[6]For ART2, vigilance $\rho = 0.90$ yielded a map of 180 stored points. For RCE, cluster size = 0.927 yielded a map of 181 points.

[7]Cluster size = 0.56 yielded a map of 180 stored points.

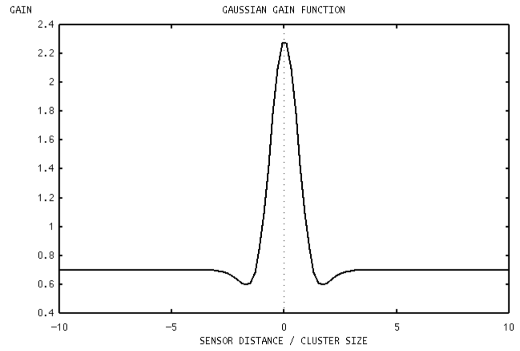

Figure 9: *Gaussian Gain Function.*

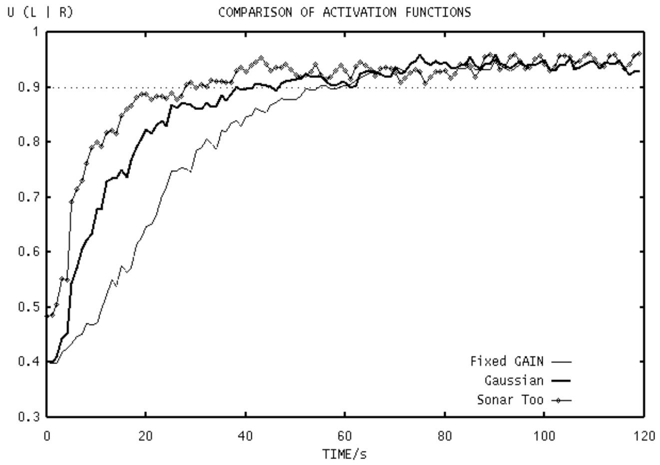

Figure 10: *The Effect of the Gaussian Function.*

4.3 Changing the Gain Function

The previous experiment fails to exploit one important characteristic in the comparison of current and stored sensor images. Here, the distance between current and stored sensor images conveys additional information about the likelihood of a particular hypothesis being the robot's actual location. In the next experiment, a Gaussian function[8] (see figure 9) was used to replace the $GAIN$ constant used up until now in the localisation algorithm. The results shown in figure 10 indicate a considerable improvement in performance at little extra computational cost.

The effect of this modification is to create a "fuzzy" matching procedure between current and stored sensor images which favours the candidate hypotheses the more closely they match the current sensor data. In addition, those hypotheses which lie on the borderline between matching and non-matching patterns are actually punished slightly more by the minima in the gain function for "not being quite close enough". This kind of Gaussian function is familiar within more neurally inspired

[8]$gain = 0.7 + 4e^{-z^2} - 2.4e^{-0.69z^2}$ where z equals the sensor distance divided by the cluster size.

Experiment	Time Taken/s for $U(L \mid R) = 0.9$
ART2 without Kalman filter	n/a
ART2	73
RCE	75
Running Average	55
Gaussian gain function	38
Sonar + Gaussian gain	28

Table 1: *Comparison of Localisation Times in the Different Experiments.* Here the localisation time is recorded as the time taken for the uncertainty coefficient $U(L \mid R)$ to reach 0.9 for the first time.

models of navigational function (e.g., see [8]). However, the localisation system presented here is computationally much cheaper than more biologically plausible models of this kind, and is therefore probably better suited for use on a mobile robot which must map large areas.

Finally, the latter modification was tested using the full range of sonar and infrared sensor data available to the robot in order to assess the full potential of the system. The sonar and infrared components were given an equal weighting in the input vector, and the clustering metric was again adjusted to keep the resolution of the map consistent with the previous experiments[9]. Figure 10 shows the corresponding increase in localisation quality.

All the results obtained are summarised in table 1, which uses the time taken for $U(L \mid R)$ to reach 0.9 for the first time as a comparative measure of localisation performance.

5 Summary

In this paper, we have presented a relocalisation algorithm that enables a mobile robot to re-establish its position within a previously explored environment after having got lost. The localisation algorithm uses sensory perception ("perceptual landmarks") and relative odometry; it is not dependent upon a global reference frame such as a Cartesian coordinate system.

In order to assess the system's ability to localise, we introduced an entropy-based performance measure which measures the dependency of the robot being at a particular location given a specific map response.

Our experiments were conducted using logged sensor and odometry data recorded by a Nomad 200 following a circular path of 118 m by wall-following. Using recorded data, obtained from a real robot, ensured that all experiments were conducted under identical conditions, whilst retaining the complexity of robot-environment interaction, rather than using simplified numerical models.

The data we obtained by this method revealed that the robot turret drifts by about $0.043^o/s$, or in other words, roughly every 260 seconds the turret drifts by half a sonar cone's width. For the proposed localisation algorithm to work, it is essential that at every location exactly one view is seen by the sensors ("canonical view"). We achieve this by leaving the turret of the robot in a fixed orientation. However, angular drift of the turret makes this strategy fail, and therefore has to be removed. In the experiments reported here, drift was removed *post hoc* by finding the corrective angle that produced the greatest correlation between successive traversals of the route (see section 6 for a discussion).

Having completed the preliminary steps just described, we then had all the required tools in place to analyse the proposed localisation algorithm with respect to its various components.

One main aspect of the localisation procedure is the use of perceptual landmarks. Three different methods for identifying such landmarks were compared; an ART2 classifier system, a Reduced Coulomb Energy (RCE) network (which stores distinct sensor readings as prototypes), and a third classifier system, similar to RCE, whose prototypes have been "smoothed" by computing the running average of all sensory perceptions within one cluster. The finding was that all three classifier systems eventually attained the same level of performance ($U(L \mid R) \approx 0.95$), but that the running average classifier was the fastest to localise. Here, the value of $U(L \mid R) = 0.9$ was reached after 55 s, whereas using ART2 or the RCE network localisation to the same quality took around 75 s.

We then investigated the influence of the mechanism of hypothesis formation on the localiser's performance. The basic version of the localisation algorithm stipulates that the strength of a hypothesis is increased whenever the robot's current sensory perception resembles the stored prototype connected with the hypothesized location closely enough, and the relative distance between current and previous hypothesized location fits the existing map data. However, this model can be improved by taking into account the degree of similarity between stored prototype and current sensory perception; if the robot's current perception is very close to a stored prototype, the likelihood for being at the mapped location is greater than if the resemblance is weak. Therefore, we used a Gaussian rather than a constant gain function, and could show that this reduces the time needed for localisation by up to 50%.

Last, we could confirm that increasing the robot's perceptual resolution reduces the time needed for localisation, a result that was to be expected. Using infrared *and* sonar sensors, the robot localises faster still compared with using only the infrared sensors and the Gaussian

[9] Cluster size = 0.90 yielded a map of 180 stored points.

gain function.

6 Conclusions and future work

We knew from previous work ([5]) that there are great discrepancies between our Nomad 200 simulator and the actual robot, but we did not know whether we would encounter considerable differences between laboratory experiments and middle-scale, "real world" experiments. After all, in both cases the underlying mechanism is that of a real robot interacting with the real world!

We did find considerable differences. Most notable was the influence of rotational drift. In particular, angular drift of the turret proved to be so pronounced that without compensation the localisation system failed very quickly. We are currently investigating the use of a compass to address this problem. Translational drift in the robot's odometry presents no problem during localisation, because only relative odometry is used here. However, during mapbuilding *absolute* coordinates are associated with prototypes, and if mapbuilding over large distances is to be achieved, drift will be a problem here as well. This is subject to ongoing research at Manchester.

The introduction of a quantitative performance measure allowed us to compare different algorithms directly, and to quantify the influence of individual process parameters upon the performance of the entire system. To use quantitative performance measures is good scientific practice, and their use in mobile robotics not only facilitates comparison, but also replication of experiments by other workers in the field. We therefore believe that using quantitative performance measures more, while using mere existence proofs less, will move mobile robotics further towards an exact science.

References

[1] G. Carpenter and S. Grossberg, "ART2: Self–Organization of Stable Category Recognition Codes for Analog Input Patterns", Applied Optics, Vol. 26, No. 23, pp. 4919-4930, 1987.

[2] T. Duckett and U. Nehmzow, "Experiments in Evidence-Based Localisation for a Mobile Robot", AISB-97 Workshop on Spatial Reasoning in Mobile Robots and Animals, University of Manchester, Technical Report Series UMCS-97-4-1, ISSN 1361-6161, 1996.

[3] T. Duckett and U. Nehmzow, "A Robust, Perception-Based Localisation Method for a Mobile Robot", Dept. Computer Science, University of Manchester, Technical Report Series UMCS-96-11-1, ISSN 1361-6161, 1996.

[4] A. Kurz, "Constructing Maps for Mobile Robot Navigation Based on Ultrasonic Range Data", IEEE Trans. Systems, Man, and Cybernetics - Part B: Cybernetics, Vol. 26, No. 2, April 1996.

[5] A. Langlois, "Quantitative Evaluation of a Mobile Robot Simulator", MSc Thesis, Department of Computer Science, University of Manchester, England, 1996.

[6] D. C. Lee, "The Map–Building and Exploration Strategies of a Simple, Sonar- Equipped Mobile Robot; An Experimental, Qualitative Evaluation", Ph.D. thesis, University College London, England, 1995.

[7] P. S. Maybeck, "The Kalman Filter: An Introduction to Concepts", in Cox and Wilfong, eds., "Autonomous Robot Vehicles", Berlin: Springer Verlag, pp. 193-204, 1990.

[8] B. L. McNaughton, C. A. Barnes, J. L. Gerrard, K. Gothard, M. W. Jung, J. J. Knierim, H. Kudrimoti, Y. Qin, W. E. Skaggs, M. Suster and K. L. Weaver, "Deciphering the Hippocampal Polyglot: The Hippocampus as a Path Integration System", Journal of Experimental Biology 199, pp 173-185, 1996.

[9] U. Nehmzow and T. Smithers, "Mapbuilding using Self–Organising Networks", in J. A. Meyer and S. Wilson (eds.), "From Animals to Animats", Proc. SAB90 Paris, pp. 152-159, MIT Press, 1991.

[10] U. Nehmzow, "Autonomous Acquisition of Sensor-Motor Couplings in Robots", Dept. Computer Science, University of Manchester, Technical Report Series UMCS-94-11-1, ISSN 1361-6161, 1994.

[11] T. Smithers, "On Quantitative Performance Measures of Robot Behaviour", Robotics and Autonomous Systems, Vol. 15, No. 1-2, pp. 107-133, 1995.

[12] R. Wehner and M. V. Srinivasan, "Searching behaviour of desert ants, genus *Cataglyphis* (*Formicidae*, Hymenoptera). Journal of Comparative Physiology, 142, pp. 315-338, 1981.

[13] U. R. Zimmer, "Self–Localization in Dynamic Environments", IEEE/SOFT International Workshop BIES'95, Tokyo, Japan, May 30-31 1995.