Location Recognition with Self-Ordering Networks

William D Smart
Department of Computer Science
Trinity College
University of Dublin
Dublin 2
Ireland
Bill.Smart@cs.tcd.ie

John Hallam
Department of Artificial Intelligence
University of Edinburgh
5 Forrest Hill
Edinburgh
United Kingdom
john@aifh.ed.ac.uk

Abstract

Location recognition is of crucial importance to mobile robots. If an autonomous robot has some concept of where it is, the varity and usefulness of the tasks which it can achieve are greatly increased. We present a system which performs location recognition in a simplified environment, building on previous work done at the University of Edinburgh. We give a summary of the previous work and describe our extensions to it, giving the results of experiments in both simulation and on a real mobile robot.

1 Introduction

Location recognition is of vital importance to mobile robotics. If an autonomous robot has some concept of where it is, the variety of tasks which it can perform is greatly increased. In this paper, we present an extension to previous work carried out in the Department of Artificial Intelligence at the University of Edinburgh, described in [6]. In this work, a small mobile robot wall-followed around a simplified enclosure, collecting information about its movements. This information was subsequently used to form topographic maps of the enclosure which were then used to perform the location recognition task.

The work presented in this paper extends the previous scheme in two ways. Firstly, it incorporates additional ideas from a theory of low-level visual processing which influenced the first system. Secondly, it further validates the ideas of the previous work by fully implementing a similar scheme in a small mobile robot, while at the same time suggesting that the previous system was overly specific in some of its requirements.

Our system was first implemented on a workstation, using data gathered from a real robot (in the fashion of [6]) to investigate the effects of parameter variation in the topological mapping routines. An analysis of the results of these experiments is given, along with a discussion of the final system implemented on the mobile robot.

2 Previous Work

Location recognition by a mobile robot in a simplified enclosure was investigated as part of the "Really Useful Robots" project at the University of Edinburgh. The set of experiments, reported in [6], upon which the work presented in this paper builds deals with a small mobile robot wall-following around an enclosure, recording information about its movements. This information, in the form of direction-duration pairs (*e.g.* `(forward 5)`) is used to train a series of Kohonen-style self-organising feature maps (SOFMs) [2] implemented on a workstation. Seven such feature maps were created, each trained using input vectors constructed from different numbers of direction-duration pairs. For example, one network was trained with vectors constructed from the two most recent direction-duration pairs, another with vectors built from the four most recent pairs, and so on. This scheme was inspired by a theory of low-level visual processing described in [3] and attempted to take into account the inherently multiscalar nature of this type of location recognition (*i.e.* locations can be identified at different scales).

The theory of low-level vision which inspired this multi-scalar approach, presented in [3] claims that the human visual field is divided into several frequency channels as one of the first stages of processings. Attempts are then made to detect objects in each of these channels. The necessary and sufficient condition for a detected object to really exist (as opposed to being noise in the signal) is that it be detected in two adjacent frequency channels. The work in [6] moved this idea from a frequency to a temporal domain, tightening the constraints by insisting that all seven f the SOFMs agree for recognition to occur.

There are two phases to the experiment. In the first, known as the learning phase, the SOFMs are trained on inputs generated from the incoming data from the robot motions. At the conclusion of this phase, after a few circuits worth of data has been used, the system is instructed to "remember" the current location by storing the activation values associated with the last input for each of the SOFMs. In the second phase, the system again builds input vectors based on the incoming motion data, but this time compares the activations generated by these with those stored. For each SOFM, when the current activation pattern is similar to the stored one[1], that SOFM signals recognition. When all seven of the SOFMs agree that their current activation patterns match those stored, the system announces that the "remembered" location has been reached.

A success rate of approximately 90% is reported in [6] for this approach, with the system never "recognising" a wrong location. That is, all failures were because *no* location was recognised, not because a wrong one was identified erroneously.

This work was continued in [5] with a system which examined the incoming direction-duration pairs and attempted to find "significant" ones as landmarks for location by comparing the current duration with a running average of all durations of the same type of action. This was claimed to give equivalent performace with less computation and was implemetned on an autonomous robot. We believe, however, that this approach was considerably less general because it makes assumptions about temporal scale and (what seems to be) an arbitrary threshold value. The multiscalar aproach used in this paper and [6] avoids making these assumptions and as a result is more generally applicable.

The main extension which we make to the previous work is by relaxing the constraints on the matching scheme. Previously, it was required that all of the SOFMs in the system agreed for location recognition to occur. Drawing more heavily on [3], our system needs only a sequential subset of the SOFMs to agree for recognition. This is both computationally cheaper and, in general, faster, since not all of the SOFMs have to be checked each time.

The other extension which we make is implementing our system on a fully autonomous mobile robot. This both validates the ideas of the previous work and lends weight to our suggestion that the constaints previously imposed can be relaxed without a negative effect on performance.

[1]Typically this involves the calculation of the Euclidean distance between the two patterns.

3 Results and Analysis

We present three sets of results from our experiments. The first explore the effects of parameter variation in the SOFMs and were carried out on a workstation with data gathered from a mobile robot in an enclosure (in the fashion of [6]). Secondly, we briefly describe the matching scheme used by our implementation. The final set of results are from implementing a fully working location recognition system on a small mobile robot.

3.1 Parameter Variation

Each of the parameters of the network (input vector length and representation, network size, initial learning gain, initial neighbourhood region size and attenuation factors, see [2]) were varied independently to examine the effects which they had on the training of the SOFMs. The system was destined to be implemented on a mobile robot with limited memory and computational abilities and this was kept in mind when selecting the final parameters to be used. This was particularly true in the case of network size and input representation, since these greatly affect the speed of the overall system and should be kept as small as practical.

It was decided to use binary vectors for input to the SOFMs to ease the computational cost of the implemented system[2]. Direction was encoded with four bits, with the remainder representing the quantised duration of the motion. Two other factors remained to be decided:

- should the representation use 0s and 1s or -1s and 1s?
- should duration be encoded with trailing 1s (00011111) or trailing 0s (00010000)?

It was found that the results produced by both representations produced qualitatively similar results. The 0/1 representation was chosen because it was more computationally efficient. The two duration representations were also found to give qualitatively similar results, with the "trailing one" representation (also known as "thermometer code") generally giving higher activation values. Because of this, the former was selected. The length of the vector generated by each

[2]By using only binary vectors the multiplication operations in the training of the networks could be replaced by computationally cheaper additions.

direction-duration pair was set at 16 bits, four for direction and twelve for duration. This was chosen fairly arbitrarily, being the length which could be stored in two bytes of memory in the robot.

Square networks of size 25 to 144 units were trained on a set of 32 input vectors and the results compared. The number of correct matches for each network was very similar, with only slightly better performance exhibited by the larger networks, as predicted by the theory. Interestingly, the smallest network, which contained fewer nodes than the number of vectors which it was supposed to learn, performed at a level similar to the larger networks. The final size chosen, after consideration of the previous work, was seven by seven units. This was selected after consideration of the work reported in [4], which successfully used a one-dimensional SOFM with 50 units. By using a similar number of units in a two-dimensional structure, equal or better performance was hoped for.

The initial learning gain and the gain attenuation factor controlled how much each new input vector affected the feature map. Varying the initial gain between 0.1 and 1.0 produced little noticable variation in the classification ability, with the general trend being that larger values produced slightly better results. However, the larger the initial gain, the longer it took for the SOFM to become stable and form recognisable regions. The same was true for small attenuation values. However, having the initial gain too low or the attenuation too high produced over-reliance on the first few input vectors and neglect of the later ones. The final choice for initial gain was 1.0 and for gain attenuation was 0.99, both of which gave the best results in the test cases.

To investigate the effects of varying the initial size of the neighbourhood region, a 13 by 13 SOFM was trained using initial neighbourhood values from 0 to 6 units[3]. The attenuation rate was also varied between 0.95 and 1.00. As with the other parameters, only slight variations were shown across the range of possible values. The initial size and attenuation rate of the neighbourhood region, along with the learning gain, had an effect on how quickly the SOFM stabilised. After considering the empirical results, an initial size of 2 units and an attenuation of 0.99 was chosen. This allowed a sufficient number of input vectors to contribute to the shape of the SOFM before the neighbourhood region shrunk to unit size.

[3] This was the number of units from the target unit to the edge of the region, *i.e.* the six unit region was actually 13 units across.

3.2 The Matching Scheme

The method of matching the current activation patterns of the networks to the stored patterns was also looked at in simulation. In [6] this matching was done by either calculating the Euclidean or "city-block" distance between the current and stored patterns and comparing this value to a previously defined threshold. This was, however, carried out on a workstation where computational expense was less of a concern than on a small robot. It was also not required to operate in real-time.

A computationally cheaper alternative involved the calculation and storing of the maximally responding unit for each network. The inputs then had the activation value for this node calculated. If the value was sufficiently close to the stored one, the current input was deemed to be the same as the memorised input.

Extensive simulations to validate this approach were carried out. In an attempt to automate the threshold setting mechanism, rather than the designer influenced method in [6], the overall activation values were examined. It seemed that the "closeness" of two activation values depended to some extent on the overall distribution of all the activation values in the network. It was seen as reasonable, therefore, that some thresholds could be set in terms of the mean and/or standard deviation of the activation values of the network.

It was found empirically that by insisting that an actication had to fall within $\pm\frac{3}{4}$ of one standard deviation of the recorded activation value, a reasonable number of candidate matches were generated. It should be noted that this stage did not have to produce unique, error free matches since, following Marr and Hildreth's theory, each network makes only a suggestion as to whether or not the location is recognised. However, by making this stage *too* lax in its selection procedure caused too many possible matches to be passed to the next stage, resulting in bad overall recognition.

As stated previously, we relaxed the necessity for all SOFMs to agree for recognition to occur. We also reduced the number of SOFMs present. This was done mainly in the interests of computational efficiency, in an attempt to get the system working in real-time. Using a system with either four or five SOFMs, experiments were run, where the required subset was between two and the number of SOFMs. Obviously, the more SOFMs required for recognition, the larger the computational load. The results from these experiments are presented in section 3.3.

Table 1: Results with four and five networks

	Four Networks		Five Networks	
thresh	enc 1	enc 2	enc 1	enc 2
2	100%	100%	100%	100%
3	100%	100%	100%	100%
4	80%	100%	80%	70%
5	n/a	n/a	50%	50%

3.3 The Robot

Once suitable parameter values were determined, the system was implemented on a small mobile robot, programmed in the spirit of Brooks' subsumption architecture [1] to follow walls and generate input vectors for the SOFMs. Two versions of the location recognition system were then implemented, using four and five networks.

Following the approach in [6], the robot was allowed to wall-follow around two simplified enclosures (similar to the one in [6]), generating input vectors and training its SOFMs. A button on top of the robot was then pressed by a human supervisior to instruct the robot to "remember" the current location. The robot then stopped and recorded the current activation values of each of the SOFMs.

The robot was then randomly relocated along the perimeter of the enclosure and the button depressed again. At this point, the robot continued to wall-follow around the enclosure, checking the activation values generated by the incoming vectors against those stored. When the recognition conditions were met, the robot stopped, signalling that it had recognised the previously "remembered" location.

The "recognition threshold" (*i.e.* the number of sequential SOFMs which had to agree for recognition to occur) was varied for both of the implementations, with results given in Table 1.

The number of direction-duration pairs used in forming the input vectors were 1, 2, 4, 8, and 12 pairs. In each case, the robot was allowed to perform two circuits of the enclosure and was instructed to "remember" a location during the third circuit. The starting point and marked location were varied each time and the result was classified as "good" if the robot correctly identified the marked location when it started looking and "bad" if it failed to recognise it within two circuits of searching. For both four and five networks, and both enclosures, each threshold level was tested 10 times.

For both systems, as the requirement for recognition becomes more stringent (*i.e.* more SOFMs are required to agree) the success rate drops off. This suggests that it is neither necessary nor desireable to have such rigid requirements. This tends to validate our earlier claims that the scheme used in [6] was overly rigorous in stipulating that all SOFMs must agree. Although our system performs at a lower level than the previous one when requiring all networks to agree (75% against approximately 90%), we believe that this is attributable to the computational "shortcut" taken in order to get a real-time system on a robot. Since [6] had fewer time and space constraints, it was able to bring the full power of a workstation to bear on the problem, using seven networks trained with inputs generated from between two and 24 direction-duration pairs. It is possible that the amount of computation available swamped the problem to some extent and although proving the method generally, hid some of the difficulties involved in fully implementing the system on a current mobile robot in real-time.

4 Conclusions

We draw three main conclusions from the work presented in this paper. Firstly, we notice that, to a large extent, variation of the parameters of Kohonen-style self-ordering feature maps, when trained on the type of data described above has little significant effect on the final abilities of the network. Small variations are apparent and the parameters change but, overall the SOFMs are very robust to such changes.

Secondly, we have shown that the location recognition system described in [6] may be simplified computationally, with little or no detrimental effect on performance. Both the number of SOFMs used and the recognition criteria were altered, reducing the amount of computation needed to perform the location recognition task. Although [5] reports a real-time system, we believe that the approach, with seemingly arbitrary thresholds and ad-hoc treatment of temporal scale is not as valid as the one in this paper and [6]. These beliefs are strengthened by the fact that the multiscalar approach can be implemented in real-time on an autonomous robot, eliminating the need to use a less appropriate method for location recognition.

Finally, we have shown that our system can be implemented on a small autonomous mobile robot and is capable of operating in real-time. In doing so, we also provide further validation of the ideas behind [6], while at the same time suggesting that it is over-constrained in some areas.

Acknowledgements

The work presented in this paper was done by WDS as part of a Masters degree in the Department of Artificial Intelligence at the University of Edinburgh. We would like to thank all of the staff there who contributed ideas to the project and helped to keep the hardware and software up and running. WDS would also like to thank Amy Montevaldo for her help with earlier versions of this paper and encouragement throughout the project.

References

[1] Rodney A Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2:14–23, April 1986.

[2] Teuvo Kohonen. *Self-Organisation and Associative Memory*. Springer-Verlag, second edition, 1988.

[3] David Marr and Ellen Hildreth. Theory of edge detection. *Proc. R. Soc. London*, 207:187–217, 1980.

[4] Ulrich Nehmzow and Tim Smithers. Mapbuilding using self-orgainising networks in "really useful robots". In Jean-Acardy Meyer and Stewart W Wilson, editors, *Proceedings of the International Conference on the Simulation of Adaptive Behaviour: From Animals to Animats, Paris, France*, pages 152–159. MIT Press, 24–28 September 1990.

[5] Ulrich Nehmzow and Tim Smithers. Using motor actions for location recognition. In Francisco J Varela and Paul Bourgine, editors, *Towards a Practice of Autonomous Systems, proceedings of the First European Conference on Artificial Life, Paris, France*, pages 96–104, 11–13 November 1991.

[6] Ulrich Nehmzow, Tim Smithers, and John Hallam. Location recognition in a mobile robot using self-ordering feature maps. In G Schmidt, editor, *Proceedings of the International Workshop on Information Processing in Autonomous Mobile Robots, Munich, Germany*, pages 267–277. Springer-Verlag, March 1991.