Door crossing behavior for a mobile robot using Bayesian Networks

Elena Lazkano[1], Basilio Sierra[1], Iñaki Rañó[1] and Iñaki Zarauz[1]

Dept. of Computer Science and Artificial Intelligence, University of the Basque Country, P.O. Box 649, E-20080 San Sebastián, Spain,
{ccplaore, ccpsiarb}@si.ehu.es, inaki_rano@terra.es, izarauz@mixmail.com
http://www.sc.ehu.es/ccwrobot

Abstract. This paper presents an attempt to learn the rotation action a robot should perform while crossing doors, using ultrasonic sensor information. Bayesian networks are used as learning technique. A Bayesian net structure is proposed based on the topological configuration of the sensors around the robot. Also, the K2 structural learning algorithm is used in order to look for a better network. This algorithm is combined with three different net evaluation metrics. Because of the large range of values the sensors provide, the *minimum description length principle cut* entropy-based discretization is applied to the readings in order to use them to learn and/or test the Bayesian networks. Some of the obtained nets are used to implement door crossing behavior in a Pioneer robot.

1 Introduction

Navigation is a basic behavior that any mobile robot needs in order to perform useful tasks. Within the context of autonomous systems, navigation can be defined as the capability of going from one objective to the next one, avoiding the obstacles that can interfere in the robot trajectory. Therefore, navigation includes not only obstacle avoidance but also objective recognition. [13] reviews and classifies, from a biological viewpoint, the navigation behaviors implemented on real mobile robots in the past years.

Indoor semi-structured environments are full of corridors that connect different offices and laboratories. Often it is necessary to cross the doors to enter into a room. Following a corridor can be approached by the task of equilibrating the free space at the right and left hand sides of the robot, and door crossings can be considered as short and very narrow passages. Our experience is that this approach does not give good results when the robot must cross a door; sensor readings can be misleading because of the panels that define the door, and the door blade also affects sensor values.

There are some references to the problem of door identification. [10] and [22] make use of visual information to identify doors. In [18] door crossing is combined together with some other behaviors in a control architecture, but doors are located in a map.

In the work described here the aim is to implement a behavior that will allow the robot to cross the doors smoothly and without bumping into the door panels when traveling towards the objective. The robot is a Pioneer 2de dual drive holonomous robot from ActivMedia Robotics. The doors that she will find in her way are 70 cm wide and her swing radius is 26 cm. We assume that the door is within the perceptual range of the ultrasonic sensor belt the robot has.

We tackle the problem as a supervised classification one [14], using Bayesian Networks. The ultrasonic sensor readings will determine the action to perform. We identify three actions: turn left, turn right and go straight. Thereupon, the action will define the sign of the rotational velocity of the robot at each moment.

In the remainder of this paper we firstly present a brief introduction to Bayesian networks, structural learning techniques and evidence propagation schemes. Subsequently, discretization of continuous data is described. Section 4 explains the experimental methodology and section 5 shows the results obtained over the cross validated data set. Next section explains how the door crossing behavior has been implemented on the robot and finally, in section 7 the main conclusions derived from the experimentation and future work are drawn.

2 Bayesian Networks

Bayesian networks are probabilistic graphical models represented by directed acyclic graphs in which nodes are variables and arcs show the conditional (in) dependencies among the variables [3][11].

There are different ways of establishing the Bayesian network structure. It can be the human expert who designs the network taking advantage of his/her knowledge about the relations among the variables. It is also possible to learn the structure by means of an automatic learning algorithm. A combination of both systems is a third alternative, mixing the expert knowledge and the learning mechanism.

Within the supervised classification area, learning is performed using a training datafile but there is always a special variable, namely the class, i.e. the one we want to deduce. Some structural learning approaches take into account the existence of that special variable [9][21], but most of them do consider all the variables in the same manner and use an evaluation metric to measure the appropriateness of a net given the data [2]. Hence, a structural learning method needs two components: the learning algorithm and the evaluation measure (score+search).

The algorithm used in the experimentation here described is the K2 algorithm [5]. This algorithm assumes an order has been established for the variables so that the search space is reduced. The fact that $X_1, X_2, \cdots, X_n$ is an ordering of the variables implies that only the predecessors of X_k in the list can be its parent nodes in the learned network. The algorithm also assumes that all the networks are equally probable, but because it is a greedy algorithm it can not ensure that the net resulting from the learning process is the most probable one given the data.

The original algorithm used the K2 Bayesian metric to evaluate the net while it is being constructed:

$$P(D|S) = \log_{10}\left(\prod_{i=1}^{n}\prod_{j=1}^{q_i}\left(\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}N_{ijk}!\right)\right) \quad (1)$$

where: $P(D|S)$ is a measure of the goodness of the S Bayesian net defined over the D dataset; n is the number of variables; r_i represents the number of values or states that the i-th variable can take; q_i is the set of all possible configurations for the parents of variable i; N_{ijk} is the frequency with whom variable i takes the value k while its parent configuration is j; $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; and N is the number of entries in the database

In addition to this metric, we have tried two more measures in combination with the algorithm.

The *Bayesian Information Criterion* [19] (BIC) includes a term that penalizes complex structures:

$$P(D|S) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\frac{N_{ijk}}{N_{ik}} - f(N)\sum_{i=1}^{n} q_i(r_i-1) \quad (2)$$

where $f(N) = \frac{1}{2}\log N$ is the penalization term.

The well known entropy [20] metric measures the disorder of the given data:

$$P(D|S) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\frac{N_{ij}}{N}\left(-\sum_{k=1}^{r_i}\frac{N_{ijk}+1}{N_{ij}+r_i}\ln\frac{N_{ijk}+1}{N_{ij}+r_i}\right) \quad (3)$$

Evidence propagation or probabilistic inference consists of, given an instantiation of some of the variables, obtaining the a posteriori probability of one ore more of the non-instantiated variables. It is known that this computation is a NP-hard problem, even for the case of a unique variable.

There are different alternatives to perform the propagation methods. Exact methods calculate the exact a posteriori probabilities of the variables and the resulting error is only due to the limitations of the computer where the calculation is performed. The computational cost can be reduced looking over the independence of the nodes in the net. The HUGIN [1] library used for part of the experimentation uses an exact propagation method.

Approximated propagation methods are based on simulation techniques that obtain approximated values for the probabilities needed. [15] proposes a stochastic simulation method known as the *Markov Sampling Method*. In the case all the Markov Blanquet[1] of the variable of interest is instantiated there is no need of the simulation process to obtain the values of the non-evidential variables

[1] The Markov Blanquet of a node is the set of nodes formed by its parents, its children and the parents of those children

and thereby, $P(x|w_x)$ can be calculated using only the probability tables of the parents and children of the node, i.e. using the parameters saved in the model specification. The method becomes for that particular case an exact propagation method and therefore, gives the same results as HUGIN.

3 Discretization of Continuous Attributes

Many classification algorithms require the classifying variables to be discretized, either because they need nominal values, or in order to reduce the computational cost and to accelerate the induction process. Discretization of the variables does not necessarily affect negatively the learning process, even more, the performance of a classifier can be improved with the discretization. [7] makes a comparison of several discretization methods. The results show that for most of the cases the discretization process improves the performance of the classifiers.

Many discretization methods can be found in the literature (see [16] and chapter three of [17] for a review). Entropy-based discretization methods have proved to be very effective for different problems. This methods pretend to divide the set of data in k subsets of minimal entropy. Given a dataset and a variable to discretize, the entries are ordered according to the values of the variable and the value that minimizes the entropy in the generated subsets is selected as a breakpoint. This criterion is applied recursively in each of the new subsets. The D-2 discretization [4] uses some stopping criteria –minimum number of instances in a region, maximum number of regions of minimal information gain– during the recursive partitioning of an attribute. Fayyad and Iranni [8] use a recursive entropy minimization heuristic together with the *Minimum Description Length Principle Cut* (MDLPC) to control the number of intervals generated.

The MDLPC method stands out in the results shown in [7]. This is the reason why we decided to try the MDLPC discretization for our experimentation.

4 Experimental Methodology

The aim of this experiment is to apply Bayesian networks to the navigation model of the robot in order to improve her behavior while crossing doors. The robot has a ring of 16 sonar sensors. During the experimentation phase the operator is always at the back of the robot. For that reason only 10 ultrasonic sensors situated in the front and left and right hands of the robot (see figure 1) have been considered.

In order to make the experiments a database of 2831 entries was created. To collect the data, the robot was pushed along different trajectories in which the action to be performed was considered to be the same. Figure 2 shows the data collection process. The easiest place from where the door can be crossed is the position just in front of the door opening; from here the robot must just go straight carefully. Thereby, while taking the sonar data we try to show to the robot how to confront the door guiding her to the point just in front of the opening.

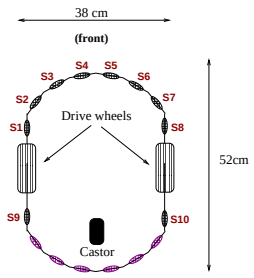

Fig. 1. Position of the sonars around the robot and variables (nodes) used

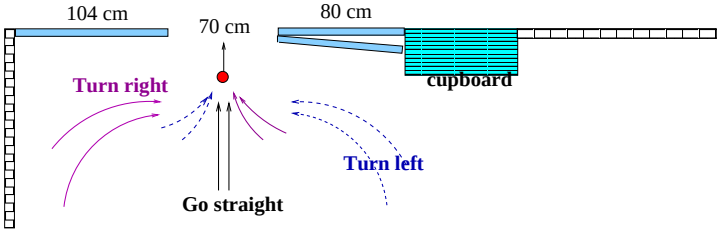

Fig. 2. Data collection process

Taking into account the large amount of values a sonar sensor (variable) can take, it is necessary to discretize those readings to obtain probability tables of an acceptable size.

Because the amount of data in the database is not very large, we decided not to divide it in two subsets, for training and testing respectively, but to use k fold cross validation [23] to evaluate the performance of the different classifiers and report the average accuracy measure from the testing k *folds*. The MDLPC discretization strategy together with the k fold cross validation sets (with $k = 10$) have been obtained with $\mathcal{MLC}$++ [12].

4.1 Bayesian net structure

As explained above, the net structure can be defined by an expert but it can also be learned using some learning mechanism or a mixed technique can be used. Our option has been to try all three different approaches.

When learning Bayesian networks two subgoals can be identified. In one hand, the structure of the network must be fixed and in the other hand, the values of the probability tables for each node must be established. In this experimental work the probability tables are always estimated from the training database.

Proposed structure (no learning) A net structure is proposed based on the sonar configuration on the robot. Figure 3 shows the proposed net structure. In this net each node or variable is linked to its immediate neighbor. These arcs aim to represent that neighbor sensors are receiving information from the same

obstacle or that a sensor can receive echoes as a result of the activation of the neighborhood. The node corresponding to the class variable ($S11$) is related to all the sensor variables because we believe that they all together contribute to the definition of the class value. It must be noted that the proposed structure does take into account the classification purpose of the net.

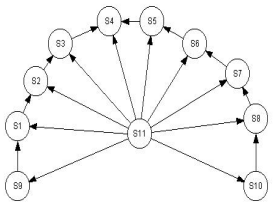

Fig. 3. Proposed net structure. There is no structural learning

Structural learning using different quality metrics In this second approximation, instead of using a fixed net structure the K2 algorithm has been used as a learning paradigm together with the three different metrics already explained. This algorithm treats all variables equally and does not consider the classifying task the net will be used for. The learned net structure depends completely on the random order generated before the learning process starts. In order to reduce the impact of the random order in the net structures learned, the experiments have been repeated 3000 times and the nets with the optimal values have been selected.

Mixed Technique In this last case, the nets have also been learned using the K2 algorithm in combination with the three different metrics but the order has not been randomly selected. Instead, a node ordering has been defined that allows the algorithm to learn the structure proposed: $S_{11}, S_{10}, S_9, S_8, S_1, S_7, S_2, S_6, S_3, S_5, S_4$. The class node can be parent of the rest of the nodes and thereby, assumes the classification objective of the net.

4.2 Robot behavior implementation

Figure 4 shows the pseudo code of the implemented behavior. Two things must be remarked on the algorithm. In one side, before propagation takes place we need to find the nearest case in the database; otherwise, if no exact match of the new sonar reading is found in the database the net will return the most frequent class in the database. The other remark concerns to the motor control section. We added a small variation factor that allows faster rotations when the classification is confirmed over time.

5 Results

We distinguish two sets of results. The first subsection refers to the results obtained off-line, that is, the cross validation results. The next subsection reports on the results obtained on the real robot.

```
Input: Bayesian network and database
Output: net structure
Begin:
   while(1)
       reading = discretize(read_sonars())
       nearest_match = find_nearest_match(reading, database)
       actual_class = propagate(nearest_match, net)
       V_left(t) = V_min − V_K × class(t) − (class(t − 1) + class(t)) × δV
       V_right(t) = V_min + V_K × class(t) + (class(t − 1) + class(t)) × δV
       set_robot_velocity(V_left(t), V_right(t))
End
```

Fig. 4. Pseudo-code of the door crossing behavior

5.1 Off-line results

Table 1 contains the average performance of the nets after the 10 fold cross validation. Although the performance differences are not dramatical, the nets learned with a random order using entropy and K2 measures give better results, followed by the net proposed by the authors based on the sonar configuration. In the other side, BIC metric gives the less accurate nets.

	k2-ran	bic-ran	entr-ran	k2	bic	entr	prop
Perf.	99.93	99.07	99.96	99.75	98.17	99.68	99.86

Table 1. Average performance obtained over the 10 folds

Table 2 shows the values we have obtained for the hamming distance between the proposed net and the learned ones, the total amount of links of the net, the number of links directly attached to the class node, and the complexity of the MB of the class node, as an attempt to measure the differences among the nets. This table reveals that the K2 nets are the most complex ones and the BIC metric gives the most simple nets because of the penalty factor. This is also remarked in figure 5 where all the nets learned using the complete database are shown.

	k2-rnd	k2-fix	bic-rnd	bic-fix	entr-rnd	entr-fix	prop
hamming	21	17	21	21	18	20	0
links	26	26	16	12	19	20	19
class links	10	10	7	3	10	10	10
links in MB	26	26	13	5	18	20	19

Table 2. Hamming distance and links

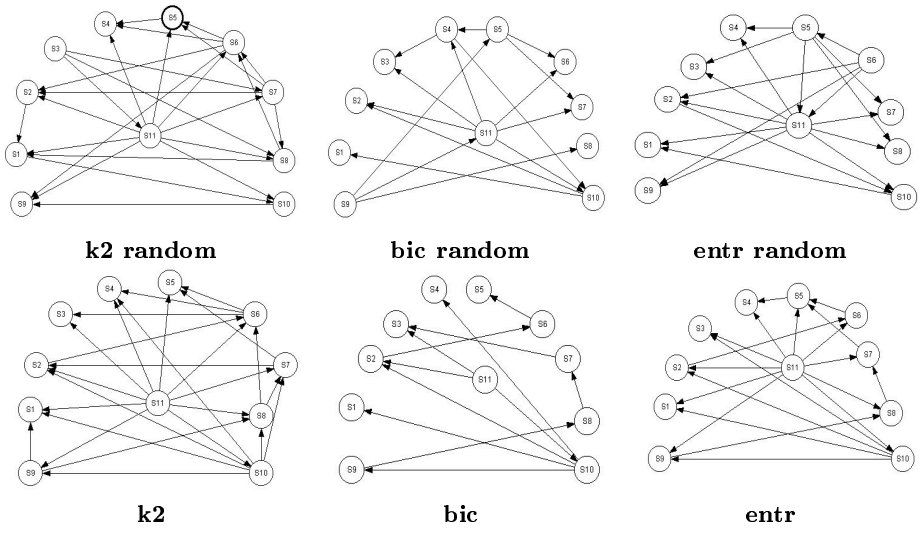

Fig. 5. The obtained nets

5.2 Behavior evaluation

It is not obvious how to proceed to evaluate the robot behavior. [6] defines behavior quality or correctness by means of factors like robustness (capability to correctly perform tasks with unknown data), adaptability (how the robot reacts to environmental dynamical changes) and flexibility (variety of tasks the robot can perform). In this paper only the door crossing problem is being tackled. Thereby, we think that the most objective way to measure the real-time performance is to just average the number of times the robot sucesfully manages to cross the door from different positions and headings.

Figure 6 show the experimental environment. The experiments were repeated 10 times for every position and for the best three nets obtained during the off-line phase. Table 3 shows these results.

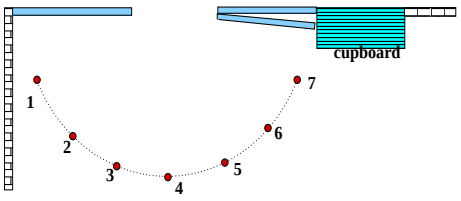

Fig. 6. Positions to measure the real-time performance of the behavior

6 Conclusions and Future Work

This paper presents the experimentation done as a first attempt to use Bayesian networks with the objective of performing the navigation behavior of a mobile robot when confronted to door crossings. A net structure has been proposed based on the topology of the sonar configuration and also a structural learning algorithm has been used in order to find a better structure, in combination with different structure quality measures to evaluate the appropriateness of the nets. Due to the large amount of values a sonar can provide, sonar data have bee discretized according to the MDLPC strategy. To alleviate the high computational cost of the exact propagation, a stochastic simulation has been used that, for this particular case where all the Markov Blanquet of the class node is instantiated becomes an exact propagation method with less computational load. Although the "off-line" results were impresive, the real-time execution of the behavior needs some improvements. Althoug position numbered 4 seems to be the easiest place from where door can be crossed, this is also the most noisy point because all the trajectories converge in that point. More data must be collected in order to behave more properly and, instead of taking into account only the reading at each time step independently, low filtered data will probably give more reliable results.

	1	2	3	4	5	6	7	%
entr-rnd	10	10	7	8	10	8	7	85.71
k2-rnd	10	8	9	7	9	6	5	77.14
prop	9	10	9	5	10	8	9	85.71

Table 3. Real-time performance of the door crossing behavior

Also, when looking for the nearest match something should be done to resolve ties. We think that applying the K-means algorithm to the database and find the nearest centroid would help not only resolving ties but also reducing the computational load. It must be said that the class probabilities can be calculated off-line and charged in memory.

The motor control must also be improved. We also think that this behavior must be combined with a door detection module to achieve more robust behavior.

Acknowledgements

We would like to thank to Anders L. Madsen, the Chief Technical Officer of the Hugin Expert, for preparing for us an extended demo version of HUGIN for Linux not so restrictive as the one available in the web. This experimentation was fullfilled thanks to the Prof. M. Graña.

References

1. S. K. Andreassen, K. G. Olesen, F. V. Jensen, and F. Jensen. HUGIN: a shell for building bayesian belief universes for expert systems. *Eleventh International Joint Conference on Artificial Intelligence*, pages 1080–1085, 1989.
2. W. Buntine. A guide to the literature on learning probabilistic networks from data. *IEEE Proceedings of the eighth Australian Joint Conference on Artificial Intelligence. World Scientific*, pages 99–106, 1996.

3. E. Castillo, J. M. Gutiérrez, and A. S. Hadi. *Expert Systems and Probabilistic Network Models.* Springer Verlag, 1997.
4. J. Catlett. On chanching continuous attributes into ordered discrete attributes. *Proceedings of the European Working Session on Learning. Springer Verlag*, pages 164–178, 1991.
5. G. F. Cooper and E. Herskovits. A bayesian method for induction of probabilistic networks from data. *Machine Learning. Kluwer Academic Publishers, Boston*, 9:309–347, 1992.
6. M. Dorigo and M. Colombetti. *Robot Shaping. An Experiment in Behavior Engineering.* A Bradford book. MIT Press, 1998.
7. J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning*, pages 194–202, 1995.
8. U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings of the 13th International Joint Conference on Artificial Intelligence. Morgan Kauffman*, pages 1022–1029, 1993.
9. N. Friedman and M. Goldszmidt. Building classifiers using bayesian networks. In *AAAI/IAAI, Vol. 2*, pages 1277–1284, 1996.
10. T. Gevers and A. W. M. Smeulders. Color based object recognition. In *ICIAP (1)*, pages 319–326, 1997.
11. F. V. Jensen. *Bayesian Networks and Decision Graphs (Statistics for Engineering and Information Science).* Springer, 2001.
12. R. Kohavi, D. Sommerfield, and J. Dougherty. Data mining using MLC++: A machine learning library in C++. In *International Journal on Artificial Intelligence Tools*, volume 6(4), pages 537–566, 1997. http://www.sgi.com/Technology/mlc.
13. H. A. Mallot and M. A. Franz. Biomimetic robot navigation. *Robotics and Autonomous System*, 30:133–153, 2000.
14. T. Mitchell. *Machine Learning.* McGraw-Hill, 1997.
15. J. Pearl. Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32 (2):247–257, 1987.
16. S. Rabaseda. *Contributions a l'extraction automatique de connaissances: application a l'analyse clinique de la marche.* PhD thesis, Université Lyon 1, 1996.
17. M. Sebban S. Rabaseda, R. Rakotomalala. Discretization of continuous attributes: a survey of methods. In *Proceedings of the second Annual Joint Conference on Information Sciences*, pages 164–166, 1995.
18. A. Saffiotti. *Fuzzy Logic in Autonomous Robot Navigation. A Case Study. Chapter 6 in Handbook of Fuzzy Computation.* Oxford University Press, E. Ruspini, P. Bonissone and W. Pedrycz edition, 1998.
19. G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.
20. C. E. Shannon. A mathematical theory of communication. *Bell Syst. Techn.*, 27:379–423, 623–656, 1948.
21. B. Sierra and P. Larrañaga. Predicting survival in malignant skin melanoma using bayesian networks automatically induced by genetic algorithms. An empirical comparison between different approaches. *Artificial Intelligence in Medicine*, 14:215–230, 1998.
22. S. Stoeter and L. M. Papanikolopoulos. Real-time door detection in cluttered environments. *Proceedings of the IEEE International Symposioum on Intelligent Control*, pages 187–192, 2000.
23. M. Stone. Cross-validation choice and assessment of statistical procedures. *Journal Royal of Statistical Society*, 36:111–147, 1974.