Sensor Planning and Bayesian Network Structure Learning for Mobile Robot Localization

Hongjun Zhou, Shigeyuki Sakane

Chuo University, Tokyo, Japan
{zhou,sakane}@indsys.chuo-u.ac.jp

Abstract

In this paper we propose a novel method of sensor planning for a mobile robot localization problem. We represent causal relation between local sensing results, actions, and belief of the global localization using a Bayesian network. Initially, the structure of the Bayesian network is learned from the complete data of the environment using K2 algorithm combined with GA (genetic algorithm). In the execution phase, when the robot is kidnapped to some place, it plans an optimal sensing action by taking into account the trade-off between the sensing cost and the global localization belief which is obtained by inference in the Bayesian network. We have validated the learning and planning algorithm by simulation experiments in an office environment.

1 Introduction

In this paper, we propose a sensor planning method for mobile robot localization. Initially, we represent causal relations between local sensing results, actions, and belief of the global localization in a Bayesian network (BN) structure. The BN structure, as well as the parameters, is learned automatically from the environment data using K2 algorithm combined with GA (genetic algorithm). In the execution phase, when the robot is kidnapped to some place, it plans an optimal sensing action by taking into account the trade-off between the sensing cost and the global localization belief which is obtained by inference in the BN[1].

2 Environment Information Gathering and BN Configuration

2.1 Path for Environment Information Gathering

We performed the simulation experiments in an office environment (Fig. 9). Initially, to obtain complete environment information, the robot must navigate in all of the corridors and intersections. We employ a framework of the *Chinese postman problem* [2]. The *Chinese postman problem* requires finding the shortest tour in a graph which visits every edge at least once. As shown in Fig. 1, we represent the topology of the environment as a graph and search a path from A to A using the *next node algorithm.* Then the robot navigates in all corridors and intersections along the path and gathers the environment information to be used for localization tasks.

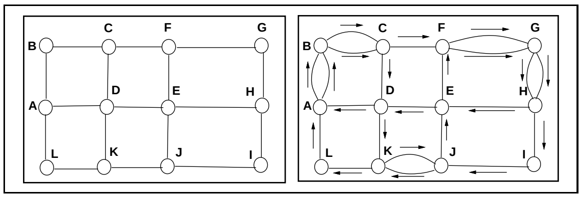

Figure 1: *(left) A graph to represent the topology of the environment. (right) A path (from A to A) obtained as a solution of Chinese postman problem.*

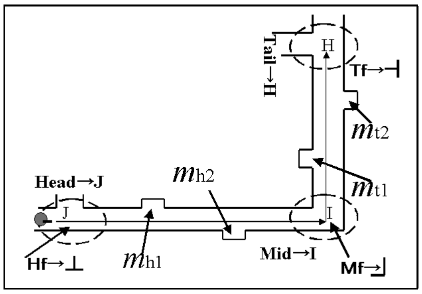

Figure 2: *Mapping the environment information of two neighboring corridors into nodes of BN.*

2.2 Environment Representation and BN Configuration

We define a *segment* (Sg) as the environment information of a corridor between two neighboring intersections. One *segment* involves four kinds of information:(1)Two intersection labels,(2) Landmarks on both sides of the corridor between two intersections,(3) Geometric features of the intersections sensed when the robot enters the intersections,(4)Action taken by the robot when it enters the corridor.

In our system, we call the environment information of two neighboring corridors an *environment information set*. The information of every *environment information set* (for example, label of an intersection, geometrical feature of an intersection, etc.) corresponds to a value of nodes in BN.

3 Learning BN Structure from Data

BN is a directed acyclic graph that represents dependencies between probabilistic variables. An arc between two nodes of BN represents the causal relation between the nodes. However, it is often difficult to determine the casual relation among nodes. In our localization tasks, we usually do not know which landmark has dependency with the other nodes, so we take a BN structure learning approach instead of designing the network structure manually.

3.1 $K2$ Algorithm Combined with GA

We apply a structure search method based on Bayesian score, named the $K2$ algorithm [4], to learn the causal relation between *local environment information*, *robot action*, and *global localization*. The Bayesian score is a joint probability $P(B_s, D)$ between BN structure (B_s) and database (D). The $K2$ algorithm is a greedy search algorithm. Ref[4] describes that the search space is too huge to evaluate all of the possible structures. To reduce the search space, the $K2$ algorithm uses a constraint of ordering of nodes (i.e., the causal attributes of a node should appear earlier in the order). However, it is often difficult to determine the order.

In our system, we employ a *genetic algorithm*(GA) to search the best ordering as described in Ref. [5]. Using this ordering, $K2$ learns the best BN structure from the data. Then the Bayesian score of $K2$ gives a *fitness* value to GA. The combination of GA and $K2$ iterates until the average fitness is improved no further.

3.2 Example of BN Structure Learning

Using the training database, we attempt to learn a structure of BN. The population size of the GA is 80 and the algorithm uses *crossover* and *mutation* operations. Figure 3 shows the convergence of fitness value with 100 generations. The dashed line and solid line in the figure show the average and the best fitness scores of each generation, respectively. By combining the $K2$ algorithm with the GA search, we can obtain a *sub-optimal* ordering of the nodes and a *semi-optimal* BN structure as shown in Fig. 4.

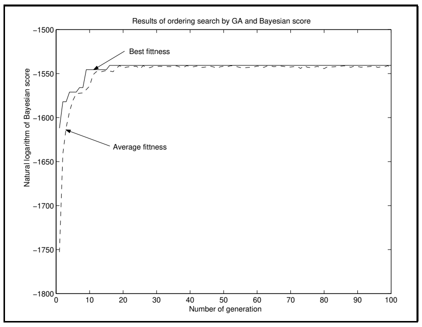

Figure 3: *The results of ordering searching by GA and Bayesian score*

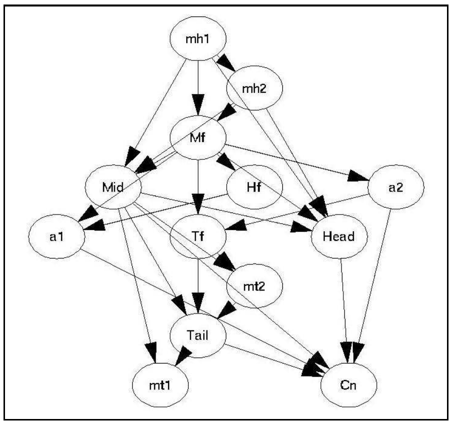

Figure 4: *Learned BN's structure by $K2$ and GA*

4 Sensor Planning for Localization

4.1 Inference for Localization

The robot starts navigation from an *unknown* position without sensor planning. The navigation basically uses a potential method in a corridor. The robot gathers sensing information events, including landmarks and geometric features of intersections, in the current corridor. Then the information events are given to the BN as evidences to infer global localization, i.e., which corridor the robot has sensed. The probability of the corridor's label is calculated as $P(Head, Mid, Tail|obtained\ sensing\ event)$ using BN.

We define belief of the global localization ($TolBef$) as follows:

$$TolBef = 0.5 * (max(P(Head)) + max(P(Mid))) \quad (1)$$

where $max(P(Head))$ and $max(P(Mid))$ are the maximum values of the probability of node $Head$ and Mid, respectively. $P(Head)$ and $P(Mid)$ are calculated by the BN inference.

If $TolBef \geq thd1$, the system terminates the localization process. Because in this case the robot can estimate the labels of the corridors only by using the current environment information, there is no need to perform sensor planning. Otherwise ($TolBef < thd1$), the robot has to move to the next corridor to perform active sensing. Therefore, the sensor planner selects an optimal sensing action for the localization.

Since the BN of our system is not a tree structure but has loops as shown in Fig. 4, we use the *Junction tree algorithm* [1] to infer probabilities of the nodes.

4.2 Prediction for Sensor Planning

The sensor planner consists of two processes: (1) prediction and (2) planning. The prediction process predicts some possible actions and sensing information expected to be obtained by these actions. The prediction algorithm has the following two steps:

(1) The first step is to search data cases, i.e., values of the node **Cn**, in the database, whose probabilitis are not zeros based on the sensing event obtained from the just-sensed corridor[1]. That is, the system stores the node **Cn**'s values, which satisfy the following condition, in a list $\mathbf{cnl} = (cnl_1, cnl_2, ...)$.

$$P(\mathbf{Cn}|obtained\ sensing\ information) \neq 0;$$

Based on the results, we can estimate which data case in the recorded sensing information database is closer to the obtained sensing information.

(2) The second step is to predict possible actions (PA) and sensor information (SI)[2] based on the obtained sensing information (OSI) and the estimated **Cn** values. The prediction is performed using the following probabilities:

$$P(PA|\mathbf{cnl}, OSI) > thd2 \quad (a)$$
$$P(SI|PA, \mathbf{cnl}, OSI) > thd2 \quad (b)$$

If the values of (a) and (b) exceed a certain threshold $thd2$, we save the possible actions in a list **actlist**, and save the predicted sensor information in a matrix $\mathbf{M_{sen}}$.

4.3 Sensor Planning Procedure

Through the prediction step, the system obtains **actlist**, a list of possible actions and also a matrix of predicted sensing information $\mathbf{M_{sen}} = (sn_1, sn_2, ..., sn_n)^T$, Each element of $\mathbf{M_{sen}}$ represents a predicted sensor information list to be obtained by a possible action (the element of **actlist**). Each predicted sensor information list of $\mathbf{M_{sen}}$ is sorted in the order of sensing cost, i.e. the distance from the current intersection to the location of the sensor information.

[1]The prediction and planning processes are performed when the robot is in the middle intersection.

[2]It includes landmarks and intersection's geometric features expected to be perceived when the robot takes the actions

Consequently, the sensor planning process selects an optimal action from **actlist** which allows the robot to acquire enough sensing events to decrease ambiguity of the global localization belief by taking into account the trade-off between the sensing cost and the global localization belief.

For example, an **actlist** and a $\mathbf{M_{sen}}$ are shown in Fig. 5. The possible actions are "*action1, action2, action3*", and the integers on the right side of Fig. 5 represent expected sensor information to be obtained by the actions. In the Fig. 5, each row of the $\mathbf{M_{sen}}$ is one set of the predicted sensor information when taking the action on the left side. Every row of the $\mathbf{M_{sen}}$ is sorted in ascending order of the sensing cost, i.e., the sensing cost of the right entry is larger than that of the left. In the evaluation process, we use the elements of the $\mathbf{M_{sen}}$ and the possible actions to estimate the labels of the intersections, and calculate the sensing cost. Since the robot uses sensing information of a set of two neighboring corridors, the *TolBef* should be defined as the sum of the maximum probabilities of the three intersection labels.

$$TolBef = (1/3) * (max(P(Head)) + max(P(Mid)) + max(P(Tail))) \quad (2)$$

Using the above possible actions and predicted sensor information, the system performs the sensor planning which has the following three steps:

(1) The first step is to use the already-obtained sensing information, the possible action, and sensing information to be obtained by the action, to infer (*TolBef*), the belief of the global localization. In this step, we must evaluate every action and every set of sensor information (every row of $\mathbf{M_{sen}}$ in Fig. 5). For example, when we evaluate "action1" of Fig. 5 and the corresponding sensor information of the three rows, the procedures are as follows ((a)∼(d)):

(a) The system creates an empty list (**SenEvn**), and pushes the left-most element of the first row's sensor information into **SenEvn**.

(b) Using the **SenEvn**, "action1" and obtained sensor information to estimate the $TolBef$ based on Eq.2 and BN.

(c)
```
IF TolBef > thd3
        OR
   all of the elements in this row
   have been pushed into SenEvn,
     THEN evaluation of the first
          row's sensor information
          is finished.
ELSE
     THEN the next element of the
          first row's sensor inf-
```

```
                 ormation is pushed into
                 SenEvn. GOTO b)
       END IF
```

(d) Using the procedure (a)$\sim$(c), the system also evaluates the other two row's sensor information corresponding to "action1". After the above procedures are finished, the number of sensor information sets (**count**), which have beliefs of the localization $TolBef > thd3$[3] will be recorded. (**count**) represents to what degree the robot could determine the *location* when it takes that action.

(2) The system sums up the sensing cost of every row's sensor information, which is used in the first step (**Cost**), and also sums up the **Cost** of each row (sensing information sets) which satisfy $TolBef > thd3$.

(3) Selects an optimal action by an *efficiency criterion*, i.e., an action which has the largest **count** and lower sensing cost.

For example, in the Fig. 5, each **count** of "action1" and "action3" is "3", and the **count** of "action2" is "1", so the optimal action can be selected from "action1" and "action3". Since the sensing cost of "action3" is lower than that of "action1", in this case, the optimal action should be "action3".

4.4 Speedup of the Sensor Planning

We explain the method using the case of Fig. 5.

(i) We compare the sensing information of the $\mathbf{M_{sen}}$ that has the same action from the left side to right side. For example, the *action1* corresponds to three sets of sensing information. In the first row of the sensing information sets (the row with $R1$ of Fig. 5), since the first element of rows $R2$ and $R3$ are $''2''$, if we get only the first sensing information $''1''$ of row $R1$, the system can distinguish the sensor information (row $R1$) from the other two sets of sensor information (rows $R2$ and $R3$) which have the same action (action1). Of course, we can also use more sensing information of row $R1$, but taking into account the sensing cost, using only the first sensing information, $''1''$, is more efficient.

(ii) We must test the *TolBef*(by Eq.2) using the selected sensing information ($''1''$) and its action, "action1". If *TolBef* $>$ *thd*3, we consider that the first sensing information ($''1''$) of row $R1$ expected by the "action1" is sufficient to uniquely determine a *location*. Otherwise, we must extend sensing information from its right side [4] and test the *TolBef* until the condition *TolBef* $>$ *thd*3 is satisfied.

(iii) Using steps (i) and (ii), we obtain the *narrowest* sensing range to distinguish the other sensing information sets which is shown in the gray region (before testing the $TolBef$) with the same action. If there are some sensing information sets, corresponding to the same action, which are identical, we cannot distinguish the information sets and cannot determine a *location* uniquely as shown in Fig. 5 by the black region.

[3]Since $TolBef > thd3$ means the robot can uniquely determine three intersections' labels (entrance, middle, and exit intersection), in other words, the robot can determine its global location by "action1" and the first row's sensor information.

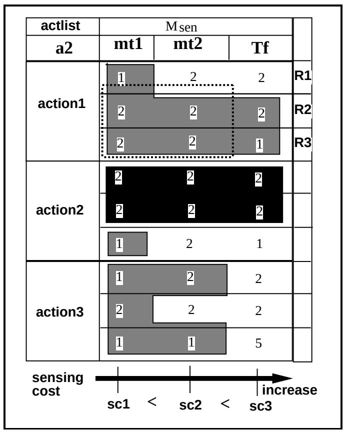

Figure 5: *An optimal action is determined by comparing the sensing information's local distance and geometric feature. The integer represents the sensing information. The numbers from left to right are values of m_{t1}, m_{t2}, and Tf, respectively. $Action1, action2, action3$ are values of a_2.*

5 Experiments

Using the above learning and planning algorithm, we performed simulation experiments in an office environment (Fig. 6). We implemented the BN learning and inference in a MATLAB BN Toolbox[6]. Note that we assume that the length of corridors $F \rightarrow G$,$E \rightarrow H$ and $J \rightarrow I$ is longer than the other corridors. In Fig. 6,8 and 9, the real numbers in parentheses, the numbers with black squares, and the numbers with hatched squares represent the probabilities of the nodes *Tail, Mid, Head*, respectively. The thresholds of the simulation experiments are defined as $thd1 = 0.9$, $thd2 = 0.9$, and $thd3 = 0.9$.

5.1 Inference for Localization

Initially, the robot starts from an unknown position of the environment. As shown in Fig. 6(a), without loss

[4]For example, in row $R1$, if $''1''$ is not sufficient, we must add the right side of the elements, $''2''$ or $''2,\ 2''$.

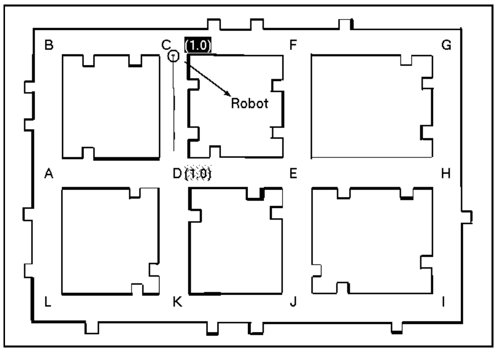

Figure 6: *Global localization using BN inference*

of generality, we assume the robot starts from an intersection D. After finishing sensing of the corridors, the robot's *global localization beliefs* are calculated by Eq.1. The probabilities of $Head$ and Mid are inferred using the learned BN and the sensor information of landmarks (m_{h1}, m_{h2}) and geometric feature Tf of the intersection. The sensor information which the robot obtained from corridors $D \rightarrow C$ is information of two landmarks, m_{h1}, m_{h2}, and the geometrical feature of intersection C. For example, information of two landmarks is denoted by number "2", and the geometrical feature is denoted by "⊤". Hence, the conditional probability of the node "Head", "Mid" is calculated as follows:

$$P(Head, Mid | m_{h1} = 2, m_{h2} = 2, Tf ='' \top'')$$

The results of the above conditional probability are shown in Table 1. Among the values (i.e., labels) of the nodes "Head" and "Mid", D and C take the maximum probabilities. Based on Eq.1, the global localization belief is calculated as $TolBef = 0.5 * (1.0 + 1.0) = 1.0$. Since $TolBef > thd1$, the start point and current position are determined as "D" and "C", respectively, and the global localization is determined. The experiment shows if the sensor information is sufficient, the robot can localize itself using the BN inference by the sensor information of only one corridor, and so sensor planning is not necessary.

5.2 Prediction for Sensor Planning

However, if the sensor information obtained from the just sensed corridor is insufficient, the robot has to perform active sensing to gather more sensor information to localize itself. In case of Fig. 9, the robot starts from intersection D (of course, the robot initially does not know its global position) and moves to intersection K, and the obtained sensing information is landmarks m_{h1}, m_{h2} and geometric feature Mf of the intersection. We use "1" and "2" to denote the landmarks m_{h1} and m_{h2}, respectively, and we denote geometric feature Mf of the intersection by "⊤". Using this sensor information, the system calculates the following conditional probability using the BN. The results of the conditional probability are shown in Table 2.

$$P(Head, Mid | m_{h1} = 1, m_{h2} = 2, Tf ='' \top'')$$

The $TolBel$ is calculated based on Eq.1, $TolBef = 0.5 * (1.0 + 0.5714) = 0.5 * 1.5714$. Since $TolBel < thd1$, the robot has two candidate locations indicated by a dot circle and a solid circle as shown in Fig. 9(a). The robot must perform sensor planning to decrease this uncertainty.

Using the sensor planning (described in Sec. 4.2), the robot predicts possible actions and sensing information expected by taking the actions, based on obtained sensing information (m_{h1}, m_{h2}, Mf). Using the *prediction algorithm*, the robot obtained **actlist** and $\mathbf{M_{sen}}$ as shown in Fig. 7. The predicted possible actions are "turn left" and "turn right", and the sensor information predicted by the possible actions has two rows, respectively.

5.3 Sensor Planning for Localization

Using the sensor planning procedure (described in Sec.4.3) and the speedup method (described in Sec.4.4), the robot can determine its location based on the sensing information which is shown by the dark background in Fig. 7. The experimental results show that we can obtain the same sensing range (marked in dark) using the *sensor planning procedure* as well as the *speedup*

nodes	probability of the intersection's labels									
	A	B	C	D	E	F	G	H	I	J
Head	0	0	0	1.0	0	0	0	0	0	0
Mid	0	0	1.0	0	0	0	0	0	0	0
	K	L								
Head	0	0								
Mid	0	0								

Table 1: *The inferred probabilities of the nodes Head and Mid in Fig.7(a).*

nodes	probability of the intersection's labels							
	A	B	C	D	E	F	G	H
Head	0	0	0	1	0	0	0	0
Mid	.5714	0	0	0	0	0	0	0
	I	J	K	L				
Head	0	0	0	0				
Mid	0	0	.4286	0				

Table 2: *The inferred probabilities of the node Head and Mid in Fig. 10(a).*

actlist	M_{sen}		
a2	mt1	mt2	Tf
turn left	2	2	3
	2	2	6
turn right	2	1	5
	1	1	5

Figure 7: *Predicted possible actions (***actlist***) and the sensing information predicted by the actions ($\mathbf{M_{sen}}$) based on sensing information of the corridor ($D \to K$). (The integers of the table represent the predicted sensor information, i.e., instantiations of the probabilistic variables (m_{h1}, m_{h2}, Tf) in the BN)*

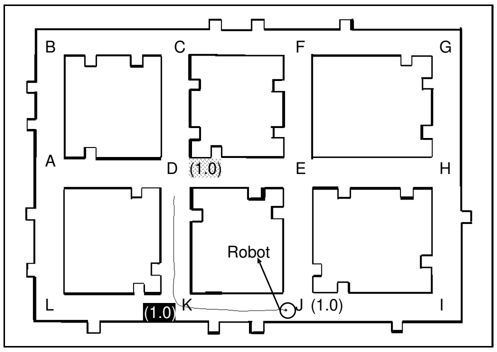

Figure 8: *The robot cannot obtain sufficient sensor information for localization until it goes to intersection J.*

method. In Fig. 9, either the "turn left" or "turn right" action can determine two possible robot *locations*, but the sensing cost of "turn right" is lower than that of "turn left" (the area of dark region expected by taking the action "turn right" is smaller than that of "turn left"). As shown in Fig. 8, if the robot takes the "turn left" action, it cannot localize itself until it goes to intersection J. Hence, the optimal action is "turn right" and the robot need not go to the next intersection for the global localization (Fig. 9(b)).

6 Conclusion

We proposed a novel sensor planning method for mobile robot localization using a Bayesian network. The BN structure is learned from environment data based on the $K2$ algorithm combined with GA. In the execution phase, the sensor planner predicts possible actions and sensing information to be obtained from these actions, and selects an optimal plan by taking into account the trade-off between the global localization belief and the sensing cost. The system are validated by simulation experiments.

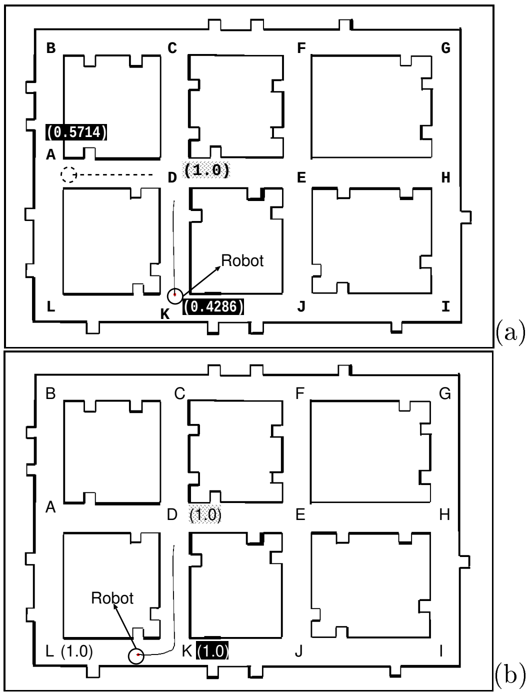

Figure 9: *Example of sensor planning experiments for robot localization. (In the figure, the real numbers in ( ), ( ) with black squares and ( ) with hatched squares represent the probability of node Tail, Mid, Head, respectively. If the intersection is the instantiation of node Tail, Mid, Head, the probability is shown at the intersection.*

References

[1] F. V. Jensen, "Bayesian Networks and Decision Graphs", *Springer*, 2001.

[2] M.Kwan, "Graphic Programming using ODD or EVEN Points", *Chinese Math*, 1, pp.273-277, 1996.

[3] N. Cristianini and J. Shawe-Taylor, "An Introduction to Support Vector Machines", *Cambridge University Press*, 2000.

[4] G. Cooper and E. Herskovits. "A Bayesian method for the induction of probabilistic networks from data". *Machine Learning*, 9:309-347, 1992.

[5] P. Larranaga, C. Kuijpers, R. Murga, Y. Yurramendi. "Learning Bayesian network structures by searching for the best ordering with genetic algorithms". *IEEE Trans. on System, Man and Cybernetics*, Vol.26, No.4, pp.487-493, 1996.

[6] Kevin Murphy, "The Bayes Net Toolbox for Matlab", *Computing Science and Statistics*, vol.33, 2001.