Place Learning and Recognition using Hidden Markov Models

Olivier AYCARD, François CHARPILLET, Dominique FOHR
& Jean-François MARI
CRIN/CNRS & INRIA-Lorraine
Bâtiment LORIA, Campus scientifique
B.P. 239
54506 Vandœuvre-lès-Nancy CEDEX
FRANCE

keywords : perception

Abstract

In this paper, we propose a new method based on Hidden Markov Models to learn and recognize places in an indoor environment by a mobile robot. Hidden Markov Models have been used for a long time in speech recognition, and we apply them to learning and recognition of places because they are well adapted to model temporal signals. The definition of Hidden Markov Models, the learning algorithm and the recognition algorithm are presented. Their applications to learning and recognition of places by a mobile robot are addressed. Results of two experiments on a real robot with five distinctive places are given.

1 Introduction

The automatic recognition of places is an important issue that determines the capability of a mobile robot to locate itself in its environment. This is the first step in the construction of cognitive maps [3].

Many researchers have studied place learning and place recognition for mobile robots. This can be done by two different ways:

- The knowledge based systems define rules to build a representation of places. For instance, [3] defines rules about the variation of the sonar sensors to learn different types of places and adds a visual information to recognize two places of the same type. [8] uses ultrasonic sensors to build evidence grids [1] associated with places, and defines an algorithm to match two places. He applied his method to spatial learning of a dynamic indoor environment and re-localization in dynamic indoor environments [9].
- The statistical systems attempt to describe the observations coming from the sensors as a random process that must be modeled in a proper way. For instance, [6] teach a neural network to extract places in an indoor environment from the data given by a distance measurement system based on a panoramic laser telemeter and use it to recognize the places previously learned.

These two approaches are opposite from each other. In the first approach, we look for understanding the observations and building a representation of the observations, whereas in the second approach, we build models that represent the statistical properties of the observations.

Stochastic modeling is a flexible method for handling the large variability of complex temporal signals. In contrast to dynamic time warping where heuristic training methods for estimating templates are used, stochastic modeling allows a probabilistic and automatic training for estimating models. A very complete tutorial on Hidden Markov Models and their application to speech can be found in [5].

In this paper, we present a new method to learn and recognize places based on second-order Hidden Markov Models (HMM2). HMM2 have been shown to be efficient models to capture temporal variations in speech [4] and in many cases they overcome first order Hidden Markov Models (HMM1). We use them to learn and recognize places by a mobile robot running in an indoor environment.

This paper is organized as follow. In section 2, we give a short presentation of our mobile robot. In section 3, we define the HMM2 and give the algorithms used for training and recognition. Section 4 is the description of our methodology. We discuss results in section 5 and give some conclusions and perspectives in section 6.

2 Description of our robot

Our robot (figure 1) is a Nomad200 commercialized by Nomadics [7]. It is composed of a base and a turret. The base is formed by 3 wheels, and tactile sensors. The turret is an uniform 16-side polygon. On each

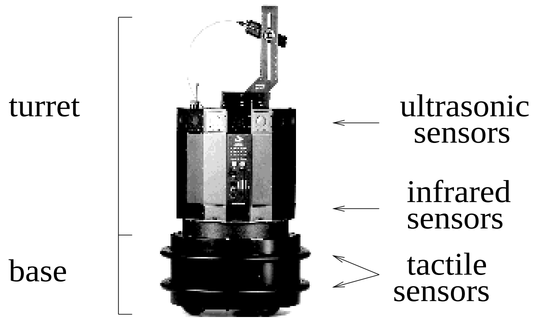

Figure 1: Our mobile robot

side, there is an infrared and an ultrasonic sensor. The turret can rotate independently of the base.

2.1 Tactile Sensors

A ring of 20 tactile sensors surrounds the base. They detect contact with objects. They are just used for the emergency cases. They are associated with low-level reflexes as emergency stop, and backward movement.

2.2 Ultrasonic Sensors

The angle between two ultrasonic sensors is 22.5 degrees, and each ultrasonic sensor has a beamed width of approximately 23,6 degree. By examining all 16 sensors, we can obtain a 360 degree panoramic view fairly rapidly. The ultrasonic sensors give range information from 17 to 255 inches. But the quality of the range information greatly depends on the surface of reflection, and the angle of incidence between the ultrasonic sensor and the object.

2.3 Infrared Sensors

The infrared sensors measure the light differences between an emitted light and an reflected light. They are very sensible to the ambient light, the object color, and the object orientation. As we assume that for short distances, the range information is acceptable, we just use infrared sensors for the areas shorter than 17 inches, where the ultrasonic sensors are not usable.

2.4 Odometry Measurements

The odometry measurement integrates the translation and rotation of the robot, and updates the position and orientation of the robot. As with all odometric systems, it accumulates errors during movements. We use it to have a coarse idea of the position and orientation of the robot.

3 The second-order Hidden Markov Models

In an HMM2, the underlying state sequence is a second-order Markov chain. Therefore, the probability of a transition between two states at time t depends on the states in which the process was at time $t-1$ and $t-2$.

A second Order Hidden Markov Model λ is specified by:

- a set of states called S;
- a 3 dimensional matrix a_{ijk} over S x S x S

$$a_{ijk} = Prob(q_t = s_k / q_{t-1} = s_j, q_{t-2} = s_i) \quad (1)$$

$$= Prob(q_t = s_k / q_{t-1} = s_j, q_{t-2} = s_i, q_{t-3} = ...)$$

  with the constraints

$$\sum_{k=1}^{N} a_{ijk} = 1 \quad with\ 1 \leq i \leq N\ ,\ 1 \leq j \leq N$$

  where N is the number of states in the model and q_t is the actual state at time t ;

- each state s_i is associated with a mixture of Gaussian distributions :

$$b_i(O_t) = \sum_{m=1}^{M} c_{im} \mathcal{N}(O_t; \mu_{im}, \Sigma_{im}), \quad (2)$$

$$with \quad \sum_{m=1}^{M} c_{im} = 1$$

  where O_t is the input vector (the frame) at time t.

The probability of the state sequence $Q = q_1, q_2, ..., q_T$ is defined as

$$Prob(Q) = \pi_{q_1} a_{q_1 q_2} \prod_{t=3}^{T} a_{q_{t-2} q_{t-1} q_t} \quad (3)$$

where Π_i is the probability of state s_i at time t = 1 and a_{ij} is the probability of the transition $s_i \rightarrow s_j$ at time t = 2.

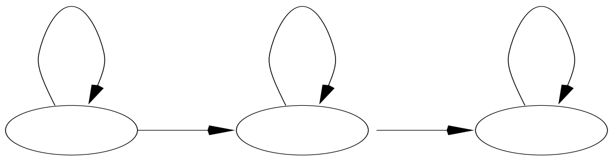

Figure 2: Topology of states used for each model of place

In this formalism, each place to be recognized is modeled by a HMM2 whose topology is depicted in figure 2. Currently, we choose to model five distinctive places that are representative of our office environment: a corridor, a T-intersection, a "starting" angle when the robot moves away from the angle, and an "ending" angle when the robot arrives at this angle and an open door (figure 3). This set of items is an extensive description of what the mobile robot can see during its run. All other unforeseen objects, like people wandering along in a corridor, are treated as noise.

In this experiment, we have to face several major issues: design efficient algorithms for training and recognition purposes; collect a corpus of observations during several runs; label this corpus by finding temporal borders of each items that the robot has observed during its run.

The recognition is carried out by the Viterbi algorithm [2] which determines the most likely state sequence given an observed sequence of observations. The learning of the models is performed using the maximum likelihood estimation criteria that determines the best models's parameters according to the corpus of items. It must be noted that this criteria does not try to separate models like a neural network does, but only tries to increase the probability that a model generates its corpus independently of what the other models can do.

3.1 The Viterbi Algorithm

In Hidden Markov Models, many state sequences may generate the same observed sequence $o_1,...,o_T$. Given one such output sequence, we are interested in determining the most likely state sequence $q_1,...,q_T$ that could have generated the observed sequence. The extension of the Viterbi algorithm to HMM2 is straightforward. We simply replace the reference to a state in the state space **S** by a reference to an element of the 2-fold product space **S x S**. The most likely state sequence is found by using the probability of the partial alignment ending at transition (s_j, s_k) at times (t-1,t)

$$\delta_t(j,k) = Prob(q_1, ...q_{t-2}, q_{t-1} = s_j, q_t = s_k, o_1, ..., o_t/\lambda) \quad 2 \le t \le T, \quad 1 \le j,k \le N. \tag{4}$$

Recursive computation is given by equation

$$\delta_t(j,k) = max_{1 \le i \le N}[\delta_{t-1}(i,j) \cdot a_{ijk}] \cdot b_k(O_t) \quad 3 \le t \le T, \quad 1 \le j,k \le N. \tag{5}$$

The Viterbi algorithm is a dynamic programming search that computes the best partial state sequence up to time t for all states. The most likely state sequence $q_1, ..., q_T$ is obtained by keeping track of back pointers for each computation of which previous transition leads to the maximal partial path probability. By tracing back from the final state, we get the most likely state sequence.

The robot's environment is described by means of a grammar that enables some sequence of models and restrict other. According to this grammar, all the HMM2 are merged in a bigger HMM on which the Viterbi algorithm is used. Then, the best sequence of states determines the ordered list of places that the robot saw during its run. It must be noted that the list of models is known when the run is completed.

3.2 The Forward-Backward Algorithm

The Forward-Backward algorithm implements the models's estimation following the maximum likelihood estimation criteria. Intuitively, this algorithm counts the number of occurences of each transition between the states in the training corpus. Each count is weighted by the probability of the alignment (state, observation). Since many state sequences may generate a given output sequence, the probability that a model λ generates a sequence $o_1,...,o_T$ is given by the sum of the joint probabilities (given in equation ??) over all state sequences (i.e, the marginal density of output sequences). To avoid combinatorial explosion, a recursive computation similar to the Viterbi algorithm can be used to evaluate the above sum. The forward probability $\alpha_t(j,k)$ is :

$$prob((O_1, ..., O_t = o_1, ..., o_t), q_{t-1} = s_j, q_t = s_k/\lambda) \tag{6}$$

This probability represents the probability of starting from state 0 and ending with the transition (s_j, s_k) at time t and generating output $o_1,...,o_t$ using all possible state sequences in between. The Markov assumption allows the recursive computation of the forward probability as :

$$\alpha_{t+1}(j,k) = \sum_{i=1}^{N} \alpha_t(i,j).a_{ijk}.b_k(O_{t+1}), \tag{7}$$
$$2 \le t \le T-1, \quad 1 \le j,k \le N$$

This computation is similar to Viterbi decoding except that summation is used instead of max. The value $\alpha_T(j,k)$ where s_k = N is the probability that the Model λ generates the sequence $o_1, ..., o_t$. Another useful quantity is the backward function $\beta_t(i,j)$, defined as the probability of the partial observation sequence from t+1 to T, given the model λ and the transition (s_i, s_j) between times t-1 and t, can be expressed as :

$$\beta_t(i,j) = Prob(O_{t+1}, ...O_T/q_{t-1} = s_i, q_t = s_j, \lambda), \tag{8}$$
$$2 \le t \le T-1, \quad 1 \le i,j \le N$$

4 Application to mobile robotics

4.1 The corpus collecting and labeling

We built a corpus to train a model for each of the 5 places. For this, our mobile robot makes 50 passes (back and forth) in a very long corridor (approximatively 30 meters). This corridor (figure 4) contains two

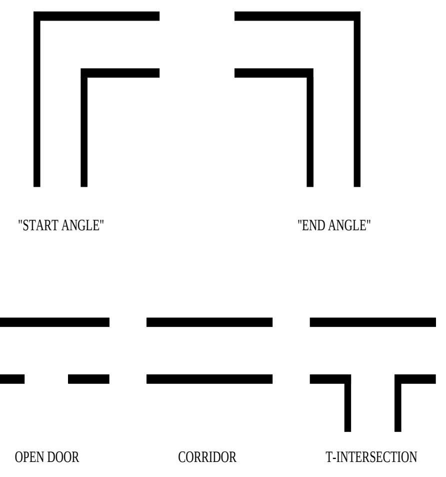

Figure 3: The places to learn

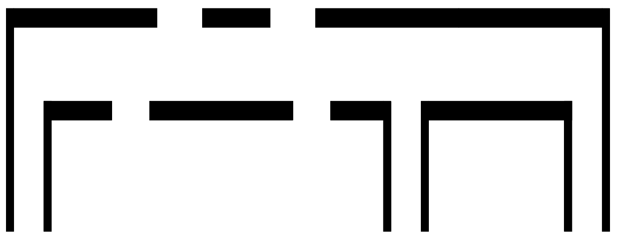

Figure 4: The corridor used to make our learning corpus

angles (one at the start of the corridor and one at the end), a T-intersection and some open doors (at least four, and not always the same). The robot ran with a simple navigation algorithm to stay in the middle of the corridor in a direction parallel to the two walls constituting the corridor. While running, we store all the ultrasonic sensors' measures of our robot. The acquisitions are done in real conditions with people wandering in the lab, doors completely or partially opened and static obstacles like shelves.

A pass in the corridor contains not only one place but all the places seen while running in the corridor. To learn a particular place, we need to segment passes in distinctive places. Moreover, we need to select the pertinent sensors' measures to observe a place. This task is more complex because the sensors' measures are noisy and when there is a place on the right side of the robot, there is an other place on the left side of the robot. For these reasons, we choose to segment passes to use for each side, the sensor perpendicular to each wall of the corridor and its two neighbor sensors. These three sensors normally gives valid measures. As all places except the corridor cause a noticeable variation on these three sensors over time, we define the beginning of a place when the first sensor's measure suddenly increases and the end of a place when the last sensor's measure suddenly decreases. The figure

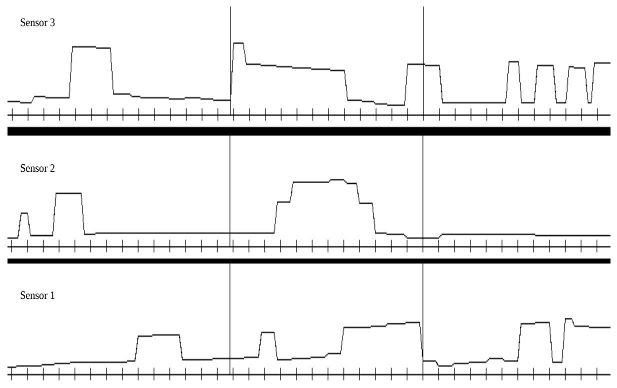

Figure 5: The segmentation corresponding to a T-intersection

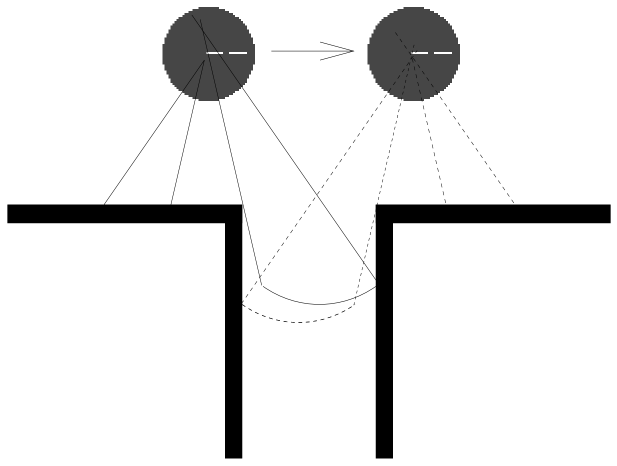

Figure 6: The three sonars used for the segmentation of a T-intersection

5 shows an example of the segmentation on the right side with these three sensors of a part of an acquisition corresponding to a T-intersection. The first line segment is the beginning of the T-intersection (sudden increase on the first sensor), and the second line segment is the end of the T-intersection (sudden decrease on the third sensor). The left part of the first line and the right part of the second line are a corridor place. The figure 6 shows position of the robot at the beginning and at the end of the T-intersection and the measures of the three sensors used at these two positions for the segmentation.

4.2 The models training

As said earlier, we choose three coefficients corresponding to the three sensors' measures. Because the segmentation was made using the variations on these three sensors' measures, we use the first derivative of the three sensors' measures. The topology used to train each model is shown in figure 2. Intuitively, we can think that the first state will contain the strong increase of the signals corresponding to the beginning of the place, the second State will contain the stationary part of the signals (where the derivative is nearly equal to zero) and the third State will contain the end of the place where the signal decreases strongly.

Two different kind of training are performed. The first training uses segmented data and each model is trained independently on these data. The second training uses the former models and estimates them on unsegmented data like in the recognition phase. It means that we merge the models seen by the robot during a complete run in a bigger model according to the sequence of observed items and train the resulting model with the unsegmented data.

4.3 The recognition phase

The recognition is performed in two steps. We took 40 acquisitions (20 passes) and used the five models trained to perform the recognition. A place is recognized if it has been detected by the corresponding model and it has been found near (less than 50 centimeters) its real geometric position. But different types of errors occured:

Insertions are places inserted in an other place. For instance, during the recognition of a corridor place (especially long corridor places), the robot sometimes recognizes the beginning of the corridor place, an inserted place (generally a T-intersection place or an open door place) and the end of the corridor places. We noticed small insertions of places (particularly in long corridor places) corresponding to reflections (when the three sensors are not perpendicular to the walls). These insertions are automatically rejected, because it makes no sense to consider an open door or a T-intersection of less than 80 centimeters (size of the smallest place of the environment). Insertions are actually considered when they have a valid size (more than 80 centimeters).

Deletions are places that the mobile has not recognized.

Substitutions are places that the mobile robot has confused with an other.

5 Discussion

The results are presented in a confusion matrix (table 1). Each column corresponds to a place seen during a pass. Each line corresponds to a place that the robot has recognized. For instance, 164 corridor places have been seen over the 40 passes. 160 corridor places have been recognized as corridor places, 1 as "end" angle place (a substitution) and 3 have not been recognized by the robot. So, 97% of the corridor places have been recognized. The last column (in table ?? corresponds to the insertions. For instance, the corridor place has been inserted 62 times.

						Ins.
corridor	160	0	0	0	7	62
"start" angle	0	15	0	0	0	8
"end" angle	1	0	19	0	0	17
T-inter.	0	1	1	16	38	67
open door	0	0	0	2	77	0
deletions	3	0	0	0	1	0
Total	164	16	20	18	122	164
% reco.	97	93	95	89	63	

Table 1: Confusion matrix of places

	number	%
Places seen	340	100
Recognized	287	85
Substitued	49	14
Deleted	4	1
Inserted	164	48

Table 2: Results of global recognition

In table 2, we present the global results. It means the number and percentage of places seen, recognized, substituted, omitted and inserted.

The places are globally well recognized (over 85% of recognition, for each place, except the open door place, and 85% of global recognition). The only problem is the percentage of insertions (48 %) of other places. Insertions are due to the navigation algorithm. Sometimes the mobile robot has to avoid people or obstacles, and in these cases it does not always run parallel to the two walls, and in the middle of the corridor. These conditions cause reflections on the three sensors which are interpreted as places. The deletions are very few (less than 1 percent).

Corridor place, "start" angle place and "end" angle place are very well recognized. Theses places have a very particular pattern, and can difficulty be confused with an other. Corridor places are characterized by a very small variation in time over the three sensors. Start angle places always begin with a progressive increase on the three sensor' measure, and end with a sudden decrease on the last sensor's measure. End angle places always begin with a sudden increase on the first sensor's measure, and the end is marked by a progressive decrease on the three sensor's measure.

T-intersections are sometimes confused with open doors and open doors with T-intersections. As we use only one model for all open doors, and in real conditions we find some different open doors (partially or completely opened, with doors opened on the right or the left), the model of open doors is representative of this diversity. Moreover, some open doors looks like (from a shape and size view) T-intersection , so the confusion is very easy, which explains the low percentage of recognition of open door places (63 %) and the high percentage of substitution of open door places by T-intersection places (31 %). If we used more specific open door places, for instance, one for doors opened on the left and one for doors opened on the right, or to use three models (or more) for a door : one when the robot is along one wall, one when the robot is along the other wall and one when the robot is in the middle of the two walls, we could have less confusion, and so decrease the percentage of substitutions between open door places and T-intersection places, which means a better percentage of recognition of these two places.

We implemented the recognition algorithm directly on the mobile robot. The mobile robot was asked to perform high level instructions like "go to the end of the corridor" or "go to the third open door on the right". The mobile robot runs memorizing its ultrasonic sensor's measurements and a recognition is periodically performed (each 70 centimeters approximatively). The results are noticeably similar to the previous experiments. The robot always stops at the end of the corridor when it is asked to. But, sometimes (approximately 25 %) the robot stops before or after the asked place.

6 Conclusion and perspectives

In this paper, we have presented a new method to learn and recognize places in an indoor environment with Hidden Markov Models. This method gives good results, and good robustness to the noise, as far as the navigation algorithm runs in adequacy. The results can be improved by adding more models to take into account different classes of the same place. As in [3], a place can be recognized when the robot has crossed it completely. It means that to turn at a T-intersection, the mobile robot has to move back.
This method actually implemented on our mobile robot is used to localize our mobile robot in its environment, and can be easily used to build cognitive maps of the environment.

References

[1] A Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer*, 22(6):46–57, June 1989.

[2] G.D. Forney. The Viterbi Algorithm. *IEEE Transactions*, 61:268–278, March 1973.

[3] D Kortenkamp and T Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *proceedings of the 1994 AAAI Conference*, 1994.

[4] J.-F. Mari, J.-P. Haton, and A. Kriouile. Automatic Word Recognition Based on Second-Order Hidden Markov Models. *IEEE Transactions on Speech and Audio Processing*, 5, January 1997.

[5] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech recognition . *IEEE Trans. on ASSP*, 77(2):257 – 286, February 1989.

[6] C Rompais, J.F Barret and G Pradet. Neural classifier for environnement recognition in mobile robotics. In *proceedings of the 1995 International Conference in Artificial Neural Networks*, pages 463–468, 1995.

[7] Nomadic Technologies. Nomad 200 usert's manual. Technical report, 1996.

[8] B Yamauchi. *Exploration and spatial learning in dynamic environnements*. PhD thesis, Case Western Reserve University, 1995.

[9] B Yamauchi. Mobile robot localization in dynamic environments using dead reckoning and evidence grids. In *proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 674–680, 1996.