

Learning Bayesian models of sensorimotor interaction: from random exploration toward the discovery of new behaviors*

Éva Simonin, Julien Diard and Pierre Bessière
Laboratoire GRAVIR / IMAG – CNRS, INRIA Rhône-Alpes, France
eva.simonin@imag.fr

Abstract— We are interested in probabilistic models of space and navigation. We describe an experiment where a Koala robot uses experimental data, gathered by randomly exploring the sensorimotor space, so as to learn a model of its interaction with the environment. This model is then used to generate a variety of new behaviors, from obstacle avoidance to wall following to ball pushing, which were previously unknown by the robot. The learned model can be seen as a building block for a hierarchical control architecture based on the Bayesian Map formalism.

Index Terms— Navigation, space representation, learning, behavior, bayesian model.

I. INTRODUCTION AND RELATED WORK

This paper is concerned with the domain of probabilistic modeling for space representation and navigation. Most of the approaches in this domain try to represent a large environment with a single global model, in the form of a map. Such approaches, because of the difficulty of this task, are generally tailored toward one specific use of this map, such as localization, planning, or simultaneous localization and mapping. For instance, on the one hand, Markov localization [1] and Kalman Filters [2] have focused on the localization process, either in a metric map such as an occupancy grid [3], or a topological map [4], or even a hybrid representation [5], [6]. On the other hand, many planning techniques have been developed in the context of probabilistic models of the environment [7], [8].

However, representing accurately and coherently a large environment with a single, flat model is still a difficult task. We follow an alternative approach and focus on modular and hierarchical models. Indeed, in previous works, we have defined the Bayesian Map formalism for defining probabilistic models of sensorimotor relationships and Operators (Abstraction, Superposition) for putting together such models [9], [10], [11]. That is why our context, in this paper, is the definition of an initial Bayesian Map based on a low-level sensorimotor relationship, in order to provide a building block for a future hierarchy.

In the Bayesian Map formalism, maps are probabilistic models that provide navigation resources in the form of behaviors. These behaviors are formally defined by probabilistic distributions computed from the map. On the other hand, the literature typically defines behaviors as goal-oriented models, and maps as goal-independent models [12], [13], without

explicating their relationship. So, in this article, we describe an experiment we have made in order to investigate how the learning of maps and behaviors can be articulated.

Indeed, this experiment demonstrates the possibility, for a mobile robot, to learn a model of its interaction with the environment (a Bayesian Map), starting from a known behavior and then to use this model to generate various behaviors *not previously known by the robot*.

The rest of this paper is structured as follows. Section II describes the scenario of our experiment, the way our probabilistic model is learned and the way it can then be used to provide behaviors. In Section III, the implementation of the experiment on a Koala mobile robot is detailed. Section IV presents the results and the obtained behaviors. Finally, Section V discusses some open issues relevant to this work, in particular, with respect to including the learned model in a subsequent hierarchy of Bayesian Maps.

II. EXPERIMENT DESCRIPTION

A. Principle

The scenario of our experiment is as follows.

A robot is given an initial behavior and is asked to apply it in the environment. While it performs this behavior, the sensory inputs and motor commands are recorded. These are the observations needed to learn the effect of an action on the perceptions of the robot. These observations are then used to build an internal representation of the environment, in the form of a probabilistic model, which encodes the correlation between the perceptions and the actions of the robot.

Once learned, this model is then used to estimate the probable consequences of an action, knowing the sensor data. Being able to estimate these consequences, the robot can evaluate which action will get it closer to the goal corresponding to the current navigation task. Defining such goals will be the means to generate different new behaviors for the robot. In other words, a behavior defines a goal in the sensorimotor representation: reaching this goal solves a particular navigation task in the environment. In order to solve several navigation tasks, we will define several behaviors using the learned model.

B. Learning the probabilistic model

The learned probabilistic model follows the dependency structure defined in the Markov localization framework [1], [14]. However, as the focus of this model is not localization

*This work is supported by the BIBA European project (IST-2001-32115).

but behavior generation, it more precisely falls into the Bayesian Map formalism.

The Markov localization model corresponds to the following probabilistic dependency structure:

$$\begin{aligned} P(P \ L_t \ L_{t+\Delta t} \ A) \\ = P(L_t)P(A)P(P \ | \ L_t)P(L_{t+\Delta t} \ | \ A \ L_t), \end{aligned} \quad (1)$$

where P is a perception variable at time t , A is an action variable at time t , and L_t and $L_{t+\Delta t}$ are instances of a location variable at times t and $t + \Delta t$, respectively. In the following, we will call $P(P \ | \ L_t)$ the *sensor model*, because it describes what should be seen on the sensors given the current location. $P(L_{t+\Delta t} \ | \ A \ L_t)$ will be called the *prediction model*, because it describes what location the robot should arrive at, given the past location and action.

We now define parametric forms for the terms in this probabilistic model. $P(L_t)$ and $P(A)$ are uniform distributions, as we have neither prior knowledge on the location of the robot at time t , nor on the action to apply. The *sensor model* and the *prediction model* are both assigned Gaussian distributions:

$$\begin{aligned} P(P \ | \ L_t) &= \mathbf{G}_{\mu_1(L_t), \sigma_1(L_t)}(P), \\ P(L_{t+\Delta t} \ | \ A \ L_t) &= \mathbf{G}_{\mu_2(A, L_t), \sigma_2(A, L_t)}(L_{t+\Delta t}), \end{aligned}$$

where μ_1 and σ_1 are respectively the mean and variance functions of the sensor model, and where μ_2 and σ_2 are respectively the mean and variance functions of the prediction model. As this paper focuses on the learning of the prediction term, the description of the sensor model is not detailed here (see Section III). To complete the model, we now show how the μ_2 and σ_2 functions are experimentally identified.

These functions can be obtained using the sensorimotor data recorded while the robot is applying the initial behavior. Indeed, at each time step, the location and the action of the robot are recorded. Thus, for each $\langle A, L_t \rangle$ pair, the mean and variance of the robot location at the next time step can be computed using the data history.

C. Using the probabilistic model

Once the probabilistic model is learned, the robot can solve navigation tasks. These tasks consist in trying to reach some goal in the sensorimotor space chosen by the programmer. To solve such a navigation task, the following question is asked to our model:

$$P(A \ | \ [L_t = l_1] \ [L_{t+\Delta t} = l_2]),$$

i.e. knowing that the current robot location is l_1 and that the location it has to reach is l_2 , what is the action it needs to apply? Therefore l_2 is the chosen goal and by computing, at each time step, the action to apply to move closer to it, the robot is able reach it eventually. We call the question $P(A \ | \ [L_t = l_1] \ [L_{t+\Delta t} = l_2])$ a *behavior*.

We can answer the question using our probabilistic dependency structure, by applying bayesian inference:

$$\begin{aligned} P(A \ | \ [L_t = l_1] \ [L_{t+\Delta t} = l_2]) \\ \propto P([L_{t+\Delta t} = l_2] \ | \ A \ [L_t = l_1]). \end{aligned} \quad (2)$$



Fig. 1. The Koala mobile robot (K-Team company).

The intuitive interpretation of this computation will best be detailed on an example (see Section IV-A).

Having computed the probability distribution over A , we can now draw at random according to this distribution an action to perform to move closer to the goal. This action is executed by the robot to move in the environment: the robot is therefore applying a behavior.

III. APPLICATION ON A KOALA ROBOT

A. Experimental platform

The experiment we describe here has been carried out on a Koala mobile robot: see Fig. 1.

This robot is equipped with two blocks of three lateral wheels. As they are independent, the robot can turn on the spot. We command the robot by its rotation and translation speed (respectively *Rot* and *Trans*) of the robot. The Koala robot is equipped with 16 infrared sensors. Each sensor can measure the ambient luminosity or the proximity to the nearest obstacle. In fact, the latter is an information on the distance to the obstacle within 25 cm which depends a lot of the nature (orientation, color, material, etc.) of the obstacle and the ambient luminosity. In our experiment, we only used the proximity measure.

The experiment took place in a large hall, on a smooth floor. The environment of the robot consisted mainly of white boards: see Fig. 6 for a typical example.

B. Probabilistic model details

In Section II, we have given the general outline of our probabilistic model. Here it is instantiated to match the features of the Koala robot.

The perception variable is a set of 16 variables Px_0, \dots, Px_{15} corresponding to the proximity sensors of the robot. Each variable has an integer value between 0 and 1023. The closer the obstacle is to the sensor, the higher the value.

The action variable represents the different rotation speeds the robot can apply. Indeed, in this experiment we have chosen to set the translation variable to a fixed value, so that the only degree of freedom we control is the rotation speed. The action variable is therefore *Rot*, which can take three different values: -1 for turning to the left, +1 for turning to the right, and 0 for not turning.

The location variables at time t and $t + \Delta t$ have the same domain. In our chosen internal representation, we assume that the robot sees exactly one obstacle at a time. A location is defined by the proximity and the angle of the robot to this

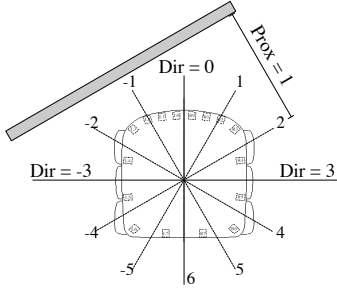


Fig. 2. The Dir and $Prox$ variables.

obstacle. Thus, the location variable at time t is a pair of variables: Dir_t , for the direction of the obstacle and $Prox_t$, for its proximity. The Dir_t variable has 12 different integer values, from -5 to 6. Fig. 2 shows the value repartition for this variable. $Prox_t$ has 3 different values: $Prox_t = 0$ (respectively 1, 2) when the robot center is roughly 44 cm (respectively 35, 24) from the wall.

The variables being now instantiated, Eq. (1) becomes:

$$\begin{aligned}
 &P(Dir_t Prox_t Rot Px_0 \dots Px_{15} Dir_{t+\Delta t} Prox_{t+\Delta t}) \\
 &= P(Dir_t Prox_t)P(Rot) \prod_i P(Px_i | Dir_t Prox_t) \\
 &P(Dir_{t+\Delta t} Prox_{t+\Delta t} | Dir_t Prox_t Rot).
 \end{aligned}$$

The 16 $P(Px_i | Dir_t Prox_t)$ terms, which define the sensor model, are Gaussian distributions defined by 16 μ_1^i, σ_1^i mean and variance functions. In our experiment, these were learned in a supervised manner, by putting the robot at a given position with respect to an obstacle and recording 50 sensor data.

The learned sensor model is used to compute, at each time step, the current location in the $Dir_t, Prox_t$ internal space, given the sensory input. This is done by computing

$$\begin{aligned}
 &P(Dir_t Prox_t | [Px_0 = px_0] \dots [Px_{15} = px_{15}]) \\
 &\propto \prod_i P([Px_i = px_i] | Dir_t Prox_t),
 \end{aligned}$$

and by drawing a value at random for Dir_t and $Prox_t$ over this distribution. The obtained sensor model is precise enough so that localization can be done directly, i.e. without using a recursive localization estimate, as would be done usually in a Markov localization framework.

C. Initial behavior and learning of the prediction term

The behavior known initially by the robot was a “random” behavior. This behavior consists in drawing a value according to a uniform distribution for Rot and maintaining the corresponding motor command for one second, before drawing a new value. During the second when the motor command is maintained, the robot has time to observe how this motor command makes Dir and $Prox$ vary, by recording their values every 100 ms.

To learn the *prediction term*, this initial behavior was applied with a zero translation speed. Thus, the learning



Fig. 3. The robot applies the initial behavior for the learning ($Prox = 0$).

required three phases, one for each possible values of $Prox$. Fig. 3 shows the learning phase for $Prox = 0$, i.e. with the robot rotating roughly 44 cm from the wall¹. Each learning phase was 5 minutes long.

During these 5 minutes, every 100 ms, the values of Dir_t , $Prox_t$ and Rot were recorded. In this data history, it is possible to know, for a given start location $\langle dir_t, prox_t \rangle$ and applied action rot , the location of the robot at the next time step. Indeed, $\langle dir_t, prox_t, rot \rangle$ are a 3-tuple in the data, and the next location $\langle dir_{t+\Delta t}, prox_{t+\Delta t} \rangle$ is given in the first following 3-tuple (with $\Delta t = 100$ ms) in the data history. Thus, a set of L_t , A and $L_{t+\Delta t}$ can be obtained.

Moreover, in this procedure for reconstructing the data, Δt needs not be 100 ms: it is possible to obtain sets of L_t , A and $L_{t+\Delta t}$ for $\Delta t = 200$ ms, $\Delta t = 300$ ms, etc. with the same data history. For instance, the second 3-tuple following the one giving L_t and A gives $L_{t+\Delta t}$ with $\Delta t = 200$ ms. The only condition to use these two 3-tuples to build the set is to make sure that the same motor command was applied during the time span separating their recording. As these data were recorded with the motor commands varying every second, $\Delta t = 1$ s at the most.

Once a Δt is chosen, the different values of $L_{t+\Delta t}$ found for a given set of L_t and A are used to compute the μ_2 and σ_2 functions defining the $P(L_{t+\Delta t} | L_t A)$ probability distribution. As $L_{t+\Delta t}$ is actually a pair $\langle Dir_{t+\Delta t}, Prox_{t+\Delta t} \rangle$, the prediction term is a set of 2D Gaussian distributions, one for each possible value of $\langle dir_t, prox_t, rot \rangle$.

IV. RESULTS

The results presented here have all be obtained with $\Delta t = 900$ ms during the learning phase. Fig. 4 illustrates some of the probability distributions obtained for the prediction term after learning. For instance, the leftmost probability distribution of Fig. 4 encodes the knowledge that, starting from $Dir_t = 0$, $Prox_t = 1$, when applying a rotation speed of $Rot = -1$, the most probable position at $t + \Delta t$ is $Dir_{t+\Delta t} = 1$, $Prox_{t+\Delta t} = 1$. In other words, if the robot had the wall in front of it, at a medium distance, and it turned to the left, then the wall would probably be on the right front side of the robot.

Having thus learned the prediction term, we have used our model with different questions. Every 100 ms, we draw a value according to $P(Rot | Dir_t Prox_t Dir_{t+\Delta t} Prox_{t+\Delta t})$ and send it to the motors. Using different values for $Dir_{t+\Delta t}$ and $Prox_{t+\Delta t}$ allow us to generate different behaviors.

¹These pictures, and the ones in subsequent figures, are taken from movies available at <http://julien.diard.free.fr/research.html>.

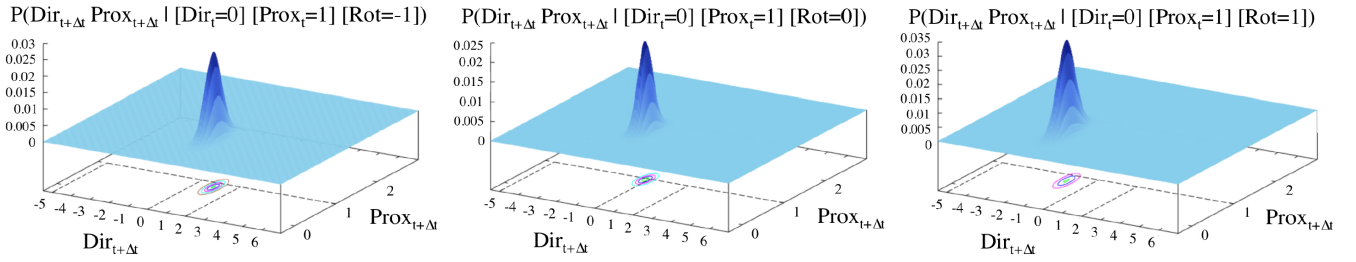


Fig. 4. Examples of probability distributions learned for the prediction term $P(Dir_{t+\Delta t}, Prox_{t+\Delta t} | Dir_t, Prox_t, Rot)$, with $\Delta t = 900$ ms. On the left (resp. middle, right), the prediction term for the starting location $Dir_t = 0$, $Prox_t = 1$ and motor command $Rot = -1$ (resp. $Rot = 0$, $Rot = 1$). For instance, the left plot shows that if the robot had the wall in front of it, at a medium distance, and it turned to the left ($Rot = -1$), then the wall would probably be on the right front side of the robot at the next time step.

A. Behaviors obtained for a zero translation speed

First, we generated behaviors with a zero translation speed. The robot was put near a wall and we set $Dir_{t+\Delta t}$ to a given value as a goal to reach, $Prox_{t+\Delta t}$ being set to the same value as $Prox_t$. In other words, the robot was asked to turn on the spot in order to reach a certain angular position with respect to the wall.

Fig. 4 and 5 illustrate how the probability distribution over the rotation speed is computed, given an initial position d_t, p_t and a position to reach $d_{t+\Delta t}, p_{t+\Delta t}$, in order to generate the behavior. This corresponds to the instantiation of (2):

$$P \left(Rot \mid \begin{array}{l} [Dir_t = d_t] [Prox_t = p_t] \\ [Dir_{t+\Delta t} = d_{t+\Delta t}] [Prox_{t+\Delta t} = p_{t+\Delta t}] \end{array} \right) \propto P \left(\begin{array}{l} [Dir_{t+\Delta t} = d_{t+\Delta t}] \\ [Prox_{t+\Delta t} = p_{t+\Delta t}] \end{array} \mid \begin{array}{l} [Dir_t = d_t] \\ [Prox_t = p_t] \\ Rot \end{array} \right).$$

The lines under the plots in Fig. 4 illustrate how the comparison between different Gaussian distributions in the prediction term is made to compute the probability distribution over Rot . For instance, the left “cut point” corresponds to the question where the goal is $Dir_{t+\Delta t} = -5$ and $Prox_{t+\Delta t} = 1$. The three probability distributions with different Rot values are evaluated at these “cut points”. Fig. 5 shows the results after normalization: for instance, the leftmost plot shows the computed distribution over Rot given that $Dir_t = 0$, $Prox_t = 1$, $Dir_{t+\Delta t} = -5$ and $Prox_{t+\Delta t} = 1$. This distribution gives a very high probability for the motor command $Rot = 1$. In other words, in order to move from a position where the wall is on the front to a position where the wall is on the rear-left side, the robot should turn on the right with a very high probability.

One should note that, whatever the distance between the current location and the goal, our model does not require any planning phase to compute this distribution. For instance $Dir_t = 0$ and $Dir_{t+\Delta t} = -5$ describe a starting and goal orientation that are very different. Nevertheless, the possible commands to reach the goal can be compared thanks to the way Gaussian distributions behave. Indeed, their numerical values, far from their means, still help identify the action that brings closer to the goal (compare the leftmost and rightmost plots of Figure 4). However, we do not need an

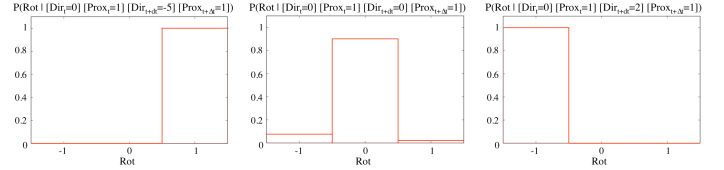


Fig. 5. Examples of probability distributions computed for generating behaviors: $P(Rot | Dir_t, Prox_t, Dir_{t+\Delta t}, Prox_{t+\Delta t})$. These situations correspond to the same starting position ($Dir_t = 0$, $Prox_t = 1$) and three different goals: on the left, $Dir_{t+\Delta t} = -5$ et $Prox_{t+\Delta t} = 1$, the middle plot is $Dir_{t+\Delta t} = 0$ et $Prox_{t+\Delta t} = 1$, and on the right, $Dir_{t+\Delta t} = 2$ et $Prox_{t+\Delta t} = 1$. For instance, the left plot shows that in order to move from a position where the wall is on the front to a position where the wall is on the rear-left side, the robot should turn on the right with a very high probability.

explicit planning phase also because the sensorimotor space in the experiment is small and concerns a simple phenomenon. It is the reason why we believe that a hierarchy of such simple models representing a complex environment is easier to handle than a complex unique model (see Section V-B).

To assess the quality of the obtained behavior, we asked the robot to reach randomly generated values of $Dir_{t+\Delta t}$. A new value for $Dir_{t+\Delta t}$ was given as a command every 10 seconds. Roughly 90% of the time, the robot successfully reached the goal and stayed in position until a new command was issued. The few cases when it did not reach the goal were due to a representation issue of the Gaussian distributions along the Dir dimension, this variable being an angle.

B. Behaviors for a non-zero translation speed

The following behaviors have been realized with a translation speed set to 20, which corresponds to a medium forward speed (roughly 6 cm/s).

Firstly, we asked the robot to apply an “obstacle avoidance” behavior. We achieved this by setting two goals, depending on the current value of Dir_t . The first one made the robot avoid the obstacles by the right, and was applied when $Dir_t \leq 0$, i.e. when the obstacle was in front or on the left of the robot. In this case, the robot turns on its right, bringing the obstacle on its back left side ($Dir = -5$). The second goal made the robot avoid the obstacles by the left and was applied when $Dir_t \geq 1$: the robot turns on its left to bring the obstacle to

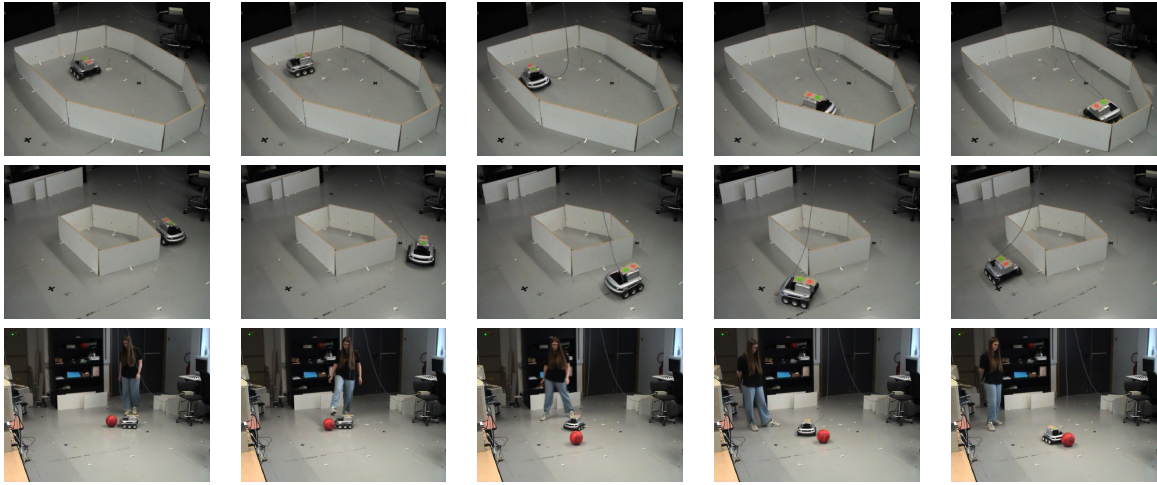


Fig. 6. Illustration of two new behaviors obtained, from top to bottom: “right wall following” (inside the arena and outside the arena) and “ball pushing”.

its back ($Dir = 6$). These goals correspond to the following questions:

$$P\left(Rot \left| \begin{array}{l} [Dir_t = d] [Prox_t = p] \\ [Dir_{t+\Delta t} = -5] [Prox_{t+\Delta t} = 0] \end{array} \right. \right), \text{ if } -5 \leq d \leq 0,$$

$$P\left(Rot \left| \begin{array}{l} [Dir_t = d] [Prox_t = p] \\ [Dir_{t+\Delta t} = 6] [Prox_{t+\Delta t} = 0] \end{array} \right. \right), \text{ if } 1 \leq d \leq 6.$$

Secondly, we asked the robot to apply “wall following” behaviors, either on the right or on the left. To obtain such behaviors, we set three different goals, depending on the value of $Prox_t$. Indeed, we wanted the robot to follow the wall without going closer to or further from it. As it made its learning with a zero translation speed, it could not learn any sensorimotor relationship for reaching values of $Prox$ different from its current value by using rotations. However, when the robot is passing corners, it does go a little closer to or a little further from the wall. To help it go back to a medium distance, we asked three different questions of the form (assuming a right wall following behavior):

$$P\left(Rot \left| \begin{array}{l} [Dir_t = d] [Prox_t = 0] \\ [Dir_{t+\Delta t} = 2] [Prox_{t+\Delta t} = 1] \end{array} \right. \right), \text{ if } Prox_t = 0,$$

$$P\left(Rot \left| \begin{array}{l} [Dir_t = d] [Prox_t = 1] \\ [Dir_{t+\Delta t} = 3] [Prox_{t+\Delta t} = 1] \end{array} \right. \right), \text{ if } Prox_t = 1,$$

$$P\left(Rot \left| \begin{array}{l} [Dir_t = d] [Prox_t = 2] \\ [Dir_{t+\Delta t} = 4] [Prox_{t+\Delta t} = 1] \end{array} \right. \right), \text{ if } Prox_t = 2.$$

We obtained “wall following” behaviors that can be applied inside or outside the arena (see Fig. 6).

Finally, we checked that, even if the robot learned the model in front of a board standing as a wall, it was possible to generalize the new behaviors to other types of environment. This is possible thanks to the fact that our model is a sensorimotor relationship rather than a precise geometric description of the environment. We therefore used a ball, and created a new “ball pushing” behavior by asking the following question:

$$P\left(Rot \left| \begin{array}{l} [Dir_t = d] [Prox_t = p] \\ [Dir_{t+\Delta t} = 0] [Prox_{t+\Delta t} = 2] \end{array} \right. \right).$$

With this question, the robot goes closer to the ball when it is in the robot proximity. Once the robot touches the ball, it pushes it and, if someone puts the ball on the side, the robot turns so as to be once again in front of it. Fig. 6 illustrates this new behavior.

It is also possible to apply the previous behaviors in this new environment. We obtained two new behaviors: “ball avoidance” as a derivative of “obstacle avoidance” and “ball orbiting” as a derivative of “wall following”.

To assess the quality of the obtained behaviors, we applied two criteria. The first one was a “recognition” criterion: the different behaviors needed to be identifiable by a human observer. The other criterion concerned the stability of the obtained behaviors: for instance, for the “obstacle avoidance” and “wall following” behaviors, we checked experimentally that the behavior could be applied for a long time without colliding with obstacles.

V. DISCUSSION

A. Open issues

1) Δt selection: The results presented Section IV have been obtained with $\Delta t = 900$ ms for the learning of the prediction term. With smaller values, the results are not as successful. Indeed, Δt is the size of the observation window for the variations of Dir , $Prox$. The smaller the Δt , the smaller the time for Dir to vary. This results in observations where the value of $Dir_{t+\Delta t}$ is the same as the value of Dir_t . Too many such data mean that the robot learns that the application of a non-zero rotation speed most of the time does not affect the value of Dir . As a result, it can not compute the correct action to perform to reach the goal, because it has learned that all the actions will probably not make it move in the Dir , $Prox$ internal space.

We have therefore studied, in the experimental data, the amount of data where $Dir_{t+\Delta t} = Dir_t$ and $Prox_{t+\Delta t} = Prox_t$, as a function of Δt (see Fig. 7). One can see that for $\Delta t = 100$ ms, roughly 70% of the observations are of this

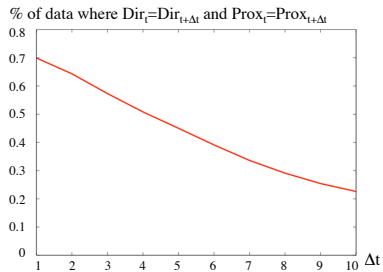


Fig. 7. Percentage of the number of data where $Dir_{t+\Delta t} = Dir_t$ and $Prox_{t+\Delta t} = Prox_t$ as a function of Δt .

type. In our experiment, we have selected empirically the Δt that gave the best results in terms of behavior quality. We are currently studying methods for comparing models with different Δt in a Bayesian manner, thus enabling inference about this parameter, in order to compute the most relevant Δt for a given situation.

2) *Choice of the initial behavior*: During our experiment, we tested several initial behaviors. We presented the one that gave the best results, i.e. that lead to a better learning of the prediction term. Indeed, as our probabilistic model is a model of the entire sensorimotor space, it can not be learned correctly if the initial behavior does not explore this space completely. For instance, we also used an “avoiding obstacle” behavior, which covered only 18,8% of the sensorimotor space for a 5-minute learning phase. Consequently, the choice of the initial behavior is not a trivial one and *is dependent of the sensorimotor space* the robot has to explore. Given a sensorimotor space, designing, in a principled manner, an initial behavior that explores it adequately is an open issue. However, in small sensorimotor spaces, as in our experiment, the random exploration strategy is a good candidate.

B. Putting together several Bayesian Maps

We have previously noted that the simplicity of our model made it tractable. In particular, no planning phase was explicitly required. But this simplicity, in our view, is not a limit, as we believe that it could be a basis for building hierarchies of models. Indeed, as previously mentioned, our long term goal is to use the Bayesian Map formalism and its Operators (Abstraction, Superposition) for putting together such models of the environment. These models are combined using behaviors as resources, which justifies our focus on behavior generation in this study.

So, a part of our current work aims at learning another probabilistic model based on the one we learned, so as to combine them into a richer model. More precisely, the new behaviors allow the robot to navigate in the environment. It can thus gather new experimental data related to another sensory modality. In a supplementary experiment, we will use the “obstacle avoidance” behavior we learned, so as to identify another internal representation modeling the influence of the Koala’s movements on its light sensors. Once the

two models are obtained, we will combine them using the Superposition operator, and study what new behaviors can be obtained.

Also, we would like to point out that the relevance of this incremental learning scenario in a biological development context is an interesting issue.

VI. CONCLUSION

In this article, we have described a robotic experiment where a probabilistic model was learned based on data gathered during the application of an initial behavior. This probabilistic model was an internal representation of the sensorimotor space of the robot, and could be used to generate new behaviors. We have described the implementation of this experiment on a Koala robot, and outlined good results, thanks to the variety of behaviors that were successfully generated out of the learned model.

REFERENCES

- [1] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proc. of the 13th Nat. Conf. on Artificial Intelligence and the 8th Innovative Applications of Artificial Intelligence Conf.*, pages 896–901, Menlo Park, August, 4–8 1996. AAAI Press / MIT Press.
- [2] J. Leonard, H. Durrant-Whyte, and I. Cox. Dynamic map-building for an autonomous mobile robot. *The International Journal of Robotics Research*, 11(4):286–298, 1992.
- [3] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.
- [4] H. Shatkey and L.P. Kaelbling. Learning topological maps with weak local odometric information. In *Proc. of the 15th Int. Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 920–929, San Francisco, August, 23–29 1997. Morgan Kaufmann Publishers.
- [5] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.
- [6] N. Tomatis, I. Nourbakhsh, and R. Siegwart. Hybrid simultaneous localization and map building: a natural integration of topological and metric. *Robotics and Autonomous Systems*, 44:3–14, 2003.
- [7] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *Proc. of the Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1080–1087, 1995.
- [8] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.
- [9] J. Diard, P. Bessière, and E. Mazer. Combining probabilistic models of space for mobile robots: the bayesian map and the superposition operator. In M. Armada and P. González de Santos, editors, *Proc. of the 3rd IARP Int. Workshop on Service, Assistive and Personal Robots - Technical Challenges and Real World Application Perspectives*, pages 65–72, Madrid, Spain, 2003.
- [10] J. Diard, P. Bessière, and E. Mazer. Hierarchies of probabilistic models of navigation: the bayesian map and the abstraction operator. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA04)*, pages 3837–3842, New Orleans, LA, USA, 2004.
- [11] J. Diard, P. Bessière, and E. Mazer. A theoretical comparison of probabilistic and biomimetic models of mobile robot navigation. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA04)*, pages 933–938, New Orleans, LA, USA, 2004.
- [12] Olivier Trullier, Sidney I. Wiener, Alain Berthoz, and Jean-Arcady Meyer. Biologically-based artificial navigation systems: review and prospects. *Progress in Neurobiology*, October 1996.
- [13] M. O. Franz and H. A. Mallot. Biomimetic robot navigation. *Robotics and Autonomous Systems*, pages 133–153, 2000.
- [14] S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21(4):93–109, 2000.