A planning map for mobile robots: speed control and paths finding in a changing environment

Mathias Quoy[1], Philippe Gaussier[1], Sacha Leprêtre[1], Arnaud Revel[1], and Jean-Paul Banquet[2]

[1] Neurocybernetics team
ETIS - Université de Cergy-Pontoise - ENSEA
6, Av du Ponceau, 95014 Cergy Pontoise Cedex, FRANCE
quoy@u-cergy.fr,
http://www-etis.ensea.fr
[2] Institut Neurosciences et Modélisation, Paris VI

Abstract. We present here a neural model for mobile robot action selection and trajectories planning. It is based on the elaboration of a "cognitive map". This cognitive map builds up a graph linking together reachable places. We first demonstrate that this map may be used for the control of the robot speed assuring a convergence to the goal. We show afterwards that this model enables to select between different goals in a static environment and finally in a changing environment.

1 Introduction

One of our research interests is to propose architectures for controlling mobile robots. These architectures are mainly constrained by the facts that we want the representations to be "grounded" on the sensor data (*constructivist approach*) and that we want an analogical computation at each stage. These requirements particularly fit in the neural network (N.N.) framework. Furthermore, the N.N. approach enables to take inspiration from neurobiological and psychological results, as well for the architectures as for the learning processes (see [Baloch and Waxman, 1991], [Millan and Torras, 1992], [Verschure and Pfeifer, 1992], [Gaussier and Zrehen, 1994] for other works on this issue). We also stress that our robot has its own internal "motivations" and that the whole learning process is under their influence. This is very important because our system has to "behave" autonomously in an a priori unknown environment (*animat approach* [Meyer and Wilson, 1991]). So we will call our robot an "animat". Our control architecture is mainly composed of two parts. The first one enables the animat to locate itself in the environment [Gaussier et al., 1997a]. We will not discuss about this part in this paper. The second one builds a graph linking successively reached places ("cognitive map" [Tolman, 1948]). It is important to note that no external a priori information about the external world is given to the system. Learning new locations and their links with each other is the core process of the system. In addition, there is also a reflex mechanism for obstacle following. This mechanism takes control

of the animat when an obstacle is encountered. The created map is typical for a given environment. It allows the animat to smartly solve various action selection problems in a static world. But the map may be updated so that the same system may work if the environment has changed (door closed or moving person for instance). The other dynamics followed by the system is given by the animat movements in the environment. These movements are either random when the animat is exploring, or given by the planning map when the animat has to reach a goal. Under the neural field formalism (*dynamic approach* to autonomous robotics [Schöner et al., 1995]), we deduce from the planning map a control variable for the animat speed (see [Goetz and Walters, 1997] for a review of control techniques). This enables a smooth approach of the goal.
In the following, we first describe the experimental environment and the planning architecture. Then we show how the planning map may be interpreted in terms of a dynamic approach for speed control. Finally, we present the results about finding a path in a static and then in a changing world.

2 Experiment

The experiments reported in this paper are performed in a simulated environment. However, some of them are already running on a real robot. The animat only sees obstacles and landmarks. It may not see the resource locations. The environment may be as large as 40 animat sizes. Three different energy levels decrease exponentially over time unless re-supplied by a proper source [Gaussier et al., 1998]. Since we are in an animat approach, these levels will be called "food", "water" and "sleep". They are linked with the motivations "hunger", "thirst" and "rest". A more robotic way of putting it in words would be to consider for instance a "charging station", and two different "delivery points", provided the robot gets rewards being at these locations. So, through random exploration, the animat has to discover "food" and "water" places where it may re-supply one of the two levels. From time to time, the animat has to go back to its nest for rest (it knows where it is at start). The animat has to go back to a previously discovered interesting place when it needs to. The animat must therefore solve the dilemma between exploring and reaching a known food and/or water place and/or rest. When on a source location, the animat source level increases by a given amount. When the source is empty, another one appears randomly in the environment. Hence, the animat always has to explore the environment in order to find new potential sources. In addition, we impose that the levels of the essential variables are maintained between two thresholds that define the comfort area. Hence, as soon as the level of one of these variables is below the lower threshold, the animat goes toward a resource place. Conversely, when the level of a particular essential variable is higher than a top threshold, the animat goes away from the resource place. The animat succeeds quite correctly in maintaining the 3 essential variables in the comfort area. A cycle of go and return takes place between the different types of sources. These cycles change only when the animat succeeds, during a random exploration phase, in

finding a new resource place, or when a location is depleted. The animat can fail only when new sources appear in areas difficult to explore so that there is a low probability for a single animat to find the solution. In this case, there is a need to have a cooperation between several animats [Gaussier et al., 1997b].

3 Model

We will not fully develop here our navigation model (see [Gaussier et al., 1998], [Gaussier and Zrehen, 1995], [Gaussier et al., 2000] for a complete presentation). What should be reminded is that a location is defined by a set of landmarks and azimuth of each landmark. As the animat explores the environment, it learns new locations when they are different enough from the previous ones. A neuron is coding for each learned position in the environment. The closer the animat to the neuron coding for a position, the higher the response of that neuron. We don't provide any external explicit description of the environment (world model), nor learn what to do for each location in the environment [Burgess et al., 1994]. In order to be able to go back to a learned location, the animat has to keep track of where it was and where it may go from any location. So we need to introduce a kind of map of the environment to be able to perform this task. To do so, we add a group of neurons able to learn the relationships between successively explored places. The temporal proximity being equivalent to a spatial proximity, the system creates a topological representation of its environment. We will call this last group our "cognitive map" (or goal map)(fig. 2 and 1) [Tolman, 1948], [Thinus-Blanc, 1996], [Revel et al., 1998].

Let W_{ij} be the weight associated with the fact that from the place i it is possible to reach directly place j. At the goal level (fig. 1), the motivations activate directly the goal neurons linked with the goal to be achieved (the goal neuron intensity is proportional to the motivation). This activity is then diffused from neuron to neuron through the synaptic weights. Other works also use a diffusion mechanism for planning paths [Connolly et al., 1990], [Bugmann et al., 1995], [Bugmann, 1997]. Our goal diffusion algorithm is the following :

- *i_0 is the goal neuron activated because of a particular motivation. G_{i_0} is its activity.*
- $G_{i_0} \leftarrow 1$ and, $G_i \leftarrow 0\ \forall i \neq i_0$
- *WHILE the network activity is not stabilized, DO:* $\forall j, G_j \leftarrow \max_i(W_{ij}.G_i)$

where G_i is the value of neuron i (see [Gaussier et al., 1998] for more details on the navigation algorithm). W_{ij} has to be strictly inferior to 1 ($0 < W_{ij} < 1$) to ensure the algorithm convergence. On figure 1 an example of activity diffusion from motivation "A" is presented. The algorithm allows to find the shortest path in the graph. The algorithm is proved to always find the shortest path in

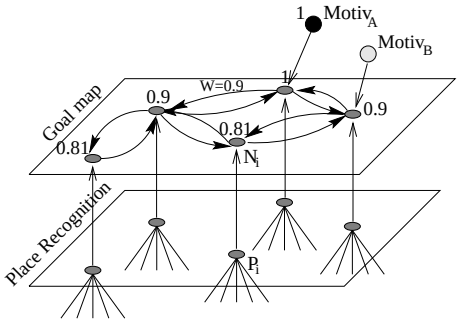

Fig. 1. Global architecture of the planning system. The recognition level allows to identify the situations when the animat arrives in the vicinity of a stored place. These situations are directly linked to the goal level which allows to plan a route from one attractor to the next until the objective. When a motivation "A" activates a goal, a propagation of the information is performed in the direction of all nodes in the graph (here all weights are equal to 0.9).

the graph (it is equivalent to the Bellman-Ford algorithm of graph exploration [Bellman, 1958], [Revel, 1997]).

The animat tries to follow the gradient of neuron activity in this cognitive map to select the next location to reach. The most activated goal or subgoal in the neighborhood of the current animat location is then selected and used to attract the animat in its vicinity. When the animat is close enough to this location, the associated subgoal is inhibited and the animat is attracted by the next subgoal and so on until it reaches the final goal. The principle of this kind of cognitive maps is not new [Arbib and Lieblich, 1977], [Schmajuk and Thieme, 1992], [Bachelder and Waxman, 1994], [Schölkopf and Mallot, 1994], [Bugmann et al., 1995], [Trullier et al., 1997], [Franz et al., 1998]. The novelty in this paper is that our algorithm allows to solve planning problems involving several moving goals in a dynamic environment (the sources may disappear and appear again elsewhere, obstacles may also be moved, landmarks may be hidden). The map is learned and modified on-line and allows to manage contradictory goals. The subgoals correspond to the following situations: the end of an obstacle avoidance (for instance, the animat stores the pathway between two rooms: location of the door) or places badly recognized (activity of all neurons below a defined recognition threshold. The higher this threshold is, the more learned places there are: denser coverage of the open space). We obtain a spatial "paving" of the environment which is almost regular (fig. 2).

Learning of the cognitive map is performed continuously. There is no separation between the learning and the test phases. The links between neurons of the graph are reinforced (hebbian associative learning) for neurons associated with successively recognized places. The learning rule is the following:

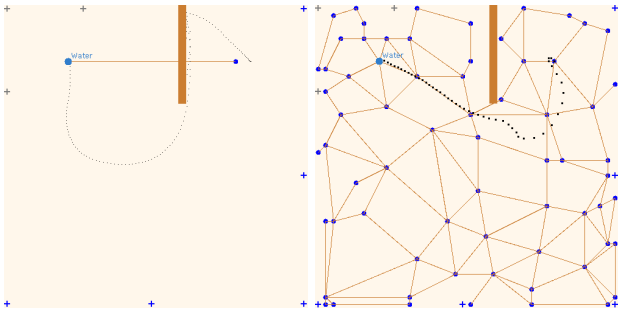

Fig. 2. On the left, reflex behavior of obstacle following for reaching the goal (Initial speed towards the upper left. The animat has a mass giving a momentum (see section 4)). The dots indicate the trajectory followed by the animat. On the right, "cognitive map" built by exploration of the same environment. The landmarks are the crosses on the border. Each circle is a subgoal. The links indicate that the two subgoals have been activated in succession. The subgoals and the learned transitions form the goal or cognitive map.

$$\frac{dW_{i,j}}{dt} = -\lambda.W_{i,j} + (C + \frac{dR}{dt}).(1 - W_{i,j}).\overline{G_i}.G_j \quad (1)$$

G_i must be held to a non null value until G_j (with $i \neq j$) is activated by the recognition of the place cell j. This is performed by a time integration of the G_i values represented in the equation by $\overline{G_i}$. $\overline{G_i}$ decreases with time and can be used as a raw measure of the distance between i and j. λ is a very low positive value. It allows forgetting of unused links. $C = 1$ in our simulations. The term $\frac{dR}{dt}$ corresponds to the variation of an external reinforcement signal (negative or positive) that appears when the animat enters or leaves a "difficult" or "dangerous" area.

4 Use of the planning map for speed control

The planning map is constructed in order to indicate which path to take to reach the goal. In addition, the map may give us another information: how to control the speed of the animat in order to assure to go to the goal and stop on it. Hence the planning map not only indicates where the goal is, but also enables to control how to reach it. For this purpose, we have to consider that the animat has a mass M. At the same time, we also add friction F. To balance this effect, an internal drive is also present. When the animat is exploring, this drive is random in strength and direction. This enables to randomly explore large areas. When it is goal seeking, the animat is going from one subgoal to another until reaching the goal. This is performed by following the gradient of the motivations on each subgoal. So when the animat is going towards the nearest goal (or sub-goal), the direction of the drive is given by the direction of the goal, and the strength may be either independent from the distance to the final goal

(hence constant), or depend on this distance (hence depend on the gradient). The equations driving the animat movement are now (we took the same form as in [Schmajuk and Blair, 1992]):

$$\begin{aligned} v_x(t+1) &= v_x(t) - \frac{dt.F.v_x(t)^2}{M} + \frac{dt.drive}{M} \\ x(t+1) &= x(t) + dt' v_x(t) \end{aligned} \tag{2}$$

where the x subscript denotes a projection on the x axis. The same equations hold for the y axis. v is the speed and (x, y) the position of the animat. $F = 5$, $M = 5$ and $drive$ a force (either random or towards the goal). The values of these constants have been chosen in order to exhibit nice "gliding" trajectories illustrating the effect of the external drive in this equation. dt and dt' are small integration constants. In order to stay at the same position (when a goal is reached for instance), the speed must be 0. Hence $drive = 0$ at the goal location. So when the drive is constant, the speed will tend to $\sqrt{\frac{drive}{F}}$ and the animat can not stop on the goal. If the drive depends on the distance to the goal, then one has to make $drive$ tend to 0 as the goal is nearer. In the case where the drive is not constant, it is the difference between the value of the nearest subgoal and the value of the second nearest subgoal which gives the strength of the drive. This value is computed during the diffusion of the motivation from the goal to the subgoals. Using equation 3, this value is $G_n = W^n$, where W is the (fixed) weight between the subgoals and n the distance (expressed in number of subgoals) to the goal. Figure 3 shows the strength of the subgoal G_n versus the distance to the goal (dotted line).

As one can see, as the animat comes close to the goal, the gradient increases. Hence as the animat approaches the goal, the drive towards it increases. This leads to a non null speed when arriving on the goal and an unstable reaching behavior (dotted line in fig. 4). In order to avoid this unstable behavior (or to stop on the goal), the speed must decrease as the animat comes closer to the goal. So the activation value used to compute the gradient is now:

$$H_n = exp - \frac{log(\frac{G_n}{W})^2}{2.\sigma^2} = exp - \frac{log(G_n) - log(W)}{\sigma^2} \tag{3}$$

where σ is chosen to be 5.

The speed near the goal is now reduced, enabling a smooth approach (fig. 3). This is particularly suitable if the animat has to stop on the goal.

Comparing the two drive politics, it appears that the first one enables to reach the goal sooner, with a high speed when arriving on the goal. The goal is reached faster because its attraction strength is high, so that the animat does not "glide" as much as in the second case. However, there may be also cases where the goal is not reached at all (see below). In the second case, the animat may take a longer time to reach the goal, but arrives there with a very low speed. However, due to the smaller strength of the attraction it may sometimes "circle around" the goal until it is reached (see fig. 5).

These results are backed by the theoretical framework of dynamical systems. In particular, the way the map is constructed fits very nicely in the "neural

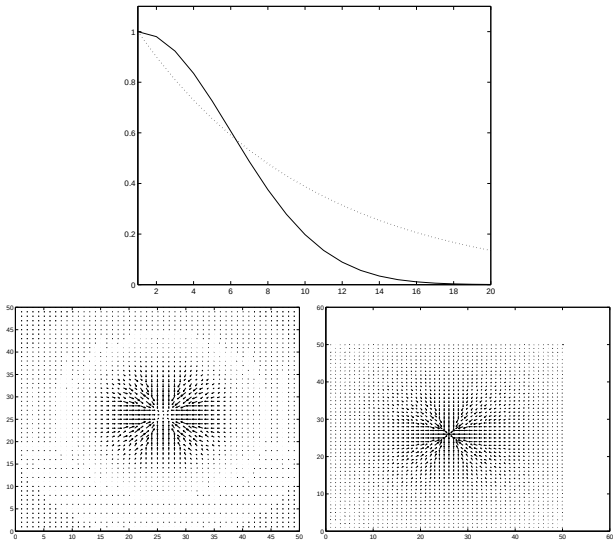

Fig. 3. H_n versus n (solid line), and G_n (dotted line). In the first case, the gradient decreases as the goal is being reached. Whereas in the second case, the gradient increases, leading to a high speed near the goal. The two figures below show left the value of the gradient in the environment when the goal is in the middle of the room. The arrows indicate the direction of the gradient and its strength (left gradient of H_n and right gradient of G_n).

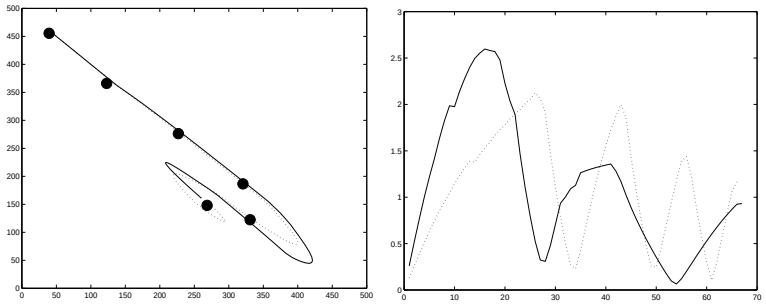

Fig. 4. Left: Trajectory of an animat driven by the gradient computed using H_n (solid line) and G_n (dotted line). The dots are the learned subgoals. The planning map links are not shown. Right: The same conventions are used for displaying the speed versus time in the two same cases. Speed is higher in the second case than in the first one.

field" framework [Amari, 1977], [Schöner et al., 1995]. Indeed, we can extend the notion of distance expressed in number of subgoals by making it continuous ("behavioral dimension" [Schöner et al., 1995]). So, one can now consider that the goal has an activation field extending over each subgoal. The value of the field at subgoal n is H_n (fig. 3). The same way it is possible to control the heading direction of an animat [Bicho and Schöner, 1997] by controlling its angular

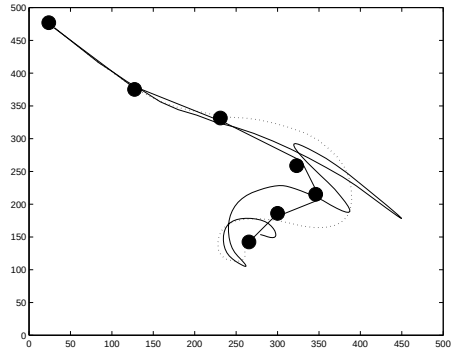

Fig. 5. Trajectory of an animat driven by the gradient computed using H_n (solid line) and G_n (dotted line). The dots are the learned subgoals. The planning map links are shown. The animat reaches the goal sooner in the second case, but with a high speed.

velocity, we control the position of the animat through modifications of its speed introduced by the gradient of H_n. In this case the goal position is a stable fixed point of the dynamics $\dot{n} = F(n)$ (fig. 6). When $n = 0$, the animat is on the goal (and its speed is 0 either). Moreover, as it is a stable fixed point, we are sure that the animat will converge on it (in asymptotical time theoretically). This is not the case when using a gradient computed with G_n. The animat may not converge on the goal at all because the goal is not a fixed point anymore (fig. 6).

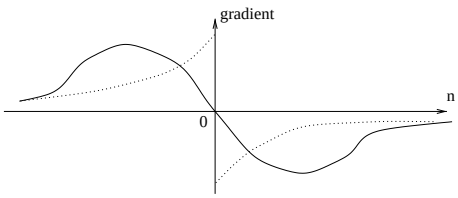

Fig. 6. Gradient of H_n (solid line) versus n and gradient of G_n (dotted line) versus n. In the first case, $n = 0$ is a stable fixed point for the dynamics $\dot{n} = F(n) = \frac{dH_n}{dn}$. In the second case, this not the case.

5 Use of the planning map for finding paths

We will now show in the next subsections that the animat is able to learn to choose between goals and that the planning map may be used in a changing environment.

5.1 Learning paths in a fixed environment

To allow a measurement of the performances, the environment is chosen in the following experiments as a T-maze. However, this maze is not discretized into squares (other than for statistical results, see below), but allows a continuous computation of the animat positions. The width of a corridor is 7 animat sizes. The need to go to the nest increases twice as fast as the need for food or water. Hence the animat has to go back very often to its nest. In order to suppress the possible biases introduced by an autonomous map building, the cognitive map is learned during a teleoperated exploration of the maze. The operator driving the animat follows the middle of the corridors. The maze and the cognitive map are displayed as the first figure of each experiment shown thereafter. In order to record the preferred paths taken by the animat, we have divided the maze into squares. Each square has the size of an animat. However note that these squares are **not** used in the computation of the movement. They are only used in this statistical analysis. The different figures show histograms of the occupation of each square in the maze. Histograms are computed from the average of 50 different runs for each experiment. One run corresponds to 20000 iterates of the animat behavior in the same T-maze. The animat needs less than 2000 iterates to construct the complete planning map.

Because the activation level of a particular subgoal is the maximum of the back-propagated motivational information, we believed at first that our algorithm was unable to choose correctly between satisfying one motivation or several simultaneous motivations. For instance, we thought it would be unable to always choose the left arm of the T maze fig. 11 that allows to satisfy at the same time 2 motivations (something that the "free flow" architecture of Tyrrell [Tyrrell, 1993] succeeds in doing at the price of local minima problems and a priori simplification in the graph). Of course, it is possible to decide that if a goal is associated with 2 motivations those two motivations are summed before being diffused with our max rule. Unfortunately, if the two interesting sources are not exactly at the same place but are in neighbor areas, this trick cannot be used. If we change the max rule into a simple addition rule we can face deadlock situations because the diffused activity can amplify itself in loop situations or when a node receives the diffusion of a lot of other nodes. So we have tried our algorithm as it is and we have verified it was nevertheless able to find the good solution after a while because of on line weight adaptation.

Figure 7 shows a symmetrical maze. What happens in this case is that the left or the right trajectory is randomly reinforced. Hence the animat goes either to the right arm or to the left arm to drink. But if during a random exploration, it goes in the other arm, it reinforces this path, and therefore enables its use again. Hence, the animat uses only one path until it goes in the other arm and then may use only the other path, and so on.

Figure 8 and 9 show what happens when one of the arms is extended. As expected, the only water source used after a while is the one in the right arm. The exploration of the left arm does not modify enough the weights between the subgoals so as to make the animat choose this water source for drinking.

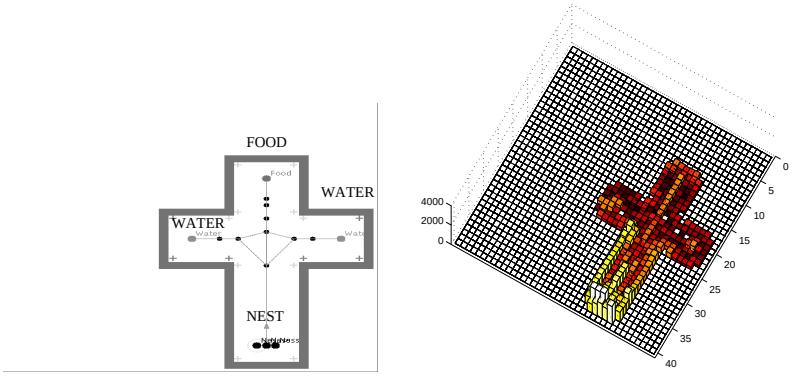

Fig. 7. The figure on the left displays the initial maze, with the nest on the bottom, water sources in the left and the right arm and a food source at the top. In the figure on the right, the height of each square shows the number of time it has been occupied by the animat (dark (resp. light) color indicates low (resp. high) occupation. The number of iterations is 20000. The values have been computed adding 50 different runs. Here, as the maze is symmetrical, no arm is preferred for going to drink.

Instead, as the animat goes more often near the water source in the right arm, these links are reinforced.

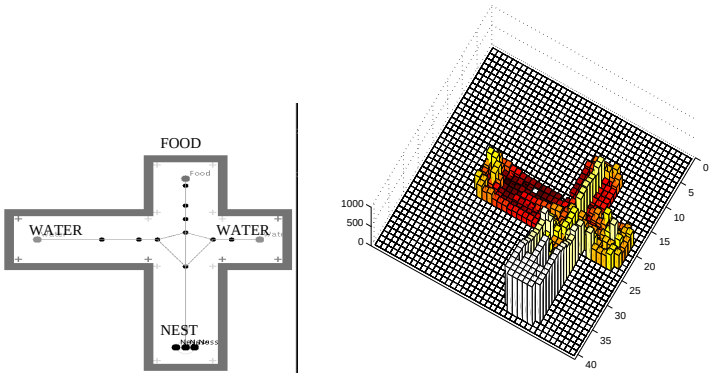

Fig. 8. Same setup as in the previous experiment. The left arm is extended, so that the left water source is farther away from the nest and from the right water source than in Figure 7. Hence, statistically, the animat prefers to go drinking in the right water source.

In the last two experiments (fig. 10 and 11), the arm where the food source is, is shifted to the left. The preferred water source is the one near the food source. When the animat goes eating, it may afterwards explore the maze. Since it is in the left arm, the animat is more likely to go in the left end. Hence the links between the food source and the left water source are reinforced. Only from time

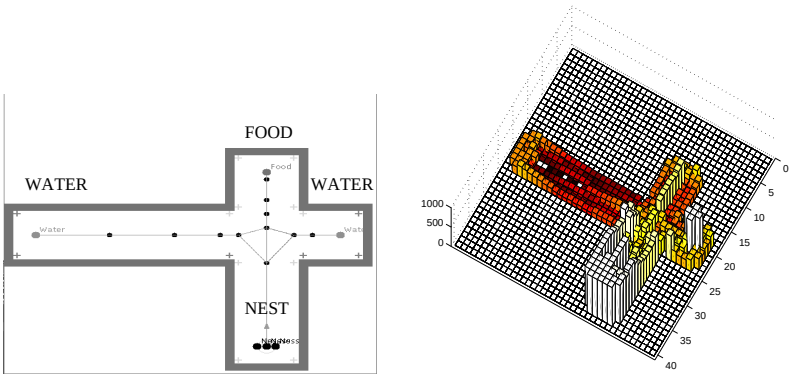

Fig. 9. Same setup as in the previous experiment. The difference with figure 8 is the farther location of the water source in the left arm in this case. As this source is moved away to the left, it gets less used by the animat for drinking.

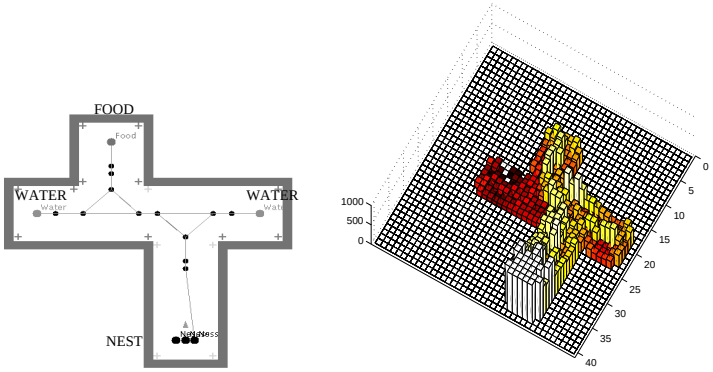

Fig. 10. Same setup as in the previous experiment. The animat visits the right source more often than the left one.

to time when the animat has enough time to go to the right end are the links to the right water source reinforced.

The difference between the experiments shown in figures 8, 9, 10 and 11 is the shifting to the left of the upper arm containing the food. This move enables the animat to use the left arm water source more often. And in the extreme (last) case, it is now the left water source which is a lot more often used than the right one. This shows the importance of the reinforcement of used links (and conversely the decrease of the unused links). This property allows the animat to "behave smartly" by not wasting its energy going to far away sources.

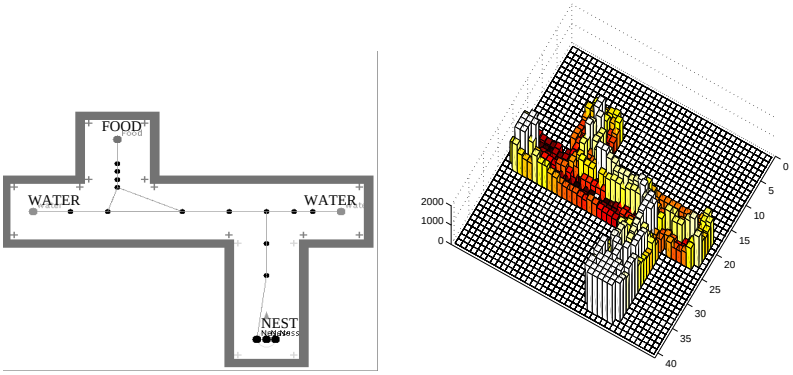

Fig. 11. Same setup as in the previous experiment. The water source on the left, near the food source, is more often used than the water source in the right arm.

5.2 Learning new paths in a changing environment: preliminary results

A real environment does not always stay the same. New obstacles may appear or disappear (somebody walking or a door opened or closed, for instance). The change in the environment may first affect the recognition of a place. But this is not the case in our system. Indeed, it stands up the loss (or the addition) of landmarks provided at least half of the landmarks used for learning a location are still visible. However, a changing environment may dramatically impair the relevance of a learned cognitive map: some paths may not be used anymore (door closed), some others may be used (door opened). If an obstacle suddenly appears, the reflex mechanism allowing to follow the obstacle enables to nevertheless reach the goal (see figure 2). On the contrary, new paths are found by random exploration of the environment when it is not goal seeking. We have tested our algorithm **as it is** in a changing environment (see figure 12). In this environment, the door opening enabling a direct path may be closed. The animat has learned a cognitive map with the door open. When there are two ways to go from one source to the other one, the direct path is preferred, but the other one is also used (left figure). In the second experiment, the direct path is closed by a door. The animat now goes through the other path.

What makes it possible to change the environment is that the map is constantly updated either by creation of new subgoals (or goals) or by increase (resp. decrease) of the value of used (resp. unused) links. We have shown in the previous section how the reinforcement of some links may lead to the emergence of preferred paths. Now, it is the decrease of the link values which will help "forget" unusable paths. For instance, if the animat has learned a path between two walls and this aperture is suddenly closed (by a door for instance), the cognitive map is not valid anymore. What happens now is a decrease to 0 of the link going "through the wall". The principle for the degeneration of a link is that it is not reinforced anymore because its two ends are not activated in succession

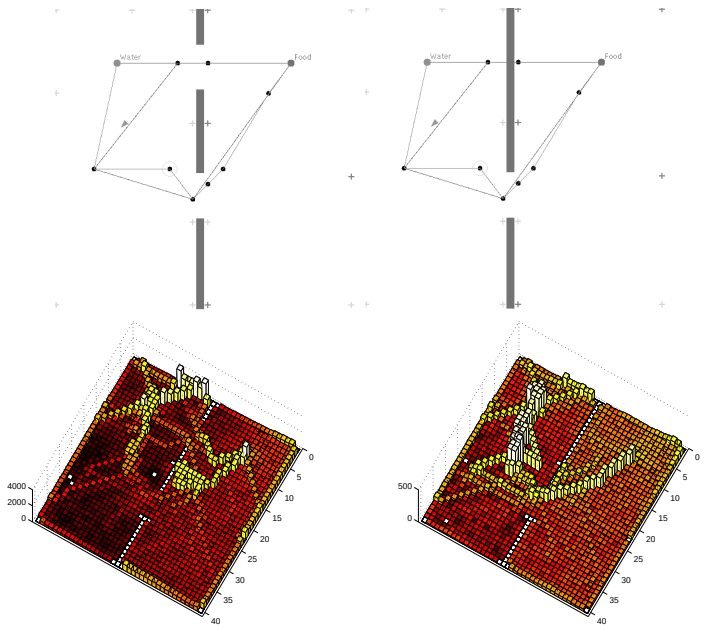

Fig. 12. Same setup as in the previous experiment. In the first experiment (left 2 figures), the animat may go from one source to the other through either door. In the second experiment, the top door is closed. The animat is able to change its behavior and goes through the last open door.

anymore. So, the link decreases to 0 due to the passive decay term in equation 1. The 3 typical cases one may encounter when a path may not be used anymore are summarized in figure 13. In the first two cases, one of the ends of the link leading through the wall may not be activated at all. Hence this link degenerates. The only problematic case is the last one presented. Indeed, there, the place cell in the wall may fire either when the animat is on the right or from the left side. Hence the left link, as well as the right link may be reinforced. The solution is the creation of a new subgoal where the animat hits the door. The situation will then be the same as case b). The creation of this new subgoal (in planning mode specifically, in exploration mode there is no need to create a new subgoal when hitting an obstacle) has not been implemented in the system for the moment.

It is worthwhile to note that after the closure of the door the environment has changed: the landmarks of the other room are not visible anymore. Hence the situation which has led to the creation of a subgoal near the door opening is not valid anymore either. In some cases, these subgoals may not be activated at all (even if the animat is exactly on them) because other subgoals, which correspond to the new environment, have been learned. Indeed, changing the environment also changes the set of locations for which a place cell responses. New place cells are learned corresponding to the new landmark configuration. Their attraction basin extends over the previously created ones. Hence, in

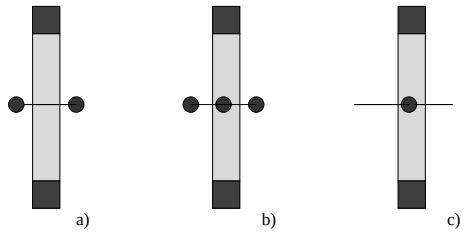

Fig. 13. A door (light hatching) has closed the pathway between two walls. There are 3 different cases depending on where the subgoals were learned when the door was open. In case a), there is no subgoal "in" the door. Since the 2 subgoals on either side of the door can not be activated successively, the link between them degenerates. In case b), for the same reason the links between the subgoals in the middle and the ones outside may not be reinforced. In case c), however the links may be reinforced.

the extreme case, a place cell learned in the old environment may not fire at all in the new one, and so no new links may be created with it. However, the old ones still exist (even if their value is close to 0), but may not be reinforced anymore. If the old environment is presented again, these links may be reactivated. The animat must first discover that the path is usable again, and use it for some time before the links values reach a high level again. Now, the same degeneration process happens for the links between place cells created in the second environment. Hence several different "layers" of cognitive maps can appear in the same physical N.N. structure. They may be linked together through some place cells valid in more than one environment, and may be activated when the environment they are coding is presented again. The animat has to try some time before building a new efficient planning map. So, even if it may be a good solution to use a very low passive decay (λ parameter) to store several different maps (memory effect), it also slowers down the process of finding new pathways, when one may not be used anymore. Indeed, the hebbian learning rule we have chosen needs some time before significantly changing the weights. Hence, in order to react faster to a change in the environment, it would be necessary to introduce an active decay mechanism decreasing unused links.

6 Discussion

The navigation and planning system we have presented is able to solve complex action selection problems. However, for the moment, it has to really perform the movements in order to reinforce particular paths. An improvement would be the possibility to internally replay the trajectory used to reach a goal.

The main drawback of the algorithm is the computation of the gradient of G_n. Indeed, in very large environments (with a great number of subgoals) the gradient may be very small. Hence, first, the drive on the speed would be very small too (leading to a long time before reaching the goal). Second, if there is

a small noise on the gradient, it would be now impossible to follow it. So there is a need to combine several maps of different joint environments. This means we have to define a planning structure, or plans of plans, in order to address large scale environments. This need is also highlighted by the use of the same neural structure for storing all maps linked with different environments. There will soon be an explosion of the number of neurons needed to code all locations, and moreover a mix up between maps coding for totally different environments. So there is a need to "transfer" the map in another structure where it may be memorized.

Concerning the interpretation of the planning map in terms of activation field, it is important to stress that this process is a top down one. Indeed, the map has first been build using the place recognition mechanism. Now, in return the use of the map gives information for the low level control of the animat. Moreover, this activation field approach could be generalized using the neural field paradigm [Amari, 1977]. Each goal can have a previous defined activation. The activations may be combined using the neural field equations. The advantage is that the animat will stay focused on a goal until either another goal becomes stronger or an external events occurs. This is also very interesting for us because it is easy to introduce in our neural architecture. Following a moving object is already coded that way [Gaussier et al., 1999].

This work was supported by two French GIS contracts on Cognitive Sciences entitled "comparison of control architectures for the problem of action selection" in collaboration with the Animat lab (J.Y. Donnart, A. Guillot, J.A. Meyer), LISC (G. Deffuant) and RFIA (F. Alexandre, H. Frezza), and "Mobile robots as simulation tools for the dynamics of neurobiological models: adaptation and learning for visual navigation tasks" in collaboration with the CRNC (G. Schöner) and the LAAS (R. Chatila).

References

[Amari, 1977] Amari, S. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87.

[Arbib and Lieblich, 1977] Arbib, M. and Lieblich, I. (1977). Motivational learning of spatial behavior. In Metzler, J., editor, *Systems Neuroscience*, pages 221–239. Academic Press.

[Bachelder and Waxman, 1994] Bachelder, I. and Waxman, A. (1994). Mobile robot visual mapping and localization: A view-based neurocomputational architecture that emulates hippocampal place learning. *Neural Networks*, 7(6/7):1083–1099.

[Baloch and Waxman, 1991] Baloch, A. and Waxman, A. (1991). Visual learning, adaptive expectations and behavioral conditionning of the mobile robot mavin. *Neural Networks*, 4:271–302.

[Bellman, 1958] Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics*, 16:87–90.

[Bicho and Schöner, 1997] Bicho, E. and Schöner, G. (1997). The dynamics approach to autonomous robotics demonstrated on a low-level vehicle platform. *Robotics and Autonomous System*, 21:23–35.

[Bugmann, 1997] Bugmann, G. (1997). *Basic Concepts in Neural Networks: A survey*, chapter Chap 5: A Connectionist Approach to Spatial Memory and Planning. Perspectives in Neural Networks. Springer.

[Bugmann et al., 1995] Bugmann, G., Taylor, J., and Denham, M. (1995). Route finding by neural nets. In Taylor, J., editor, *Neural Networks*, pages 217–230, Henley-on-Thames. Alfred Waller Ltd.

[Burgess et al., 1994] Burgess, N., Recce, M., and O'Keefe, J. (1994). A model of hippocampal function. *Neural Networks*, 7(6/7):1065–1081.

[Connolly et al., 1990] Connolly, C., Burns, J., and Weiss, R. (1990). Path planning using laplace's equation. In *International Conference on Robotics and Automation*, pages 2102–2106.

[Franz et al., 1998] Franz, M., Schölkopf, B., Mallot, H., and Bülthoff, H. (1998). Learning view graphs for robot navigation. *Autonomous Robots*, 5:111–125.

[Gaussier et al., 2000] Gaussier, P., Joulain, C., Banquet, J., Leprêtre, S., and Revel, A. (2000). The visual homing problem: an example of robotics/biology cross fertilization. *Robotics and Autonomous Systems*, 30:155–180.

[Gaussier et al., 1997a] Gaussier, P., Joulain, C., Zrehen, S., Banquet, J., and Revel, A. (1997a). Visual navigation in an open environment without map. In *International Conference on Intelligent Robots and Systems - IROS'97*, pages 545–550, Grenoble, France. IEEE/RSJ.

[Gaussier et al., 1998] Gaussier, P., Leprêtre, S., Joulain, C., Revel, A., Quoy, M., and Banquet, J. (1998). Animal and robot learning: experiments and models about visual navigation. In *Seventh European Workshop on Learning Robots- EWLR'98*, Edinburgh, UK.

[Gaussier et al., 1997b] Gaussier, P., Moga, S., Banquet, J., and Quoy, M. (1997b). From perception-action loops to imitation processes: A bottom-up approach of learning by imitation. In *Socially Intelligent Agents, AAAI fall symposium*, pages 49–54, Boston.

[Gaussier et al., 1999] Gaussier, P., Moga, S., Banquet, J., and Quoy, M. (1999). From perception-action loops to imitation processes. *Applied Artificial Intelligence*, 1(7).

[Gaussier and Zrehen, 1994] Gaussier, P. and Zrehen, S. (1994). A topological map for on-line learning : Emergence of obstacle avoidance in a mobile robot. In *From Animals to Animats: SAB'94*, pages 282–290, Brighton. MIT Press.

[Gaussier and Zrehen, 1995] Gaussier, P. and Zrehen, S. (1995). Perac: A neural architecture to control artificial animals. *Robotics and Autonomous Systems*, 16(2-4):291–320.

[Goetz and Walters, 1997] Goetz, P. and Walters, D. (1997). The dynamics of recurrent behavior networks. *Adaptive Behavior*, 6(2):247–283.

[Meyer and Wilson, 1991] Meyer, J. and Wilson, S. (1991). From animals to animats. In Press, M., editor, *First International Conference on Simulation of Adaptive Behavior*. Bardford Books.

[Millan and Torras, 1992] Millan, J. R. and Torras, C. (1992). A reinforcement connectionist approache to robot path finding in non-maze-like environments. *Machine Learning*, 8:363–395.

[Revel, 1997] Revel, A. (1997). *Contrôle d'un robot mobile autonome par une approche neuromimétique*. PhD thesis, U. de Cergy-Pontoise.

[Revel et al., 1998] Revel, A., Gaussier, P., Leprêtre, S., and Banquet, J. (1998). Planification versus sensory-motor conditioning: what are the issues ? In Pfeifer, R., Blumberg, B., Meyer, J., and Wilson, S., editors, *From Animals to Animats : Simulation of Adaptive Behavior SAB'98*, pages 129–138. MIT Press.

[Schmajuk and Blair, 1992] Schmajuk, N. and Blair, H. (1992). Place learning and the dynamics of spatial navigation: a neural network approach. *Adaptive Behavior*, 1:353–385.

[Schmajuk and Thieme, 1992] Schmajuk, N. and Thieme, A. (1992). Purposive behavior and cognitive mapping: a neural network model. *Biological Cybernetics*, 67:165–174.

[Schölkopf and Mallot, 1994] Schölkopf, B. and Mallot, H. (1994). View-based cognitive mapping and path-finding. Arbeitsgruppe Bülthoff 7, Max-Planck-Institut für biologische kybernetik.

[Schöner et al., 1995] Schöner, G., Dose, M., and Engels, C. (1995). Dynamics of behavior: theory and applications for autonomous robot architectures. *Robotics and Autonomous System*, 16(2-4):213–245.

[Thinus-Blanc, 1996] Thinus-Blanc, C. (1996). *Animal Spatial Navigation*. World Scientific.

[Tolman, 1948] Tolman, E. (1948). Cognitive maps in rats and men. *The Psychological Review*, 55(4).

[Trullier et al., 1997] Trullier, O., Wiener, S., Berthoz, A., and Meyer, J. (1997). Biologically based artificial navigation systems: review and prospects. *Progress in Neurobiology*, 51:483–544.

[Tyrrell, 1993] Tyrrell, T. (1993). *Computational Mechanisms for Action Selection*. PhD thesis, University of Edinburgh.

[Verschure and Pfeifer, 1992] Verschure, P. and Pfeifer, R. (1992). Categorization, representation, and the dynamics of system-environment interaction. In *From Animals to Animats: SAB'92*, pages 210–217.