The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Allostatic Control for Robot Behaviour Regulation: an extension to path planning

Marti Sanchez Fibla [‡], Ulysses Bernardet [‡] and Paul F.M.J. Verschure [‡,§]
{marti.sanchez,ulysses.bernardet,paul.verschure}@upf.edu

Abstract— **Rodents are optimal real-world foragers that can smoothly regulate behaviors like homing and exploration combined with more elaborate abilities as food source localization. Here we investigate a robot based model that implements the self-regulatory processes that underly optimal foraging of rodents in unknown environments and is also able to combine it with goal directed behaviors. Behavior is decomposed into minimal homeostatic subsystems that regulate themselves through the local perception/detection of a gradient. On a higher level, the allostatic control orchestrates the interaction of the different homeostatic modules allowing it to dynamically manage the interactions between the desired values of each subsystem to achieve stability on a meta behavioral level. In this case, we show that overall behavioral stability can be achieved. We validate our model by comparing the behavior of both simulated and real robots with that of rodents. Our next step is then to justify gradients as a valid biological assumption by giving a biologically plausible process for generating them from a cognitive map, in this case, a set of approximated hippocampal place cells. We finally formulate path planning (used for goal reaching, e.g. food source localization) in the same context of a gradient map generation that can be then inserted as an additional subsystem of the higher meta level allostatic control.**

I. INTRODUCTION

Behavior is motivated by the needs of the agent defined by internal variables such as hunger, thirst, temperature, security, etc. Indeed a number of psychological theories of learning and behavior, ranging from psychoanalysis to cognitive behaviorism of the 1930's, propose that drive reduction underlies experience and behavior. A classic example of such a perspective is hierarchy of needs proposed by Maslow [1]. In general this perspective proposes that behavior results from processes that generate behavior in order to maintain a number of drive states, or essential variables, that support the integrity of the organism within certain specific predefined limits. A typical example would be the regulation of fight or flight behaviors with respect to the distance to a threatening stimulus [2]. The concept of self-regulation goes back to the study of sleep regulation of Walter Cannon and has been further explored in the notions of feedback, explored in cognitive behaviorism and cybernetics [3]. Indeed more recently it has been argued that any process trying to approximate animal behavior should be derived from this ability of self-regulation and self-generated internal motivation (e.g. [4]). Here, we are interested in the complexity of rodent behavior originating from self-regulation and how to emulate it with a computational neuronal model. For this purpose, behavior is then decomposed into minimal homeostatic subsystems that regulate themselves through a local perception of a gradient map. Homeostasis is a useful concept to apply to individual drive systems but it does not generalize to the control of behavior in case multiple drive systems need to interact. We argue that the process of achieving stability between an organism and its environment must involve an additional higher layer of regulation that controls the different objectives of each homeostatic subprocess and is able to integrate and weight its outputs.

We propose that each of these systems is not static but that the parameters that control its dynamics such as gain and threshold can be dynamically adjusted consistent with the physiological notion of *allostasis* [5]. We augment these individual control loops with a regulatory system that adjusts the individual control systems dependent on the dominant drive state of the animal. In order to design such a modular allostatic control system (in which other subsystems can be added or suppressed without affecting the fundamental properties of achieving stability), each subsystem operates on a gradient (or vector field). Thus we envision a nested structure where homeostasis acts in a closed loop that regulates the actual value of each subsystem to bring it to stability, that is, close to its desired value in the gradient. At the next level, allostatic control changes the set points and gains of individual homeostatic loops, while the integrative loop orchestrates the interaction of the different allostatic modules and allowing to dynamically manage the interactions between the desired values of each subsystem to achieve stability at a meta level that ultimately defines the relationship between the organism and its environment. Thus where homeostasis achieves stability through constancy inside a closed loop and allostasis achieves stability through change [5], we propose that in order to achieve dynamic stability in the coordination of drive based behavior a third level of integration among allostatic sub-systems must be considered.

The idea of using gradient maps or vector fields as a way of combining forces to drive an agent through an unknown environment is not new, neither from the pure robotics side ([6], [7], [8], [9], [10]), nor from the neurobotic perspective where usually some kind of reinforcement learning is employed to associate some preferred directions to a place cell system ([11], [12], [13], [14], [15], [16]).

[‡] SPECS, Technology Department, Universitat Pompeu Fabra, Carrer de Roc Boronat 138, E-08018 Barcelona, Spain.
[§] ICREA, Institució Catalana de Recerca i Estudis Avançats, Passeig Lluís Companys 23, E-08010 Barcelona
This work was supported by Synthetic Forager FP7-ICT-217148.

We make three new contributions. The first is the design of a meta-control represented by the allostatic regulation able to dynamically change the desired values of each subsystem (section II-B) and dynamically change the gradients themselves (section II) and we prove that stability at this meta-level can be achieved. The second one is that we present a biological and plausible way to generate these gradients by a neural optimization process that could give insights in how place cells are used in the hippocampus to compute spatial properties of the environment. In this sense it is really innovative as place cells (approximated here by Gaussians) take an active role and are used to compute properties of the environment (section III). The third contribution is that we formulate path planning in terms of gradient learning. Subsequent activations of place cells of a cognitive map are used to generate a gradient driving to a goal. We describe a procedure that exploits the fact that certain place cells play a special role in path planning (like the ones representing corners or bifurcations) and is able to actively use them to speed up path planning (section IV).

Each of these three contributions are tested either using a simulated or real robot with the general aim of approximating certain characteristics of rodent behavior: free exploration of different types of arenas (for this purpose we present a novel mixed reality robot arena in section V), goal directed behavior driven by visual cues and path planning. In [17] we focus on the comparison of the model (without the path planning component) to several different rodent experimental setups.

II. The Basis of Allostatic Control

Animals are driven by internal variables such as hunger, temperature, security, etc. which have to be maintained within certain limits in order to be stable and predictive over changing environments. The first issue that we tackle is to decompose self-regulation into a minimal set of homeostatic subsystems (such as arousal, security, energy, etc.) that can be plugged together orchestrated by what we call allostatic control. In figure 1 we show the schematic view of a minimal allostatic control including two subsystems, each one including a gradient (or vector field), the actual value aV of the agent in that gradient and a desired value dV in that gradient. The regulator works as a gradient ascent/descent given a local view of the gradient around the agent.

The main contributions of the allostatic control as a meta-level self-regulation are depicted in figure 1 as the connections labeled (a), (b) and (c). Connection (a) is the responsible of manipulating the gradients of each sub-system. In the present work, the main manipulation of the gradients is for learning them, for example, to represent the open space map of an environment (see section III). Connection (b) controls the desired values of each subsystem and is able to change the valence of each gradient from appetitive to aversive, making the regulator to ascend or descend the corresponding gradient (see section II-B). Connection (c) controls the integration of the gradients that in our case is a weighted sum (see section II-A).

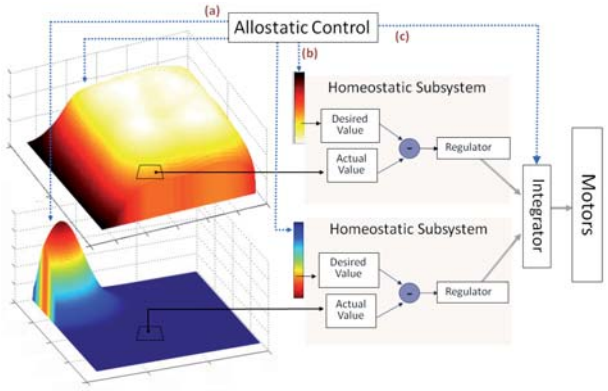

Fig. 1. A minimal allostatic control integrating two homeostatic subsystems. These subsystems can be related to a security subsystem (on bottom) with a preferred location on the top left corner and an arousal subsystem with its maximum in the center and dropping to zero close to the walls.

A. Gradient integration

We explain here how the gradient integration is done and how this integration is mapped to the motor commands of the robot. Each gradient is sensed locally as a matrix of values between 0 and 1. In figure 2 we show a representation of this local view. q^0, q^1, q^2, q^3 are the mean values of the four quadrants of the local view.

The actual gradient value aV corresponds to the mean of the whole local view. $hSign$ controls for the sign of the difference between the upper horizontal quadrants, it controls for the direction of increase or decrease of the gradient. When the right upper quadrant is greater than the left one (above a threshold th), $hSign$ is set to 1, on the opposite case it is set to -1:

$$hSign_i = \begin{cases} 1 & \text{if} \quad q_i^0 < q_i^1 - th \\ -1 & \text{if} \quad q_i^0 > q_i^1 + th \\ 0 & \text{otherwise} \end{cases}$$

$adSign$ controls for the sign of the direction of increase or decrease between the actual and desired values aV and dV also in a similar way than $hSign$:

$$adSign_i = \begin{cases} 1 & \text{if} \quad aV_i < dV_i - th \\ -1 & \text{if} \quad aV_i > dV_i + th \\ 0 & \text{otherwise} \end{cases}$$

So the mapping is done by summing up the contribution for all the gradients to the left and right motors. In the horizontal case, for example, if $hSign$ is 1, means that the gradient increases from left to right. If $adSign$ is 1, means that dV is greater than aV, that is, there is an intention to increase the actual value. Concluding, if we want to increase the dV and this increase is from left to right we should then turn right (positive contribution in the left motor and negative in the right one). The formulas used for this integration follow:

$$left_h = c + (\sum_{i=1}^{numGrad} hSign_i * adSign_i * f_i) * \frac{1}{activeGrad}$$

$$right_h = c - (\sum_{i=1}^{numGrad} hSign_i * adSign_i * f_i) * \frac{1}{activeGrad}$$

c is a constant that assures the default action of going forward. f_i is the total force for every gradient i. It is computed integrating a weighting factor k_i, different for every gradient and ranging from 0 to 1 to weight the contribution of every gradient, the speed and the difference between the actual value and the desired one:

$$f_i = k_i * speed * |aV_i - dV_i|$$

The last term of the motors integration, $\frac{1}{activeGrad}$, is a normalization factor that takes into account the number of gradients that are active (that is, the counting of subsystems with $k_i > 0$). A randomized factor can also be added besides the constant c to the left and right contributions to assure some variability and tie breaking. Summarizing, the robot, having a default action of going forward (constant c) is influenced by each gradient to turn left or right depending on wether this turn brings the actual value closer to the desired one. The influence of this turn is weighted by a factor k_i that can be different for every gradient and also the difference between the actual and desired values. If the total contribution is then normalized by the sum of all k_i the controller makes the robot move forward at constant speed affecting only the turns.

Until now the motor integration only includes the, so called, horizontal contribution. The $left$ and $right$ motor integration can also include a vertical correction of each gradient (the difference between the upper quadrants q^0,q^1 and the lower ones q^2,q^3). In a similar way than $hSign$, a $vSign$ correction can be defined using quadrants q^0 and q^2, or the sum of $q^0 + q^1$ compared to $q^2 + q^3$. We are not going to detail the formulas because they look very similar than the previous case. Including the vertical correction $vSign$, we now can have an influence on the backward and forward speed of the robot. Consider $left_v$, $right_v$ to be the forces for stabilizing the gradient with respect to the vertical correction (like in previous case).

Now the robot varies its speed and stops at particular positions of equilibrium of the desired and actual values for a gradient. This is a way of adding variations of speed in the model that correlates with the speed maps of a rat in a square arena. If now a gradient can attract the robot up to the point of balancing the motors to stop it when the desired value dV is similar to the actual one aV it becomes necessary to include a mechanism to change the desired values over time.

B. Regulating desired values

Conventionally the desired value in a homeostatic system is seen as being constant. Yet, this does not need to be so; the desired value can change over time in a cyclic fashion

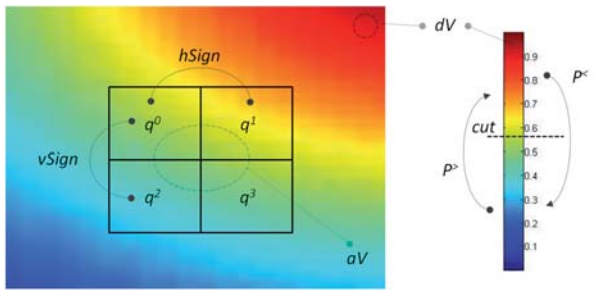

Fig. 2. Gradient Local Sensing. Left: Notation when referring to the local sensing of each gradient. Right: When regulating desired values two probabilities come into play: $p_i^<$ as the probability the dV will lower and $p_i^>$ as the probability the dV will get higher.

(circadian rhythm [18]), subject to habituation if the difference between actual and desired value is too large over an extended period of time, or influenced by other homeostatic systems. The dynamic regulation of the desired values is one of the key elements of allostasis. Stability in allostasis lies one level above homeostasis and is considered to be achieved through change, dynamicity and not constancy. For regulating the desired values we put in place the simplest probabilistic automata that defines a probabilistic switching mechanism. Two probabilities are involved for each gradient for regulating the changes of its associated desired value. Let's assume the simplest case where both probabilities are 1, for a particular gradient. This setting generates an intermittent behavior that switches the behavior once the homeostatic objective is achieved. If one of the probabilities is lower than 1 the robot can accept to stay a bit longer in a state of equilibrium before changing its desired value. Each of these probabilities is tested throwing dice at each time step whenever the actual value and the desired value are close enough. Two probabilities are involved for each gradient for regulating the changes of the desired values. When the aV is close to the dV (that is, $|aV_i - dV_i| < th$) if the desired value is low there is a probability $p_i^>$ of changing it to a higher value. If the desired value is higher then there is a probability $p_i^<$ of changing it to a lower value. Figure 2 illustrates this mechanism. The switching happens between a cutting point, cut_i, that divides the aV scale in two states. This cutting point can be estimated from the histograms of aV_i temporal series of a rat (see figure 3) as we will suggest in the following case study. We are seeing the problem of achieving stability as a multi-objective optimization problem in the landscape of the dynamic weighted (signed by $vSign$ and $adSign$) sum of the different involved gradients.

C. Case study: free exploration

As a specific case study of this minimal system, we pose the following question: is it possible, with this minimal allostatic control, to generate a similar behavior than the one elicited by rodents in free exploration of a squared arena? We choose these subsystems to be related to the variables security and arousal (also taking inspiration from [19]) as

the behavior of a rat in a squared arena may be driven by the constant equilibrium of its feeling of security (the distance to the home base) and the need for exploration modulated by its arousal and motivation (exposure to open space) [20]. In figure 3 we show the trajectory of a rat when freely exploring a square arena. We observe that the rat moves around the walls establishing a preferred top left corner, also called home base. The rat does also many traversals of the center of the arena, indicative of its aroused state. It seems feasible to approximate this behavior with the previous described model (section II-A) using only two subsystems: security and arousal. We define their two gradients: the security gradient will have its maximum at the top left corner (home base) and the arousal gradient, representing the open space, with a flat maximum in the whole central area of the arena and dropping to zero towards the borders. The surfaces showed in the schema of figure 1 are representing these gradients (top is arousal, bottom is security). Once the gradients are defined (notice that in section III we describe a method for learning them) we can extract from the trajectory of the rat the temporal series of the actual values (shown in figure 3(b)). Two main series will be involved: aV_{aro} series (for arousal, plotted in red) and aV_{sec} (for security, plotted in green). From the rat aV series we extract several facts: the rat spends most of the time in a low aroused state (aV_{aro}'s have a mean of 0.18) and it seldom does traversals of the arena (red peaks of the aV_{aro} series). The green peaks of the security aV_{sec} series show regular trips to the home base with sometimes longer stays (around time step 6000 of the rat's aV_{sec} plot in figure 1, the rat spends a long time at the home base).

We will then fit the model using the horizontal and vertical correction ($left_h$, $right_h$ and $left_v$,$right_v$, see section II-A) thus being obliged to fix a policy for dynamically changing the desired values. Let $p^{>}_{aro}$ and $p^{<}_{aro}$ be the probabilities of changing the arousal desired values as described in section II-B and $p^{>}_{sec}$ and $p^{<}_{sec}$ the same probabilities for security.

We set $p^{<}_{aro} = 1$ because the probability of going to a lower desired arousal, once a high arousal is reached, is one (the traversals of the arena are instantaneous). We set $p^{>}_{aro} = 0.21$ in concordance with the number of occurred traversals in the rat aV_{aro} temporal series. Similarly we set $p^{>}_{sec} = 0.95$ and $p^{<}_{sec} = 0.5$.

Figure 3 compares the rat and robot behavior: observed trajectories (figure 3(a)), the aV time series (figure 3(b)) and region analysis of the occupancy maps of the rat and the robot (figure 3(c)). We observed experimentally that the robot behavior can be fitted to the one elicited by the rat quite accurately. The mean of the aV series can be made similar (arousal is also low in the robot, having a lower mean of $aV_{aro} = 0.15$). The region histograms are conceptually similar with a longer stay in region 2 (the home base) and a relative small stay in the center of the arena (region 3).

D. Case study: a cue following task

We now modify the allostatic control to perform a cue following task in a y-maze, also taking inspiration from a rat setup. The details of the original rat experiment together with a complete analysis of how to approximate the exact rat behavior in this task, are out of the scope of this paper. Three reward ports are placed in each of the arms of the y-maze with a blue light above each. When a blue cue is delivered if the rat reaches it on time, it is rewarded and a new cue lights up. Rats are able to learn the task and approach the ports very efficiently.

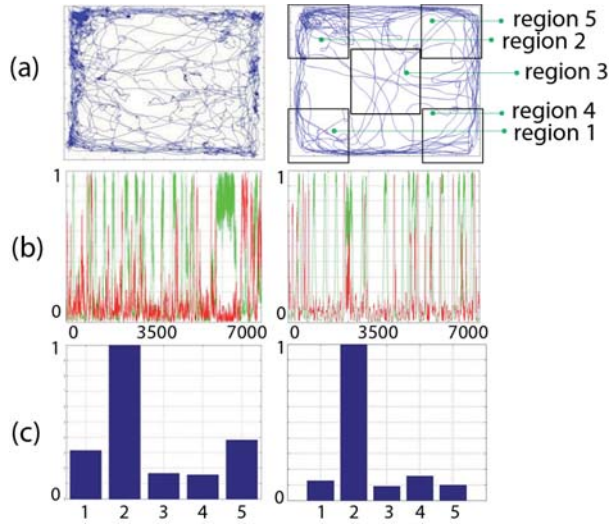

Fig. 3. Left Column: Rat 20 minutes session in a squared arena. (a) trajectory of the rat (b) arousal(red)/security(green) actual values over time. The arousal/security actual values are extracted from the gradients using the rat trajectory and they range from 0 to 1 (y axis). The x axis plots 7000 time steps. (c) histogram of the region occupancy. Region numbers are indicated in the robot trajectory plot. The region histograms are normalized to 1. Right Column: The same plots for a robot simulation.

We just prove that the allostatic control can be modified to perform the cue following task together with the needed navigation capabilities. We now involve also two subsystems: arousal and cue subsystems. The arousal gradient (as in previous case) has a maximum in the center and drops to zero towards the walls. The cue gradient, can now change dynamically, and is setup with a Gaussian centered on the port that has its light turned on.

We then fit the model using only the horizontal correction ($left_h$, $right_h$ of section II-A). In this case, the robot is forced to move forward and can only correct its left and right turn. We permanently set the desired values to $dV_{cue} = 1$, $dV_{aro} = 0.8$ and we set a lower strength influence for the arousal gradient: $k_{aro} = 0.4$ and $k_{cue} = 1$. The desired values are static, $p^{>} = 0$ and $p^{<} = 0$ for both subsystems. This simplified setting allows the robot to navigate towards the cue while also being able to use the arousal gradient to help following the middle of the alleys. To strongly prove that in such a simple control both gradients are needed, we apply the same control to a t-maze where the cues are delivered at the end of the arms, thus preventing the robot to follow a straight line when a cue lights up. In the snapshots of figure 4 we show the e-puck robot in the mixed reality robot setup

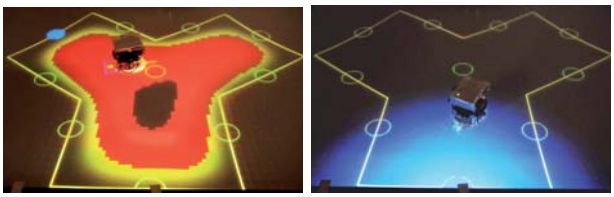

Fig. 4. Allostatic Control for an e-puck robot performing a cue following task in a y-maze. Left: e-puck navigating towards a blue cue. The open space gradient is being showed in the surface of the mixed reality robot arena. For being able to visually identify high values of the gradient, the color scale was set to black when the gradient was greater than 0.9: this fact explains the black spot in the middle of the image. Usually, gradients are shown in another color scale like the t-maze in figure 5. Right: the gradient of the blue cue is showed.

(see section V). The blue gradient corresponds to the cue and the yellow and red gradient to the open space. The task performed by an epuck robot is also showed in the online video [**?**] in both the y-maze and the t-maze.

In this section we simplified the allostatic control to only take into account the horizontal correction having also a static policy of desired values to perform a cue following task. Nevertheless, for fulfilling the task, we had to control for the integration weight of the considered gradients (k_{cue} and k_{aro}). These weights refer to connection labeled (c) in figure 1. In the previous section we demonstrated the allostatic control in a more general framework: both vertical and horizontal corrections where considered and thus obliging to setup a dynamic policy of desired values. The desired value policy of the allostatic control is captured by the connection labeled (b) in figure 1, which controls for changing the dVs. In the next section we deal with the last connection labeled (a) in figure 1 dealing with the dynamic manipulation of the gradients themselves.

III. Learning Gradients from a Cognitive Map

The different gradients that are used by each homeostatic subsystem (home base, open space, cue, reward, visited,...) can be learned or generated. We show in this section how this is done. A gradient will be generated by a sum of two dimensional Gaussians:

$$g_{\sigma,x_0,y_0}(x,y) = e^{-(\frac{(x-x_0)^2}{2\sigma^2}+\frac{(y-y_0)^2}{2\sigma^2})}$$

where (x_0, y_0) is the position and σ the standard deviation. The security subsystem gradient of figure 1 was generated adding a single Gaussian in the top left corner of a 120x90 matrix. All the entries of the matrix were initially set to zero. We address now the issue of how can we generate the open space gradient in a biologically plausible way. We present a simple algorithm to generate the open space gradient denoted by the $n * m$ matrix M_{aro}:

The algorithm 1 loops until convergence of the gradient adding Gaussians to the M_{aro} matrix. The loop in line labeled 1 assures that the generated Gaussian is a valid one, representing a valid possible Gaussian area inside the arena. In practice the function $isValidPlaceCell(g)$ can be easily implemented by testing that the center of the Gaussian is inside the maze by a standard inside polygonal test. An arrow in $\overrightarrow{M}$ indicates that the gradient is normalized, that is, divided by its maximum value. It is worth noting that instead of randomly generating Gaussians and testing its validity one could generate place cells from sensory inputs as in [21].

Algorithm 1: Generate open space gradient.

while $convergenceValue > 0.1$ **do**
1 **while** $not\ isValidPlaceCell(g)$ **do**
$g \leftarrow random(g_{\sigma,x_0,y_0})$
$M_{aro} \leftarrow M_{aro} + g$
2 $convergenceValue = \sum_{i,j} \mid \overrightarrow{M}_{ske}(i,j) - \overrightarrow{M}_{aro}(i,j) \mid$

A way of assessing convergence is comparing the resulting gradient with a skeletonization computed with the straightforward method of considering the minimum distance of every point to the closest wall. Let M_{ske} be a matrix representing the skeleton of the environment where each position (i, j) is set to the minimum distance to the closest wall. This last method gives as result an skeleton with abrupt changes, so the comparison is restricted where the value of the skeleton with this method is greater than a certain value. Thus the sum $\sum_{i,j}$ in previous algorithm is restricted to values where $M_{ske} > 0.8$ and $convergenceValue$ is finally normalized by all the elements of the sum. This computation is the one performed in line labeled 2 in the algorithm.

In figure 5 we show the plot of the evolution of this convergence value while Gaussians are added into M_{aro} for different values of σ. Not surprisingly, with bigger σ we converge much rapidly to the skeleton, but with less accuracy. Figure 5 shows that the algorithm converges to the skeleton but the real underlying theoretical proof of this is related with the central limit theorem which states that if one sums a sufficiently large number of Gaussians randomly distributed in a circular region of a two dimensional space the resulting surface will also have the shape of a Gaussian centered in the center of the circle. We consider that the learning process has converged when the $convergenceValue$ is below a predefined significant level.

A. Towards an hippocampal model

Are gradients relevant in biological terms? [22] discusses how pigeons that elicit homing behavior after being released far away from the home base, could be using and integrating gradient maps of different kinds to navigate: geomagnetic field, visual landscape, coriolis force, sun, atmospheric chemosignals, infrasounds, ... It could also be the case that gradients are computed from joint activity of place cells. We discuss in this section how close is the gradient learning approach to an hippocampal model. Place cells [23] have taken a main role in trying to understand how a cognitive map could be operating in the hippocampus. What we call here a cognitive map is a pool of place cells representing the environment. Place cells where first discovered in rats and are neurons that are recruited during the exploration of a new environment and acquire rapidly a directional independent

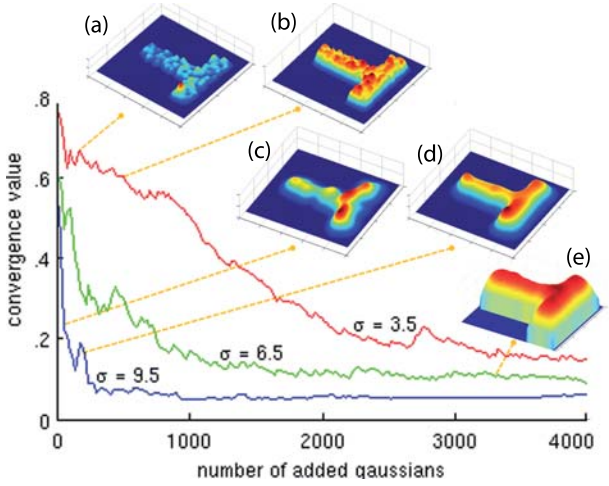

Fig. 5. Convergence of the open space gradient of a t-maze shaped environment. Different values of σ are used (the standard deviation of the added Gaussian). (a,b) show different stages of the gradient being learned for a t-maze $\sigma = 3.5$. (c,d) show different stages of the gradient for $\sigma = 9.5$. (e) corresponds to an advance stage of $\sigma = 6.5$.

activity that correlates with a particular spatial position. Once acquired, the correlation is observed for long periods of time. The global characteristics of this new recruited cell population are not known: how many cells encode the new environment?, is their overlapping important?, what are their scale distributions? We also don't know if place cells are used for something else than encoding the current position in a new environment. [24] show that place cell activity can also encode paths forward of the rat in decision points. Similar type of result is found in [25], [26], [27]. These evidences prove that place cells are actively used to compute at least highly localized possible forward paths. Whether place cell activity is used to compute other spatial computations is not known. Several neuronal based computational models exist that can acquire place cells from sensory input [21] and that make use of place cells for spatial navigation system [11], [12], [13], [14], [15], [16]. In these works, it is common to approximate place cells by Gaussians.

IV. PATH PLANNING

Our problem formulation is as follows: we have as input an arbitrarily large graph of cells (that can change over time) whose activity is correlated with different positions of an environment covering it all. The environment can dynamically change and this will be reflected in the cells. Neighboring cells are connected to each other if their activity overlaps in the spatial correlation. Cells can be arbitrarily small and have a maximum space coverage corresponding to the biggest area in the environment. The graph of place cells provides us a discretization in the wanted scale of the continuous space represented by the spatial characteristics of the environment. We then specify a starting cell and a destination one, the aim being to learn a path that connects both. A path between cells could be computed by reinforcing the synapses between the corresponding cells using a hebbian-like learning rule (like in [28]). We will compute such a path, learning a gradient between the covered regions of the two cells.

A big gap exists between the cited biological approaches (section III-A) and the pure robotic ones in what refers to path planning. Probabilistic approaches (like the sampling methods) have lead to efficient solutions for complex configurations spaces and robots with many degrees of freedom, sometimes biassing the sampling towards the obstacles or the medial axis [29], a solution that is similar to the method presented here. The main difference is that we base our method in a biological plausible solution, linking path planning with a self-regulation model. All the computation is reduced to learn a gradient that is constituted by Gaussians. The learned open space map (see previous section III) is the perfect candidate to guide both the optimization process and reduce space dimensionality. A second novelty is that Gaussians are used to interpret the semantics of the environment, giving a special role to corners and bifurcations which will be probabilistically detected to drive the search. Doing so, the efficiency of a search becomes independent of the size of the environment (a property called abstraction in incremental search algorithms [30]).

Algorithm 2: Generate a single forward path of Gaussians.

$M_{bif} \leftarrow M_{bif} + g_{\sigma,x_{robot},y_{robot}}$
3 $\langle x_0, y_0 \rangle \leftarrow randomPosIn(M_{bif} > 0.1)$
while $(notfound)$ *and* $(ngaussians(M_{path}) < nGauss)$ **do**
4 $\quad \langle x'_0, y'_0 \rangle \leftarrow \langle x_0 \pm rand(range), y_0 \pm rand(range) \rangle$
5 $\quad$ **if** $isValidPlaceCell(g_{\sigma,x'_0,y'_0})$ *and* $\overrightarrow{M}_{aro}(x'_0, y'_0) > 0.8$
6 $\quad$ *and* $\overrightarrow{M}_{vis}(x'_0, y'_0) < 0.5$ *and* $\overrightarrow{M}_{path}(x'_0, y'_0) = 0$ **then**
$\quad\quad M_{path} = M_{path} + g_{\sigma,x'_0,y'_0}$
$\quad\quad \langle x_0, y_0 \rangle \leftarrow \langle x'_0, y'_0 \rangle$
$\quad\quad seq \leftarrow seq \cup \langle x_0, y_0 \rangle$
$M_{vis} \leftarrow M_{vis} + M_{path}$
7 **if** $dirChange(seq)$ **then**
8 $\quad \langle x_0, y_0 \rangle \leftarrow pointOfDirChange(seq)$
$\quad M_{bif} \leftarrow M_{bif} + g_{\sigma,x_0,y_0}$
$found \leftarrow isGoalIn(M_{vis} > 0)$

Algorithm 2, called *forward paths* in inspiration from [24], corresponds to a single generation of a path of Gaussians. The algorithm can be iteratively used until a feasible path is found. M_{path} is the gradient that accumulates the Gaussians of the generated path and is initialized to 0 at each single path generation. M_{vis} accumulates all the generated Gaussians, thus acts as a mark of the visited paths. M_{bif} is the gradient that accumulates the evidence of special locations like corners and bifurcations. M_{bif} is used to initiate the search (see labeled line 3). The function $randomPosIn$ just selects a position at random where M_{bif} is greater than 0.1. When the search starts it contains no suggested corners or bifurcations, M_{bif} is then initialized with the robot position (first line of the algorithm). We then activate at random neighboring Gaussians (line 4) and filter them by several conditions: the Gaussian must be a valid place cell (as in previous algorithm 1), the Gaussian must be in a high area of the open map

gradient (line 5), the Gaussian must be in a low visited area (line 6) and it must not fall into the current generated path (line 6). If all the conditions are fulfilled we add the Gaussian in the path M_{path} and we retain the sequence. When we reached a desired number of generated Gaussians (variable $nGauss$) we check if the generated sequence (variable seq) contains an abrupt angle change (this is done by checking a change greater than 85 degrees in 4 consecutive Gaussians), thus being a candidate for a corner or a bifurcation (line 7). If it is the case we accumulate this evidence in the point of change (label 8). A possible generated path of Gaussians by a forward path iteration is shown in figure 6(d).

In figure 6 we show the results of the forward paths algorithm 2. As in [28], the time to reach the target is plotted against the optimal length of the path in grid units (computed using Dijkstra algorithm in the underlying grid). The target is proven to be always reached, which is not the case if we don't apply the restart strategy guided by M_{bif}. The algorithm has a number of free parameters: the range (variable $range$ controlling the randomized distance to the next Gaussian), the thresholds of the conditions in lines 5 and 6, the threshold angle in $dirChange(seq)$ for accumulating M_{bif}. Our belief is that we can setup those to find the optimal configuration to make the solving time more dependant on the number of bifurcations instead of the optimal distance. The majority of solving times in figure 6 follow a linear increase with respect to the shortest path distance. This fact supports the assumption that recognizing the semantics that some Gaussians have because of its localization (corners, bifurcations) is important.

From the biological side, the forward-paths algorithm presents a new ingredient for interpreting path planning using a cognitive map. This fact is adding abstraction capabilities to the algorithm presented in [28] and also to other Reinforcement Learning (RL) approaches to solve this problem [12], [14]. Usually, these suffer from the handicap of starting from a predefined state space representation and also from the inability to operate in different state space granularity or resolution: state space representation matters and can lead to a combinatorial explosion. A recent work exists that uses RL [15] and addresses this problem by indexing the continuous spatial environment using the place cells representation and associating to each place cell a weighting of eight possible direction cells. [15] solves the combinatorial explosion associated to the spatial states by indexing space with the place cells but then introduces another discretization in the space of possible directions. Our algorithm solves this problem by sequentially activating neighboring place cells to generate another gradient towards the goal, thus not representing the directions explicitly.

V. Mixed Reality Robot Arena

Mixed Reality Robot Arena is a mixed real/virtual environment for epuck robots. The system is able to track the robots and deliver to them real and virtual stimulus. It consists of a retro projected surface on top of which a real robot navigates. It is equipped with an infrared camera and tracking system.

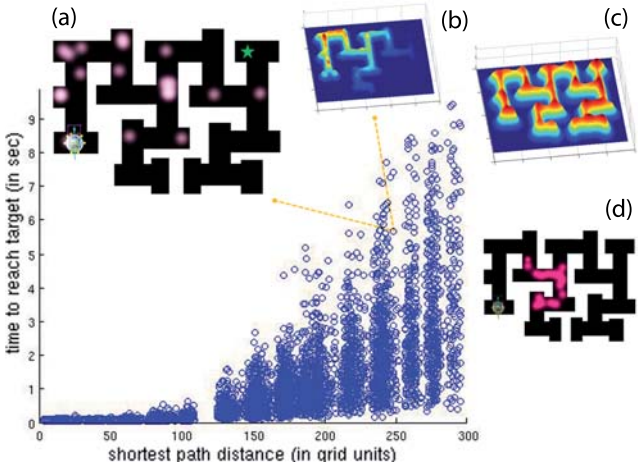

Fig. 6. Path finding results. The time to reach a randomized target is plotted against the optimal path (in grid units) computed with a standard Dijkstra algorithm. Labels (a,b) are example gradients after reaching a target location placed at 250 grid units distance (denoted by a green star). The position of the robot is indicated in (a) with a miniature photo of the epuck. In (a) we show the M_{bif} gradient superimposed to the maze. The gradient is stronger in bifurcations and corners. (b) corresponds to the M_{vis} gradient. (c) M_{aro} after convergence. (d) a possible generated single path of Gaussians, M_{path}.

Its main purpose is to be able to generate dynamic cues to the robot without much effort. With a click of the mouse we can displace the robot into a different arena. We can also display information relative to the solving process (see figure 4 and the delivered video [1]. We place the robot in half the way between reality and a simulated environment.

VI. Conclusions

We have described a computational model able to approximate certain characteristics of rodent behavior: free exploration, goal directed behavior driven by visual cues and path planning towards a target destination. The main principle of the model is based on decomposing behavior into different homeostatic subsystems, orchestrated by the allostatic control achieving stability at a meta-level. Each subsystem receives local input of a gradient and regulates itself according to the difference between the actual value and the desired one. These gradients are then combined and weighted by the internal relevance. The meta-control dynamically changes the desired values of each subsystem and the gradients themselves. In this sense we go one step further than the potential field approaches [6], [7], [10]. More specifically, in the context of the motor schemas based behaviors a schema-based homeostatic control has been described in [7]. Compared to the motor schemas approach, we are contributing in several aspects: firstly due to the nature of the meta-control a gradient can change its valence (from appetitive to aversive, and vice versa) via the desired value. Moreover from the fact that gradients can be learned we make much less assumptions about the environment. Specifying the capabilities of the meta-control we gain

[1]Video material is available at `http://specs.upf.edu/sf/`

scalability and we are able to include more subsystems to account for more complex behaviors as proven in the jump from the square arena task II-D to the y-maze II-D.

We then describe a biological plausible process to dynamically learn, generate and modify the gradients. This dynamical process may have an impact in explaining how place cells are used in the hippocampus to compute properties of the environment, or for explaining how they can be used to goal directed navigation. With respect to place cells neuronal models used for navigation [11], [12], [13], [14], [15], [16] our contribution is to make use of place cells in a way that is new to our knowledge, computing properties of the environment.

A third contribution is that we formulate path planning in terms of gradient learning. Subsequent activations of place cells of a cognitive map are used to generate a gradient driving to a goal. We have described a procedure that exploits the fact that certain place cells play a special role in path planning (like the ones representing corners or bifurcations) and is able to actively use them to speed up path planning. In this sense, we give new insights to approaches like [28], making the time to reach the goal dependant on the number of bifurcations rather than the size of the maze, thus accounting for abstraction of the environment.

We believe that the latter contributions generalize, can be applied to other kind of robots than the epuck, and remain fundamental to understand how a cognitive map can be exploited at a higher cognitive level to compute properties of the environment and perform path planning. These contributions, together with the fact that we combine them with self-regulation, achieved by the principles of *allostasis*, give certainly insights on how to implement an artificial rat.

VII. ACKNOWLEDGMENTS

This work was supported by Synthetic Forager FP7-ICT-217148. We thank TelAviv University (TAU), for providing us with the rat data used for comparing to the robot. Sincere thanks to anonymous reviewers for detailed and relevant comments.

REFERENCES

[1] A. Maslow, "A theory of human motivation," *Psychological Review*, vol. 50, no. 4, pp. 370–96, —1943—.

[2] R. J. Blanchard and C. D. Blanchard, "Antipredator defensive behaviors in a visible burrow system," *Journal of Comparative Psychology*, vol. 103, no. 1, pp. 70–82, —1989—.

[3] C. Hull, *Principles of Behavior*. New York: Appleton-Century-Crofts, —1943—.

[4] E. A. Di Paolo, "Organismically-inspired robotics: Homeostatic adaptation and natural teleology beyond the closed sensorimotor loop," in *Dynamical Systems Approach to Embodiment and Sociality*, K. M. . T. A. (Eds), Ed., —2003—, vol. Advanced Knowledge International, pp. 19 – 42.

[5] B. S. McEwena and J. C. Wingfield, "The concept of allostasis in biology and biomedicine," *Hormones and Behavior*, vol. 2, pp. 2–15, —2003—.

[6] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, —1986—.

[7] R. C. Arkin, *Behavior-Based Robotics*. MIT Press, —1998—.

[8] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *Transactions on Robotics*, vol. 21, no. 3, pp. 354–363, —2005—.

[9] D. Wooden, *Graph-based Path Planning for Mobile Robots*, ser. Dissertation. Georgia Institute of Technology, —2006—.

[10] S. R. Lindemann and S. M. LaValle, "Simple and efficient algorithms for computing smooth, collision-free feedback laws over given cell decompositions," *International Journal of Robotics Research*, vol. 28, pp. 600–621, —2009—.

[11] R. Chavarriaga, T. Strsslin, D. Sheynikhovich, and W. Gerstner, "A computational model of parallel navigation systems in rodents," *Neuroinformatics*, vol. 3, no. 3, pp. 223–241, —2005—.

[12] J. L. Krichmar, A. K. Seth, D. A. Nitz, J. G. Fleischer, and G. M. Edelman, "Spatial navigation and causal analysis in a brain-based device modeling cortical-hippocampal interactions," *Neuroinformatics*, vol. 3, no. 3, pp. 197–221, —2005—.

[13] A. V. Samsonovich and G. A. Ascoli, "A simple neural network model of the hippocampus suggesting its pathfinding role in episodic memory retrieval," *Learn Mem.*, vol. 12, no. 2, pp. 193–208, —2005—.

[14] T. Strosslin, D. Sheynikhovich, R. Chavarriaga, and W. Gerstner, "Robust self-localisation and navigation based on hippocampal place cells," *Neural Networks*, vol. 18, no. 9, pp. 1125–1140, —2005—.

[15] M. Tamosiunaite, J. Ainge, T. Kulvicius, B. Porr, P. Dudchenko, and F. Worgotter, "Path-finding in real and simulated rats: assessing the influence of path characteristics on navigation learning," *Journal of computational neuroscience*, vol. 25, no. 3, pp. 562–582, —2008—.

[16] D. Sheynikhovich, R. Chavarriaga, T. Strosslin, A. Arleo, and W. Gerstner, "Is there a geometric module for spatial orientation? insights from a rodent navigation model," *Psychological Review*, vol. 116, no. 3, pp. 540–566, —2009—.

[17] M. Sanchez-Fibla, U. Bernardet, E. Wasserman, T. Pelc, M. Mintz, J. C. Jackson, C. Lansink, C. Pennartz, and P. F. Verschure, "Allostatic control for robot behavior regulation: a comparative rodent-robot study," *Advances In Complex Systems*, vol. In Press, 2010.

[18] A. G. Watts, "Motivation," in *Handbook of brain theory and neural networks*. MIT Press, —2003—, pp. 680–683.

[19] N. Bischof, "Die regulation der sozialen distanz - von der feldtheorie zur systemtheorie," *Zeitschrift fr Psychologie*, vol. 201, pp. 5–43, —1993—.

[20] A. Dvorkin, Y. Benjamini, and I. Golani, "Mouse cognition-related behavior in the open-field: emergence of places of attraction," *PLoS Comput Biol*, vol. 4, no. 2, p. e1000027, —2008—.

[21] R. Wyss, P. Konig, and P. F. M. J. Verschure, "A model of the ventral visual system based on temporal stability and local memory," *PLoS Bio*, vol. 4, no. 5, p. e120, —2006—.

[22] G. Wallraff, *Avian navigation: Pigeon homing as a paradigm*. Springer-Verlag, —2005—.

[23] J. O'Keefe and J. Dostrovsky, "The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat," *Brain Res.*, vol. 34, no. 1, pp. 171–175, —1971—.

[24] A. Johnson and D. Redish, "Neural ensembles in ca3 transiently encode paths forward of the animal at a decision point," *Journal of Neuroscience*, vol. 27, no. 45, pp. 12 176–12 189, —2007—.

[25] J. A. Ainge, M. Tamosiunaite, F. Woergoetter, and P. A. Dudchenko, "Hippocampal ca1 place cells encode intended destination on a maze with multiple choice points," *Journal of Neuroscience*, vol. 27, no. 36, pp. 9769–9779, —2007—.

[26] R. Muller and J. Kubie, "The firing of hippocampal place cells predicts the future position of freely moving rats," *Journal of Neuroscience*, vol. 9, pp. 4101–4110, —1989—.

[27] R. A. Koene and M. E. Hasselmo, "Reversed and forward buffering of behavioral spike sequences enables retrospective and prospective retrieval in hippocampal regions ca3 and ca1," *Neural Networks*, vol. 21, no. 2-3, pp. 276–288, —2008—.

[28] M. Toussaint, "A sensorimotor map: Modulating lateral interactions for anticipation and planning," *Neural Computation*, vol. 18, no. 5, pp. 1132 –1155, —2006—.

[29] L. Jaillet and T. Simeon, "Path deformation roadmaps : compact graphs with useful cycles for motion planning," *International Journal of Robotics Research*, vol. 27, no. 11-12, —2008—.

[30] N. Sturtevant and M. Buro, "Partial pathfinding using map abstraction and refinement," in *AAAI Conference*, vol. 3. AAAI Press, —2005—, pp. 1392–1397.