Procedia Computer Science

Volume 71, 2015, Pages 178–184

2015 Annual International Conference on Biologically Inspired Cognitive Architectures

Respective advantages and disadvantages of model-based and model-free reinforcement learning in a robotics neuro-inspired cognitive architecture

Erwan Renaudo[1,2], Benoît Girard[1,2], Raja Chatila[1,2], and Mehdi Khamassi[1,2]

[1] Sorbonne Universités, UPMC Univ Paris 06,
UMR 7222, Institut des Systèmes Intelligents et de Robotique, F-75005, Paris, France
[2] CNRS, UMR 7222, Institut des Systèmes Intelligents et de Robotique, F-75005, Paris, France
renaudo@isir.upmc.fr

Abstract

Combining model-based and model-free reinforcement learning systems in robotic cognitive architectures appears as a promising direction to endow artificial agents with flexibility and decisional autonomy close to mammals. In particular, it could enable robots to build an internal model of the environment, plan within it in response to detected environmental changes, and avoid the cost and time of planning when the stability of the environment is recognized as enabling habit learning. However, previously proposed criteria for the coordination of these two learning systems do not scale up to the large, partial and uncertain models autonomously learned by robots. Here we precisely analyze the performances of these two systems in an asynchronous robotic simulation of a cube-pushing task requiring a permanent trade-off between speed and accuracy. We propose solutions to make learning successful in these conditions. We finally discuss possible criteria for their efficient coordination within robotic cognitive architectures.

Keywords:

Robotic Cognitive Architecture ; Reinforcement Learning ; Biological Inspiration

1 Introduction

Psychology and neuroscience have highlighted that learning and decision-making in mammals result from the combination of two types of behaviors: goal-directed and habitual behaviors, that are respectively exhibited after moderate and extensive training on a given task. When rodents, monkeys or humans start a new decision-making task, they appear to initially rely on their goal-directed system, taking time to analyze the structure of the task in order to build an internal model of it, and make slow decisions by planning and inferring the long-term consequences of their possible actions before deciding what to do next. Then as their

 Selection and peer-review under responsibility of the Scientific Programme Committee of BICA 2015

doi:10.1016/j.procs.2015.12.194

performance gradually improves, they appear to make quicker and quicker decisions, putatively relying on their habitual system which slowly acquires simple stimulus-response associations to solve the task. Finally, when subjects restart to make errors after a task change, they appear to restart planning within their internal model and thus slow down their decision process before acquiring the new task contingencies [2]. The coordination of these two learning systems may be evolutionarily advantageous by allowing mammals to avoid linejours og and costly computations when the environment is sufficiently stable to allow habit learning, while still enabling animals to detect environmental changes requiring to update their internal model and replan.

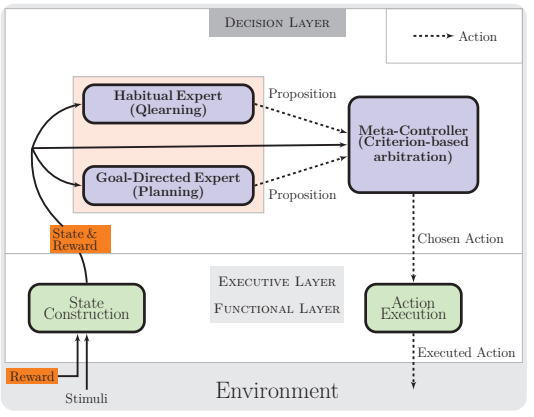
Computational modelling work has shown that the model-based (MB) / model-free (MF) reinforcement learning (RL) framework can capture these different types of learning behaviors [4], the internal model being in this case: (1) a transition function giving for each (state, action) couple a probability to end up in each possible state of the environment; (2) a reward function explicitly storing which (state, action) couples trigger a reward from the environment. However, the principles for the coordination of MB and MF RL are still debated. In [4], the coordination depends on the uncertainty of each system. This works well in simple simulated tasks with less than 10 states but does not scale-up to real-world robotic tasks involving hundreds of states. Keramati and colleagues [7] proposed to avoid estimating the uncertainty of the MB system by considering that it always has "perfect information", which again cannot be true in tasks where the large number of states imposes to compute approximations of the transition function, as we previously found in a robotic navigation implementation of these learning processes [3].

In a recent work, we proposed a new neuro-inspired cognitive architecture combining MB and MF RL applied to a robotic cube-pushing task [11] and a human-robot cooperation task [10]. We found in these cases that the combination of these two learning systems (Experts) performs better than each learning system alone, which validates the relevance of the approach for robotics. Nevertheless, we also found that the large number of states in these robotic tasks makes the MB system reach poor performance, preventing a number of tested criteria for systems coordination from achieving the task [12].

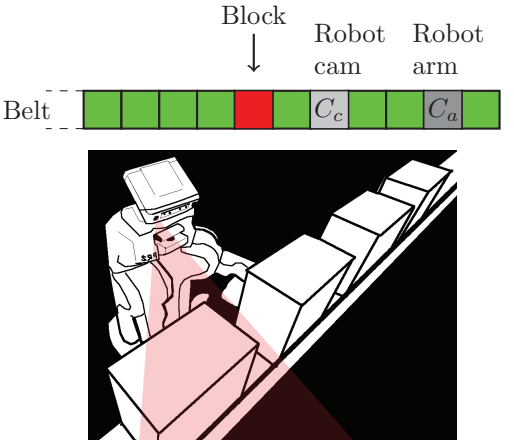
Here we present new simulations of our cognitive architecture in the robotic cube-pushing task in order to precisely analyze the respective performance of MB and MF RL and understand why classical neuro-inspired criteria for their coordination are not appropriate in this paradigm. While the small number of states in computational neuroscience tasks enables the MB to replan from scratch at each new decision (e.g. [5]), here we find that it can reach a good performance only when starting from the previous plan at each new decision, which to our knowledge constitutes a new proposal for computational neuroscience MB/MF RL models. We then show in which cases MB and MF are advantageous and finish by discussing which criteria could be efficient in these conditions.

2 Materials and Models

We study Experts from the Decision Layer [1] of the robotic cognitive architecture (figure 1a) proposed in our previous work [11, 12, 10], implemented in ROS (Robot Operating System, [9]) and simulated with move3D [13] in [10] (ROS in other works). The Decision Layer receives a state built from perception and a reward from the environment, that gives a feedback on the previous action choice. This information is sent to Experts and a Meta-Controller (MC) in charge of selecting the most relevant Expert in the current situation, according to a criterion (see [12] for a study of criteria). It delivers the final action to the Executive Layer, that recruits the right skills (eg. SLAM, obstacle avoidance and control laws for navigating) from the Functional Layer to actually perform the action. In this work, these layers are kept simple and the action's



(a) Experts and Meta-Controller in the Decision Layer exchange state, reward and action with the environment through the abstraction levels of the Executive and Functional Layers.



(b) The experiment used for evaluating the architecture as described in [11]. Blocks are coming towards the robot which can visually perceive the presence of a block in C_c and by touch in C_a.

Figure 1: *1a Control architecture and 1b simulated task*

effect is directly solved by the simulation.

2.1 Habitual Expert (model-free RL)

The habitual behavior is implemented as in [11]. It is a 1-layer neural network where weights W between state input S_t and action output encode the value $Q_t(s, a)$ of doing a certain action in the considered state. Thus, estimating action values consists in propagating input activity into the network to get output activity, from which is computed an action selection probability distribution used for decision. It is done with Eq. (1) (b_t^i is the bias for action a^i).

$$Q_t(s_t, a_t^i) = W_t \cdot S_t + b_t^i \ . \tag{1}$$

W_t is updated incrementally according to the Qlearning rule (Eq. (2), [14]) and the reward r_t obtained (γ_{MF} : discount factor).

$$\delta = r_t(s_{t-1}, a_{t-1}) + \gamma_{MF} \cdot \max_a Q_{t-1}(s_t, a) - Q_{t-1}(s_{t-1}, a_{t-1}) \ . \tag{2}$$

Learning these associations is a long process as only the experienced states can be updated, and this process is even longer in case previous knowledge biases the behavior which makes the agent only explore a subpart of the task's space.

The conversion from $Q_t(s_t, a_t)$ to the probability $P_t(a_t \mid s_t)$ is done using the softmax function and the decision is drawn from the resulting distribution. Thus, the initial policy is random as all $Q_t(s, a)$ are equal to zero before a reward is received.

2.2 Goal-directed Expert (model-based RL)

The goal-directed behavior, as in [11], incrementally learns transition and reward models (T, R) of the task after each interaction with the environment. The reward model is a collection of

(s, a) pairs directly associated to the instant reward r_t experienced. The transition model is incrementally learned storing connection strengths of $s \xrightarrow{a} s'$ according to Eq. (3). The strength $T(s, a, s')$ is updated at learning rate α_{MB} and converted to a transition probability $Pr_t(s, a, s')$ when planning. These updates happen when a change is experienced by the robot.

$$T_t(s, a, s') = T_{t-1}(s, a, s') + \alpha_{MB} \cdot (1 - T_{t-1}(s, a, s')) \ . \tag{3}$$

$$Q_t(s, a) = \max \left(r_t(s, a), (\gamma_{MB} \cdot \sum_{s'} Pr_{t-1}(s, a, s') \cdot \max_{a'} Q_t(s', a')) \right) \ . \tag{4}$$

Action values $Q_t(s, a)$ are computed by planning (i.e. propagating the known rewards in the graph generated by the transition function using Eq. (4)), and are used to decide an action in each state. Planning is done with a limited time budget per decision. When time expires, the estimation of $Q_t(s, a)$ is stopped and the current estimates are used for decision. Because of this time constraint, planning is focused on the N most visited states, hypothetized to be the most interesting in a stable environment. This is done by computing the ratio R_n between the current entropy of the state visits distribution and its maximal entropy and selecting $N = R_n * N_{max}$ states. See [11] for the implementation details.

Changes in the environment (ie. a transition leads to a new state or reward's value has dropped) modify the models when experienced. The next computation of $Q_t(s, a)$ takes the model updates into account and adapts the behavior accordingly. Decision is taken as for the habitual expert.

Here, we consider two versions of planning: the "drop" version clears its $Q_t(s, a)$ estimation and compute from scratch; the "keep" version computes starting from the previous estimation.

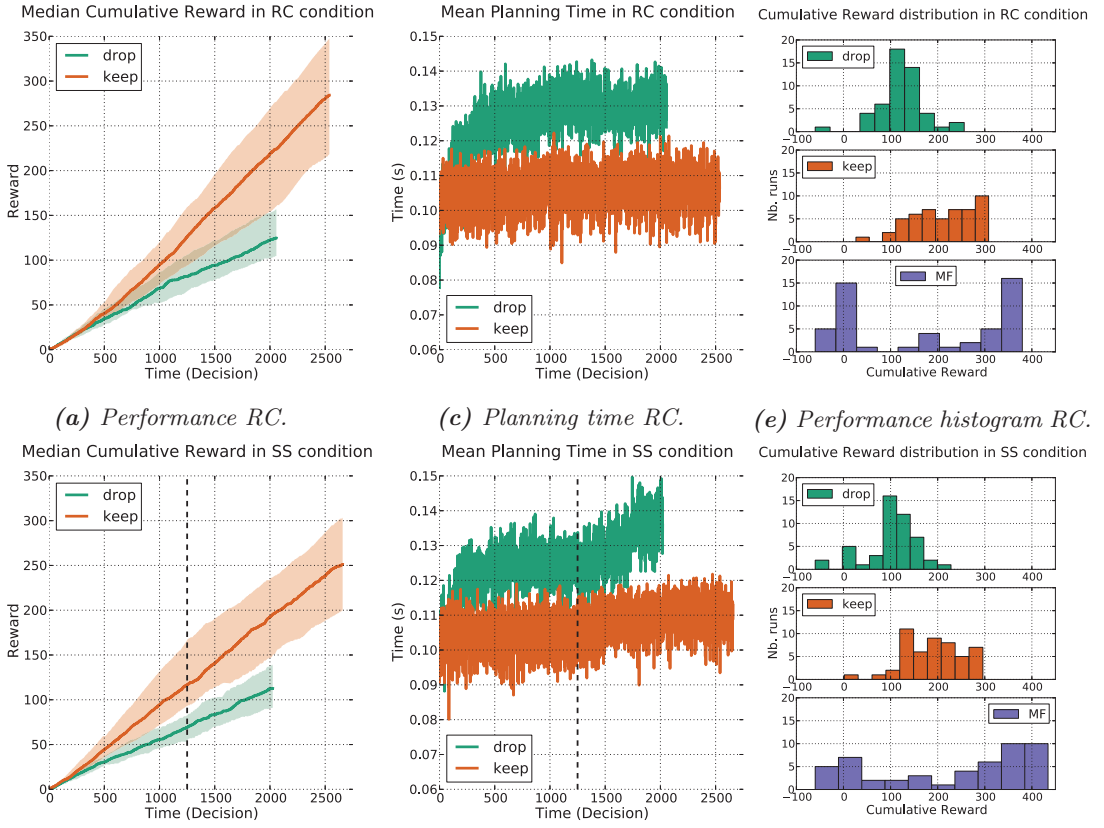
2.3 Task

We evaluate each Expert controlling individually the robot in the cube-pushing task from [11] (see fig 1b). A simulated robot is facing a conveyor belt with evenly spaced blocks coming at speed v_{bs}. The robot has 3 actions ("Do Nothing", "Look Cam", "Push Arm") that can modify the environment and update its binary perception of blocks. As time elapses, previous perceptions are kept in memory (separately in a visual memory and a tactile one) and the state of memories produces the state s_t sent to Experts.

Two conditions are defined in the task: the Regular Condition (RC), where v_{bs} is kept constant and the Speed Shift condition (SS) where v_{bs} increases after a certain number of decisions. If the task in itself, especially in the Regular Condition, is deterministic and predictable by the robot, its representation by the robot is more complex. Because no synchrony is forced between robot actions and the world, aliasing can happen in the state experienced by the robot. Thus, erroneous transitions can be experienced and the models built include stochastic transitions. We do not address the question of representations in this work, but previously pointed out the importance of that question for real autonomous robots in [12].

3 Results

In these conditions, we found that keeping the previous plan between decisions in the model-based system dramatically improved the performance (figures 2a & 2b), compared to dropping the plan and replanning from scratch at each new decision as it is proposed in computational neuroscience models (eg. [5]). The improvement of performance was true in both the regular



(a) *Performance RC.* ***(c)*** *Planning time RC.* ***(e)*** *Performance histogram RC.*

(b) *Performance SS.* ***(d)*** *Planning time SS.* ***(f)*** *Performance histogram SS.*

Figure 2: *Left column: Model-based performance in both conditions. The performance is evaluated as the median of the cumulative reward with its* 1^{st} *and* 3^{rd} *quartiles. Middle column: Model-based time of planning, in seconds. Right column: Experts performance histogram showing the distribution of runs. The dashed line indicates the occurrence of the speed shift.*

case and the speed shift case, showing that keeping the previous plan does not reduce the ability of the model-based system to flexibly adapt to environmental changes in our task. In addition, keeping the previous plan also allowed the model to reduce decision-making time in the two studied experimental cases (figures 2c & 2d). Indeed, starting with at least partially relevant information at each new decision may have allowed the planning process to converge more quickly. Moreover, shifting the speed of the conveyor belt (figure 2d) resulted in an increase in decision-making time for the model-based system which drops the previous plan at each new decision. This is because in this case the model had both to adapt its internal model to the new environment and to replan from scratch in this model. In contrast, the model-based system which keeps the previous plan only marginally increased its decision-making time in response to a speed shift. The model-based system in this case had to adapt its internal model but could benefit from the previous plan in the major part of the model where this plan was still relevant after the environmental change.

After simulating 50 experiments with the goal-directed Expert (MB drop the plan, MB keep the plan) and with the habitual Expert in each condition (regular condition, speed shift), we

found that the distribution of performances of the model-based system was consistently shifted to the right when keeping the plan (2^{nd} row in figures 2e & 2f) compared to dropping it (1^{st} row in figures 2e & 2f). This again argues in favor of keeping the plan in the model-based system in robotic tasks involving hundreds of states, in contrast with computational neuroscience implementations using less than ten states. Interestingly, we also found that the used parameterization of the model-free system produced a somehow bimodal distribution of performances (3^{rd} row in figures 2e & 2f): a subset of simulated MF reached poor performance compared to MB, as classically considered in computational neuroscience; in contrast, another subset of simulated MF reached better performance than the model-based system. This variability was partly due to the parameterization we used for the MF system which was optimized on average performance. When optimizing to reduce variability, the average performance of MF is intermediate between the two distributions but still not dramatically poorer in performance than the MB system (data not shown). This suggests that the MF system can sometimes reach a better or at least similar performance to the MB system, especially in robotic tasks where the large number of states make it difficult to learn and exploit an internal model of the environment. Thus in contrast to the previously proposed criteria for the coordination of MB and MF RL in computational neuroscience and considering that the MB system is always efficient [4, 7, 5], an efficient cognitive architecture should rather monitore on-line the respective performance of MB and MF system and give the hand to the one which reveals the most advantageous at a given moment.

4 Discussion

We presented an analysis of the behavior of Experts used in our cognitive architecture and showed that for the goal-directed Expert (model-based RL), overall robot performance can be improved in both conditions of our task by keeping the plan (i.e. action value estimation) from one decision to the other, instead of replanning from scratch as most of the work [4, 5] we take inspiration from do. We also showed that the planning time is shorter when keeping the plan, and stays more stable when the environment changes. Finally we showed that the task can sometimes make the habitual Expert (model-free RL) more efficient than the goal-directed Expert, contradicting the hypothesis that due to the flexibility allowed by its internal model, the latter is always more efficient than the former.

These results give a second chance to the criteria proposed in [12], as several are strongly relying on the goal-directed Expert performance. Improving the Expert can allow some criteria to perform better than a random mixture of model-free and model-based RL that we initially tested as proof-of-concept for their coordination [11].

Moreover, this analysis highligths that monitoring more accurately Experts can bring more information on their current adequation to the task: the evolution of the number of states and transitions in the goal-directed Expert can provide an analogous measure to the variation of action values in the habitual Expert on the status of learning. These measures could in addition constitute indirect measures of uncertainty within the model-based system, thus avoiding to compute computationally expensive explicit Bayesian measures of it as it has been previously proposed [4].

Although we mainly studied the planning capabilities of the goal-directed expert, studying how the model is learnt more precisely could also lead to improvements in performance: in order to keep a relevant model, having a forgetting mechanism or pruning transitions [6] could remove the transitions coming from erroneous perceptions that are rarely experienced.

The bimodal performance of the habitual behavior and the difficulty to optimize both the

performance and the variability on this task raises the idea that the role of the Meta-Controller could also be to adapt Experts' parameters online, as we previously proposed in a robotic implementation of meta-learning applied to a single model-free learning process [8]. In future work, we will test these different online performance monitoring and meta-learning processes applied to MB/MF RL in our cognitive architecture confronted to a variety of robotic tasks to assess its robustness.

Acknowledgements

This work has been funded by a DGA (French National Defence Armaments Procurement Agency) scholarship (ER) and by the French Agence Nationale de la Recherche (Grants ANR-12-CORD-0030, ANR-11-IDEX-0004-02) and by Sorbonne-Universités SU-15-R-PERSU-14 PERSU.

References

[1] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An architecture for autonomy. *IJRR Journal*, 17:315–337, 1998.

[2] B. W. Balleine and J. P. O'Doherty. Human and rodent homologies in action control: corticostriatal determinants of goal-directed and habitual action. *Neuropsychopharmacology*, 35:48–69, 2010.

[3] K. Caluwaerts, M. Staffa, S. N'Guyen, C. Grand, L. Dollé, A. Favre-Félix, B. Girard, and M. Khamassi. A biologically inspired meta-control navigation system for the psikharpax rat robot. *Bioinspiration & Biomimetics*, 7:025009, 2012.

[4] N.D. Daw, Y. Niv, and P. Dayan. Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nat. Neurosci.*, 8(12):1704–1711, 2005.

[5] A. Dezfouli and B. W. Balleine. Habits, action sequences and reinforcement learning. *Eur. J. Neurosci.*, 35(7):1036–1051, 2012.

[6] Q.J. Huys, N. Eshel, E. O'Nions, L. Sheridan, P. Dayan, and J.P. Roiser. Bonsai trees in your head: how the pavlovian system sculpts goal-directed choices by pruning decision trees. *PLoS Comp. Biol.*, 8(3), 2012.

[7] M. Keramati, A. Dezfouli, and P. Piray. Speed/accuracy trade-off between the habitual and goal-directed processes. *PLoS Comp. Biol.*, 7(5):1–25, 2011.

[8] M. Khamassi, S. Lallée, P. Enel, E. Procyk, and P.F. Dominey. Robot cognitive control with a neurophysiologically inspired reinforcement learning model. *Frontiers in Neurorobotics*, 5:1, 2011.

[9] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[10] E. Renaudo, S. Devin, B. Girard, R. Chatila, R. Alami, M. Khamassi, and A. Clodic. Learning to interact with humans using goal-directed and habitual behaviors. *RoMan 2015, Workshop on Learning for Human-Robot Collaboration.*, 2015.

[11] E. Renaudo, B. Girard, R. Chatila, and M. Khamassi. Design of a control architecture for habit learning in robots. In *Biomimetic and Biohybrid Systems, LNAI Proceedings*, pages 249–260, 2014.

[12] E. Renaudo, B. Girard, R. Chatila, and M. Khamassi. Which criteria for autonomously shifting between goal-directed and habitual behaviors in robots? In *5th International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EPIROB)*, pages –, Providence, RI, USA, 2015.

[13] T. Siméon, J-P. Laumond, and F. Lamiraux. Move3d: a generic platform for path planning. In *in 4th Int. Symp. on Assembly and Task Planning*, pages 25–30, 2001.

[14] C. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, 1989.