Reducing computational cost during robot navigation and human-robot interaction with a human-inspired reinforcement learning architecture

Rémi Dromnelle, Erwan Renaudo, Mohamed Chetouani, Petros Maragos, Raja Chatila, Benoît Girard, Mehdi Khamassi

REDUCING COMPUTATIONAL COST DURING ROBOT NAVIGATION AND HUMAN-ROBOT INTERACTION WITH A HUMAN-INSPIRED REINFORCEMENT LEARNING ARCHITECTURE

PREPRINT OF THE PAPER PUBLISHED IN IJSR (2022), SPECIAL ISSUE ON 'HUMAN-LIKE BEHAVIOR AND COGNITION IN ROBOTS'

Rémi Dromnelle
Institute of Intelligent Systems and Robotics
Sorbonne Université, CNRS
Paris, France
remi.dromnelle@gmail.com

Erwan Renaudo
Intelligent and Interactive Systems Group,
Universität Innsbruck
Innsbruck, Austria
erwan.renaudo@uibk.ac.at

Mohamed Chetouani
Institute of Intelligent Systems and Robotics
Sorbonne Université, CNRS
Paris, France
mohamed.chetouani@sorbonne-universite.fr

Petros Maragos
1 Athena Research and Innovation Center
2 School of ECE, National Technical Univ. of Athens
Athens, Greece
maragos@cs.ntua.gr

Raja Chatila, Benoît Girard, Mehdi Khamassi
Institute of Intelligent Systems and Robotics
Sorbonne Université, CNRS
Paris, France
firstname.lastname@sorbonne-universite.fr

November 24, 2022

ABSTRACT

We present a new neuro-inspired reinforcement learning architecture for robot online learning and decision-making during both social and non-social scenarios. The goal is to take inspiration from the way humans dynamically and autonomously adapt their behavior according to variations in their own performance while minimizing cognitive effort. Following computational neuroscience principles, the architecture combines model-based (MB) and model-free (MF) reinforcement learning (RL). The main novelty here consists in arbitrating with a meta-controller which selects the current learning strategy according to a trade-off between efficiency and computational cost. The MB strategy, which builds a model of the long-term effects of actions and uses this model to decide through dynamic programming, enables flexible adaptation to task changes at the expense of high computation costs. The MF strategy is less flexible but also 1000 times less costly, and learns by observation of MB decisions. We test the architecture in three experiments: a navigation task in a real environment with task changes (wall configuration changes, goal location changes); a simulated object manipulation task under human teaching signals; and a simulated human-robot cooperation task to tidy up objects on a table. We show that our human-inspired strategy coordination method enables the robot to maintain an optimal performance in terms of reward and computational cost compared to an MB expert alone, which achieves the best performance but has the highest computational cost. We also show that the method makes it possible to cope with sudden changes in the environment, goal changes or changes in the behavior of the human partner during interaction tasks. The robots that performed these experiments, whether real or virtual, all used the same set of parameters, thus showing the generality of the method.

Keywords strategy coordination, cognitive monitoring, reinforcement learning, robot cognitive architecture, navigation, HRI, neuro-inspiration

1 Introduction

The field of robot reinforcement learning (RL) has seen a fast growth in the last decade [Kober et al., 2013, Khamassi et al., 2018, Ibarz et al., 2021]. In particular, notable progresses have been made with the use of deep RL algorithms [Mnih et al., 2015], which enable to deal with large continuous state and action spaces. Nevertheless, these methods are computationally very costly, requiring millions of iterations before convergence [Justus et al., 2018, Strubell et al., 2019]. Moreover, they are most of the time designed specifically for a given scenario, thus preventing generalization. More precisely, the human designer either goes for a model-based (MB) RL, when it seems feasible for the robot to try and estimate a model of the effect of its actions, or for a model-free (MF) RL one, when it does not seem feasible [Wang et al., 2019]. Overall, a wide variety of algorithmic solutions exist (some being value-based, other being policy-based), each being more appropriate to specific experimental scenarios [Kober et al., 2013]. While recent hybrid MB/MF robot learning methods have been proposed [Caluwaerts et al., 2012b, Chebotar et al., 2017], it is not clear if they could cope on-the-fly with the high degree of variability and non-stationarity of human-robot interaction (HRI), and at the same time minimize computational cost. To our knowledge, no generic solution exists that enable robots to automatically choose the most efficient and least costly learning algorithm in a variety of contexts depending on the characteristics of the task at hand.

In contrast, humans, and more generally mammals, are endowed with behavioral flexibility which enable them to adapt to a variety of contexts and situations. One of the key ingredients of this behavioral flexibility is thought to be a certain degree of modularity within their cognitive architecture, so that learning and decision-making processes rely on the alternation and sometimes combination of different learning strategies [Hikosaka et al., 1999, Daw et al., 2005, Dollé et al., 2008, 2010, Khamassi et al., 2011, Khamassi and Humphries, 2012, Van Der Meer et al., 2012, O'Doherty et al., 2017]. In other words, humans have different cognitive tools within their mental toolbox, and can reuse the tools they think are appropriate in new situations while minimizing cognitive effort [Shenhav et al., 2013, Zenon et al., 2019]. More precisely, it has been shown that humans rely on a mixture of MB and MF RL processes when facing contexts requiring repeated decisions [Daw et al., 2011, Lee et al., 2014, Viejo et al., 2015]. They are moreover able to recognize the degrees of stability and familiarity of a given task to decide when to shift between these two behavioral modes. Importantly, these human cognitive abilities have recently been modeled with the deep reinforcement learning framework [Wang et al., 2018]. However, these approaches still rely on task-specific parameterization and computationally heavy pretraining, and do not explicitly address genericity nor cost reduction.

The idea of taking inspiration from how the brain coordinates multiple learning systems to enable more flexibility in robots has received increased attention in the robotics community during the last couple of decades [Girard et al., 2005, Meyer and Guillot, 2008, Caluwaerts et al., 2012b, Zambelli and Demiris, 2016, Banquet et al., 2016, Lowrey et al., 2019]. Furthermore, robot cognitive architectures combining both MB and MF learning processes have started to be studied in recent years [Caluwaerts et al., 2012b, Jauffret et al., 2013, Renaudo et al., 2014, 2015b, Llofriu et al., 2015, Maffei et al., 2015, Chatila et al., 2018, Sheikhnezhad Fard and Trappenberg, 2019, Hafez et al., 2019, Rojas-Castro et al., 2020, Hangl et al., 2020]. Among these proposals, we have previously proposed a way to implement these principles within a classical three-layered robot cognitive architecture, to facilitate integration with other sensing and control components, as well as to permit future transfer to different robotic platforms [Renaudo et al., 2015c]. Nevertheless, to our knowledge, none of these recent projects have studied (1) the extent to which combining MB and MF RL can provide *behavioral flexibility* and simultaneously *reduce computational cost*, by enabling robots to autonomously determine when to avoid the high cost of MB planning when an MF strategy is considered sufficient; and (2) the extent to which such a multi-strategy architecture is effective in a variety of tasks, including social and non-social ones, and thus can be generalized to different scenarios and situations.

Here, we present a novel robot reinforcement learning architecture which display behavioral flexibility by dynamically shifting between MB and MF RL through the arbitration of a trade-off between performance and computation cost. We test the new algorithm during simulated and real robot experiments, and test its generalizability without parameter re-tuning in three different scenarios: a navigation task involving paths of different lengths to the goal, dead-ends, and non-stationarity; a human-robot interaction task where the robot learns to put objects in the rights containers under human teaching signals; a human-robot cooperation task where both human and robot have to hand-over some objects to the other agent in order to put them in their respective containers. We find that the proposed architecture flexibly and consistently switches to MB control after environmental changes in any of the three scenarios. It moreover efficiently switches to MF control when the task is recognized as stationary. Overall, the robot achieves the same performance as optimal MB control in the three scenarios, while dividing computation time by more than two.

Part of the results in the navigation scenario (Experiment 1), those with change in reward location, but not those with change in the wall configuration, have been published in a conference paper [Dromnelle et al., 2020b]. Part of the results in the HRI scenario (Experiment 2) have been published in a second conference paper [Dromnelle et al., 2020a]. We present new unpublished results in both experiments, new extended analyses of the properties of the robotic architecture which explain these results, and a thorougher description of the methods. Experiment 3 is completely new.

In summary, we propose an original and efficient human-inspired mechanism for the coordination of robot learning systems in a variety of scenarios. To our knowledge, this is the first robotic implementation of a hybrid MB/MF algorithm that efficiently reduces computation cost while maintaining performance, and which can cope with human behavioral variability during HRI. This feature can be a key advantage from an ecological point of view and for robots that can only rely only on their limited internal computational and energetic resources to achieve their objectives.

2 Material and Methods

2.1 Markov Decision Problem

In the three scenarios considered in this work, we systematically consider the robot as an RL agent facing a Markov decision problem (MDP) [Sutton and Barto, 1998]. This means that the robot will experience a series of discrete states $s \in \mathcal{S}$, choosing what to do at each iteration t (*i.e.,* timestep) within a finite set of discrete actions $a \in \mathcal{A}$, with the goal of maximizing the sum of cumulative reward $r \in \mathbb{R}$ over a potentially infinite horizon (the robot does not know in advance how long the task will last): $f(t) = \sum_{t=0}^{\infty} \gamma^t r_t$ with $0 \leq \gamma \leq 1$.

The MDP can be described by the n-uplet (S, A, T, R, γ) where $T : (\mathcal{S}, \mathcal{A}) \rightarrow \mathcal{S}$ is the transition function, which represents the probability $P(s'|s, a)$ of arriving in state s' after executing action a in state s, and $R : \mathcal{S} \rightarrow \mathbb{R}$ is the reward function, which represents the scalar reward r that the robot can get after reaching state s'.

It is important to note that using a discrete state space does not necessarily mean that the human designer always has to pre-define in advance the decomposition of the task into discrete states. As we will see in the navigation scenario (Experiment 1), we propose a method for the autonomous decomposition of states from the data acquired through a Simultaneous Localization and Mapping Algorithm (SLAM, Grisetti et al. [2007]) by the real robot during initial random navigation within the environment. In that case, the states will represent unique locations in space, and the actions allowed to the robot represent moves in eight cardinal directions: north, north-east, east, etc. In the Human-Robot Interaction (HRI) scenarios (Experiments 2 and 3), the states will represent the configuration of cubes on a table and the possible actions will be: pick a cube, place a cube in a container, hand-over a cube to the human, take the cube that the human is handing over. Moreover, we will present our method for the robot to autonomously learn a world model from the data it collects during initial exploration, this model consisting in the estimations $\hat{T}$ and $\hat{R}$ of the transition and reward functions T and R, respectively. The robot will then use this learned world model to perform mental simulations through Dynamic Programming [Sutton and Barto, 1998], and hence bootstrap learning within a few hundreds of iterations, thanks to such an MB strategy.

The rationale here for using discrete state and action spaces, and addressing them with a hybrid MB/MF learning strategy, is to test in a robot the performance, computational cost and generalizability of a human-inspired model. We thus want to evaluate to which extent it enables robot fast adaptation and quick (in the order of thousands of iterations) reaching of an optimal performance at a low computational cost, inspired by human ability to quickly adapt in new situations. This human ability is currently thought to rely on the combination of MB and MF RL applied to such discrete representations of the task at hand [Daw et al., 2011, Lee et al., 2014, Viejo et al., 2015]. In contrast, current deep RL methods are computationally heavy and cannot achieve an optimal performance in these simple tasks within a few thousands of iterations (we will even show cases of adaptations to task changes within a few hundreds of iterations), but rather require millions of iterations [Wang et al., 2019]. We will illustrate in the navigation scenario that at the end of the experiment, after the robot has performed 6400 actions, that a Deep Q-Network (DQN) [Mnih et al., 2015] barely had time to slightly improve its performance, compared to the other tested algorithms.

2.2 A robot cognitive architecture with a dual decision-making process

The present work implements a classical three-layer robot cognitive architecture [Gat, 1998, Alami et al., 1998] composed of a decision, an executive and a functional layer. The decision layer of the proposed architecture (Fig. 1) is composed of two competing experts which generate action propositions, each with its own method and with its own advantages and disadvantages. These two experts are directly inspired by current computational neuroscience models which combine MB and MF RL strategies for navigation [Khamassi and Humphries, 2012], and more generally for decision-making tasks [Daw et al., 2005, 2011]. Hereafter, we follow the decomposition of the computations of each

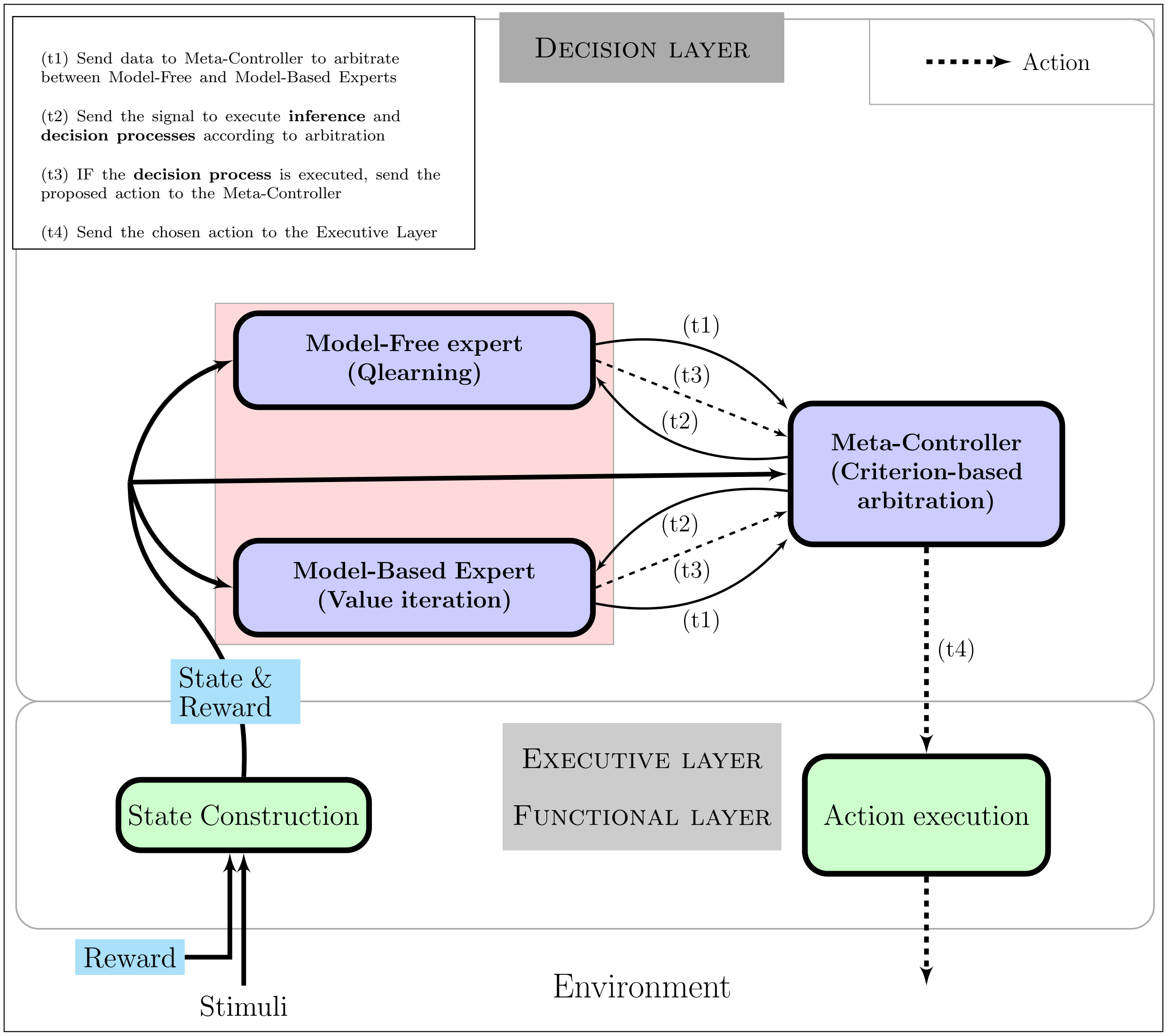

Figure 1: General structure of the architecture. Two experts having different properties are computing the next action to do in the current state s. They each send monitoring data to the meta-controller (MC) about their learning status and inference process (t1). The MC chooses an expert according to a criterion that uses this data and authorizes it to carry out its inference and decision processes (t2). After the decision, the chosen expert sends its proposition to the MC (t3), which sends the action to the Executive Layer (t4). The effect of the executed action generates a new perception, transformed into an abstract Markovian state, and eventually a non null reward r, that are sent to the experts. Each expert learns according to the action chosen by the MC, the new state reached and the reward. Figure by Dromnelle, Renaudo, Khamassi and Girard (2022); available under a CC-BY4.0 licence (https://doi.org/10.6084/m9.figshare.21031723).

expert into three processes, namely learning, inference and decision [Cazé et al., 2018], in order to more clearly identify what is the respective computational cost of each of these processes.

The decision layer is also equipped with a meta-controller (MC) in charge of arbitrating between experts. The MC determines which expert will perform inference and decision steps in the current state, according to an arbitration criterion. After that, the decision layer sends the chosen action to the executive layer, who ensures its accomplishment by recruiting robot's skills from the functional layer. The latter consists of a set of reactive sensorimotor loops that control actuators during interaction with the environment. The robot reaches a new state and obtains or not a reward. The two experts use the new state and the reward information to update their knowledge about the executed action. This allows MB and MF experts to cooperate by learning from each others' decision.

Compared to our previous architecture [Renaudo et al., 2015b], several changes have been made: The overall organization of the decision-making layer and the prioritization of communication between modules have been changed; The MF expert is no longer built as a neural network but as a tabular algorithm; The MC chooses which expert is the most suitable at a given time and in a given state, and no longer simply at a given time; And above all, we have defined a novel arbitration criterion that not only compares experts' performance, but also their estimated computational cost.

2.3 The decision layer

2.3.1 Model-based (MB) expert

The MB expert learns a transition model T and a reward model R of the problem, and uses them to compute the values of actions in each state. These models allow to simulate over several steps the consequences of following a given behavior and to look for desirable states to reach. Consequently, when the task changes, the robot can use this knowledge to find the new relevant behavior with little actual interactions with the world. However, this search process is costly in terms of computation time as it needs to simulate several value iterations [Sutton and Barto, 1998] in each state to find the correct solution.

Learning process. The learning process of the MB consists in updating the reward and the transition models by interacting with the world. The transition model T is learnt by counting occurrences of transitions (s, a, s'). A pretraining phase can take place to improve the robot's transition model before the beginning of task. Nevertheless, the transition model is updated all along the experiment, so that the robot can adapt to task changes.

The transition model T is updated using the number of visits $V_N(s, a)$ of state s and action a. $V_N(s, a)$ has a maximum value of N and $V_N(s, a, s')$ is the number of visits of the transition (s, a, s') in the last N visits of (s, a). The transition probability $T(s, a, s')$ is defined in Eq. 1. This leads to an estimation of the probability to the closest multiple of $1/N$:

$$T(s, a, s') = \frac{V_N(s, a, s')}{V_N(s, a)} \tag{1}$$

The reward model R stores the most recent reward value r_t received for performing action a in state s and reaching the current state s', multiplied by the probability of the transition (s,a,s').

Inference process. Performing the process of inference consists in planning using a tabular Value Iteration algorithm [Sutton and Barto, 1998]:

$$Q(s, a) \leftarrow \sum_{s'} T(s, a, s') \left[R(s') + \gamma max_{k \in \mathcal{A}} Q(s', k)\right] \tag{2}$$

$Q(s, a)$ is the action-value estimated by the agent for performing the action a in the state s, $R(s')$ the probabilistic reward of the reward model R associated with the state (s') and γ the decay rate of future rewards.

Decision process. Performing the decision process consists in converting the estimation of action-values into a distribution of action probabilities using a Boltzmann softmax function, and drawing the action proposal from this distribution. We moreover introduce the possibility of human interventions under the form of a bias $Q_H(s, a)$ representing the human's preferences for action (these will be used for HRI tasks in Experiments 2 and 3, but not in the navigation task of Experiment 1):

$$P(a|s) = \frac{\exp((Q(s, a) + \alpha_H * Q_H(s, a))/\tau)}{\sum_{b \in \mathcal{A}} \exp((Q(s, b) + \alpha_H * Q_H(s, b))/\tau)} \tag{3}$$

where τ is the exploration/exploitation trade-off parameter, and where the human-predicted preference (bias) $Q_H(s, a)$ equals 1 if the human praised the robot the last time it performed the action a in state s, and 0 otherwise. For the sake of parsimony, the weight of the human bias α_H is identical to the learning rate of the robot α.

2.3.2 Model-free (MF) expert

The MF algorithm does not use models of the problem to decide which action to do in each state, but directly learns the state-action associations by caching in each state the earned rewards in the value of each action (action-values). Because updating the action-values is local to the visited state, the learning process is slow and the robot cannot learn the topological relationships between states. Consequently, when the task changes, the robot takes many actions to adopt the new relevant behavior. On the other hand, this method is less expensive in terms of inference duration.

Learning process. Performing the learning process consists in estimating the action-value $Q(s, a)$ using a tabular Q-learning algorithm:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R(s) + \gamma \max_k Q(s', k) - Q(s,a)] \tag{4}$$

where α is the learning rate, $R(s)$ is the scalar reward received for reaching the state s, γ is the decay rate of future rewards (same as γ used by MB in Eq. 2), and s' is the state reached after executing a.

Inference process. Since the MF expert does not use planning, its inference process consists only in reading from the table that contains all the action-values the one that corresponds to performing the action a in the state s.

Decision process. The decision process is the same as for the MB expert (Eq. 3).

2.3.3 Meta-controller and arbitration method.

The MC is in charge of selecting which expert will generate the behavior. For each state s, it computes the entropy of the action probability distribution $H(s, E)$ of expert E [Viejo et al., 2015], which is close to the notion of trust in [Rutard et al., 2020]:

$$H(s,E,t) = -\sum_{a=0}^{|\mathcal{A}|} g(P(a|s,E,t)) \cdot \log_2\left(g(P(a|s,E,t))\right) \tag{5}$$

where $g(P(a|s, E, t))$ is a low-pass filtered action probability distribution, estimated from the past inferences performed by expert E, with time constant $\tau = 0.67$, which has previously been found to reflect the quality of learning in humans [Viejo et al., 2015]. The lower the entropy, the lower the uncertainty of the agent about the action to choose. So the lower the entropy, the higher the quality of learning. The action selection probabilities used to compute the entropy are averaged over time, per state, using an exponential moving average.

For each state, the MC also computes the low-pass filtered duration of the previous inference processes $C_T(s, E, t)$ of expert E, measured in actual simulation time. The novel arbitration criterion that we propose here is a trade-off between the quality of learning and the cost of inference. By using it, the MC can decide between favouring the most certain expert (the most efficient) and the cheapest expert in terms of computations. Note that the inference process of an expert does need to be run before the meta-controller's arbitration since it relies on a low-pass filtered memory of the past costs of each expert in each state. The meta-controller computes the expert-value $Q(s, E)$ for each expert as following:

$$Q(s,E,t) = -\left[H(s,E,t) + \exp(-\kappa H(s,MF,t)) C_T(s,E,t)\right] \tag{6}$$

where the term $\exp(-\kappa H(s, MF, t))$ allows to weight the impact of computation costs in the criterion: The lower the entropy of the MF distribution of action probabilities, the more the computation cost of the inference process weights in the equation. We have chosen the value (here κ = 7) of the weighting of $-H(s, MF, t)$ according to a Pareto front analysis [Powell and Sammut-Bonnici, 2015] (Figure 2, left). We were looking for a κ that minimizes the cost of inference, while maximizing the agent's ability to accumulate reward over time (here we tried to loose less than 1% of the maximum, dashed line on fig. 2, left), in the two non-stationary navigation tasks detailed in the next section. Figure 2, right, illustrates this process by showing the way $\exp(-\kappa H(s, MF, t))$ evolves as a function of the value of the entropy $H(s, MF, t)$ and parameter κ.

Finally, the MC converts the estimation of expert-values $Q(s, E)$ into a distribution of expert probabilities using a softmax function (Eq. 3), and samples the activated expert from this distribution. The inference process of the unchosen expert is inhibited, which thus allows the system to save the corresponding computation time.

2.4 World-model building

In this work, we alternate experiments in simulation and with the real robot. This is to enable the robot to learn a world model of the task in reality, then use this world model for simulations permitting to tune the parameters and evaluate the

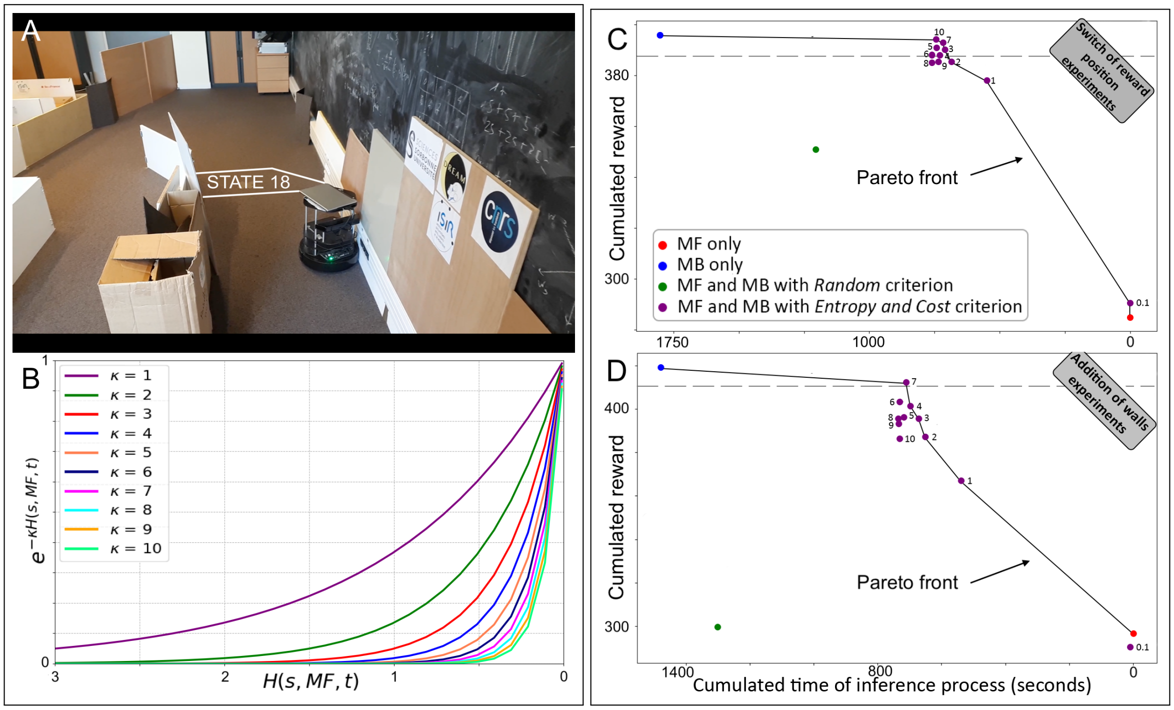

Figure 2: Selection of the value of the κ parameter in simulations of the navigation task (Experiment 1). **A**. Indoor arena used for the navigation task with the real robot. State 18 depicts the initial reward location. The robot learned a discrete map of the environment which was then used for parameter optimization in simulation. **B**. Shape of the $\exp(-\kappa H(s, E, t))$ function for various values of the κ parameter. **C,D**. Cumulated reward and cumulated computational cost obtained with various values of κ (Eq. 6) in the MC-EC architecture (purple), versus the MF-only (red), MB-only (blue) and MC-Rnd (green) controls. The dashed line represents 0.99% of the maximal cumulated reward measured. The analysis was performed on data collected in the two non-stationary navigation scenarios (top: displace reward scenario; bottom: added wall scenario). Figure by Dromnelle, Renaudo, Khamassi and Girard (2022); available under a CC-BY4.0 licence (https://doi.org/10.6084/m9.figshare.21031723).

proposed robot cognitive architecture. And finally perform the learning experiments with the real robot under various conditions: Change in the reward function R of the MDP, change in the transition function T of the MDP.

Figure 3 illustrates the method. The robot first learns a world model from real data collected during initial exploration. Then the world model is used as a new approximate but realistic MDP to perform offline simulations. These simulations serve to evaluate the robot cognitive architecture, measure its performance and cost in different conditions, and optimize its parameters in simulation, thus more quickly than with a real robot. Finally, the parameterized architecture can be tested again on the real robot, where MB and MF RL strategies can learn in parallel the new task conditions imposed to the robot.

The method is here illustrated with a navigation scenario, easy to conceptualize and visualize. But it is a generic method which can be used in other scenarios, such as MDPs for HRI with humans.

2.5 General information

Similarly to the Rmax algorithm [Sutton and Barto, 1998], we initialized the action values to non-zero values so to help exploration of non-previously selected actions, since the action values are updated according to the previous ones. Thus, in any non-rewarded states, having previously selected at least one action results in a non-flat action probability distribution, and thus more chances to select another one (exploration). More precisely, the initial action values are set to 1 for both experts.

For the MF expert, we conducted a grid search to find the best parameter-set, *i.e.,* parameters maximizing the total accumulated reward over a fixed duration of 1600 timesteps (which is the duration of the first phase of the navigation phase, before task changes occur). As this expert is very slow to learn compared to the MB expert, it is important to ensure that it can display a beginning of performance improvement within this duration. We found α = 0.6, γ = 0.9 and

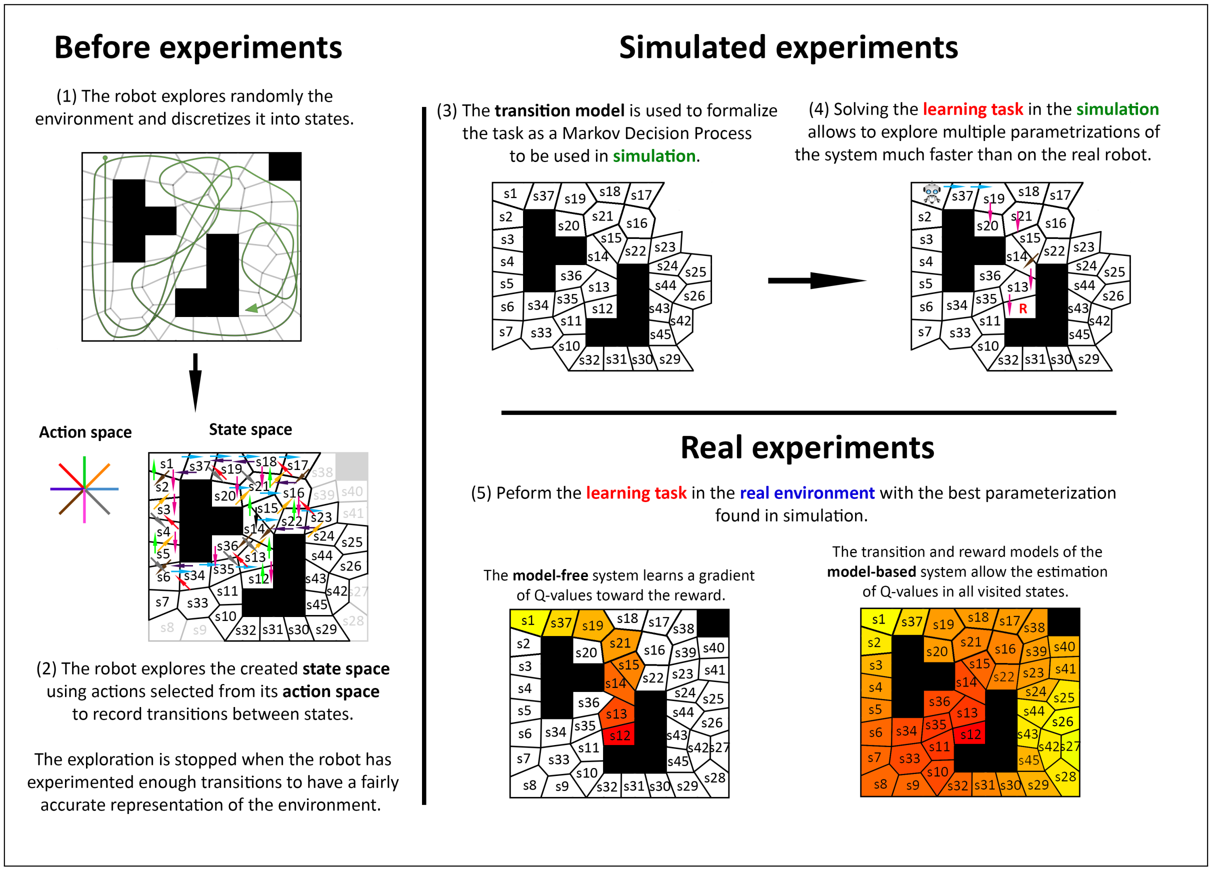

Figure 3: The different phases of the method used for world model building and offline usage. We illustrate the method with a navigation scenario, easy to conceptualize and visualize, but the method is generic and can be used in other scenarios, such as MDPs for HRI. Figure by Dromnelle, Renaudo, Khamassi and Girard (2022); available under a CC-BY4.0 licence (https://doi.org/10.6084/m9.figshare.21031723).

$\tau = 0.02$. For the MB expert, we chose $\gamma = 0.95$. For the MB expert and the MC, we chose the same value of τ as the MF expert. Finally, for the MC, we choose a gating parameter $\kappa = 7$.

3 Experiment 1: Navigation task

The work described in this section presents extended analyses of the results of Dromnelle et al. [2020b], plus unpublished results in a new condition of the task (changes in wall configuration). Finally, we also provide more details about the world model building method, because it will also be used in Experiments 2 and 3. We will refer to Dromnelle et al. [2020b] for previously published results, which can be accessed from: `https://hal.archives-ouvertes.fr/hal-02883717v3/document`.

3.1 Methods

We first evaluated our cognitive architecture in a navigation task. Since running 1600 actions on the robot takes about six hours, we have created a simulation of the task where the probabilities of transitions are derived from a world model learned by the real robot during a 13 hours exploration of the real arena (Section 2.4). This simulation allowed us to quickly test multiple coordination criteria and parameterizations, before evaluating them on a real robot.

We used a 2.6 m x 9.5 m arena containing obstacles (Fig 2A), and a turtlebot. The computer uses ROS [Quigley et al., 2009] to process the signals from its sensors, controls the mobile base and interfaces with our architecture. A Kinect-1 sensor returns an estimate of distance to obstacles in its field of view, completed by contact sensors at the front and sides of the mobile base. The robot localizes itself using the gmapping Simultaneous Localization and Mapping Algorithm (SLAM, [Grisetti et al., 2007]). During a preliminary environmental exploration phase, the robot incrementally builds a discretized map by creating a new nodes every time its minimal distance with all existing nodes is larger than 35 cm, and thus autonomously creating new Markovian states (Fig. 4). The current state (of the corresponding MDP) is the closest node from the robot when its previous action is completed and it evaluates the consequences. We chose to build this map beforehand and to reuse it for each of the learning experiments, so as to reduce the sources of behavioral variability. However, note that with the present method the system could start with an empty map and build it incrementally, and that a new map could be used for each experiment.

In this experiment, the robot must learn to reach a specific state of the environment (state 18 – see Fig. 2A). When it succeeds, it receives a unitary reward and is randomly returned to one of the two initial positions, located in the extremities of the arena (states 0 and 32), to start over. The goal of the robot is first to reach state 18. Thus the reward used here could represent the energy that the robot gets when it reaches its battery recharging station, or it could represent the success for achieving the instruction given by a human to the robot to go to its home base.

Performing an action consists of moving in a certain direction and changing state. The robot can move along 8 equally distributed allocentric directions (Fig. 4). When the contact sensors are activated, the robot moves back 0.15 meters. Finally, according to the exact position in which the robot is located within a state, the arrival state will not necessarily be identical for the same action performed. The environment is therefore probabilistic, which multiplies the possibilities for the robot. For the MB expert, this specificity implies that the transitions $T(s, a, s')$ and the rewards $R(s, a)$ are stored respectively in the model of transition T and the model of reward R as probability distributions.

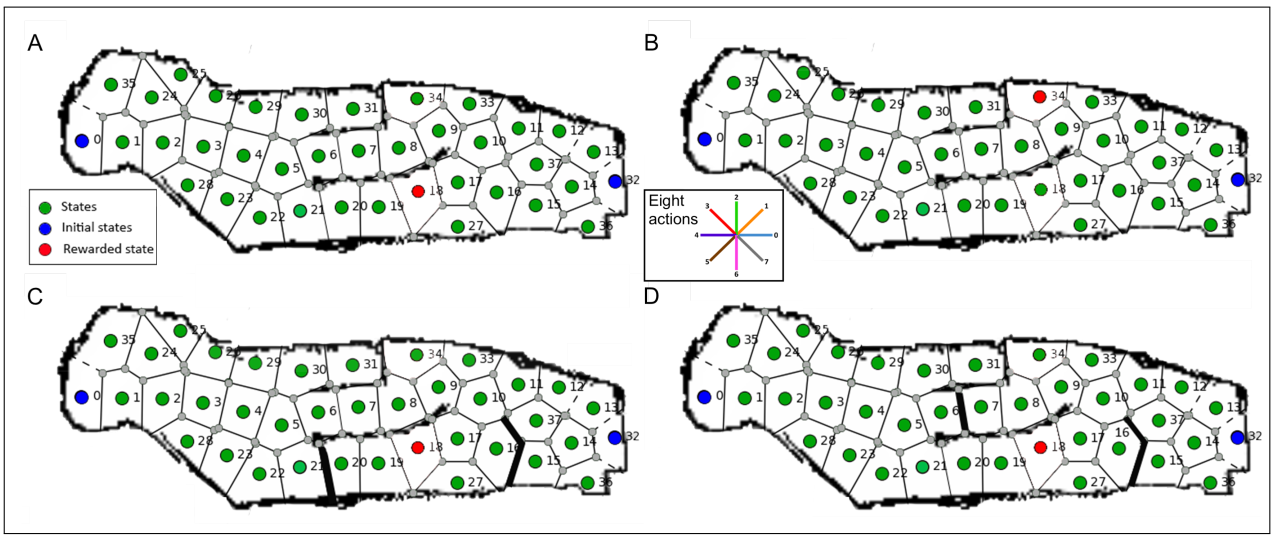

Figure 4: Configurations of the navigation task. **A**. Starting condition: The rewarding state is state #18 (red), the departure states are #0 and #32 (blue), all other states are in green. **B**. Goal-location change condition (after 1600 actions) used in [Dromnelle et al., 2020b]: The reward location is moved to state #34. The inset figure shows the eight actions available to the robot. **C&D**. Wall configuration change conditions (after 1600 actions): Obstacles are added that forbid the transitions between state #16 and states #15 and #37 (C&D), and either between states #20 and #21 (C) or states #6 and #7 (D).

The experiment involves a stable period during which the environment and reward do not change (Fig. 4A). Then, after the 1600th action a task change is imposed where the reward is moved from state 18 to state 34 (Fig. 4B). We also made a second series of experiments where the reward is fixed but some wall configurations are changed in the environment, either in the lower corridor (Fig. 4C) or in the middle corridor (Fig. 4D) depending which of these is preferentially used by the robot, when starting from state 0, so as to maximize the induced perturbation. We chose this duration of 1600 actions (in the order of a few hours with the real robot, as mentioned above), so as to represent a realistic scenario in the context of HRI. In this situation, the human's instructions to the robot may change during the day: the robot may have to complete a task with a specific configuration of the environment in the morning, and then in the afternoon it has to learn a new goal location, or the configuration of the environment changes (*e.g.,* one of the corridors is obstructed while a human is repairing a light in the ceiling). Under these conditions, we cannot afford to use a learning algorithm which requires millions of actions before converging.

To evaluate the performance of the virtual robot, we studied four combinations of experts : (1) a MF-only robot using only the MF expert to decide, (2) an MB-only robot using only the MB expert to decide, (3) a random coordination robot which coordinates the two experts randomly and (4) an Entropy and Cost robot which coordinates the two experts using the model of arbitration presented in 2.3.3. In Dromnelle et al. [2020b], we also compared our algorithm to a reference learning algorithm in the literature, a DQN deep neural network [Mnih et al., 2015], to show that our method outperforms it in terms of cumulated reward with very limited computational cost.

We define the "optimal behaviour" as the behaviour that allows the robot to accumulate the most reward over time.

The navigation task does not involve any human intervention, in contrast to the HRI tasks of Experiments 2 and 3. Thus, all the results of Experiment 1 were obtained with $\alpha_H = 0$ in the robot's decision-making equation through softmax (Eq. (3)).

3.2 Results

Overall, the navigation experiment (Experiment 1) consists of two conditions:

- Condition 1 (simulation + real robot): initial learning followed by changes in goal location (published in Dromnelle et al. [2020b]).
- Condition 2 (simulation + real robot): initial learning followed by changes in wall configuration (unpublished).

We mainly focus on the presentation of the new results in Condition 2, while referring to Dromnelle et al. [2020b] and to the supplementary material to show that the global pattern of the results is similar between the two conditions. We moreover show replications of the simulated results in the real environment with a Turtlebot.

3.2.1 Trade-off between learning flexibility and computational cost

The first important result that we illustrate here with the wall configuration change condition (Fig. 5A,B) is that the MB and MF expert show complementarity in the trade-off between learning flexibility and computational cost:

- The MF-only robot (red) takes longer to reach the optimal behaviour during initial learning, is even slower to adapt to the task change after the 1600th action (Fig. 5A), but achieves this performance at a negligible computational cost (Fig. 5B). This is because its inference process simply consists in reading from the table that contains all the actions-values.
- In contrast, the MB-only robot (blue) has the best performance (Fig. 5A), but also the highest computational cost due to the planning process (about 1000 times higher than the MF-only robot) (Fig. 5B).

The Entropy and Cost (EC) robot (purple), which combines MB and MF experts through the meta-controller proposed in the present cognitive architecture (Fig. 1), manages to reach a non-significantly different performance from the MB-only robot (Mann-Whitney test, df = 1, p = 0.171), showing that our coordination method does not penalize the robot in terms of cumulated reward. This good performance is obtained despite the fact that the EC robot chooses the MF strategy more than 50% of the time after the 800th action (Fig. 5.C). This means that the MF strategy in the EC robot has learned faster than in the MF-only robot, taking advantage of the demonstrations provided by the MB expert. The activation of the MB expert is thus limited, which drastically reduces the computation cost (more than two times smaller than the MB-only robot at the end of the experiment, Fig. 5B). In addition, the EC robot performs better than the random coordination robot (green) suggesting that our coordination method is more efficient than randomly alternating between MB and MF control.

Thus in this task, the proposed architecture enables to benefit from the high learning flexibility of the MB-RL expert, with a limited computational cost thanks to the cheap MF-RL expert. These results replicate what we previously obtained in the change in goal location condition [Dromnelle et al., 2020b], and show similar properties when tested in the real robot (Online Resource Suppl. Fig. S4).

3.2.2 Emergent temporal pattern of expert selection

The second important result is the consistent temporal pattern of expert selection that emerges from the meta-controller's expert selection rule (Equation 6). This pattern was observed (1) in the change in goal location condition [Dromnelle et al., 2020b], (2) in the simulated version of the change in wall configuration condition (Fig. 5.C), and (3) in the version with the real robot (Fig. 5.D), thus showing the robustness of the pattern. This pattern consists in:

- **The MF exploring phase (1 on Fig. 5.C):** Before the discovery of the position of the reward, the robot uses mainly the MF expert. This is due to the difference in the method for updating action-values between the two experts. With the same initial values and the set of parameters we have defined, the action-values of the MF expert decrease slightly more than those of the MB expert, which drives a more pronounced decrease of

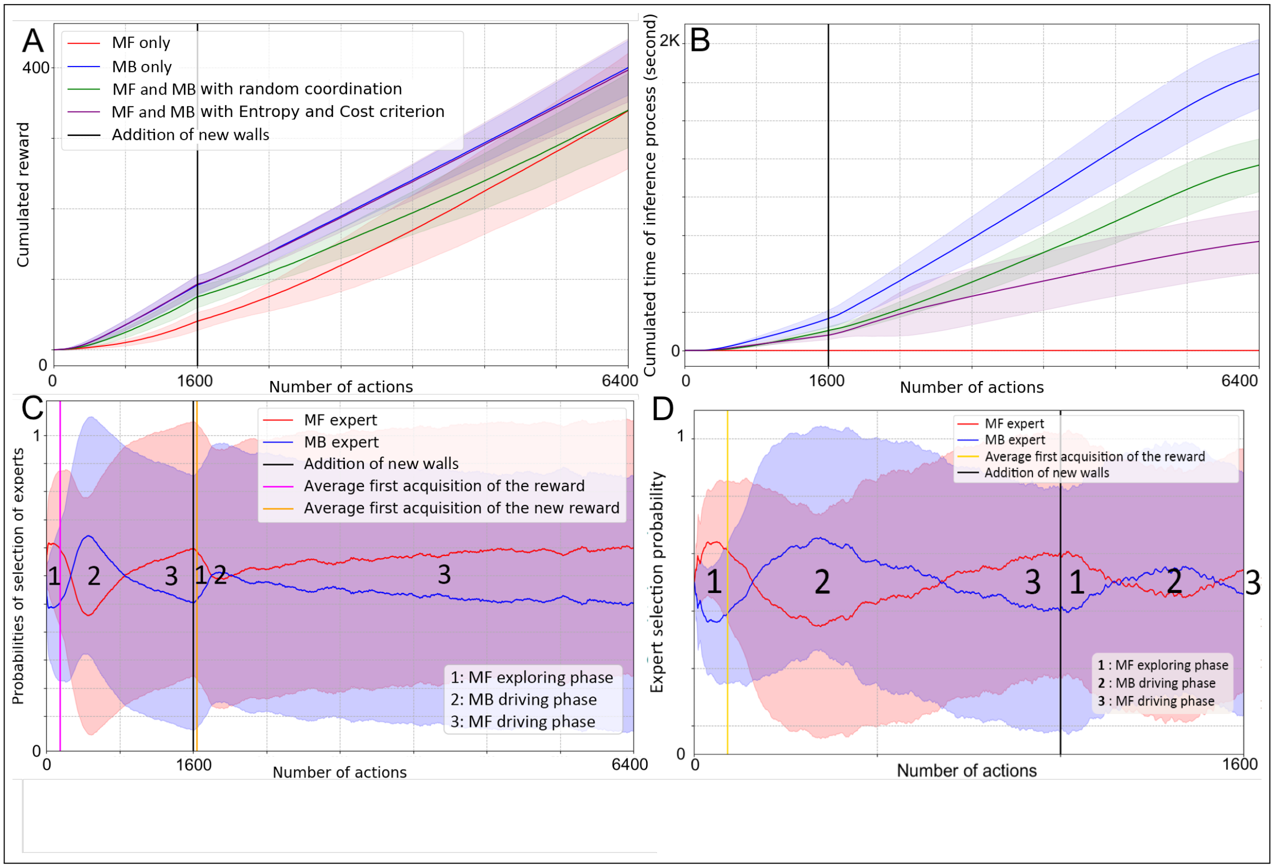

Figure 5: Simulation results of the wall configuration change condition of the navigation experiment: **A**. Mean performance for 100 simulated runs of the task. The performance is measured as the cumulative reward obtained over the duration of the experiment. The duration is represented as the number of actions performed by the robot. We use standard deviation as dispersion indicator. At the 1600th action, new walls are introduced in the arena, as illustrated in Fig. 4C-D. **B**. Mean computational cost for 100 simulated runs of the task. The computational cost is measured as the cumulative time of the inference process over the duration of the experiment in seconds. The duration is represented as the number of actions performed by the robot. **C**. Mean probabilities of selection of experts by the MC using the Entropy and Cost criterion for 100 simulated runs of the task. These probabilities are defined by the softmax function of each expert. The duration is represented as the number of actions performed by the robot. We use standard deviation as dispersion indicator. **D**. Mean probabilities of selection of experts by the MC-EC robot for 10 runs of the wall configuration change task with the real robot.

the entropy of the action probability distribution. In addition, since we do not have an expert specialized in exploration, it makes sense to use the computationally cheapest expert until the position of the reward has been discovered.

- **The MB driving phase (2 on Fig. 5.C):** After finding the first reward the MB expert progressively takes the lead on the decisions because its inference process needs only to find the reward once to spread action-values to all states of the environment thanks to its transition model. It can thus find the reward more easily than the MF expert, and so, its performance increases.

- **The MF driving phase (3 on Fig. 5.C):** The MF expert learns by demonstration from the MB expert, and thus spreads action-values from state to state and eventually, towards the 800th action, it reaches the performance of the MB expert. Because the MF expert is less expensive, the arbitration criterion (Eq. 6) gives it the lead over decisions.

- Interestingly, when a change in the task occurs (At the 1600th action on Fig. 5.C), the sequence of three phases appears again.

The large standard deviation shown in the figures is explained by the fact that for each experiment, the robot's strategy and behaviour can be very different, notably due to the large number of states and possible actions, but also to the probabilistic nature of the environment. As a result, the time of the switches from one phase to another varied a lot from one individual to another. Nevertheless the individual behavior of each run is consistent with the average behavior presented here (Online Resource Suppl. Fig. S1.B). Importantly, experiments with the real robot replicated the expert selection pattern obtained in simulation (Fig. 5D).

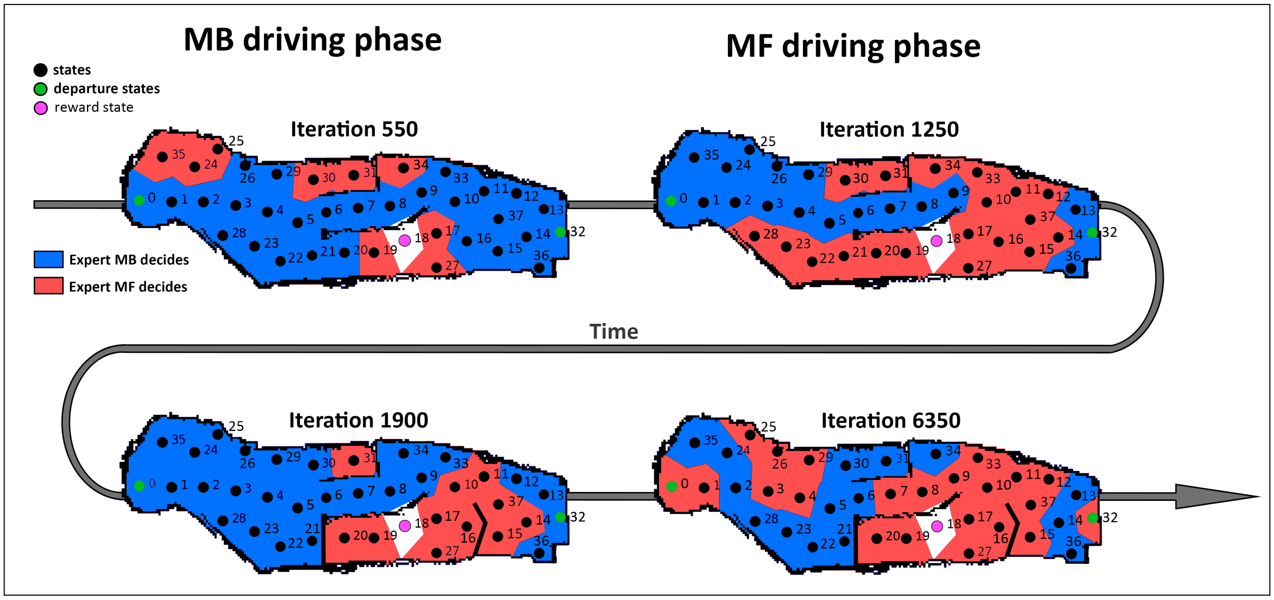

Figure 6: Evolution of the expert spatial preferences in the wall configuration change condition of the navigation experiment. Expert selection maps of the MC-EC robot for one of the hundred simulations: in red, states where the MF was the last chosen expert, in blue, where the MB was last chosen. after 1600 actions, new walls are introduced that, here, forbid the transitions between states between state #16 and states #16 and #37, and between states #20 and #21. The MF driving phase and the MB driving phase correspond to the behavioral phases identified in Fig. 5C.

3.2.3 Spatial pattern of expert selection

The last important result is the spatial pattern of expert selection: The MB and MF selection probabilities reported earlier were not the same in all states of the environment; The meta-controller (MC) turned out to stably prefer the MB expert in specific parts of the environment at different stages of learning, and preferred the MF expert in other parts or at different stages.

Figure 6 illustrates the expert selection map by the MC of the EC robot at different periods of the experiment. These maps show the relative dominance of MB and MF experts over the robot's decisions in different parts of the environment. They enable us to shed a different light on the emergence of the temporal pattern of expert selection reported in the previous subsection. During the MB driving phase, the map is mainly colored in blue, indicating a dominance of MB decisions, while during the MF driving phase, it is the opposite and the states are mostly colored in red. Interestingly, we can see with these maps how a spatial coordination pattern of MB and MF experts evolves with time: during the MF driving phase, paths composed of mostly red states start to appear. These paths approximately end up connecting the departure states to the rewarding state, although the states at the extremities of this path (states 0, 1, 2 and 32) are still preferentially controlled by the MB expert at the 1250th iteration in the example shown in Fig. 6. After the 1600th action, where a change in the wall configuration along the south corridor occurs in the example shown in the figure, the extremities of the red path vanish progressively, before re-forming themselves along the central corridor. This illustrates the new preference of the robot for the central corridor instead of the south one, because it is now the optimal path to the reward.

This leads to the distinction between two types of states: (1) states located on the optimal path, where the MF expert is well trained, and where the robot often goes; (2) states located at the border of the optimal path, where the MF expert received little training, and thus where the MB expert remains dominant. Because the robot does not often go outside the optimal paths after learning, the MF expert remains the most often selected. Nevertheless, when occasionally the robot gets outside the optimal path, the MC reacts by giving the lead to the MB expert which will bring the robot back on track. This illustrates another important aspect of the behavioral flexibility produced by the architecture, which could contribute in explaining flexibility in humans, while neuroscience experiments usually cannot tell whether the biological "MB expert" is completely deactivated after learning or whether it remains potentially reactive to similar situations. This leads to a model-driven prediction which could be tested with future human experiments: An MB process should guide humans back to their familiar sequence of states and actions, after they got out of their optimal path in a given task.

Similar results were obtained in the change in goal location condition of the task (Online Resource Suppl. Fig. S2). Finally, Online Resource Suppl. Figures S5 and S6 show that the same pattern of spatial coordination of experts that we observed previously in simulation, also emerged over time with the real robot in the two types of experiments. However, one can note that the red paths are less complete than they were in the simulation results. This is a sign of a reality gap [Koos et al., 2012], meaning that the experiments with the real robot were more difficult, which impacted the robot's ability to achieve the task.

Another interesting prediction for neuroscience from these results is that a situation with more difficulty, more volatility and uncertainty, could involve a more intertwined contribution of both MB and MF experts, even after a long training. In such cases, rather than observing a continuous activation, from departure until reward, of a putative MF expert in the brain, one would expect to observe intermittent activations of a putative MB expert along the robot's trajectory.

Overall, the important thing to note is that the proposed robot architecture enables to adapt to different situations (different types of task changes), with different degrees of difficulty and uncertainty (simulation versus reality), with the same principle for expert coordination by the meta-controller. This enables to achieve a performance in these simple navigation tasks which is not different from optimality, at a drastically reduced computational cost.

4 Experiment 2: Human-robot interaction with human as teacher

In this section, we evaluate our robotic architecture and coordination system in a human-robot interaction task. First, we present the simulated task, consisting in putting colored cubes in colored containers on a table. Then we present the two types of simulated humans that we defined to interact with the robot. In the second part, we present the results obtained and show how our coordination system allows the robot, in a task with more states, and without major change in our architecture, to maintain again a high level of performance while decreasing greatly its computational cost, but also to deal with the volatility of human behavior. The work presented in this section is an extended version of the publication Dromnelle et al. [2020a], to which we will refer when mentioning previously published results. The pdf of the publication can be accessed from: `https://hal.archives-ouvertes.fr/hal-02899767v2/document`.

4.1 Material and Methods

4.2 Simulated environment and robot

Unlike Experiment 1, this experiment was performed only in simulation. Here, a robot having at least one mobile arm, a visual sensor and a sound sensor faces a table. On the table, three containers and three cubes of different colors are placed. The robot is able to distinguish the colors of cubes and containers, and to manipulate each of the cubes. On the other side of the table, a human can interact verbally with the robot, but can also take control of the robot's arm. We consider that the robot is able to interpret the very simple human messages consisting in either congratulating it, thus constituting a reward signal for the robot, or telling it to observe human demonstrations, thus constituting an observation of action by the robot. Figure 7 illustrates the experiment.

As for the navigation task, we represent the environment by a model of transitions between Markovian states. The transition model representing the simulated environment is not generated by a robot in the real world, since there is no real experience, but predefined by the experimenter. This model is deterministic: Each action carried out in each state by the robot leads to a single terminal state. It would undoubtedly be more complex if it had been generated by a robot carrying out this task in the real world, as for the navigation task of Experiment 1 (Section 3). Initially, we had planned to carry out the task with real human subjects and a *Baxter* robot, but the various lockdowns and the sanitary conditions in 2020 made us abandon this project and stick to simulations [Feil-Seifer et al., 2020]. Nevertheless, this HRI task model is in a sense already more complex than the navigation environment, as we will see in the next two subsections.

In this HRI task, the robot's objective is to learn how to put each of the cubes, initially placed on the table, in the container of the corresponding color. When this is done, the robot gets a scalar reward, and the cubes are automatically put back on the table. Because real naive humans playing with the robot could have wanted the robot to achieve any possible configuration (*i.e.,* not always simply to put the red cube into the red container, and so on, as required here, but also sometimes to put the red cube into the blue container, the blue one into the green container, etc., or to put all cubes into the red container), the robot will have to learn by trial and error the configuration desired by the human.

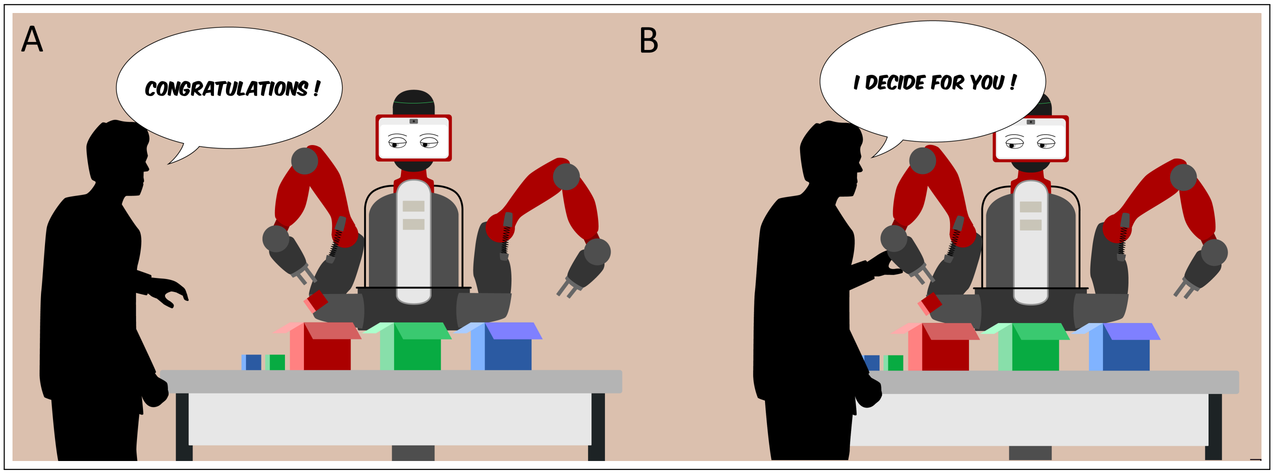

Figure 7: Human-Robot interaction task teaching signals. **A**. Human provides the robot with evaluative feedback (Human intervention type: *Congratulation*). **B**. Human provides the robot with demonstrations (Human intervention type: *Takeover*). Adapted from Dromnelle et al. [2020a], with permission from IEEE.

Importantly, the robot will have to learn this quickly, and to maintain a correct performance throughout the trials, in order to make the duration of the experiment consistent with real human-robot interactions, and to prevent humans from getting bored. Thus, even if the task is simple, we want the robot to quickly achieve an optimal performance at a low computational cost. This is the reason why we are interested in testing whether the same generic robot cognitive architecture can produce human-inspired behavioral flexibility also in this HRI task.

4.3 State and action spaces

As for the navigation experiment, the robot state space is discrete. Here, a state represents the position of the three colored cubes: In the red container, in the green container, in the blue container, on the table, in the robot's hand, or in the human's hand. If we remove the states where the robot and the human hold several cubes at the same time, there remains a total of 112 states, *i.e.,* three times as many states as in the navigation experiment. These 112 states correspond to 5x5x5-13, because the 3 cubes can be put in 5 different positions (hand, table, red container, blue container, green container), from which we subtract the 13 configurations corresponding to the robot's hand having several cubes simultaneously.

Regarding the action space, the robot can perform 7 different actions: Take the red cube, take the green cube, take the blue cube, put the cube held in its hand into the red container, into the green container, into the blue container and onto the table.

While other ways of modeling the task would have been possible, such as with relational RL [Džeroski et al., 2001], we chose this state decomposition for several reasons: To remain in line with the representation used in the navigation experiment; For its ease of use; As a proof of concept of the interest of combining MF and MB learning strategies also in the field of human-robot interaction.

4.4 Pre-experimental babbling phase

A babbling phase precedes the experiment, where the robot can manipulate the cubes without getting rewarded. We defined this pre-experimental phase because in this task, the robot explores its environment much less than in the navigation task (at an equivalent exploration parameter τ), which may have significant repercussions on the performance of the robot. The reasons for this less extensive exploration are as follows:

- The environment of this HRI task is defined by approximately three times as many states as in the navigation task (112 states for the former, 38 for the latter),
- Only 6 actions must be performed from the initial state to reach the final state (*i.e.,* approximately 5% of the total number of states), against 9 in the navigation task (*i.e.,* approximately 24% of the total).

- The environment is not probabilistic, each action performed by the robot in each state of this task leads to a single terminal state. If the probabilistic environment in the navigation task made it more complicated for the robot to traverse, it also allowed it to discover unexplored states by chance.

First, we evaluated the performances of the robot in the HRI task after several babbling durations, using our arbitration criterion (MC-EC) and without human intervention (Fig. 8). We found an optimal babbling duration of 1200 iterations. Beyond that, babbling no longer improved the performance of the robot. Of course, we could also choose to give the robot a more or less complete transition model before the start of the experiment. We consider here the case where the robot has no a priori knowledge about the environment, apart from predefined state and action spaces. In the same way, we could very well imagine that the transition model built by the robot before the first experiment could be reused for all the following experiments. This would be particularly useful in the case of real experiments, where pretraining the robot can accelerate its performance for the next interactions with human participants. Nevertheless, in the present simulations, including a babbling phase enables to estimate how many iterations are required by the robot to learn a correct transition model.

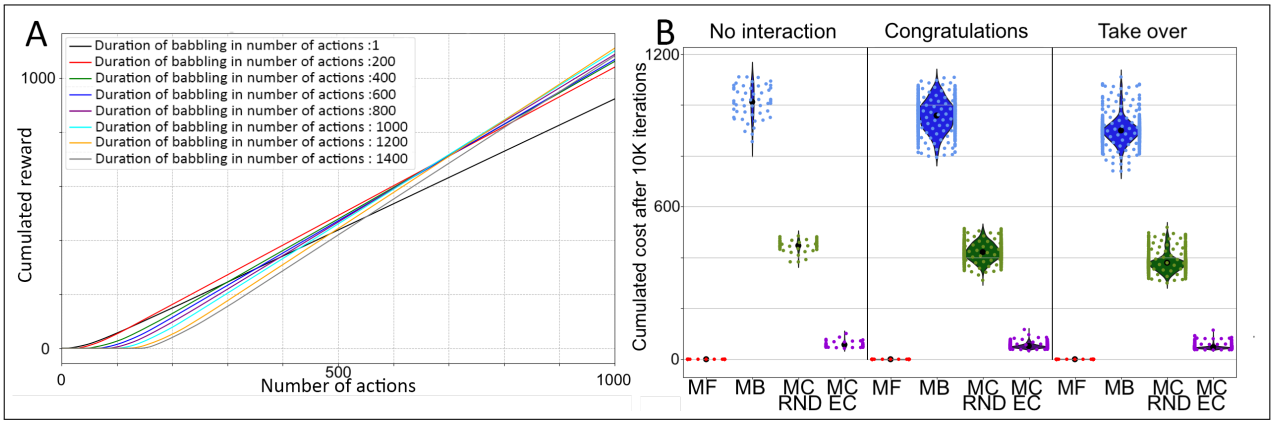

Figure 8: Results in the HRI teaching task. **A**. Average performance of the MC-EC robot for different babbling durations. For each duration, 50 simulated experiments were performed. Performance is defined as the robot's ability to accumulate reward over the duration of the experiment that follows the babbling phase. The duration is represented by the number of actions performed by the robot. **B**. Costs of the inference processes accumulated at the 10000th iteration by the different robots and for the different types of intervention. The colored dots represent the unit performances of the different experiments and the black dots the average performances for all the experiments and all durations of interventions combined, that is to say 600 experiments per type of robot.

4.5 Simulated humans

A simulated human able to interact with the robot faces the table. We have defined two ways for the robot to learn from humans, drawing inspiration from the concepts of *learning by evaluative feedback* and *learning by demonstration* [Knox and Stone, 2009, Judah et al., 2010, Griffith et al., 2013]. We name respectively the two types of underlying interventions: Intervention of the type *congratulation* and intervention of the type *takeover*. More precisely:

- In the case of the *congratulation* type intervention, the human can congratulate the robot after it has put a cube in the correct container, for example the red cube in the red container (Fig. 7A). The effect of the intervention will be effective the next time the robot is again in the same situation (when it holds the red cube again). Knox and Stone [2012] have previously shown that the more human praise directly affects the robot's action selection process, the better the robot. Conversely, the more human praise affects the update of state-action values for each experienced transition, the worse it is. Thus, in our work, we model the human's congratulation, and therefore his/her preference, as a positive bias (a bonus) of an state-action value valid only during the decision process, rather than as a direct modification direct of state-action values. Concretely, we are inspired by the *policy shaping* method named *Action Biasing* [Knox and Stone, 2012], and thus use a non-null parameter α_H to weight the human-predicted preference (bias) $Q_H(s, a)$ in the *softmax* function (Eq. (3)).
- In the case of the *takeover* type intervention, the human can override the choice of the robot, when a cube is held by it, by choosing the place where it will be placed (Fig. 7B). As for the congratulation, the demonstration of the human is associated with a single state-action pair (s_0, a_0). Note that compared to the congratulation,

the demonstration has an instantaneous effect on the robot. And even if it cannot act during these moments, the robot still learns from observing the consequences of the actions chosen by the human.

We note that in both cases, no human intervention memorization process was modeled. By interacting with the robot to influence its decisions, the human biases the updating of its action-state values. Therefore, the consequence of the intervention is incorporated into the robot's state-action value model, which illustrates both the robot's choices and the human's preference, even if it is not possible to separate them.

4.6 Expert parameters

In order to show the generic and task-independent nature of our learning and meta-control system, we reused the same set of parameters as the one used in the navigation task for each of the experts and for the meta-controller (Table 1).

Table 1: Chosen values of experts and meta-controller parameters in the cube ordering task.

Param	MB	MF	MC
α	n.a.	0.6	n.a.
τ	0.02	0.02	0.02
γ	0.9	0.9	n.a.
κ	n.a.	n.a.	7.0

In contrast to the navigation task, the state-action values of the experts are not initialized to a positive value, and are worth 0.0 at the start of the experiment.

4.7 Results of the experiments without human intervention

To evaluate the performance of the simulated robots, we reuse the color code of the navigation experiment: Red for the MF-only robot, blue for the MB-only robot, green for the random coordination robot (MC-Rnd) and purple for the robot that coordinates the two experts using the arbitration criterion that we have proposed (MC-EC).

The interest of this experiment is to evaluate the contribution of meta-control in a task where a robot can interact with a human. We will start by evaluating the performance of the robots without human intervention, then with the two types of human intervention defined above.

In Dromnelle et al. [2020a] we studied the evolution of the average performance of the different robots when the human does not interact with them. As in the navigation experiments, the MF-only robot was the one with the worst performance. Interestingly and in contrast with the navigation experiment, we had observed that the maximum performance was achieved by robots doing meta-control (MC-EC and MC-Rnd) rather than by the MB-only robot. Importantly, the MC-EC robot displayed a much lower computational cost than that of the MC-Rnd robot. Finally, we found that these properties where obtained through a different temporal pattern of expert selection: We observed a very short guidance phase by the MB expert, followed by the guidance phase of the MF expert. Because the state-action values were initialized to 0.0 at the beginning of the experiment, we did not observe the exploratory phase of the MF expert that we observed during the navigation experiment.

These results thus constituted a first step of validation of the genericity of the proposed method in a simple HRI task. In such a case, when the robot has to learn on its own without human intervention, it can be useful to combine MB and MF RL to get an optimal performance while minimizing the computational cost.

4.8 Meta-control provides robustness to errors in humans' teaching signals

Next, we evaluate the architecture when the human intervenes in the form of two possible types of teaching signals: *Congratulations* or *Takeover*. The main messages from the analyses that will be presented hereafter are that:

- The meta-controller of MC-EC robots enables them to get a robust performance in the task independent from whether the human intervenes or not. Only MF-only robots require human intervention to bootstrap their learning performance in this task, while all robots with an MB expert can already learn fast (but note that human interventions are still beneficial in the *Takeover* case, see Fig. S10).
- The meta-controller of MC-EC robots provides them with robustness with respect to errors that humans can make during their interventions (Fig. 9): We tested different percentages of errors made by the humans when congratulating the robot or when taking-over to show the robot was is the right action to perform; We also tested different omission rates in human's teaching signals. The deterioration of performance caused by

omitted (Fig. 9C) or misleading (Fig. 9B) interventions was mostly penalizing the MF-only robot, while being mitigated in the MB-only, MC-Rnd and MC-EC robots, thanks to the MB expert.

- The meta-controller of MC-EC robots minimizes computational cost: Its cost was more than four times lower that of the MC-Rnd, and ten times lower than the one of the MB-only. (Fig. 8B).
- Finally, overall the *takeover* human interventions were more efficient than the *congratulation* ones (compare Online Resource Suppl. Fig. S10 with Online Resource Suppl. Fig. S8), as they allowed to reach larger cumulated reward levels for all the configurations of the architecture (MF-only, MB-only, MC-Rnd and MC-EC). This required 300 iterations in the worse case (MF-only) but was faster for robots incorporating a MB expert (150 interactions). Quite naturally, increasing the number of such interventions increased the cumulated reward up to a ceiling value (Online Resource Suppl. Fig. S10).

In the next subsections, we present more detailed analyses of these results to illustrate the task-independent nature of our coordination model, its generalization to an environment composed of about three times more states than for the navigation task (Section 3), as well as its ability to cope with the volatility of human behavior. Despite these many differences, we reused the same parameters that were optimized for the navigation task, in order to show the generic and task-independent nature of our learning and meta-control system.

4.8.1 Results with human intervention of the *Congratulation* type

Cumulative reward. In Online Resource Suppl. Fig. S8, we can visualize the performance of the different robots at the last iteration (the 10000th) for different durations of human interventions of *Congratulation* type. The human begins to intervene directly after the end of the babbling period. We notice that only the MF-only robot seems to be strongly impacted by human intervention. The other robots have their performance slightly improved for long human interventions, but not for null and short human interventions. A *Krustal-Wallis* test determined that, for the MB-only and MC-Rnd robots, at least some performances for different intervention durations were significantly different (Kruskal-Wallis test, p-value MB-only = 5.66×10^{-5} and p-value MC-Rnd = 0.002). In order to identify which performances were significantly different from the others, we performed multiple comparison procedures through the *Dunn* test [Dunn, 1964] with *Bonferroni corrections* (Online Resource Suppl. Fig. S7). If four performance comparison tests for the MB-only and MC-Rnd robots indeed had a p-value below the significance threshold of 0.05, we note that the effect seems above all to be due to the variability of the data. This is evidenced by the proximity of these p-values to the threshold of 0.05 compared to those of the MF-only robot. For example, for the MB-only robot, the performance relative to the duration of 10 interventions stands out, for no specific reason. Conversely, the effect of the *Congratulation* type intervention on the performance of the MF-only robot had an effect proportional to the duration of the intervention, which makes sense.

We then compared the performance between MF-only, MB-only, MC-EC and MC-Rnd robots. A *Krustal-Wallis* test between the performances of the four robots for an intervention duration of 500 iterations confirms that at least one of the performances was significantly different from the others (p-value = 2.99×10^{-7}). Finally, a *Dunn* test allows us to see that the performance of the MC-EC robot at an intervention time of 500 iterations was significantly different from the performance of the MC-Rnd robots (p-value = 0.0408), MB-only (p-value = 9×10^{-5}) and MF-only (p-value = 3.94×10^{-7}) at the same duration of intervention. The performance of the MC-Rnd robot was also significantly different from the performance of the MF-only robot (p-value = 0.042) while the MF-only and MB-only robots had indistinguishable performances (p-value = 1.0).

For the moment, we have therefore shown that human intervention of the *Congratulation* type seems to be useful only to the MF-only robot, which only embeds a model-free expert. In contrast, only the MC-EC robot achieves maximal performance. Importantly, the MB-only, MC-Rnd and MC-EC robots, which all embed a model-based expert, do not need human intervention to improve their performance. In other words, the interest of the hybrid MB-MF architecture that we propose here is to be more robust to short human teaching interventions, and thus to produce optimal performance in this simple cube tidying task even for cases where real human participants were bored to provide the robot with a long supervision.

Computational cost. Next, we examine the advantages of the proposed architecture in terms of computational cost reduction. Figure 8B allows us to compare the cumulative costs of the inference processes of the different robots at the end of the experiment in the case where the human does not interact with the robot, and in the case where the human congratulates the robot or takes over. Overall, we can say that the help provided by the human seems to slightly offload the robot in computational cost. This is especially observable for the MB-only robot (which in fact performs more expensive computations than the other robots). In any case, the displayed cost of the MC-EC robot is again extremely low compared to those of the MB-only and MC-Rnd robots.

Overall, we can conclude that the MC-EC robot is capable, at minimal cost, of compensating for the absence of human intervention. When the human is present and interacts with the robot, the cost of the MB expert decreases, a sign that it performs less expensive computation. When the duration of the intervention is long, the MF-only robot is fully capable of performing the task efficiently at a very low computational cost. However, as soon as the duration of the intervention decreases, its performance drops. This is when the MB expert behaves like a "backup expert", which allows the robot not to be dependent on the human. In a situation where the presence of the human is uncertain, the MC-EC robot is therefore the ideal robot.

Humans that make omissions. In order to confirm this reasoning, we performed another set of simulations where the simulated humans had a tendency to omit to congratulate the robot from time to time. In other words, the human behavior is now simulated with a certain degree of stochasticity, so that the robot is rewarded by the human only a proportion of the required feedback (from 0%, 10%, .. up to 100% of the time). If omitting has a clear effect on the performance of the MF-only robot (Online Resource Suppl. Fig. S9, first row), bringing it back to the performance of non-intervention, the other robots deal with it without much concern (Online Resource Suppl. Fig. S9, three bottom rows). This is because, as we have previously seen, their performance is already high without intervention, and remains here largely unaffected by the intermittent absence of human feedback.

Humans that make mistakes. Finally, to test the adaptability of these different robots to slightly more realistic humans, we made a last series of simulations where humans could make errors. Within the framework of the *Congratulation* type intervention, an error consists in congratulating a bad action of the robot (for example putting the red cube into the green container). All the system configurations suffer a performance degradation (Fig. 9A), the MF-only configuration is the most affected one. This corroborates our previous observations regarding the dependence of the MF-only robot, and therefore that of the MF expert, on human intervention. Again, using an MB expert is very beneficial for the robot. In all four cases, and even if the performance degradation of the other robots is minimal, we observe that at very high human error rates, the quantity of cumulative rewards at the end of the experiment remains lower than when the human never makes mistakes or never interacts with the robot. This is because during this 500 iterations period of interventions, all the system configurations struggle to accumulate the reward despite human detrimental interventions, which therefore creates a performance delay compared to the robots not interacting with the human or with a human not making mistakes.

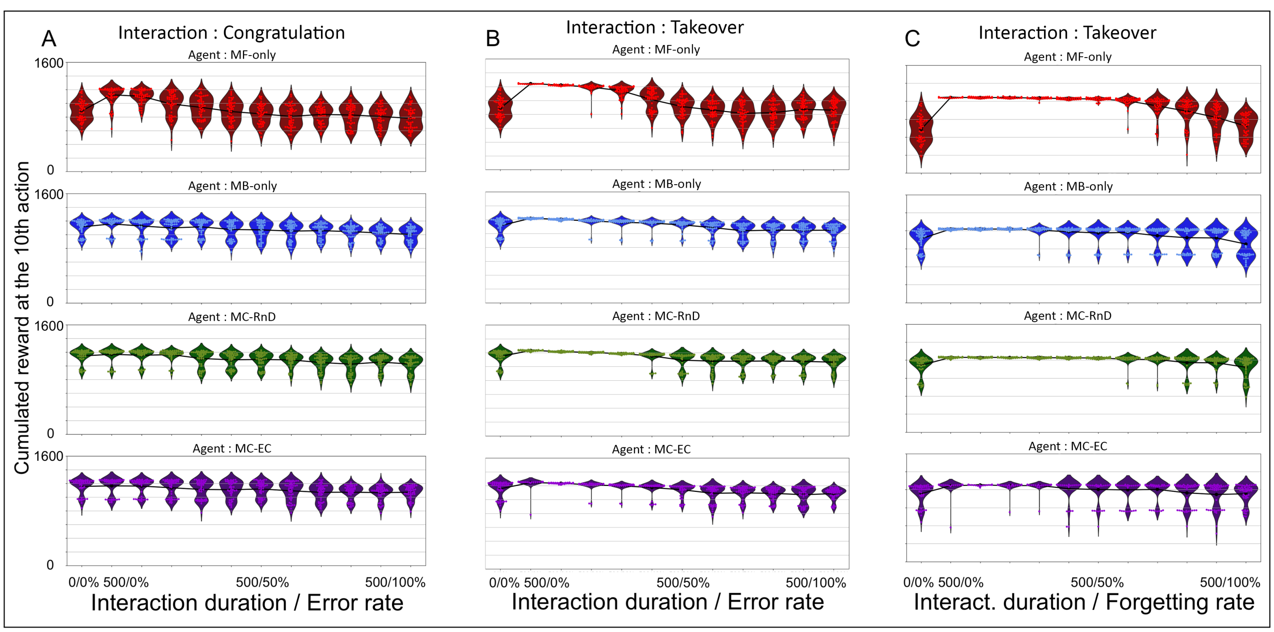

Figure 9: Reward accumulation results in the HRI teaching task. **A**. Case where humans provide erroneous *congratulation* feedback with increasing error rates. **B**. Case where humans provide erroneous *takeover* feedback with increasing error rates. **C**. Case where humans omit to provide *takeover* feedback with increasing omission rates. Dots report the accumulated after 10,000 simulation timesteps, for 50 simulations. First row (red): MF-only robot; second row (blue) MB-only robot; third row (green): MC-Rnd robot; fourth row (purple): MC-EC robot.

Importantly, using our arbitration criterion allows the MC-EC robot not to be dependent on the human to achieve the objective that has been set for it, but also to absorb its potential errors more effectively. In other words, the proposed architecture allows the simulated robot to be more robust to human errors in this task.

4.8.2 Results with human intervention of the *Takeover* type

Unlike the *Congratulation* type intervention, we can see in Online Resource Suppl. Fig. S10 that the *Takeover* type intervention has an effect on the performance of each robot, although the performance effect on the MF-only robot remains larger. For the other three robots, we can see that intervening over a period of more than 100 iterations no longer significantly increases performance. A *Krustal-Wallis* test between the performances of the four robots for an intervention duration of 500 iterations confirms that at least one of the performances is significantly different from the others (p-value = 6.10×10^{-35}). A *Dunn* test finds that at an intervention time of 500 iterations the performance of the MC-EC robot is significantly different from the performance of the MC-Rnd robot (p-value = 5.84×10^{-16}), MB-only (p-value = 1.73×10^{-32}) and MF-only (p-value = 2.50×10^{-04}). The performance of the MC-Rnd robot is also significantly different from the performance of the MF-only (p-value = 1.53×10^{-04}) and MB-only (p-value = 1.25×10^{-03}), which both also have a significantly different performance (p-value = 1.44×10^{-04}). These performances exceed on average the 1200 accumulated rewards, *i.e.,* more than the maximum performances obtained by the robots within the framework of the *Congratulation* type intervention (Online Resource Suppl. Fig. S8). In summary, all robots have different performances, and again, the MC-EC robot is the best of all.

We can explain the high performance of the *Takeover* type intervention by the fact that the decision of the human replaces that of the robot in 100% of cases, whereas in the case of the intervention of *Congratulation* type, the decision-making process, although biased in favor of the human, is still subject to a probabilistic treatment through the *softmax* function (3), which can at times select a non-optimal action. In addition, the *Takeover* type intervention acts on the behavior of the robot at the iteration on which it is performed, while the *Congratulation* type intervention has an influence on the robot behavior only the next time the robot performs the state-action combination that the human praised.

In Figure 8B, we can see that the cumulative cost values are as low as in the *Congratulation* type intervention: The more efficient the human intervention, the less the MB expert needs to do expensive calculations. Finally, in Dromnelle et al. [2020a] we observed the same guidance phases of the two experts as for the Congratulation and No-intervention cases.

If we observed previously that the robots MB-only, MC-Rnd and MC-EC were not impacted by humans omitting to intervene, because the human did not provide any significant assistance to the robots equipped with an MB expert, things are logically different here since the intervention brings clearer help. Indeed, we can see in Figure 9C that at high omission rates, the performance of all the robots degrades, even if again, the degradation of the performance of the robot MF-only remains much more important. Of the three other robots, the MC-EC robot seems to be the one doing the best when faced with the oversights of its human partner.

Finally, we again put the robots in front of humans making mistakes (Fig.9B). In the context of the *Takeover* type intervention, this means that the human takes control of the robot arm to put the cube in the wrong container, or to remove the cubes from the containers of the right color. Here the results are quite close to those observed in Figure 9A: we observe an overall degradation of the robots' performance, again much more intensive in the case of the MF-only robot. As before, at a very high human error rate, the quantities of cumulative rewards at the end of the experiment are lower than these same quantities when the human never interacts with the robots. This is due to the performance lag accumulated during the 500 iterations of erroneous interventions.

With our arbitration criterion, the robot benefits from the human performing a *Takeover* to even better achieve the objective that has been assigned to it, contrarily to *Congratulation* interventions, that are less effective. This superiority of *Takeover* over *Congratulation* has been observed in other studies [Knox et al., 2011]. It is therefore to be preferred. Nevertheless, as with the *Congratulation* type intervention, the combination of MF and MB experts can absorb human errors more effectively.

5 Experiment 3: Human-robot interaction with human as cooperator

In the third experiment, we evaluate our coordination system in a human-robot cooperation task different from the previous one: While in Experiment 2 the robot could learn with or without human intervention, here the robot necessarily needs help from the human. All the following results are previously unpublished.

We first present the new version of the simulated cube storing task, and the way in which we modeled the human partner with whom the robot must now cooperate to achieve its goal. In the second part, we present the results obtained and show that in a situation where the partner can turn into an adversary, our coordination system is no longer able to maintain a high level of performance. To circumvent this problem linked to a natural algorithmic asymmetry between

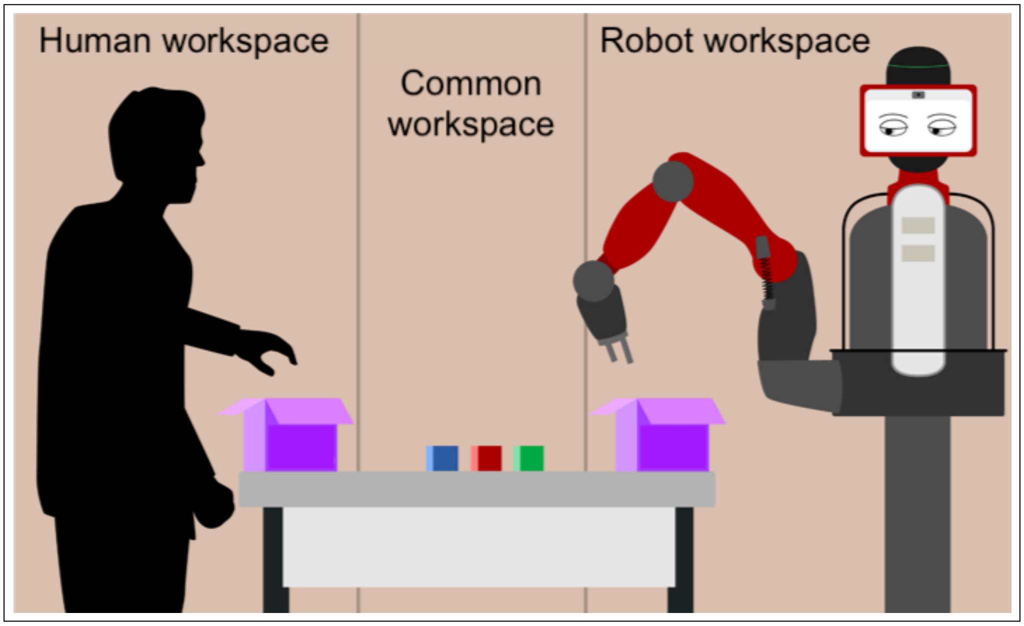

Figure 10: Illustration of the Human-Robot Cooperation task. Figure by Dromnelle, Renaudo, Khamassi and Girard (2022); available under a CC-BY4.0 licence (https://doi.org/10.6084/m9.figshare.21031723).

the MF and MB experts, and not to the human partner, who is only the revealer, we propose an inexpensive solution, under the form of adding a context switching detection mechanism to the robot. With this mechanism, the robot is again able to maintain a high level of performance while still greatly reducing its computational cost.

5.1 Material and methods

5.1.1 Simulated environment and robot

This experiment is also carried out in simulation only. The same robot as the one presented in Experiment 2 faces a table. This time, the table is divided into three distinct spaces: A space accessible to the human only, a common space and a space accessible to the robot only. The human space and the robot space each contain a container, referred to as the human's container and the robot's container. Three colored cubes are available on the table (Fig. 10). This task is inspired by those of Alami et al. [2011] and Renaudo et al. [2015a].

Unlike in Experiment 2, here the robot's first objective is to learn how to put each cube in its own container. When this is done, the robot gets a scalar reward, and the cubes are automatically returned to the human's container. Like in Experiment 1, we make the task non-stationary by introducing a change of objective during the experiment. More precisely, at the 5000th iteration, the robot must now learn to put each cube in the human's container. When this is done, the cubes are automatically returned to the robot's container.

We also test a variant of this experiment with another pair of objectives. The cubes' position has to be swaped: first, the red and the blue start in the robot container and have to be put in the human container, while the green starts in the human container and must end in the robot container; then, the starting position is reversed (red and blue in the human container, green in the robot container) and positions still have to be swaped.

Unlike the task in Experiment 2, where the robot could carry out the experiment without the help of the human, the participation of the human is essential here, since the robot does not have access to the human's side of the table. For this reason, we speak here of *cooperation with humans*, and no longer just of *human intervention*.

5.1.2 Robot state and action spaces

The state space is again a discrete state space. A state always represents the position of the three colored cubes. Each of the cubes can be located: In the human's container, in the common space, in the robot's container, in the human's hand and in the robot's hand. If we remove the states where the robot and the human are holding several cubes at the same time, this represents a total of 99 states, which is 13 less than the task of Experiment 2.

Concerning the action space, the robot can perform 6 classic actions: take the red cube, take the green cube, take the blue cube, place the cube held in hand in its container, place the cube held in hand in the common area, skip its turn. In addition, there are 2 interactive actions, allowing the robot to give the cube held in hand directly to the human (if his hand is empty) or, conversely, to ask the human to give the cube he is holding (if the robot's hand is empty), leading to a total of 8 actions.

As we will see in the next subsection, the human is considered in this experiment as a decision-making agent, and therefore has its own state space equivalent to that of the robot.

5.1.3 Simulated human

In Experiment 2, the human could from time to time interact with the robot. Here, its participation in the task is essential to the success of the robot. To model human behavior, we opted for a version of our MB-only robot with a complete transition model. We consider that if the robot must first learn the consequences of its actions during the babbling phase, the human already knows, for example, that when he takes the red cube from his container, the cube is now located in his hand.

5.2 Pre-experimental babbling phase

A babbling phase, where the robot and the human can manipulate the cubes in the absence of reward precedes the experiment. We chose to add this pre-learning phase for the same reasons as those mentioned in Experiment 2. This time, on the other hand, rather than evaluating the robot's performance using our arbitration criterion (MC-EC) at different babbling durations, we evaluate them at different percentages of transitions explored (Fig. 11). We choose an exploration percentage of 80% (yellow curve) for the first pair of objectives and an exploration percentage of 70% (orange) for the second. These values correspond to those above which continuing to explore no longer allows the reward to accumulate quicker over time. Again, we could choose to give the robot a more or less complete transition model before the start of the experiment or to reuse the transition model built by the robot before the first experiment for all subsequent ones, in the case of real experiences where time is not an unlimited resource.

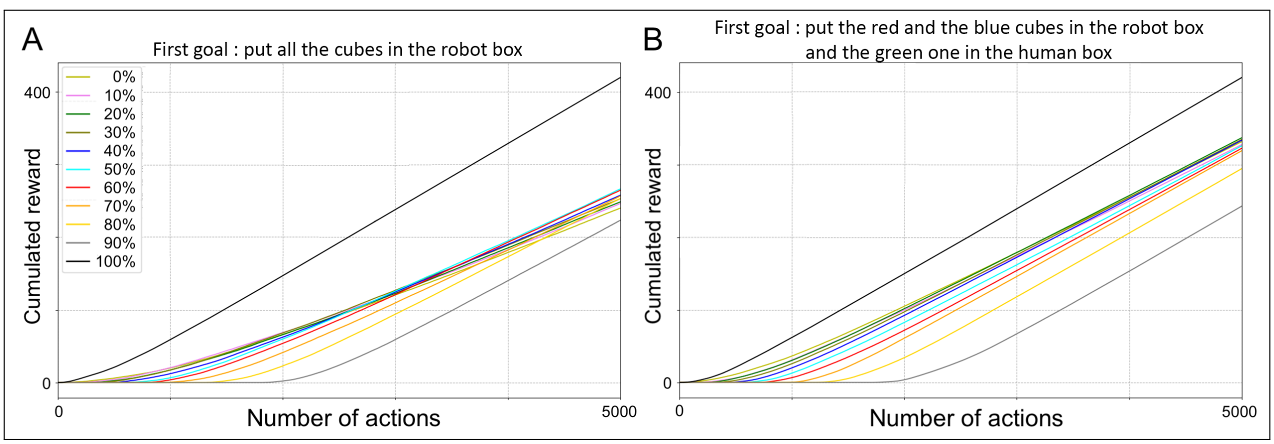

Figure 11: Sizing the babbling phase. **A**. Average performance of 50 simulations of the MC-EC robot for different percentages of transitions explored during the babbling phase and for the first combination of objectives (tidying task). **B**. Average performance of 50 simulations of the MC-EC robot for different percentages of transitions explored during the babbling phase and for the second combination of objectives (swapping task). Performance is defined as the robot's ability to accumulate reward over the duration of the experiment (5000 actions).

5.2.1 Expert parameters

We reuse again the same set of parameters used in the navigation task and the human-robot interaction task for each of the experts and for the meta-controller (Table 2), in order to show the robustness of our learning and meta-control system. The parameters of the simulated human are identical to those of the robots.

The action-state values of the experts and the human are again initialized to 0.0 at the start of the experiment.

Table 2: Selected values of expert and meta-controller parameters in the tidying task in cooperation with a human.

Param	MB	MF	MC
α	n.a.	0.6	n.a.
τ	0.02	0.02	0.02
γ	0.9	0.9	n.a.
κ	n.a.	n.a.	7.0

5.3 Results

To evaluate the performance of simulated robots, we reuse the color code from previous experiments: Red for the MF-only robot, blue for the MB-only robot, green for the random coordination robot (MC-Rnd) and purple for the robot that coordinates the two experts using the arbitration criterion that we have proposed (MC-EC).

The interest of this experiment is to evaluate the contribution of meta-control (expert coordination) in a task where a robot must necessarily cooperate with a human to progress, but also to push our architecture to its limits.

5.3.1 When the partner becomes an adversary

With the first pair of objectives (tidying task) during the first phase of the experiment, the performance of the MC-EC robot again equals that of the MB-only robot (Fig. 12.B), for a computational cost divided by three (Fig. 12.D). Unfortunately, as soon as the objective changes, the MC-EC robot no longer manages to accumulate as many rewards as the MB-alone robot, and is even caught up by the MF-only robot, hitherto considered to be the less efficient. We observed exactly the same tendencies with the second pair of objectives (swapping task, Online Resource Suppl. Fig. S11.A and B). In previous experiments, we had never faced such a drop in performance of the MC-EC robot. To explain it, we need to look at what exactly happens at the 5000th iteration.

For the robot and the human, the 5000th iteration is just another iteration: The objective changes without them being informed. Not knowing that the objective has changed, the two partners will continue to pass the cubes as if nothing had happened. When they finally manage, for example, to put all the cubes in the robot's container (in the case of the first pair of objectives), no reward is issued to them and their R reward models are therefore modified accordingly. Following this, as soon as the inference processes of the MB experts of the MC-EC robot and the human are activated, the state-action values of the MB experts get reset to 0.0 via the natural action of the dynamic programming algorithm *Value Iteration* (Eq. 2).

However, before the 5000th iteration, the behavior of the MC-EC robot is mainly directed by the MF expert (Fig. 12F and Online Resource Suppl. Fig. S11.C), which is not able to reset its action-state values in one go. Indeed, it will take many iterations and passages through the states leading to the rewarded state for the action-state values to decrease following the absence of reward. The problem is therefore the following: after realizing that the objective has changed, the simulated human will go back to exploring the environment in order to find the new rewarded state, or even try to fulfill the new objective if he succeeds. To discover it, while the robot MC-EC, whose behavior is directed at this moment of the experiment mainly by its expert MF, will continue to try to achieve the first objective, resulting in destructive interferences. The robot will, for example, ask the human to give the currently held cube, so as to put it in the robot's container, before the human can put it in its own container, therefore preventing the obtention of reward (and thus the identification of a new goal). On the contrary, the human may manage to put some cubes in his own container, preventing the robot to reach the previously rewarded state, where it would observe the absence of reward, generating large negative reward prediction errors that would start to modify the behavior of his MF expert. Here, the partner turned adversary highlights an algorithmic difference whose effect we had already observed in the navigation task of Experiment 1.

Indeed, this inability of the MF expert to reset his state-action values in the same way as the MB expert was the cause of a "spike" in the selection probability of the MF expert (Fig. 12C) which correlated with the very slight lag in reward accumulation that the MC-EC robot took on the MB-only robot (Fig. 12A). As a reminder, our arbitration criterion is a compromise between the cost of the inference process and the quality of the learning defined as the entropy of the distribution of the probabilities of selection of actions. Concretely, the closer the state-action values of a state are to each other, the greater the entropy will be, and the lower the learning quality will be. When the MB expert resets his state-action values, he also resets his learning quality. The MF expert not being able to do so, he will de facto become the expert with the best learning quality, and therefore the expert controlling the behavior of the robot, whereas the judicious behavior would be precisely to stop playing the first objective.

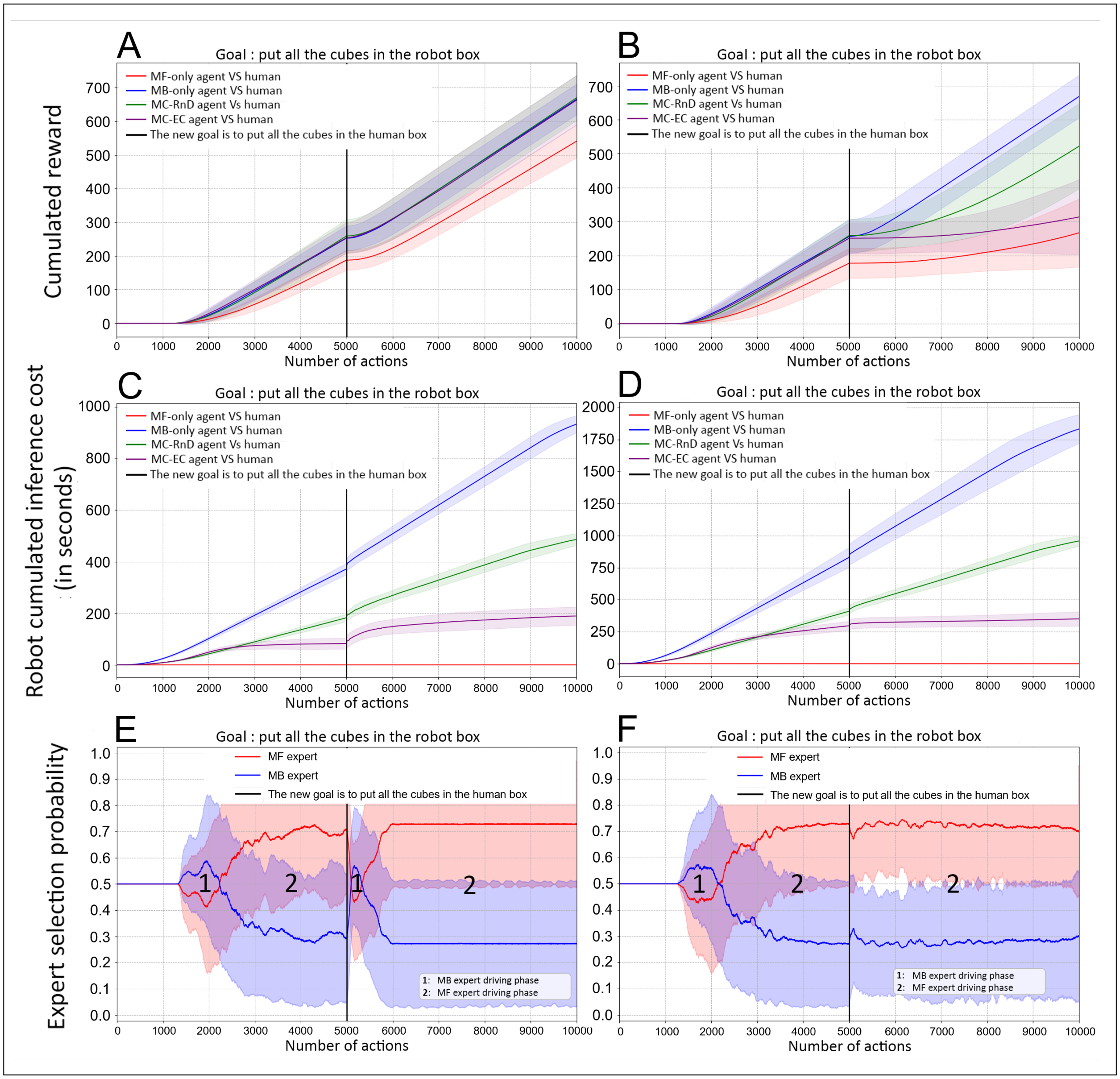

Figure 12: Tidying task results with (A, C,E) or without (B, D, F) a context change detection mechanism: **A,B**. Average performance for 50 simulated experiments. **C,D**. Average computational cost for 50 simulated experiments. **E,F**. Average probability of selection of experts by the meta-controller of the MC-EC robot for 50 simulated experiments. We use standard deviation as an indicator of dispersion in all three figures.

In both experiments, the observation is therefore the same: if the environmental change implies a modification of the reward models of the MB experts, the algorithmic asymmetry of the MF and MB experts gives rise to a period when the MF expert directs the behavior of the robot more than it should. If this did not prevent the robot from maintaining good performance in the navigation task, the MB expert is no longer able to regain control of the robot's behavior here (Fig. 12F and Online Resource Suppl. Fig. S11.C) and therefore remains stuck in MF expert guidance phase 2.

Note that, compared to the navigation task of Experiment 1, we do not observe here the exploratory phase of the MF expert. As a reminder, the existence of this phase was due to the difference in learning methods of the two experts, at the origin of the fact that the state-action values of the expert MF decreased slightly more than those of the expert. MB expert. Here, the state-action values of the experts being initialized at 0.0 at the start of the experiment, and not at 1.0 as in the browsing experiment, this effect of algorithmic asymmetry is not observed.

5.3.2 Context change detection

To counter this problem, we equipped our robot with a mechanism allowing it to automatically detect changes in goals by taking into account only the evolution of its action-state value models. To do this, we relied on the *cosine similarity* to evaluate the similarity of two n-dimensional vectors by determining the cosine of their angle. Generally used as a measure of similarity between two documents, we use it here to measure the similarity between two vectors of state-action values:

$$cos(\theta) = \frac{A.B}{\|A\| \, \|B\|} \tag{7}$$

where A is the state-action value vector of the previous state before it was updated by the MB expert and B is the state-action value vector of the previous state after its update by the MB expert. Note that the vector values have all been multiplied by 100 and the null values have been replaced by very small values to avoid division by 0. If the two vectors are identical, θ is 1.

We already used this measure in [Caluwaerts et al., 2012b], where the *cosine similarity* was computed on vectors containing the Q-values of the MB expert. Concretely, every time the MB expert carries out its inference process, it also computes the *cosine similarity* θ of the Q-value vectors before and after this update, and compares it to a threshold. If θ is lower than this threshold, the MB expert then sends an additional signal to the meta-controller (arrow t1 in Fig. 1), which will take care of sending a signal to the MF expert to request a reset of its Q-values (arrow t2). The value of θ will necessarily decrease due to updates of state-action values in three cases:

- When the robot first finds the reward. In this case, resetting the state-action values of the MF expert is not a problem, since all of them are already null.
- When the robot reaches the previously rewarded state and does not obtain a reward, the moment we are most interested in.
- When the robot first finds the new reward. In this case, resetting the state-action values of the MF expert again is not a problem, since they have all been reset previously.

In the end, more than a mechanism allowing it to automatically detect a change of objective, the *cosine similarity* also allows the robot to detect the appearance of a new objective: it is therefore a mechanism for detecting changes of context, as pointed out by Caluwaerts et al. [2012b]. In our algorithm, when the robot discovers that the rewarded state no longer yields a reward, the action-state values of its expert MF are reset. Instead, we could allow it to store them in memory, so that we can potentially reuse them if the formerly rewarded state becomes rewarded again later in the experience, which is not the case here.

Of course, the functionality of the mechanism depends on the threshold against which the value of the *cosine similarity* θ will be compared. To define it, we looked over 200 simulations at the value of the *cosine similarity* at the iteration following that in which the robot reaches the formerly rewarded state for the first time. θ was in 100% cases less than or equal to 0.611 for the first pair of objectives (100 simulations), and 0.706 for the second (100 simulations). We have chosen a common threshold of 0.7. The histograms of the frequencies of the different values obtained from θ for an experiment of each of the pairs of objectives (Fig. 13) reveal that most of the time, the values of θ are worth 1.0, a sign that during the experiments, the values of the state-action pairs of the MB expert do not evolve much. In the tidying task (Fig. 13A), the values of θ were lower than 0.7 four times (3 of 0.558 and 1 of 0.611), and in the swapping task (Fig. 13B), it happened five times (2 of 0.61 and 3 of 0.666). In both cases, this therefore corresponds to more event than the 3 ones we identified above as being actual context changes (discovery of reward, discovery of the disappearance of the reward, discovery of the new reward). This means that sometimes, the values of state-actions of the expert MB strongly evolve without this being linked to a change of context, but for example rather to the discovery of a new unexplored state or transition. Depending on the defined threshold, the robot can therefore trigger false alarms, mistaking this "brutal" update for a change of context and reset the state-action values of the MF expert when it should not.

However, a these rare false alarms do not seem to have any negative effect on the robot's performance: With the context change detection mechanism and a threshold of 0.7, the performance of the MC-EC robot is now identical to that of the MB-only robot (Fig. 12A). Just after the goal change, the computational cost of the inference process increases (Fig. 12C), a sign that the MB expert takes control of the robot's behavior to enable it to better cope with environmental change. Figure 12E confirms this with the reappearance of the second guidance phases of the MB expert, absent from the experiments carried out without the detection mechanism context changes. Again, these results were replicated with the second pair of objectives (swapping task, Online Resource Suppl. Fig. S12).

5.4 Conclusion

In this last experiment, we evaluated our learning expert coordination model in a simulated human-robot cooperation task where the robot must actively cooperate with the human to achieve its objectives. The human is no longer simply present to help the robot improve its performance, but becomes a real partner. Again, we reused the parameters optimized for the navigation task, in order to show the robustness of our learning and meta-control system.

In this experiment, the robot was confronted with a problem already observed in the navigation task, but which until now did not prevent it from progressing: the inability of the MF expert to reset its action-state values after the change of objective compared to the MB expert. Here, due to the presence of a human not being affected by this problem, the two partners can become adversaries for a time, which leads to a drastic drop in the robot's performance. Here, the human is not the problem, but simply its revelator. To counter this, we have therefore added a mechanism to detect context switches, allowing the robot to automatically reset the state-action values of its MF expert when necessary. With this mechanism, the robot using our arbitration criterion, once again obtains the same level of performance as that of a robot controlled solely by a model-based learning algorithm, while drastically reducing its computational cost (Fig. 12B,D).

Finally, we illustrated again with this human-robot cooperation task the generic and task-independent nature of our coordination model, and an efficient and inexpensive solution allowing it to circumvent a problem that can arise during abrupt changes in the task objectives. These results further highlight the robustness of the proposed method.

6 Discussion

We analyzed the behavior of a three-layered robot cognitive architecture integrating human-inspired mechanisms for the coordination of model-based (MB) and model-free (MF) reinforcement learning modules. Its main novelty lies in the use of the explicit online measure of both performance and computational cost of each system, so as to give control to the system with the best current trade-off between the two. The goal of this approach is to maximize behavioral flexibility, while enforcing computational (and thus energetic) frugality.

Behavioral flexibility was assessed in three main experiments: an indoor navigation task, a HRI task where the human teaches the robot and a HRI task where the human and the robot must cooperate. All these tasks were non-stationary, as an unsignalled change of the goal or of the available transitions, always happened in the course of learning. We kept the parameters of the system identical from one task to another.

Heavy computations consume both time and energy, resources that can be essential for robots: autonomous robots that rely on their sole (and usually limited) computational resources cannot always afford the time required by a complex computation, fast reactions can be necessary in many realistic settings, to avoid damaging the environment or oneself; even when time is not a crucial issue, heavy computations consume energy, a resource that is even more crucial to a mobile robotic platform. Our RL module coordination system is the first one in robotics, to our knowledge, to explicitly take into account the actual computational costs to arbitrate between modules. In computational neuroscience, some earlier models [Keramati et al., 2011, Pezzulo et al., 2013] proposed to evaluate the value of gaining better information from a MB module, versus the cost of performing inference with this MB module, but they were tested in toy problems, with shallow MDPs, with deterministic transitions, and with the model already knowing the transition function. Here we used a more empirical approach, by evaluating the real temporal costs induced by the use of MF and MB learning modules.

The comparison with DQN, made in the navigation experiment, showed that using end-to-end RL has a computational cost not compatible with robotic constraints, and that thus building and using a data representation adapted to the task at hand reduces the burden on the RL part of the system, allowing for low-cost on-the-fly learning. Nevertheless, the

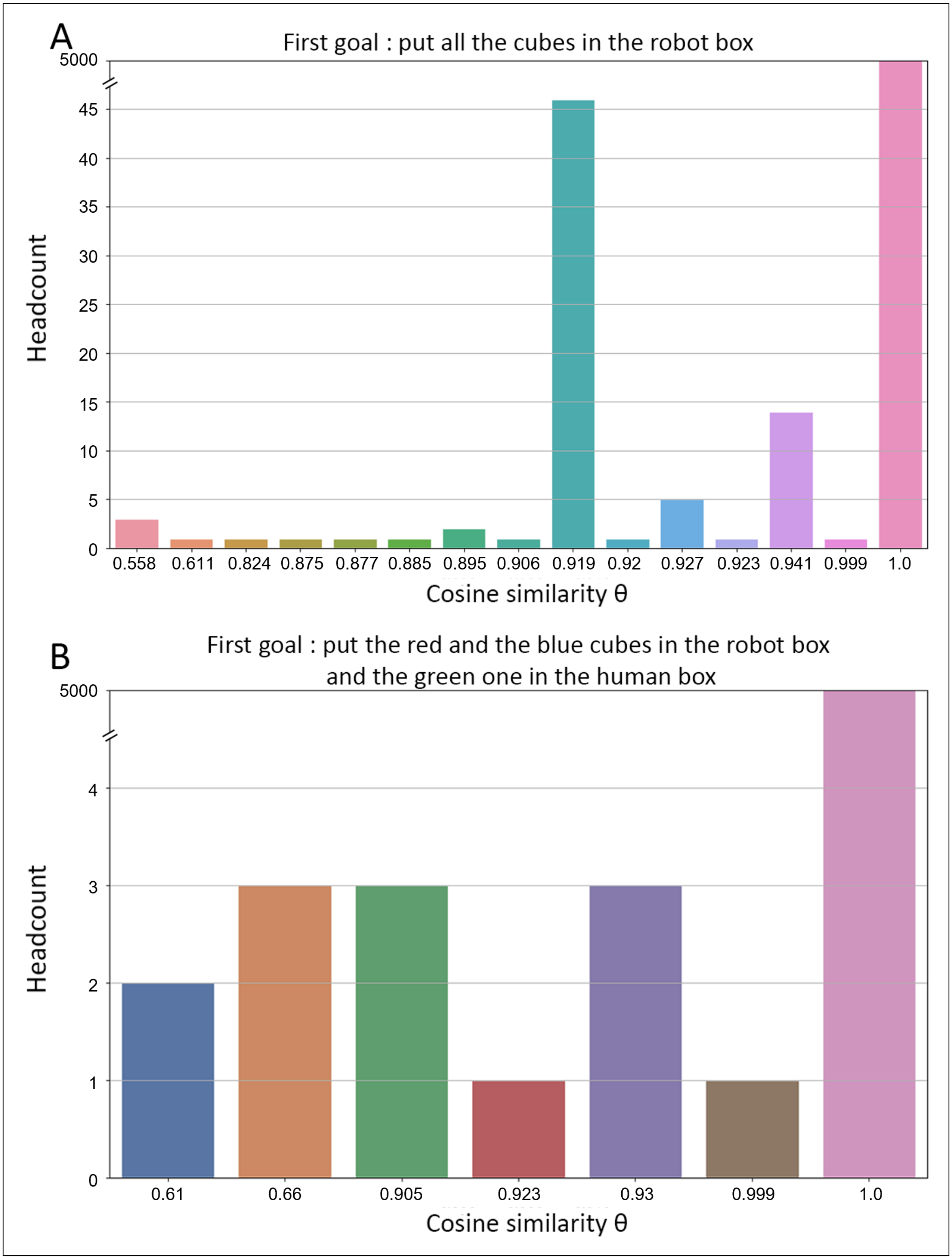

Figure 13: Values taken by the cosine similarity θ, used to parameterize the context change detection threshold. Histograms report the frequency of θ values measured in two 10,000 iteration-long simulations, using: in **A**, the first pair of objectives; in **B**, the second pair of objectives. The robot and the human play on a turn-based basis, so that makes a total of 5000 values of θ per experiment.

discrete state and action spaces used here for RL may partly limit the generality of the method, and prevent it from tackling more complex high dimensional problems. Indeed, as designers of the system, we chose a representation (discretization of the output of a SLAM algorithm) adapted to the problem at hand (a navigation problem). However the context of this proposal is to build on the representation redescription framework [Doncieux et al., 2018, 2020] to ultimately design systems that autonomously determine the representations adapted to the task. The modularity of the present architecture also enables to extend it to the continuous case by replacing tabular value functions with neural network implementations. Nevertheless, there is actually a trade-off between quickly learning an efficient (even if not optimal) solution to coarsely represented or even discretized problem, versus slowing acquiring a more precise and optimal solution using continuous representations and deep function approximators. In particular, humans are able to alternate between contexts in which learning a discrete action plan is sufficient, versus contexts requiring the slower acquisition of more fine-grained plans, especially motor plans like riding a bicycle, learning to play a music instrument, etc [Haruno and Kawato, 2006, Hikosaka et al., 1999]. Thus, rather than having robots tackle any new problem with computationnally heavy deep RL methods, a promising direction for future work could be to add yet another expert to our architecture, composed of a deep network, that the meta-controller will coordinate and compare to the other experts. This way, when the meta-controller detects that a simpler solution is sufficient, it could avoid heavy computation and would both reduce learning time and energy consumption. Moreover, because in our architecture each expert learns from observing what the other experts are doing, initial MB control could bootstrap initial learning and exploration in the deep network composing the new expert.

The arbitration criterion proposed in this work allowed the robots to autonomously determine when to shift between systems during learning, generating coherent temporal decision-making patterns that alternates between strategies over time. This promoted more flexibility than pure MF control in response to task changes, and permitted to reach the same level of performance than pure MB control, while drastically reducing the computational cost. The HRI teaching task revealed an interesting property of our system: Its ability to compensate for the imperfections of the human feedbacks (when they were either omitted or erroneous). This suggests that our method is promising for experiments involving interactions between robot and naive human users. In that case, our architecture can automatically cope with human errors by relying more on its MB component. This enables to avoid redesigning or retuning the robot learning parameters to different situations, and thus make the approach more realistically applicable to real-world HRI.

The meta-controller proposed here often produced a sequence of three behavioral phases with different expert selection patterns: Initial MF-driven exploration, MB-driven decisions once the internal model has included reward information, MF-driven less costly decision-making once the MF expert has been sufficiently trained. Such a pattern is similar to the one observed in humans in an instrumental task [Viejo et al., 2015]. In that task, humans had to learn through trial-and-error to associate different colored stimuli (considered as Markovian states) to different fingers of the hand (considered as actions). After learning and stabilizing these associations (exploitation), the task conditions were changed so that the humans had to learn new associations. Different computational models had been fitted to human subjects' behavior, in order to determine the best model: An MF-only model, and MB-only model, and different ways of coordinating MB and MF. Not only did the authors find that an entropy-based MB-MF coordination model best explained humans' behavior in this task. They also found during subsequent analysis of the model fitted to human behavior that it displayed a sequence of three behavioral phases: Initial quick responses by the humans when exploring (where both MF and MB experts contributed), then an increase in decision time due to the MB contribution, and then a progressive reduction of decision time as the MF increased its contribution. It is thus striking that despite a task difference between humans and robots, and despite the fact that the present entropy-based coordination method has been extended from [Viejo et al., 2015] by adding a cost term, we can still replicate on the robot a similar behavioral pattern than the one experimentally observed in humans.

A system able to detect context changes was added in the last experiment, in order to allow for re-learning when the goal-change occurred. It was inspired by such a system developed in our previous MF-MB coordination system [Caluwaerts et al., 2012a]. Explicitly detecting task changes did not prove necessary in the navigation nor in the teaching task, nevertheless, it should also improve the performance in these two tasks. In future work, we could study to which extent the context change detector produces similar performance in these other tasks, and whether it allows in general to cope with a wider variety of non-stationary tasks.

Acknowledgements

This work was supported by the Délégation Générale de l'Armement (ER, RD), by the Agence Nationale de la Recherche (ANR-12-CORD-0030 Roboergosum Project), by joint funding from ANR and the Austrian Science Fund FWF (ANR-21-CE33-0019-01), by the Centre National de la Recherche Scientifique (INS2I Appel Unique programme; MK), and by the European Union Horizon 2020 research and innovation programme under grant agreement No 761758

"HumanE-AI-Net" (H2020-ICT-48 Network of Centers of Excellence). The authors would like to thank Romain Retureau and Camille Lakhlifi for their help with some of the figures.

Data and code availability

The code related to this work will be made available in an open source repository like github upon publication. The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

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

R. Alami, M. Warnier, J. Guitton, S. Lemaignan, and E. A. Sisbot. When the robot considers the human... In Proceedings of the 15th International Symposium on Robotics Research, 2011.

Jean-Paul Banquet, Souheïl Hanoune, Philippe Gaussier, and Mathias Quoy. From cognitive to habit behavior during navigation, through cortical-basal ganglia loops. In International Conference on Artificial Neural Networks, pages 238–247. Springer, 2016.

K. Caluwaerts, A. Favre-Félix, M. Staffa, S. N'Guyen, C. Grand, B. Girard, and M. Khamassi. Neuro-inspired navigation strategies shifting for robots: Integration of a multiple landmark taxon strategy. In T.J. et al. Prescott, editor, Living Machines 2012, LNAI, volume 7375/2012, pages 62–73. 2012a.

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, 2012b.

Romain Cazé, Mehdi Khamassi, Lise Aubin, and Benoît Girard. Hippocampal replays under the scrutiny of reinforcement learning models. Journal of neurophysiology, 120(6):2877–2896, 2018.

Raja Chatila, Erwan Renaudo, Mihai Andries, Ricardo Omar Chavez-Garcia, Pierre Luce-Vayrac, Raphaël Gottstein, Rachid Alami, Aurélie Clodic, Sandra Devin, Benoît Girard, and Mehdi Khamassi. Toward self-aware robots. Frontiers in Robotic and AI, 5(1):88–108, 2018.

Yevgen Chebotar, Karol Hausman, Marvin Zhang, Gaurav Sukhatme, Stefan Schaal, and Sergey Levine. Combining model-based and model-free updates for trajectory-centric reinforcement learning. In International conference on machine learning, pages 703–711. PMLR, 2017.

Nathaniel D Daw, Samuel J Gershman, Ben Seymour, Peter Dayan, and Raymond J Dolan. Model-based influences on humans' choices and striatal prediction errors. Neuron, 69(6):1204–1215, 2011.

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.

L. Dollé, D. Sheynikhovich, B. Girard, R. Chavarriaga, and A. Guillot. Path planning versus cue responding: a bioinspired model of switching between navigation strategies. Biological Cybernetics, 103(4):299–317, 2010.

Laurent Dollé, Mehdi Khamassi, Benoît Girard, Agnes Guillot, and Ricardo Chavarriaga. Analyzing interactions between navigation strategies using a computational model of action selection. In International Conference on Spatial Cognition, pages 71–86, 2008.

Stephane Doncieux, David Filliat, Natalia Díaz-Rodríguez, Timothy Hospedales, Richard Duro, Alexandre Coninx, Diederik M Roijers, Benoît Girard, Nicolas Perrin, and Olivier Sigaud. Open-ended learning: a conceptual framework based on representational redescription. Frontiers in neurorobotics, page 59, 2018.

Stephane Doncieux, Nicolas Bredeche, Léni Le Goff, Benoît Girard, Alexandre Coninx, Olivier Sigaud, Mehdi Khamassi, Natalia Díaz-Rodríguez, David Filliat, Timothy Hospedales, et al. Dream architecture: a developmental approach to open-ended learning in robotics. arXiv preprint arXiv:2005.06223, 2020.

Rémi Dromnelle, Benoît Girard, Erwan Renaudo, Raja Chatila, and Mehdi Khamassi. Coping with the variability in humans reward during simulated human-robot interactions through the coordination of multiple learning strategies. In 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 612–617. IEEE, 2020a.

Rémi Dromnelle, Erwan Renaudo, Guillaume Pourcel, Raja Chatila, Benoît Girard, and Mehdi Khamassi. How to reduce computation time while sparing performance during robot navigation? a neuro-inspired architecture for autonomous shifting between model-based and model-free learning. In Conference on Biomimetic and Biohybrid Systems, pages 68–79. Springer, 2020b.

OJ Dunn. Multiple comparisons using rank sums technometrics 6: 241–252. Find this article online, 1964.

Sašo Džeroski, Luc De Raedt, and Kurt Driessens. Relational reinforcement learning. Machine learning, 43(1-2):7–52, 2001.

David Feil-Seifer, Kerstin S Haring, Silvia Rossi, Alan R Wagner, and Tom Williams. Where to next? the impact of covid-19 on human-robot interaction research, 2020.

E. Gat. On three-layer architectures. In Artificial Intelligence and Mobile Robots. MIT Press, 1998.

B. Girard, D. Filliat, J-A. Meyer, A. Berthoz, and A. Guillot. Integration of navigation and action selection functionalities in a computational model of cortico-basal ganglia-thalamo-cortical loops. Adaptive Behavior, 13:2, 2005.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. Advances in neural information processing systems, 26, 2013.

G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. Trans. Rob., 23(1):34–46, February 2007. ISSN 1552-3098. doi: 10.1109/TRO.2006.889486.

Muhammad Burhan Hafez, Cornelius Weber, Matthias Kerzel, and Stefan Wermter. Curious meta-controller: Adaptive alternation between model-based and model-free control in deep reinforcement learning. In 2019 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2019.

Simon Hangl, Vedran Dunjko, Hans J Briegel, and Justus Piater. Skill learning by autonomous robotic playing using active learning and exploratory behavior composition. Frontiers in Robotics and AI, 7:42, 2020.

Masahiko Haruno and Mitsuo Kawato. Heterarchical reinforcement-learning model for integration of multiple cortico-striatal loops: fmri examination in stimulus-action-reward association learning. Neural networks, 19(8):1242–1254, 2006.

Okihide Hikosaka, Hiroyuki Nakahara, Miya K Rand, Katsuyuki Sakai, Xiaofeng Lu, Kae Nakamura, Shigehiro Miyachi, and Kenji Doya. Parallel neural networks for learning sequential procedures. Trends in neurosciences, 22 (10):464–471, 1999.

Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. How to train your robot with deep reinforcement learning: lessons we have learned. The International Journal of Robotics Research, 40(4-5): 698–721, 2021.

Adrien Jauffret, Nicolas Cuperlier, Philippe Gaussier, and Philippe Tarroux. From self-assessment to frustration, a small step toward autonomy in robotic navigation. Frontiers in neurorobotics, 7:16, 2013.

Kshitij Judah, Saikat Roy, Alan Fern, and Thomas Dietterich. Reinforcement learning via practice and critique advice. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 24, pages 481–486, 2010.

Daniel Justus, John Brennan, Stephen Bonner, and Andrew Stephen McGough. Predicting the computational cost of deep learning models. In 2018 IEEE international conference on big data (Big Data), pages 3873–3882. IEEE, 2018.

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.

M. Khamassi and M.D. Humphries. Integrating cortico-limbic-basal ganglia architectures for learning model-based and model-free navigation strategies. Frontiers in Behavioral Neuroscience, 6:79, 2012.

Mehdi Khamassi, Charles Wilson, R Rothé, René Quilodran, Peter F Dominey, and Emmanuel Procyk. Meta-learning, cognitive control, and physiological interactions between medial and lateral prefrontal cortex. Neural basis of motivational and cognitive control, pages 351–370, 2011.

Mehdi Khamassi, George Velentzas, Theodore Tsitsimis, and Costas Tzafestas. Robot fast adaptation to changes in human engagement during simulated dynamic social interaction with active exploration in parameterized reinforcement learning. IEEE Transactions on Cognitive and Developmental Systems, 10(4):881–893, 2018.

W Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In Proceedings of the fifth international conference on Knowledge capture, pages 9–16, 2009.

W Bradley Knox and Peter Stone. Reinforcement learning from simultaneous human and mdp reward. In AAMAS, pages 475–482, 2012.

W Bradley Knox, Matthew E Taylor, and Peter Stone. Understanding human teaching modalities in reinforcement learning environments: A preliminary report. In IJCAI 2011 Workshop on Agents Learning Interactively from Human Teachers (ALIHT), 2011.

Jan Kober, Andrew J. Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. IJRR Journal, 32(11): 1238–1274, 2013. doi: 10.1177/0278364913495721.

Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. The transferability approach: Crossing the reality gap in evolutionary robotics. IEEE Transactions on Evolutionary Computation, 17(1):122–145, 2012.

Sang Wan Lee, Shinsuke Shimojo, and John P O'Doherty. Neural computations underlying arbitration between model-based and model-free learning. Neuron, 81(3):687–699, 2014.

Martin Llofriu, Gonzalo Tejera, M Contreras, Tatiana Pelc, Jean-Marc Fellous, and Alfredo Weitzenfeld. Goal-oriented robot navigation learning using a multi-scale space representation. Neural Networks, 72:62–74, 2015.

Kendall Lowrey, Aravind Rajeswaran, Sham Kakade, Emanuel Todorov, and Igor Mordatch. Plan online, learn offline: Efficient learning and exploration via model-based control. In International Conference on Learning Representations, 2019.

Giovanni Maffei, Diogo Santos-Pata, Encarni Marcos, Marti Sánchez-Fibla, and Paul FMJ Verschure. An embodied biologically constrained model of foraging: from classical and operant conditioning to adaptive real-world behavior in dac-x. Neural Networks, 72:88–108, 2015.

J.-A. Meyer and A. Guillot. Biologically-inspired robots. In B. Siciliano and O. Khatib, editors, Handbook of robotics, pages 1395–1422. Springer-Verlag, Berlin, 2008.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, February 2015. ISSN 00280836.

John P O'Doherty, Jeffrey Cockburn, and Wolfgang M Pauli. Learning, reward, and decision making. Annual review of psychology, 68:73–100, 2017.

Giovanni Pezzulo, Francesco Rigoli, and Fabian Chersi. The mixed instrumental controller: Using value of information to combine habitual choice and mental simulation. Frontiers in Psychology, 4(92), 2013. ISSN 1664-1078. doi: 10.3389/fpsyg.2013.00092. URL http://www.frontiersin.org/cognition/10.3389/fpsyg.2013.00092/abstract.

Taman Powell and Tanya Sammut-Bonnici. Pareto Analysis. 01 2015. ISBN 9781118785317. doi: 10.1002/9781118785317.weom120202.

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.

Erwan Renaudo, Benoît Girard, Raja Chatila, and Mehdi Khamassi. Design of a control architecture for habit learning in robots. In Biomimetic and Biohybrid Systems, LNAI Proceedings, pages 249–260, 2014. doi: 10.1007/978-3-319-09435-9_22.

Erwan Renaudo, Sandra Devin, Benoît Girard, Raja Chatila, Rachid Alami, Mehdi Khamassi, and Aurélie Clodic. Learning to interact with humans using goal-directed and habitual behaviors, 2015a.

Erwan Renaudo, Benoît Girard, Raja Chatila, and Mehdi 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 254–260, Providence, RI, USA, 2015b.

Erwan Renaudo, Benoît Girard, Raja Chatila, and Mehdi Khamassi. Respective advantages and disadvantages of model-based and model-free reinforcement learning in a robotics neuro-inspired cognitive architecture. In Biologically Inspired Cognitive Architectures BICA 2015, pages 178–184, Lyon, France, 2015c.

Dalia Marcela Rojas-Castro, Arnaud Revel, and Michel Menard. Rhizome architecture: An adaptive neurobehavioral control architecture for cognitive mobile robots—application in a vision-based indoor robot navigation context. International Journal of Social Robotics, 12(3):659–688, 2020.

Felix Rutard, Olivier Sigaud, and Mohamed Chetouani. Tirl: enriching actor-critic rl with non-expert human teachers and a trust model. In 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 604–611. IEEE, 2020.

Farzaneh Sheikhnezhad Fard and Thomas P Trappenberg. A novel model for arbitration between planning and habitual control systems. Frontiers in neurorobotics, (13):52, 2019. doi: 10.3389/fnbot.2019.00052.

Amitai Shenhav, Matthew M Botvinick, and Jonathan D Cohen. The expected value of control: an integrative theory of anterior cingulate cortex function. Neuron, 79(2):217–240, 2013.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. arXiv preprint arXiv:1906.02243, 2019.

Richard S. Sutton and Andrew G. Barto. Introduction to Reinforcement Learning. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981.

Matthijs Van Der Meer, Zeb Kurth-Nelson, and A David Redish. Information processing in decision-making systems. The Neuroscientist, 18(4):342–359, 2012.

Guillaume Viejo, Mehdi Khamassi, Andrea Brovelli, and Benoît Girard. Modelling choice and reaction time during arbitrary visuomotor learning through the coordination of adaptive working memory and reinforcement learning. Frontiers in Behavioral Neuroscience, 9(225), 2015. ISSN 1662-5153. doi: 10.3389/fnbeh.2015.00225.

Jane X Wang, Zeb Kurth-Nelson, Dharshan Kumaran, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Demis Hassabis, and Matthew Botvinick. Prefrontal cortex as a meta-reinforcement learning system. Nature neuroscience, 21(6): 860–868, 2018.

Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. arXiv preprint arXiv:1907.02057, 2019.

Martina Zambelli and Yiannis Demiris. Online multimodal ensemble learning using self-learned sensorimotor representations. IEEE Transactions on Cognitive and Developmental Systems, 9(2):113–126, 2016.

Alexandre Zenon, Oleg Solopchuk, and Giovanni Pezzulo. An information-theoretic perspective on the costs of cognition. Neuropsychologia, 123:5–18, 2019.

Supplementary Information (SI) for the manuscript 'Reducing computational cost during robot navigation and human-robot interaction with a human-inspired reinforcement learning architecture'

Preprint of the paper published in IJSR (2022), special issue on 'Human-like Behavior and Cognition in Robots'

Rémi Dromnelle **Erwan Renaudo** **Petros Maragos** **Mohamed Chetouani** **Raja Chatila**

Benoît Girard **Mehdi Khamassi**

November 24, 2022

1 Experiment 1: Navigation task

1.1 Additional simulation results of the task with change in goal location

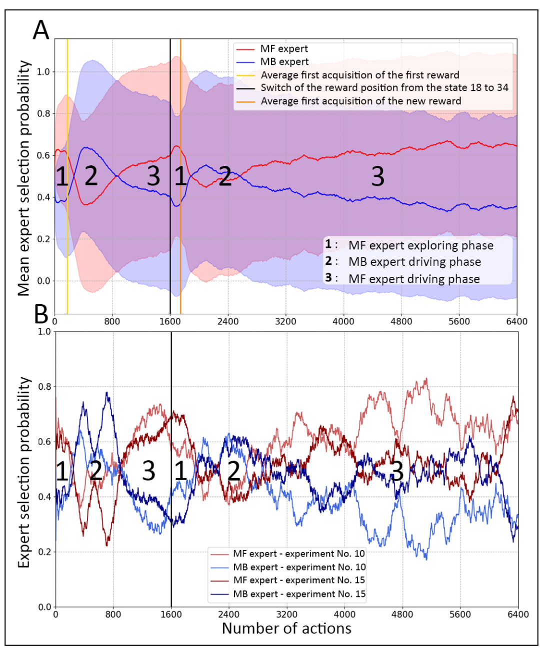

Figure 1: **Simulation results of individual runs of the navigation task with change in goal location. A**. Mean probabilities of selection of experts by the MC using the Entropy and Cost criterion for 100 simulated runs of the task. These probabilities are defined by the softmax function of each expert. The duration is represented as the number of actions performed by the agent. We use standard deviation as dispersion indicator. **B**. Probabilities of selection of experts by the MC using the Entropy and Cost criterion for 2 simulated runs of the task.

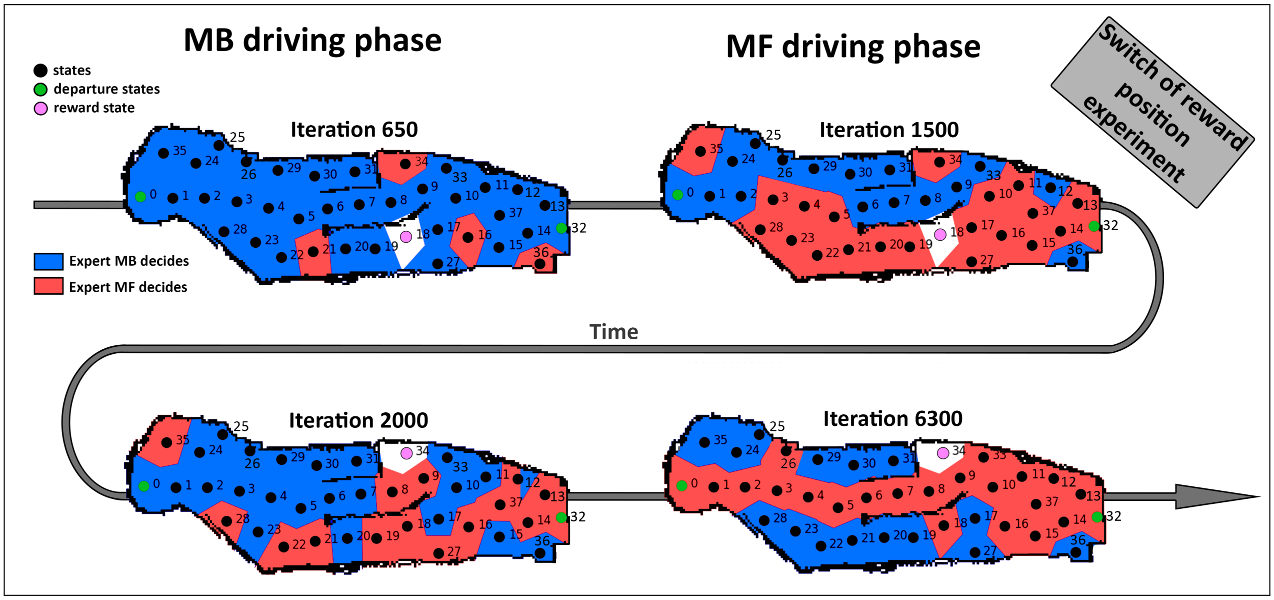

Figure 2: Evolution of the expert spatial preferences in the reward location change navigation experiment. Expert selection maps of the MC-EC agent for one of the hundred simulations: in red, states where the MF was the last chosen expert, in blue, where the MB was last chosen. The MF driving phase and the MB driving phase correspond to the behavioral phases identified in Fig. 5C in the main manuscript.. Same conventions as in Fig. 6 in the main manuscript.

1.2 Additional simulation results of the task with change in wall configuration

1.3 Additional results of the navigation task with the real robot

2 Experiment 2: Human-robot interaction with human as teacher

2.1 Results with human intervention of the type *Congratulations*

2.2 Results with human intervention of the type *Takeover*

3 Experiment 3: Human-robot interaction with human as cooperator

3.1 When the partner becomes an adversary

3.2 Context change detection

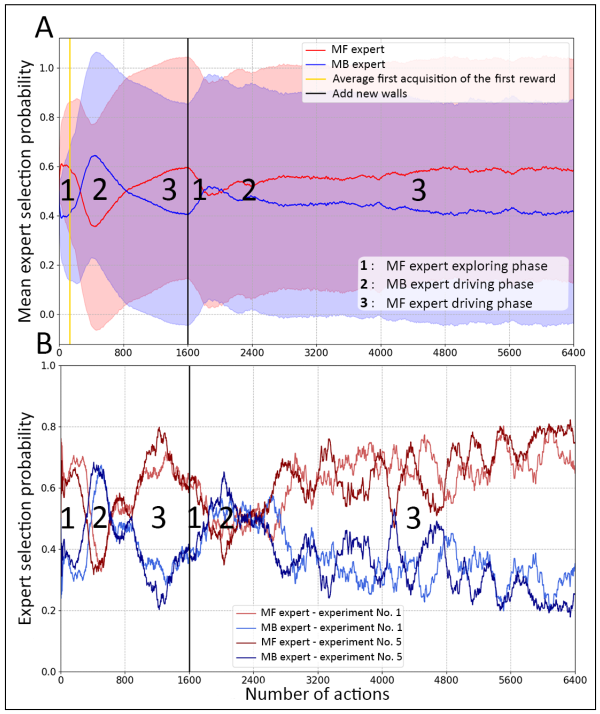

Figure 3: **Simulation results of individual runs of the navigation task with change in wall configuration. A**. Mean probabilities of selection of experts by the MC using the Entropy and Cost criterion for 100 simulated runs of the task. **B**. Probabilities of selection of experts by the MC using the Entropy and Cost criterion for 2 simulated runs of the task. Same conventions as Suppl. Fig. 1.

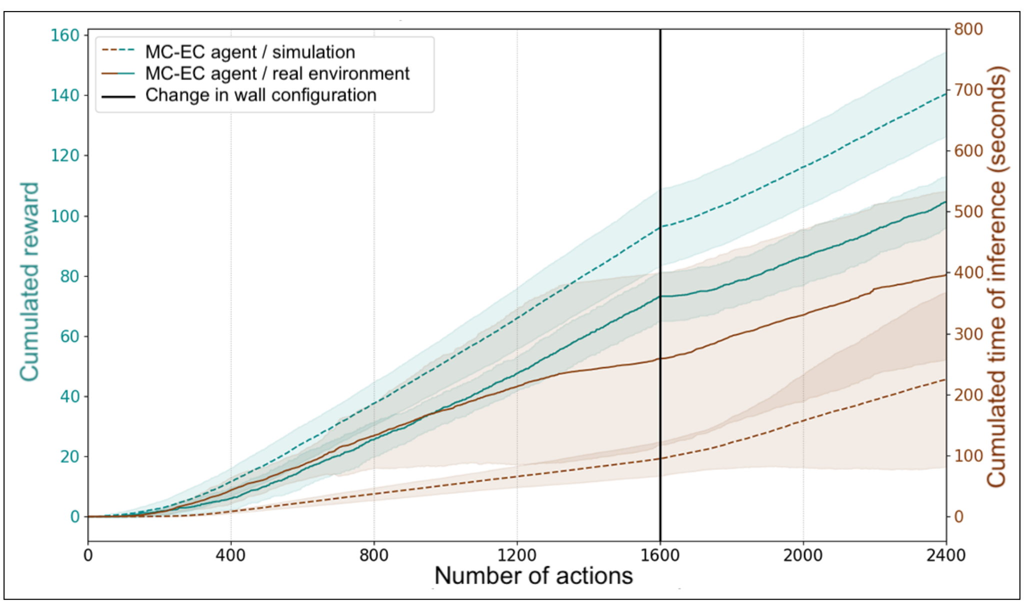

Figure 4: Real robot dynamics of expert selection in the wall configuration condition of the navigation task: Mean performance (in cyan) and computational cost (in brown) of the MC-EC robot. Dashed lines: simulation results; full lines: real robot results.

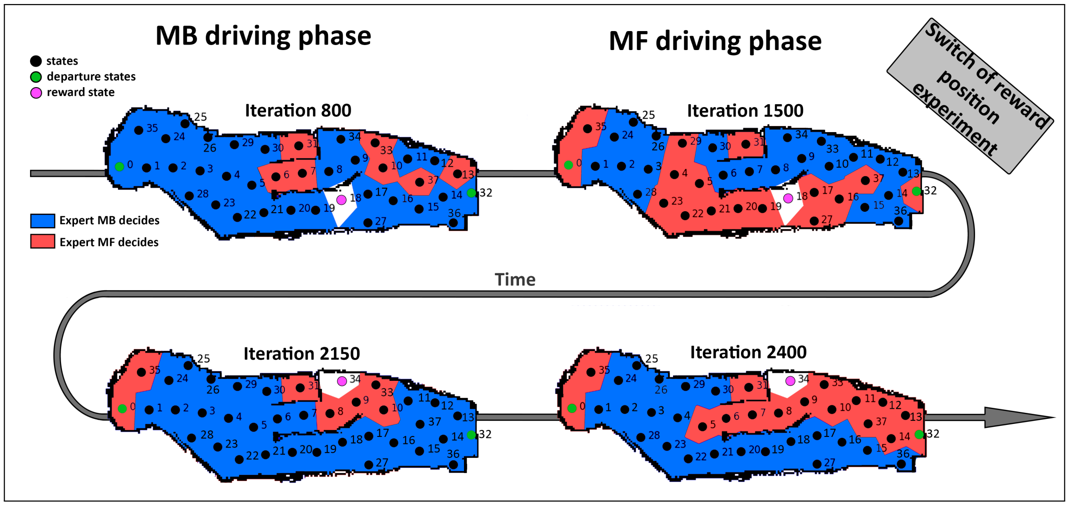

Figure 5: **Expert selection map by the MC of the MC-EC robot for one of the navigation experiments with the real robot and with change in reward location.** Same conventions as in Fig. 6 in the main manuscript.

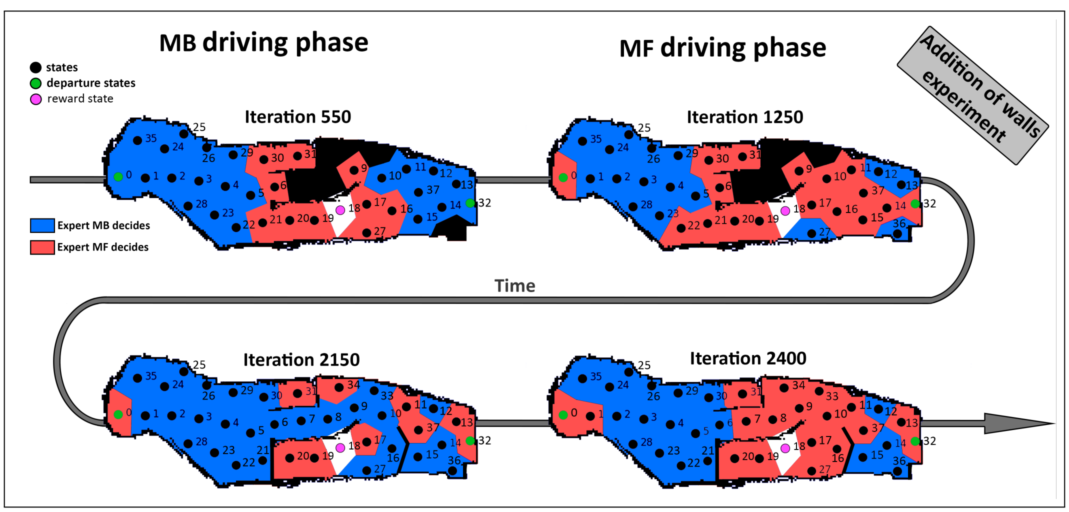

Figure 6: **Expert selection map by the MC of the MC-EC robot for one of the navigation experiments with the real robot and with change in wall configuration.** Same conventions as in Fig. 6 in the main manuscript.

MF / MB	0	10	20	30	40	50	100	150	200	300	400	500
0	0	1.0	1.0	1.0	1.0	1.0	1.0	0.408531	0.030455	0.000016	1.626891e-08	1.572634e-13
10	1.0	10	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.008622	3.915252e-05	3.025572e-09
20	1.0	1.0	20	1.0	1.0	1.0	1.0	0.563601	0.045360	0.000029	3.248472e-08	3.714620e-13
30	1.0	1.0	1.0	30	1.0	1.0	1.0	1.0	0.180889	0.000215	3.829072e-07	8.160585e-12
40	1.0	1.0	1.0	1.0	40	1.0	1.0	1.0	1.0	0.155930	1.696102e-03	4.402584e-07
50	1.0	1.0	1.0	1.0	1.0	50	1.0	1.0	1.0	0.203656	2.423178e-03	7.127423e-07
100	1.0	1.0	1.0	1.0	1.0	1.0	100	1.0	1.0	0.074458	6.376559e-04	1.186862e-07
150	1.0	1.0	1.0	1.0	1.0	1.0	1.0	150	1.0	1.0	2.169030e-02	1.448447e-05
200	0.568655	0.033144	0.406211	1.0	1.0	1.0	1.0	1.0	200	1.0	3.101433e-01	6.592969e-04
300	0.132280	0.005327	0.090044	1.0	1.0	0.478832	1.0	1.0	1.0	300	1.0	3.831200e-01
400	0.474306	0.026334	0.336678	1.0	1.0	1.0	1.0	1.0	1.0	1.0	400	1.0
500	0.172429	0.007403	0.118358	1.0	1.0	0.605852	1.0	1.0	1.0	1.0	1.0	500

RND / EC	0	10	20	30	40	50	100	150	200	300	400	500
0	0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.890272	1.0	1.0
10	1.0	10	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.455387	1.0	0.632360
20	1.0	1.0	20	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
30	1.0	1.0	1.0	30	1.0	1.0	1.0	1.0	1.0	0.029717	0.655518	0.044854
40	1.0	1.0	1.0	1.0	40	1.0	1.0	1.0	1.0	0.017303	0.431479	0.026507
50	1.0	1.0	1.0	1.0	1.0	50	1.0	1.0	1.0	0.819644	1.0	1.0
100	1.0	1.0	1.0	1.0	1.0	1.0	100	1.0	1.0	1.0	1.0	1.0
150	1.0	1.0	1.0	1.0	1.0	1.0	1.0	150	1.0	1.0	1.0	1.0
200	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	200	1.0	1.0	1.0
300	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	300	1.0	1.0
400	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	400	1.0
500	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	500

Figure 7: **Results of *Dunn*'s multiple comparison tests for the performance of the four robots in the *congratulation* type intervention of Experiment 2**. P-values below the significance threshold 0.05 are colored in red. The significance level has been corrected with the *Bonferroni correction*.

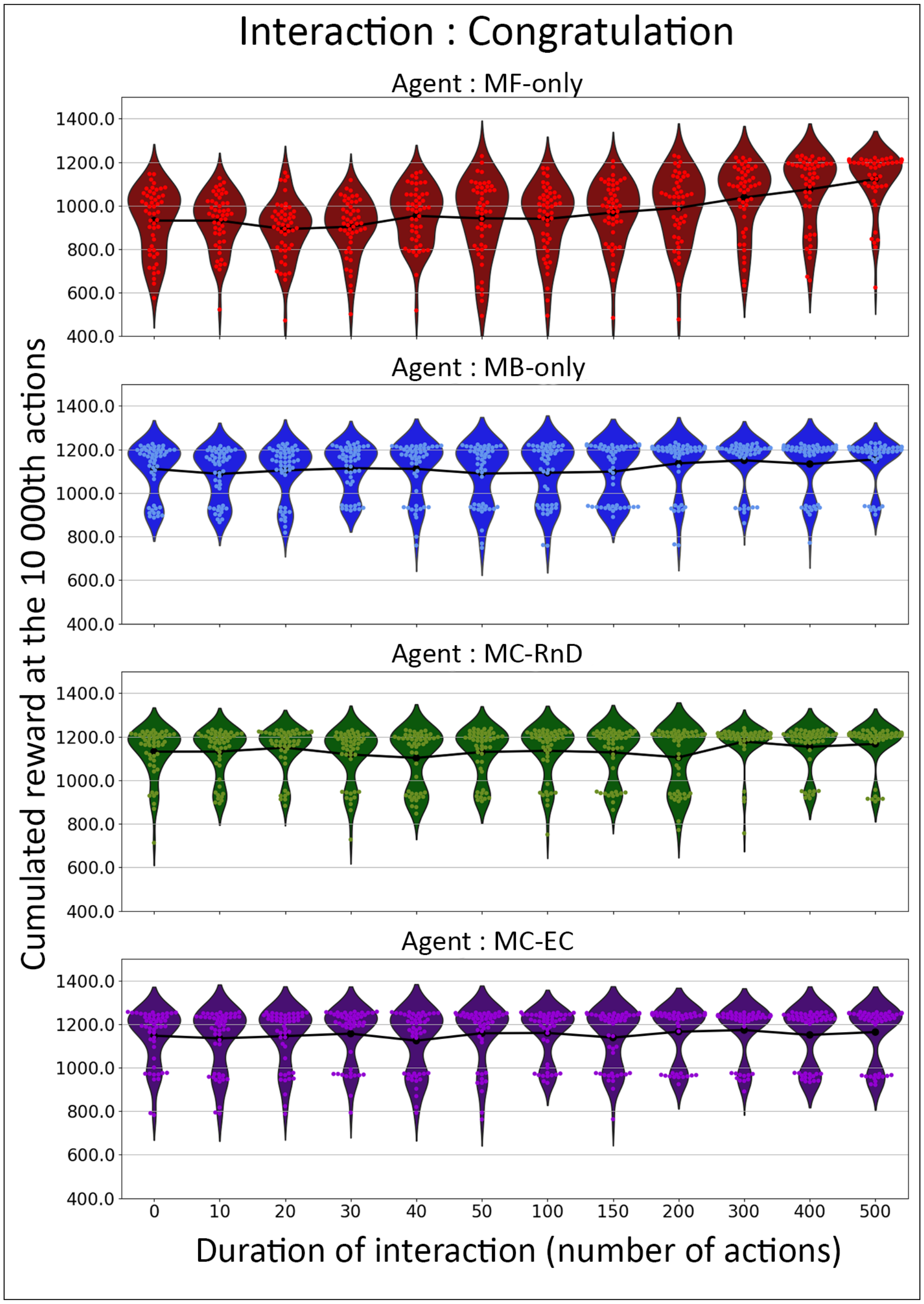

Figure 8: Reward accumulation in Experiment 2 when humans provide *congratulation* feedback for various durations (from 0 to 500 timesteps). Dots report the accumulated after 10,000 simulation timesteps, for 50 simulations. First row (red): MF-only agent; second row (blue) MB-only agent; third row (green): MC-Rnd agent; fourth row (purple): MC-EC agent.

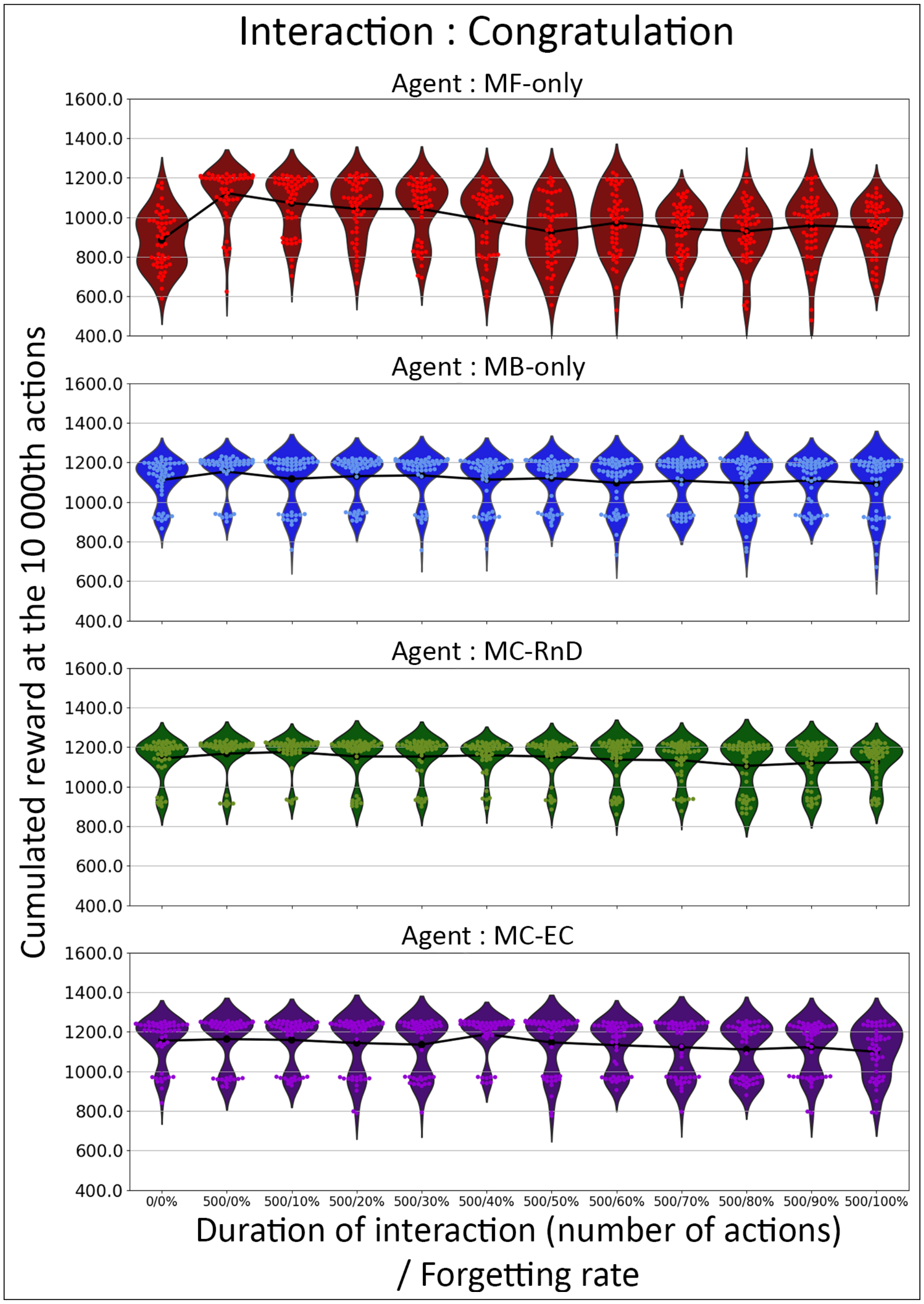

Figure 9: Reward accumulation in Experiment 2 when humans omit to provide *congratulation* feedback with increasing omission rates. Dots report the accumulated after 10,000 simulation timesteps, for 50 simulations. First row (red): MF-only agent; second row (blue) MB-only agent; third row (green): MC-Rnd agent; fourth row (purple): MC-EC agent.

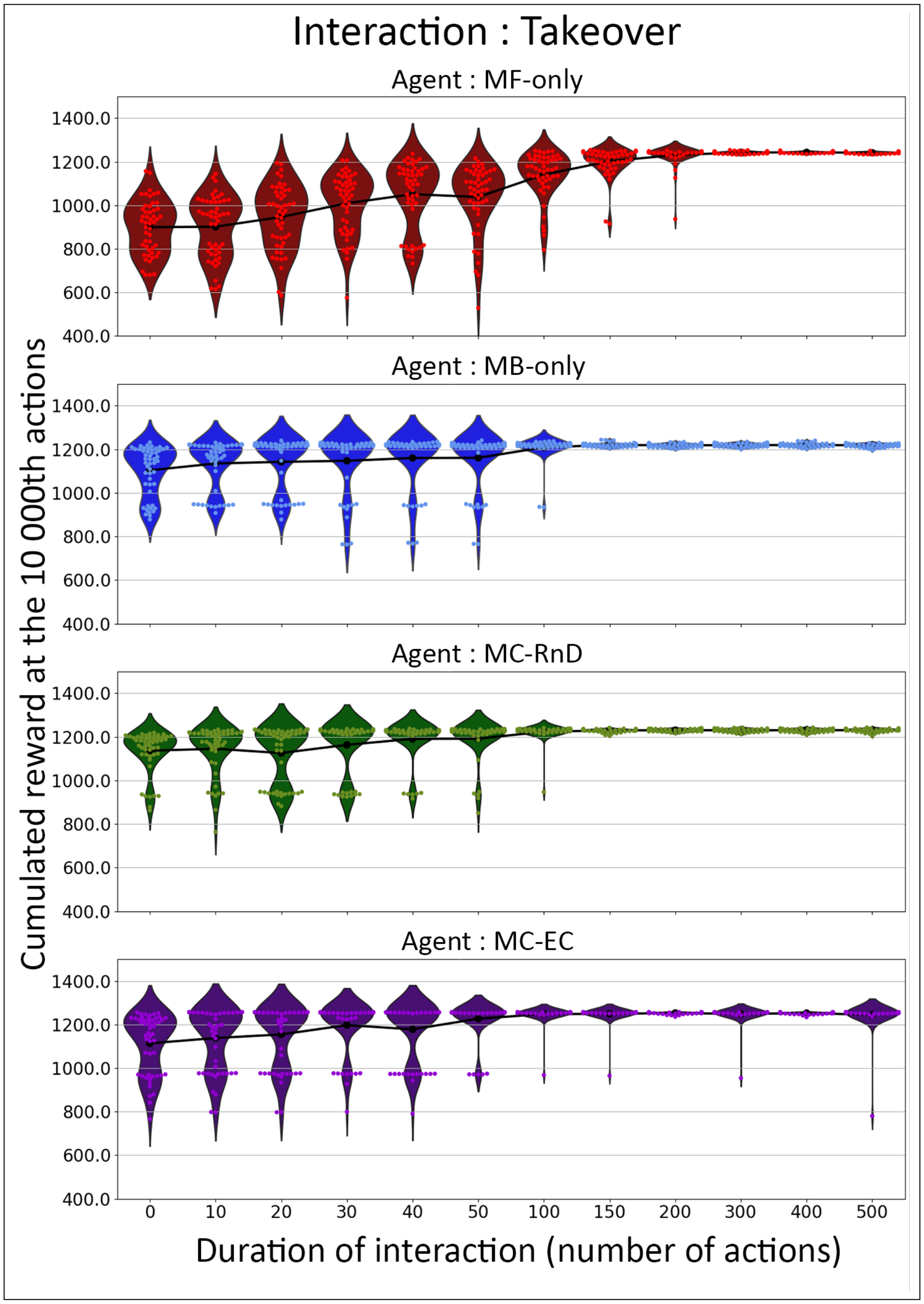

Figure 10: Reward accumulation in the HRI teaching task, when humans provide *takeover* feedback for various durations (from 0 to 500 timesteps). Dots report the accumulated after 10,000 simulation timesteps, for 50 simulations. First row (red): MF-only agent; second row (blue) MB-only agent; third row (green): MC-Rnd agent; fourth row (purple): MC-EC agent.

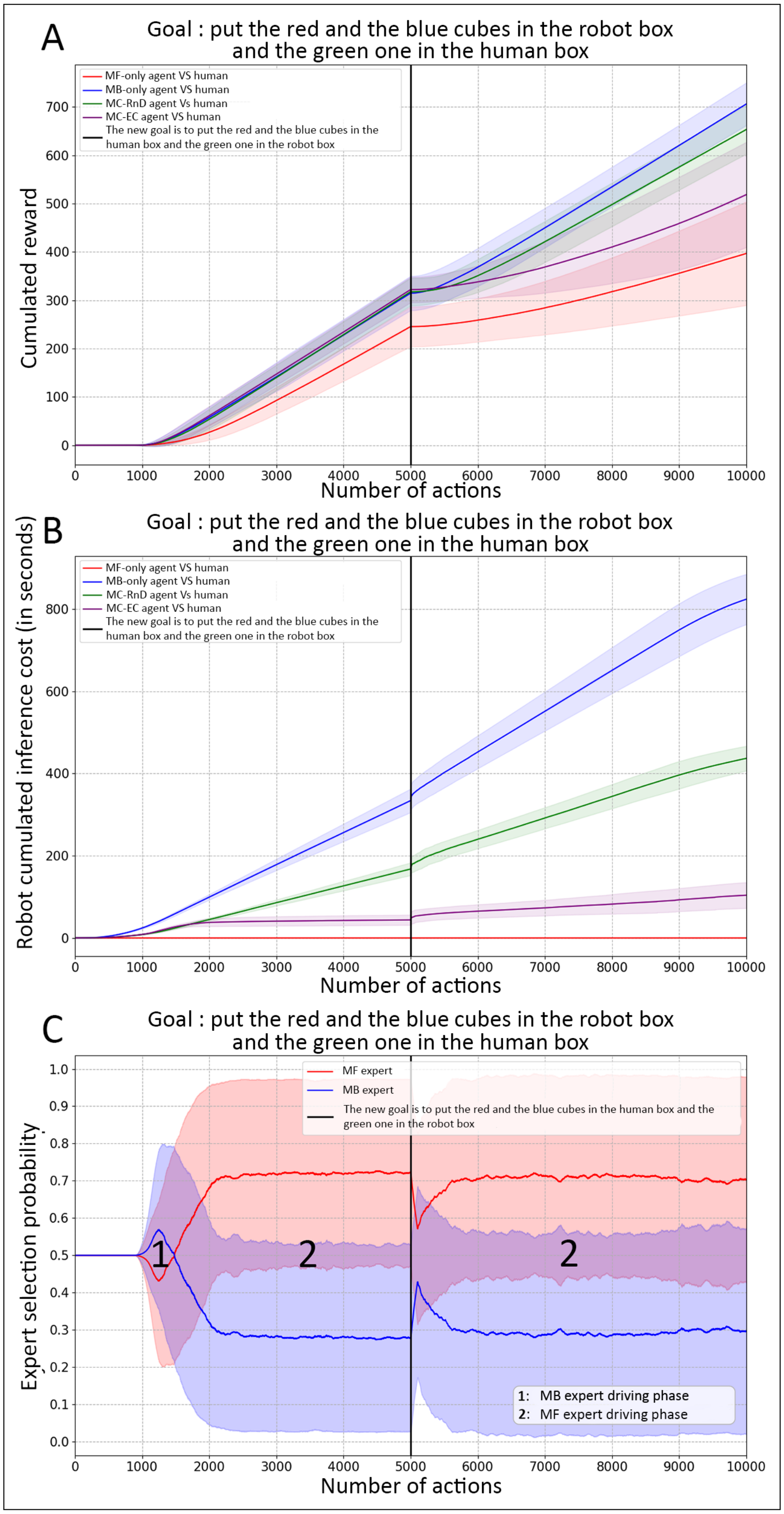

Figure 11: **Simulation results of the human-robot cooperation task (Experiment 3) with the second pair of objectives. A**. Average performance for 50 simulated experiments. **B**. Average computational cost for 50 simulated experiments. **C**. Average probability of selection of experts by the meta-controller of the MC-EC robot for 50 simulated experiments. We use standard deviation as an indicator of dispersion in all three figures.

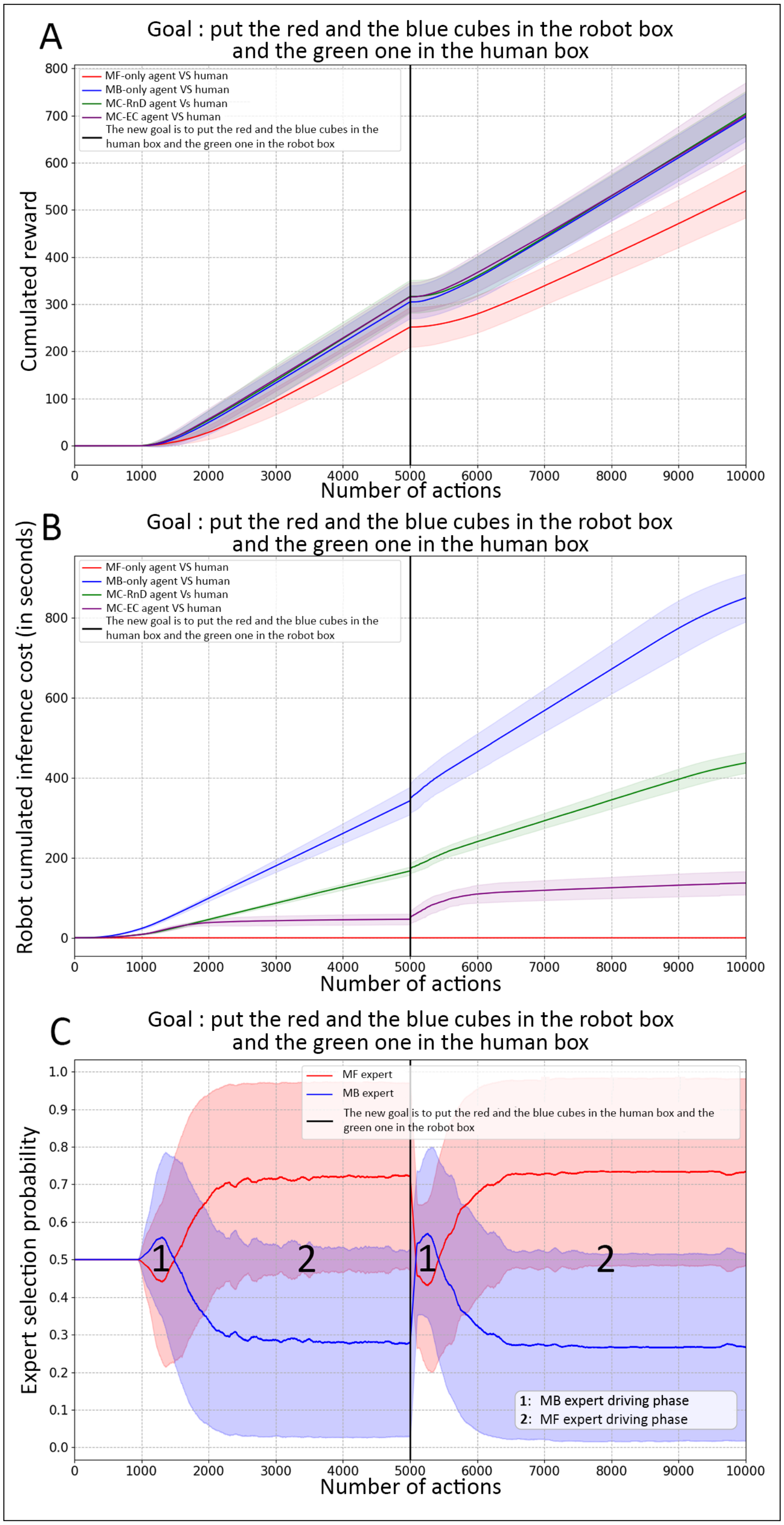

Figure 12: **Simulation results of the human-robot cooperation task (Experiment 3) with the second pair of objectives with context change detection.** **A**. Average performance for 50 simulated experiments. **B**. Average computational cost for 50 simulated experiments. **C**. Average probability of selection of experts by the meta-controller of the MC-EC robot for 50 simulated experiments. We use the standard deviation as an indicator of dispersion in all three figures. In these experiments, robots are able to detect context switches.