A Hybrid Controller based on the Egocentric Perceptual Principle

Zinovi Rabinovich*, Nicholas R. Jennings

Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, UK

Abstract

In this paper we extend the control methodology based on Extended Markov Tracking (EMT) by providing the control algorithm with capabilities to calibrate and even partially reconstruct the environment's model. This enables us to resolve the problem of performance deterioration due to model incoherence, a problem faced in all model-based control methods. The new algorithm, *Ensemble Actions EMT (EA-EMT)*, utilises the initial environment model as a library of state transition functions and applies a variation of prediction with experts to assemble and calibrate a revised model. By so doing, this is the first hybrid control algorithm that enables on-line adaptation within the egocentric control framework which dictates the control of an agent's perceptions, rather than an agent's environment state. In our experiments, we performed a range of tests with increasing model incoherence induced by three types of exogenous environment perturbations: *catastrophic* – the environment becomes completely inconsistent with the model, *deviating* – some aspect of the environment behaviour diverges compared to that specified in the model, and *periodic* – the environment alternates between several possible divergences. The results show that EA-EMT resolved model incoherence and significantly outperformed its EMT predecessor by up to 95%.

Keywords: hybrid control, perceptual control, dynamics based control, Kullback-Leibler divergence

1. Introduction

Egocentric perceptual control (EPC) formulates a control problem in terms of an agent's perceptions, i.e. its internal interpretation of sensory input, rather than the actual environment state [1]. As a direct outcome of this representation, any task

*Corresponding author

Email addresses: zr@ecs.soton.ac.uk (Zinovi Rabinovich), nrj@ecs.soton.ac.uk (Nicholas R. Jennings)

that an agent performs is expressed as a preference over perceptions, and the optimality criteria follows suit. In fact, from this egocentric point of view, changes in the environment are simply a means to alter and control the agent's perceptions. As a technical example consider instrument flight rules (IFR), the regulations and procedures for flying aircraft by referring only to the aircraft instrument panel for navigation. These rules describe the instrument readings that a pilot (and hence the auto-pilot control algorithm) has to maintain, therefore referring to the perceived flight parameters, rather than the factual physical state of the plane. Notice that the instrument readings are indeed *perceptions*, the interpretations of the automated sensors, rather than the observations or measurements that they make. To date, EPC has been used in a variety of domains, including sensory-based navigation of autonomous robots, where all the necessary information is represented through perceptions, such as maps or landmarks (see e.g. [2, 3]). In fact, one of the most successful control approaches in robotics, the behaviour-based control (BBR) [4], can be seen to be a particular instantiation of the EPC. In more detail, in BBR a complex behaviour with desired properties is obtained by means of arbitration and fusion of a set of simple mappings (basic behaviours) from perceptions to actions [1]. Starting from the simplest basic behaviours, that are enacted once some key perception is formed, and ending with complex arbitration of a BBR scheme, all key features of decision making are based on perceptual information, therefore conforming BBR to the EPC view. Moreover, EPC is inherent to behaviour patterns found in nature or based on human intuition and psychology (e.g. [5, 6] and references therein). It enables, for instance, a quick design of individual behaviours in BBR, as well as the interpretation and explanation of the final outcome in human understandable terms. Unfortunately, with a few exceptions, most current EPC approaches are not universal. In BBR, for example, the elementary behaviours are commonly designed off-line for a specific domain or learned from scratch, a significant shortcoming in dynamic or only partially known environments.

On the other hand, classical control theory has been explicitly developed to find universal control solutions with an explicit environment model as input [7]. It was also readily extended to hybrid models, where several discrete and continuous components interact in a non-trivial manner (see e.g. overview in [8] and references therein). In particular, model predictive (or model-following) methods have been found to be applicable to a wide range of control problems and to be efficient at dealing with modelling errors (see e.g. [9, 10]). These methods use a system

[1] Notably, BBR is also inherently hybrid, since distinct behaviours can be designed using completely different methodologies: while some of them can use fuzzy logic, others may include a learning algorithm or simply be reactive. However, *EPC* and *hybrid* are, in general, distinct properties.

model to generate predictions on the system development, and compute a control signal to optimise this predicted behaviour. Furthermore, the methodology readily accepts various learning techniques, both to calculate the control signal and to adaptively calibrate the model in dynamic or partially known environments. However, the detail of the model calibration may vary according to the imposed system structure and dynamics assumptions. For instance, in reinforcement learning architectures, such as Dyna [11], model corrections are local to the current environment state. Dyna's principles are also echoed by the modern Bayesian techniques where a POMDP model is recovered while finding the reward maximising policy (e.g. [12, 13]). However, the success of these works has been conditioned on the domain being well factored or on the presence of an oracle to query for the true system state. Furthermore, these approaches can not address the problem of an environment that drifts through a continuous range of models due to their rigid assumptions on system structure. To address this issue, much stronger, hybrid control methods have been constructed, usually based on the model predictive (or model-following) principle (see e.g. [14–16]). Some methods even provide theoretical guarantees [14], however at the price of requiring additional modifications to work with discrete space domains or losing this capability entirely.

Given these complementary strengths, the fusion of EPC with model-based control can potentially lead to an extremely powerful framework. It would combine the egocentric autonomous representation, i.e. dynamic system without external control input, of a task and the capability to incorporate high level environment knowledge in the form of a system model. Unfortunately, various as they are, classic control theory approaches have an important underlying assumption: the subject of the optimality criteria are the state and the dynamics of the environment. Be that the expected accumulated cost of the state variation (e.g. the classic work of Stengel [7]), be that the proximity to an ideal distribution over system trajectories (e.g. [17]) or be that the cost of system stability (e.g. [18]), the optimality criteria always comes back to consider the underlying system state transitions as the utility source, even if the environment model contains observed quantities only (e.g. [19]). By so doing, this assumption explicitly contradicts the EPC point of view, which hinders the aforementioned fusion of the two control principles.

In fact, the only control algorithm that possesses a complete fusion of both the model-based control principles and the EPC view is the Extended Markov Tracking (EMT) algorithm [20] and its descendants (e.g. [21, 22]). However, as our experiments have revealed, the standard EMT can not cope well with model incoherences. To this end, in this paper we propose an extended EMT algorithm that has all the aforementioned capabilities: it is an egocentric perceptual control algorithm, it is a universal model-based controller, it is adaptive to environment changes by means of an on-line model calibration, it is a hybrid controller capable

of operating in mixed discrete-continuous domains or domains with a hierarchical abstraction of actions. In more detail, for each action available to the agent, we deploy an experts ensemble [23] to learn a good estimate of an action's effects. Such ensembles are known to provide highly flexible and dynamic estimates, which in our case corresponds to fast estimation and calibration of a system model. Notice that this estimate is with respect to the predictive capabilities of the action effects on the agent's perceptions. Now, the expert ensemble is composed of a finite set of potential effects an action may have, mined from an initial environment model, which are dynamically merged together into a single estimate of an action's effect. The new control algorithm, the Ensemble Action EMT (EA-EMT) then uses the collection of these estimates to form a complete environment model and proceeds to follow the normal EMT flow of action selection.

To demonstrate the adaptive efficacy of the EA-EMT algorithm we have devised a set of experiments with various incoherences of the initial system model. In a discrete state environment we have investigated the effects of exogenous perturbations of three types: *catastrophic* – the environment becomes completely inconsistent with the model, *deviating* – some aspect of the environment behaviour diverges compared to that specified in the model, and *periodic* – the environment alternates between several possible divergences. The results show that EA-EMT resolved model incoherence and outperformed its EMT predecessor by up to 95%. To clearly demonstrate the hybrid nature and capabilities of the EA-EMT algorithm, we have devised an additional experiment with a continuous state environment, where a task had to be achieved by switching between several pre-specified subcontrollers. In this continuous state environment we have also compared the effects a deviating inconsistency has on EMT-based approaches (both the standard EMT and the EA-EM) and the classical model-following approach. In our experiments, EMT has outperformed the model-following controller under model incoherence, and both have been outstripped by EA-EMT by at least 40% in error rate.

To summarise, the contributions of this paper are as follows. First, we introduce a new hybrid control method that is equally applicable in environments with discrete, continuous or mixed environment state. This enables the algorithm to serve both as a universal low level mechanism of action selection, and as a high level switching mechanism between separate tuned controllers in a hybrid architecture. In particular, the algorithm is resistant to switching noise, the capability well beyond even the most modern switching methods (e.g. [14]). Second, our approach provides, for first time, an adaptive controller version of the model-based EPC paradigm, enabling in observable terms. Third, EA-EMT is the first algorithm that, without sacrificing its generality with respect to its environment's continuity, is capable of composing a good control signal even if the underlying environment dynamics are non-stationary, and change over time.

The rest of the paper is organised as follows. In Section 2 we detail the operation of the standard EMT Control algorithm. Section 3 follows with the description of our new EA-EMT algorithm, detailing how it reconstructs and calibrates the environment model through the use of expert ensembles. Experimental support for the effectiveness of our approach in handling various model incoherences is given in Sections 4, while the experiments of Section 5 are designed to expose the hybrid nature of our algorithm. To underline the algorithm's capability to work in environments with changing behavioural trends, our experiments take a special focus on the on-line property of the EA-EMT model calibration. Section 6 summarises the results and gives future directions of this research.

2. EMT Control

EPC controllers are constructed around some perceptual concept, and necessarily include a subsystem that creates and maintains these perceptions by accumulating and interpreting the observed data. In the case of an EMT Controller the perception is that of the autonomic system dynamics, where the system state appears to stochastically develop over time without external influence. The convenience of this choice is made apparent by the following observation. Assume that some control has been plugged into the environment. The resulting overall system is autonomic, and describes the behaviour of the control-augmented environment in all possible states. Furthermore, although we may not know what specific control law will bring it about, we frequently can describe the autonomic dynamics that we would consider to be ideal or optimal. For example, in IFR, the behaviour of instrument gauge is described without specifying what actions the pilot has to take to achieve this behaviour. This approach is adopted by the EMT controllers, the control task is described by a perception of an idealised autonomic system dynamics, and the algorithm has to sequence actions to achieve the perception of this ideal. To do so, however, the controller requires a subsystem that creates and maintains the necessary perception, and in this paper the subsystem is the Extended Markov Tracking (EMT) algorithm, that also lends its name to the entire control scheme.

Formally, the EMT algorithm produces and maintains an estimate of a stochastic state transition function that models the autonomic system behaviour. It does so by performing a conservative update, specifically it minimises the Kullback-Leibler divergence between the new and the old estimate, with the limitation that the new estimate has to match the most recently observed system transition. In more detail, assume that two probability distributions over the system state, p_t and p_{t+1}, are given that describe two consecutive states of knowledge about the system, and τ_t^{EMT} is the old estimate of the system dynamics. Then the EMT update, abbreviated by $\tau_{t+1}^{EMT} = H\left[p_t \to p_{t+1}, \tau_t^{EMT}\right]$, is the solution of the optimisation problem

depicted in Fig. 1, where D_{KL} is the Kullback-Leibler divergence. The optimisation can be recast as finding joint distribution with given marginals, and solved by an Iterated Proportional Fitting (IPF) procedure [24]. In fact, the EMT update, H, can be calculated for any practical set of distributions that describe p_t, p_{t+1} and τ_t^{EMT}, although in more general situations approximate representations, such as particle filters or unscented transforms, may be necessary. This enables EMT to uniformly treat both discrete, continuous and hybrid state spaces. However, to ease the exposition, in this paper we concentrate on two simplest distribution families, namely the discrete and the Gaussian distributions. For these families the IPF has been well studied and needs not to be proof-checked [25, 26].

$$\begin{aligned} \tau_{t+1}^{EMT} &= \arg\min_{\tau} D_{KL}(\tau \times p_t \| \tau_t^{EMT} \times p_t) \\ \text{s.t. } p_{t+1}(x') &= \sum_x (\tau \times p_t)(x', x) \\ \text{and } p_t(x) &= \sum_{x'} (\tau \times p_t)(x', x) \end{aligned}$$

Figure 1: The EMT Update

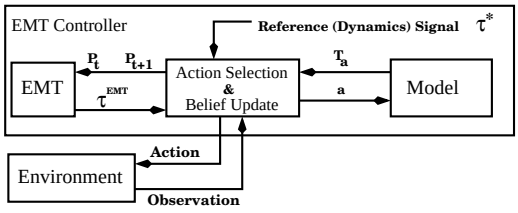


Figure 2: The closed loop of EMT Control

To complete the EMT control loop, however, we still need to describe how the observation data is accumulated to form the perception of the system dynamics. To this end, we have to address the type of environment models we will be working with. Although EMT can work with more general environmental descriptions (see e.g. [22]), it has been more commonly used with a discrete Markovian environment with partial observability, described by a tuple $MEnv =< S, s_0, A, T, O, \Omega >$, where: S is the set of all possible environment states; $s_0 \in \Delta(S)$ is the initial state distributions of the environment, where $\Delta(S)$ is a family of distributions over S; A is the set of all actions applicable in the environment; $T : S \times A \to \Delta(S)$ is the environment's probabilistic transition function, where $T(s'|a, s)$ is the probability that the environment will move from state s to state s' under action a; O is the set of all possible observations; $\Omega : S \times A \times S \to \Delta(O)$ is the observation probability function, where $\Omega(o|s', a, s)$ is the probability that o will be observed given that the environment moved from state s to state s' under action a.

This naturally connects with the EMT algorithm, as knowledge about the system is summarised by a distribution vector over the system states $p_t \in \Delta(S)$, in which case the system dynamics estimator created by EMT has the form of a conditional probability $\tau : S \to \Delta(S)$.

Given this, the overall control algorithm, termed *EMT Control*, forms a closed loop control with a reference signal [7]. Fig. 2 depicts the resulting scheme. Three sub-modules form the *EMT Controller* that interacts with an *Environment* by ap-

plying actions in and receiving observations from it: the *Model*, the *EMT* estimator, and the decision making module of *Action Selection and Belief Update*. The *Model* module is queried for the effects T_a of an action a on the real system state. These effects are used both in predicting future perceptions, and in filtering the observed data to maintain system state beliefs. The *EMT* module is used to estimate the perceived dynamics τ^{EMT} that explain the change in beliefs about the system from p_t at time t to p_{t+1} at time $t+1$. The central, decision making module, interconnects the *Model* and the EMT estimator, and implements the EMT Control algorithm, the detail of which we describe below. Finally, the reference signal, τ^*, encodes the task to be performed and formally takes the form of the conditional probability $\tau^* : S \rightarrow \Delta(S)$.

Notice that τ^* represents the ideal autonomic system dynamics we would like to obtain by exercising control. From the EPC point of view, this is the target perception that we would like to achieve and maintain, hence the standard EMT Control (see Fig. 3) can be described as a greedy one-step look ahead correction action selection, and it follows a closed loop structure. In more detail, at every point in time, the algorithm attempts to predict the reaction of an estimation algorithm (EMT in this case) to the changes induced by an action (lines 12-16 of the algorithm), and then chooses the action that shifts the EMT estimator closest (line 17) to the reference dynamics τ^*. Once the action has been applied, the response of the EMT estimator to the changes in the environment is registered (line 20), and the control loops to make its next decision.

At this point, we would like to underline the strength of the task representation by the autonomic system dynamics τ^*. First, while deterministic dynamics are a way to concisely represent feasible sequences of states, probabilistic dynamics can also engender a preference over such sequences. Thus, system dynamics τ^* can encode a richer variety of preferences and tasks for EMT control, than, for instance, a reward function over states would. Second, in Markov chains, system dynamics completely determine the system state in the long run. As an outcome, the knowledge about the initial system state is not essential to EMT control operation, expanding its applicability. Furthermore, although over the given state space the τ^* transition is Markovian, the task it describes needs not be Markovian within the environment itself. This is due to the fact that the model's state space is abstract, and each state can serve as a tag for complex, time extended events. Notice, however, that the controller action selection in lines 12-16 is heavily dependent on the environment model, as it uses the mapping T_a to predict action effects. However, if the model is incoherent the reaction of EMT can not be estimated correctly, which, in turn, will lead to selection of a suboptimal action. Thus, in what follows, we modify the action selection process to vary the environment model it uses.

Require:
Set the system state estimator: $p_0(s) = s_0 \in \Delta(S)$
Set the system dynamics estimator: $\tau_0^{EMT}(\bar{s}|s) = prior(\bar{s}|s)$
Set time to $t = 0$.
11: **loop**
12: **for all** $a \in \mathcal{A}$ **do**
13: **Set** $\bar{T}_a = T_a$ {**use transition model** T **directly**}
14: Set $\bar{p}_{t+1}^a = \bar{T}_a * p_t$
15: Set $D_a = H\left[p_t \rightarrow \bar{p}_{t+1}^a, \tau_t^{EMT}\right]$
16: Set $V(a) = \langle D_{KL}\left(D_a \| \tau^*\right)\rangle_{p_t}$
17: Select $a^* = \arg\min_a V(a)$
18: Apply a^*, receive observation $o \in O$
19: Compute p_{t+1} due to the Bayesian update: $p_{t+1}(s) \propto \Omega(o|s,a) \sum_{s'} \bar{T}(s|a,s') p_t(s')$
20: Compute $\tau_{t+1}^{EMT} = H\left[p_t \rightarrow p_{t+1}, \tau_t^{EMT}\right]$
21–25: {**no model update**}
26: Set $t := t + 1$

Figure 3: The standard EMT control algorithm. Note: EA-EMT will modify lines 13,21-25.

3. Ensemble Action EMT

Although the standard EMT Control is attractive in its combination of the egocentric control perspective and the task description by the perceived system dynamics, our experiments (see Section 4) have revealed that its performance deteriorates significantly if the environment model is incoherent. However, we believe (and will subsequently demonstrate) that, by providing the algorithm with an additional method to correct model incoherences, it is possible to rectify the deterioration. Now, there are many incoherences a Markovian model, $MEnv =< S, s_0, A, T, O, \Omega >$, may have. Specifically, while the choice of the state, action and observation spaces, as well as the observability function, may be dictated by subjective considerations (e.g. to make it more readable for the human domain designers), the transition function T is always dictated by the environment. Thus, in this work we choose to concentrate on the quality of the transition function T. This function maps actions into stochastic matrices, so that for each action $a \in A$ the matrix $T_a = T(\cdot|\cdot, a)$ models the effects of that action on the system state. The difference between the matrix T_a and the true effects of the action $a \in A$ is the incoherence type we have resolved in the EA-EMT algorithm (Fig. 4). Thus, while the standard EMT Control views the transition mapping, $a \mapsto T_a$, to be constant, the EA-EMT algorithm modifies its transition mapping over time, reducing the mapping's incoherence. However, before we go into the details of how it was implemented, we need to explain the principles of the approach taken by EA-EMT.

EA-EMT assumes that, although the mapping $T : A \rightarrow \Delta(S)^S$ is incoherent, the set of matrices $T_A = \{T_a = T(\cdot|\cdot, a)\}_{a \in A}$ represents feasible effects that the actions may have. The algorithm then attempts to assemble a better mapping,

$\bar{T} : A \to \Delta(S)^S$, based on the set T_A. More specifically, for each action $a \in A$ the transition matrix $\bar{T}_a$ is a weighted linear combination of matrices in the set T_A, that is $\bar{T}_a = \sum_{b \in A} T_b * w_a(b)$. Intuitively, the weight $w_a(b)$ represents the similarity between the matrix $T_b \in T_A$ and the effects that the action $a \in A$ has on on the environment state. As the interaction between the EA-EMT algorithm and the environment progresses, the weights $w_a(\cdot)$ are updated, modifying the mapping $\bar{T} : A \to \Delta(S)^S$ to reduce its incoherence with the environment.

The intuition behind this approach stems from *Polytopic Linear Models (PLM)* with continuous state, where a complex non-linear system is represented as a combination of a finite set of simpler linear sub-systems [15]. Similarly, in our formalism, an action $a \in A$ may be more than a primitive operation. Rather, it may represent a subsystem with a complex underlying controller, which forces the system to follow dynamics described by T_a. In fact, by enriching the set T_A, one can guarantee that environment incoherences of interest will be well captured. As an utterly extreme example consider a dynamic system with a discrete state space. By setting T_A to be the set of permutation matrices, we essentially allow $\bar{T}_a$ to be any matrix from the polytope of stochastic matrices, and endow EA-EMT with the capability to capture any environment disturbance, be it a randomly reoccurring one or be it a disturbance localised to a particular system state. Although the relationship between the composition of T_A and its expressiveness needs not be this extreme, and in practice only small sized T_A is required, its exact properties are non-trivial. In fact, it forms a separate branch of research, where the works by Angelis [15] and Cesa-Bianchi [23] are only few representatives of a vast literature, that falls out of scope of this paper. Nevertheless, we can safely assume that T_A forms a sufficiently large polytope that includes all relevant system dynamics.

Now, the update of the weights $w_a(\cdot)$ is based on the approach of predictions with expert ensembles [23]. The intuition behind this approach is that, when making a prediction or a decision, a readily available set of feasible alternatives (the expert ensemble) can be merged together to form a prediction which is potentially better than any of the alternatives standing alone. The dynamic properties of this merger are such, that it can be readily applied even if the best prediction (or the best decision) is not stationary, but rather changes over time. This made the choice of expert ensembles particularly attractive to maintain a system model in varying, unstable environments. Specifically, in our algorithm the *expert ensemble* is the set T_A, where each expert attempts to predict the effects an action would have on the environment state. From this point of view, the weight $w_a(b)$ expresses how much the expert $T_b \in T_A$ is trusted to capture the effects of the action $a \in A$ correctly. Once EA-EMT has applied an action, a^*, it measures the discrepancy between the effect a^* had and the effect predicted by expert T_b. The lower the discrepancy, the higher will be the weight $w_{a^*}(b)$ when the next control decision is made.

Given the above principles, we have modified the standard controller algorithm. Specifically, line 13, previously directly substituted into the calculations the transition function from the provided model. Whereas now it uses a weighted combination of the matrices in T_A, which is continually tuned by the expert ensemble to improve its representation of an action's effects. The rest of the computations proceed as before until the EMT estimate, τ_{t+1}^{EMT}, of the action outcome is computed in line 20: the algorithm predicts the effects of each action on the EMT estimate, chooses the action that would bring τ_{t+1}^{EMT} closest to the reference signal τ^*, applies the action and receives an observation. At that point, the algorithm has to measure the performance of each expert, and update the weights. Now, recall that the algorithm operates in terms of subjective beliefs, the relevant effects of the action are thus those expressed in the EMT estimate τ_{t+1}^{EMT}. This means that the performance of each expert can be expressed by the distance between the estimate τ_{t+1}^{EMT} and the estimate that would have been obtained based on the expert prediction. This distance is computed in lines 22-24, and the weight of the expert is updated accordingly. Specifically, the old weight of the expert is multiplied by β^d, where $\beta \in (0, 1)$ is the parameter of the update and d is the distance above. Once all weights are updated, they are normalised to sum to 1, so that $\overline{T}_a$ at the next step will be a stochastic matrix. Notice that all these operations take time polynomial in the model parameters, such as the size of state, action and observation spaces. This makes EA-EMT a computationally efficient and scalable algorithm, an attractive property systems where environment models tend to be large.

Require:
...
Set action weight vectors: $w_a(a') \propto \delta_a(a') + \epsilon$
Set time to $t = 0$.
11: **loop**
.....
13: Set $\overline{T}_a = \sum_{a'} T_{a'} * w_a(a')$
.....
21: **for all** $a \in \mathcal{A}$ **do**
22: Set $\overline{p}_{t+1}^a = T_a * p_t$
23: Set $D_a = H\left[p_t \rightarrow \overline{p}_{t+1}^a, \tau_t^{EMT}\right]$
24: Set $V(a) = \left\langle D_{KL}\left(D_a \| \tau_{t+1}^{EMT}\right)\right\rangle_{p_t}$
25: Set $w_{a^*}(a) \propto w_{a^*}(a)\beta^{V(a)}$
26: Set $t := t + 1$

Figure 4: The EA-EMT control algorithm: only changes to the standard EMT control are shown.

4. Experimental Evaluation: Discrete State Space

To test the effectiveness of the EA-EMT algorithm, we have devised a set of comparative tests with the standard EMT Controller. The latter is a natural baseline,

as it is the only other universal control algorithm capable of complete fusion of the EPC and the model-based paradigms. In discrete state systems this is also the only baseline, as no other control algorithm can reproduce the action selection sequence of EMT. Fortunately, in the environments with a continuous state space, which we tend to in Section 5, the sequence of actions selected by EMT can be at least partially reproduced by model-following control algorithms, and we immediately use it to provide an additional baseline comparison. In all cases, we have preferred a simulated system so that the true effects of our control algorithm will not be confused with the properties of an embodied physical system.

Now, to support comparability with previous work on EMT variations, all tests were based on modifications of the Drunk Man (D-Man) domain: a controlled random walk over a linear graph (see Fig. 5 for the principle structure) with actions weakly modulating the probability (only a small discrete set of probabilities in the range $(\epsilon, 1-\epsilon)$ with $\epsilon \gg 0$ is attainable) of the left and the right steps. The domain is also partially observable, namely, instead of its true position on the graph, an agent receives as an observation a random position within the two step neighbourhood of agent's location. In turn, a task within the domain is represented by a conditional probability $\tau^*(s'|s)$, the reference signal for the controller, specifying what sort of motion through the state space has to be induced. During an experiment run, the control algorithm was provided with a Markovian environment model, $MEnv =< S, s_0, A, T, O, \Omega >$, incoherent with the true behaviour of the domain. The incoherences were created by introducing exogenous perturbations to the behaviour of the D-Man domain. In particular, three perturbations, making the model of the standard D-Man domain increasingly incoherent with the actual environment behaviour, were used: **Deviating**, where an additional deterministic step (to the right) was done; **Periodic**, where the direction of an additional deterministic step changed over time; and **Catastrophic**, where a random permutation of actions was selected $\sigma : A \rightarrow A$, so that when the controller applied action $a \in A$, the environment responded instead to $\sigma(a)$. Three baselines where obtained in various

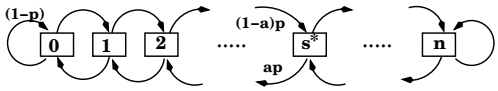


Figure 5: Principle structure of the Drunk Man domain.

combinations: standard EMT Control algorithm operating in a perturbed environment, standard EMT Control operating within an unperturbed environment, and standard EMT Control operating in a perturbed environment with its model correctly encoding the environment perturbation. At least two baselines are present in each experimental setting to provide comparative performance bounds and the

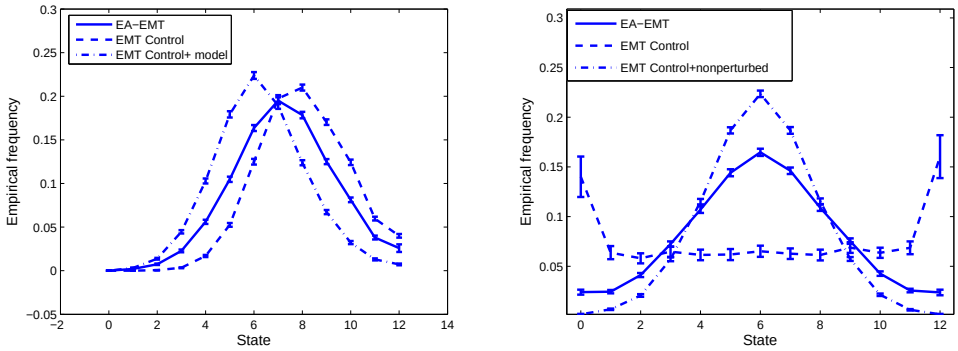
99.5% confidence envelope is depicted in all plots. In all our experiments the reference dynamics for the controller is given by $\tau^*(s'|s) \propto \delta_{s^*}(s') + \epsilon$, where $\epsilon > 0$ is small. In other words, the target prescribes that the environment should almost surely move to the ideal state s^* from any other state. In our experiments the state space was $S = \{0, ..., 12\}$, and the ideal state $s^* = 6$. Notice that, due to the probabilistic nature of the domain, any reasonable[2] control scheme set to accomplish the task would result in a bell shaped empirical distribution of the system state. Success of the control scheme can then be readily appreciated visually by the difference of the expected value and the ideal system state, as well as the standard deviation of the empirical state distribution. The empirical distribution was taken over a 200 decision step *sliding window*, to obtain statistically significant distribution shape. In turn, the overall length of experimental runs was then chosen to be sufficiently large to enable analysis of stable trends of the empirical 200-step distribution. In particular, for the *catastrophic* and the *deviating* perturbations each experiment run was 1000 steps. The necessity to obtain statistical significance while preventing the algorithm from completely stabilising, has also led to the choice of the 500 step period for the *periodic* perturbation experiments, accompanied by the 5000 step total length of each experiment run. Although alternative experimental setups were also run, varying both the sliding window size and the experiment length, their results were similar, we, therefore, omit them due to space limitations. Nevertheless, the aforementioned sequence of choices is reflected in the way our experimental results are presented: *deviating*, *catastrophic* and then *periodic* perturbations. Furthermore, to present an overall evaluation of a control scheme's performance, rather than a comparison of multiple parameters, we also measured the distance between the empirical distribution and δ_{s^*} using l_1 norm.

To further the intuition of this domain, consider once more the IFR example where the pilot has to maintain flight level within the air corridor prescribed by the ground control. If we discretise the space of possible flight levels we will obtain a linear graph depicted in Fig. 5. The transitions between the states are controlled, but are also subject to random changes in the air density or wind gusts. Ideally, the auto-pilot will need to actively return the airplane to the ideal, centre flight level.

4.1. Deviating Perturbation

In this experiment we introduce a deviating perturbation. That is, beyond the usual probabilistic step, the environment has also deterministically shifted in one direction along the linear graph. For example (referring to Fig. 5) if the system reached

[2]Unreasonable, for instance, would be choosing a constant action to equalise the left and the right step probabilities, as this would result in an almost uniform distribution, utterly defeating the controller purpose.

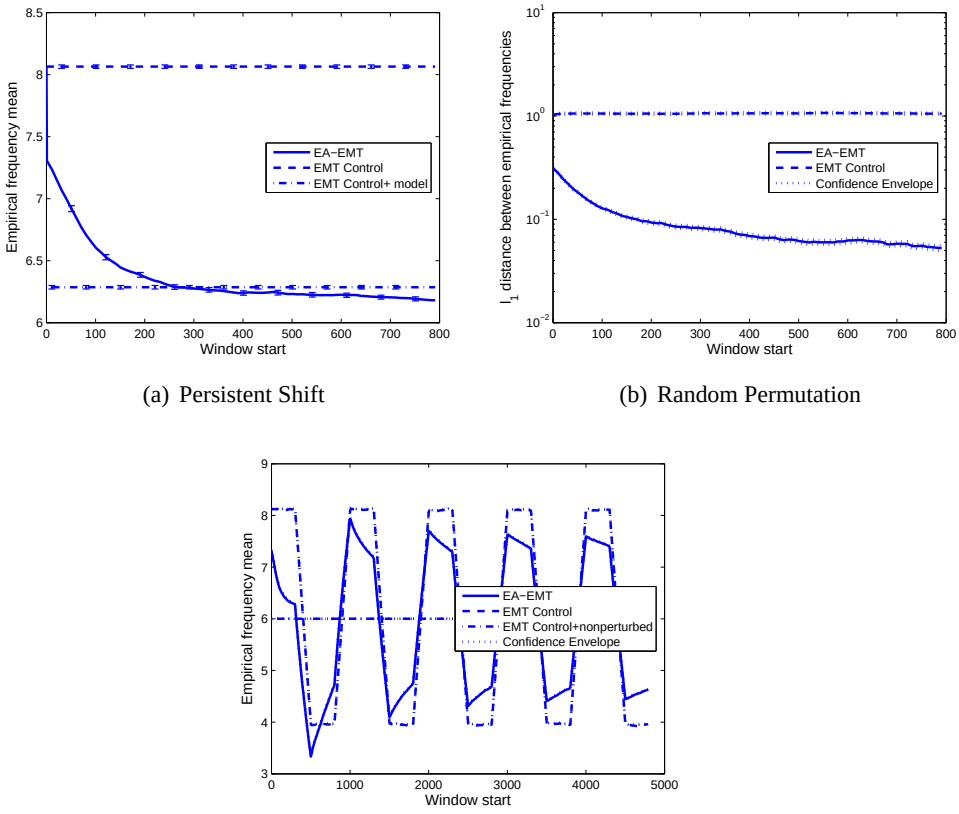


(a) Persistent shift (b) Random permutation

Figure 6: EA-EMT performance under (a) deviating and (b) catastrophic perturbations

state $k \in \{0, ..., n-1\}$, the additional step will shift it to state $k+1$. In this context, Fig. 6(a) shows the empirical distribution of system states under three control strategies: the EA-EMT controller and the standard EMT Controller equipped with the standard D-Man model (thus excluding the shift modelling), and the standard EMT Controller equipped with the environment model that explicitly captures the additional shift. The figure shows the complete empirical distribution of the EA-EMT obtained during the first 200 control choices made in this experiment, and marks a definitive improvement in performance. This can be seen from the fact that the standard EMT Control fails to enforce the reference dynamics τ^*, with the system spending the majority of its time away from the ideal state, $s^* = 6$, while EA-EMT manages to force the state distribution to concentrate closer to s^*. In fact, the distance between δ_{s^*} and the EA-EMT distribution induced in the first 200 steps is 40% less than the comparable distance for the EMT controller. This, however, does not fully reflect the adaptability of EA-EMT. To this end, Fig. 7(a) shows how the mean of the empirical distributions of the 200 step windows behave. The distributions induced by EMT Control do not change over time, resulting in straight horizontal lines depicting the constancy of the mean. On the other hand, the data shows that EA-EMT quickly adapts, the algorithm induces the empirical state distribution with the mean approaching the ideal state $s^* = 6$. In this respect, EA-EMT even slightly surpasses the performance of the standard EMT algorithm with the correct environment model. This is due to the adaptive portion of EA-EMT contributing to the tie breaking when considering similar actions – this tie breaking is rigid in EMT Control. Similar pictures occur with respect to the variance of the empirical distributions. This means that EA-EMT overcomes the model

incoherence and increasingly concentrates the state empirical distribution around the ideal state, which is exactly what the reference dynamics, τ^*, requires.



(a) Persistent Shift

(b) Random Permutation

(c) Switching Shift

Figure 7: EA-EMT adaptation to various perturbations. Notice the log-scale of the Y axis in (b).

4.2. Catastrophic Perturbation

The action space of the D-Man domain has a simple intuitive interpretation – the action sets how quickly the system state will shift left or right. The deviating perturbation did not exceed this interpretation, it simply meant that the system will naturally move in one direction faster than the other. In a way it also meant that the perturbation induced a very mild model incoherence – principally the model remained correct. However, EA-EMT can adapt to much more severe model incoherences. In fact, in the next set of experiments the environment model is completely incorrect. For each run in this experiment set a random permutation $\sigma : A \rightarrow A$

was selected. Then, when action $a \in A$ was applied, the environment reacted as if the action was $\sigma(a)$.

In more depth, Fig. 6(b) shows the empirical distributions obtained in the first 200 steps of decision making. Permuting the action breaks any connection between what EMT Control expects the action to do and what actually occurs in the environment, essentially the actions are scrambled and the EMT Control chooses a random action. This results in the algorithm's failure – the empirical state distribution is equivalent to that of applying no control at all, with higher probability of terminal states due to the failure of the respective left and right steps. In contrast, EA-EMT easily adapts and performs increasingly well, as can be seen in Fig. 7(b). Following the development of the empirical distribution within a 200 step sliding window, the figure shows the l_1 distance from the distribution formed by the standard EMT algorithm in the non-perturbed environment. This data demonstrates that EA-EMT exponentially quickly discovers the true effects of actions and approaches the performance of the EMT control in a non-perturbed environment. Even though the empirical distribution of the first 200 steps includes the first decisions made based on the scrambled model, it already recovers 70% of the performance lost due to the model incoherence and, through further adaptation, it reaches 95% recovery.

4.3. Periodic Perturbation

Finally, it is important to ensure that the algorithm can perform well in a dynamically changing environment. For example, a robot's body is subject to environmental effects, and its response to control will change accordingly. Some environment parameters, like the daily temperature variation on Lunar surface, may be extreme and persistently reoccurring. To test EA-EMT in such environments, we consider yet another perturbation: an additional deterministic step is made, and the direction of the step switches between left and right with constant period (500 control steps in our experiments). The shape of the distributions formed by the controllers are equivalent to those in the persistent shift experiment (see Fig. 6(a)), and we omit the respective graph. On the other hand, the development of the empirical distribution over time is quite different. In particular, Fig. 7(c) shows the behaviour of the mean value for empirical distributions calculated within a 200 step sliding window. While the standard algorithm literally switches from one value to another, depending on the direction of the shift, the performance of EA-EMT always shows recovery after a direction switch occurs. Notice also, that the magnitude of the mean variation at the switch point becomes significantly (25%) less for EA-EMT than the standard EMT. This suggests that, beyond its ability to recover from irrelevant adaptations, the adaptive controller version learns to reduce the control inertia. In other words the algorithm reduces the impact of the sudden change in the environment behaviour, stabilising the overall performance.

5. Experimental Evaluation: Continuous State Space

To complete the demonstration of our algorithm, we apply EA-EMT to a continuous state environment, where a task is achieved by switching between pre-specified sub-controllers. This combination of the discrete switching and the continuous switching components, clearly show EA-EMT to be a hybrid controller. Notably, neither the structure nor the principle of application change with the transition from a discrete to a continuous state space. The transition is achieved simply by replacing the finite dimensional vector, that has represented state probability distribution in the discrete case, by a Gaussian distribution to represent the state distribution of the continuous domain. Similarly, stochastic transition matrix is replaced by a conditional Gaussian distribution to capture system dynamics. Furthermore, the amount of underlying calculations grows only polynomially with the dimension of the state and the observation spaces. It is this computational scalability, and the fact that no modification is required nor made to the reasoning of the action selection procedure, which remains fully and completely intact whatever the environment dimensionality is, that grant EA-EMT almost universal applicability. It allows our algorithm to be deployed both as a direct low level controller, and as a part of a complex hierarchical hybrid controller with multiple levels of abstraction.

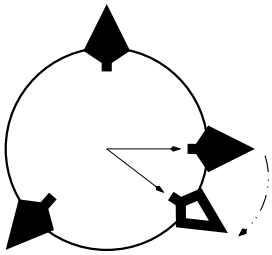


Figure 8: Hovercraft scheme.

The specific domain we chose is that of a hovercraft with three thrusters depicted in Fig. 8. Solid arrows show thruster directions, while the hollow arrow denotes a potential mistake in that thruster's model. From the perspective of our IFR example, such modelling mistake would correspond to a sudden change in the plane's responses, for instance due to a collision with a bird or a mechanical malfunction. The system generically develops in discrete time using the equation:

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \\ y_{k+1} \\ \dot{y}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & h & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \\ y_k \\ \dot{y}_k \end{bmatrix} + \begin{bmatrix} \frac{h^2}{2} & 0 \\ h & 0 \\ 0 & \frac{h^2}{2} \\ 0 & h \end{bmatrix} [v_1, v_2, v_3] \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

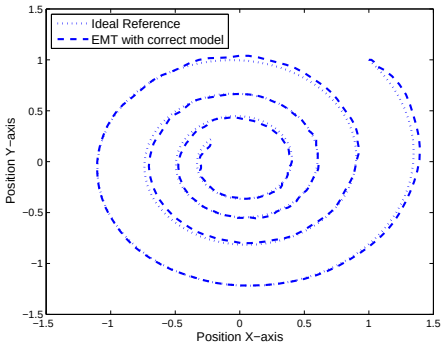
In the equation, v_i denotes the directional force distribution of a thruster, u_i

its basic level of activity, and h denotes the time span during which the thrust was applied. To further underline the use of EA-EMT as a switching mechanism of a hybrid controller, we restrict u_i in our experiments to a finite discrete set. Specifically, we used 5 activation levels between 0.2 and 1.0 in equal intervals, and only one thruster could have a non-zero activation at any time, so that total of 15 distinct joint activations were possible. This naturally simulates the situation that occurs in hybrid systems, where an action corresponds to the application of a distinct sub-system controller, rather than a choice of a continuous value control signal. Two configurations of thrusters were used. Configuration A, $[v1, v2, v3] = \left[\begin{smallmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{smallmatrix}\right]$, that corresponds to the solid arrows in Fig. 8; and configuration B, $[v1, v2, v3] = \left[\begin{smallmatrix} 1 & 0 & -1 \\ -0.3 & 1 & -1 \end{smallmatrix}\right]$, that corresponds to a structural failure of a thruster depicted by the hollow arrow. In all experiments, while the controller algorithm was given either the environment model with thruster Configuration A or B, the actual motion of the hover craft was always simulated using Configuration A. This discrepancy allowed us to test the performance of our algorithm under a deviating modelling incoherence.

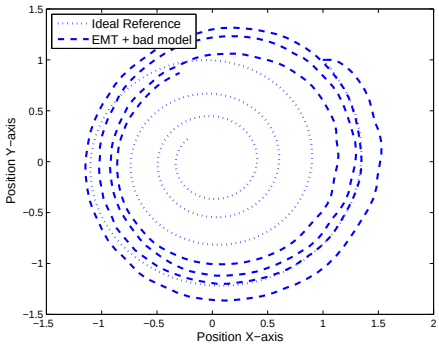
Now, to provide a quantitative performance measure, we have set several control algorithms with the task to simulate a gradual spiralling descent towards zero from rest at coordinates $[1, 1]$, which we have described by an autonomic linear system with the equation given below. Recalling once more our IFR scenario, such system would correspond, for example, to the necessary relative properties of the altitude and speed of the airplane, as well as their development in time, during a landing procedure. As before, h denotes the time span of a single step, while λ denotes the decay of the spiral and θ the rotation angle of a single step of the system.

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \\ y_{k+1} \\ \dot{y}_{k+1} \end{bmatrix} = \begin{bmatrix} \lambda\cos(\theta) & 0 & -\lambda\sin(\theta) & 0 \\ \frac{2}{h}(\lambda\cos(\theta)-1) & -1 & -\frac{2}{h}\lambda\sin(\theta) & 0 \\ \lambda\sin(\theta) & 0 & \lambda\cos(\theta) & 0 \\ \frac{2}{h}\lambda\sin(\theta) & 0 & \frac{2}{h}(\lambda\cos(\theta)-1) & -1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \\ y_k \\ \dot{y}_k \end{bmatrix}$$

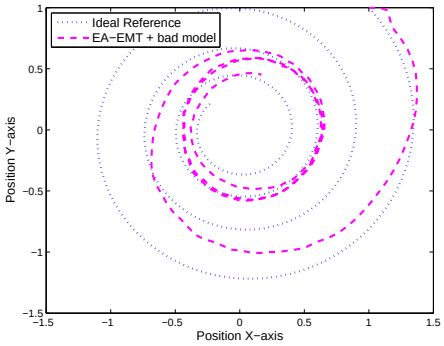
In more detail, the algorithms we have considered were EMT, EA-EMT and a discrete Model Follower Controller (MFC). The latter algorithm has been selected for its robustness and ubiquity of its principle (see e.g. [7, 14, 16]), making it suitable to produce a baseline comparison. The MFC algorithm operated in the usual manner, specifically, given the current hover coordinates, the algorithm selected thrust to minimise the discrepancy between the outcome predicted by the task's equation and the equation of the hovercraft's model. We have tuned EMT initialisation and task representation parameters so that, for the Configuration A thrusters model, its decisions coincide with MFC. We conjecture, in fact, that EMT is formally a more general approach than MFC, in that EMT can always be tuned to reproduce MFC's behaviour. The resulting hovercraft trajectory is depicted in



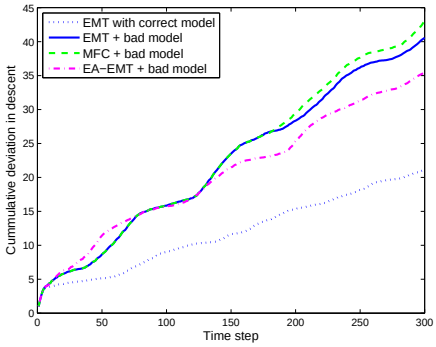
(a) EMT (and MFC) with Configuration A (correct) model



(b) EMT with Configuration B (wrong) model



(c) EA-EMT with Configuration B model



(d) Cumulative error of controllers

Figure 9: Hovercraft trajectories under various controllers algorithms and controller models and their cumulative error. In all cases physical simulation adopts thruster Configuration A.

Fig. 9(a). The dotted line represents the ideal trajectory that could have been obtained if the thrusts u_i where continuous, rather than discretised. The Fig. 9(a) also demonstrates that the task we posed can be indeed solved by an application of the standard EMT controller or MFC, forming a performance baseline where the environment develops exactly as the controller's model describes it. In turn Fig 9(b) and Fig 9(c) depict the performance of the EMT and EA-EMT algorithms provided with Configuration B (wrong) thruster model. Due to the aforementioned EMT tuning, even under Configuration B the trajectories of EMT and MFC are extremely similar, and we omit the latter due to space limitations.

However, Fig.s 9(a), 9(b), 9(c) can only provide an intuition as to how various algorithms cope with the task. To clearly distinguish and evaluate the control algo-

rithms' performance we have calculated the cumulative error of these trajectories. That is, for each experiment run at each time step we have computed the difference between the system state that results from the discrete level of thrust chosen by a control algorithm and the system state that resulted from the application of the analytically computed continuous thrust. Fig. 9(d) depicts the accumulation of that discrepancy over time. Initially slightly worse, due to slack expert ensemble initialisation, over time EA-EMT significantly outperforms both EMT and MFC. Perhaps to further underline the strength of the EMT-based approach in general, notice that, under model incoherence, even the standard EMT outperforms MFC, and aggregates trajectory error at a lower rate. Notice that due to thrust discretisation zero error is unachievable, as is witnessed by the error accumulation of EMT (and MFC since they coincide in this case) with the correct Configuration A thrusters model. Furthermore, we have calculated the accumulated thrust utilised by all algorithmic solutions when faced with the bad Configuration B model. The results are given in Table 1. The data confirms that EA-EMT recovers significant portion of losses due to model incoherence. Furthermore, to complete our investigation, we have also measured the amount of energy consumed by the control algorithms in terms of the applied thrust vector norm (see the third column of Table 1). Although at first sight it may look that EA-EMT has conserved some energy by a faster move to a lower spiral loop, in fact, and unlike a passive descent under a gravitational pull, maintaining a tighter trajectory at the same speed necessitates ever higher energy levels to counter the centrifugal force. We are, therefore, inclined to conclude that the energy conservation is an algorithmic property of EA-EMT.

Algorithm/Thruster Configuration	Total Energy	Total Trajectory Discrepancy
EMT(MFC)/Configuration A	132.6	21.067285
MFC/Configuration B	150.8	43.036976
EMT/Configuration B	140.4	40.56558
EA-EMT/Configuration B	117.6	35.400698

Table 1: Total trajectory discrepancy and energy consumption over 300 steps

6. Conclusions and Future Work

In this paper we present the Ensemble Action EMT algorithm – a control solution that has three important properties: it is an *egocentric perceptual controller*; it is *a universal model-based controller*; it is an *on-line model calibrating controller*; and it is a *hybrid controller* capable of operating in mixed discrete-continuous or hierarchical action abstraction domains. As an EPC solution, EA-EMT describes the control task and the optimality criteria in terms of the agent's interpretation

of sensory input, thus enabling an autonomous agent to formulate internal control tasks, rather than just following an external command. Being a universal model-based solution, EA-EMT is capable of utilising a given environment model, but is not bound to one model or one environment in particular. Finally, on-line model calibration enables EA-EMT application to changing or simply poorly modelled environments.

EA-EMT is unlike other adaptive control algorithms based on expert ensembles, where experts directly produce actions or plans to be fused (e.g. [16, 27, 28]) [3]. Rather, EA-EMT operates in two distinct modules: the expert-based model estimation and a control algorithm that utilises that model. This enables greater design flexibility, and generalisation, particularly with respect to the model type that experts produce. For instance, in robotic soccer – a domain well known to attract hybrid control solutions – environment models are frequently found at the edge of logic and probability-based approaches, especially in opponent plan recognition [29–31]. Nevertheless, because of the employed probabilistic notions, these models can still be successfully weighted and fused, albeit necessitating an inference process to do so [29–31]. Furthermore, they still can be evaluated and compared via the Kullback-Leibler divergence. As a result, EA-EMT can be expanded to operate even in such a highly complex and dynamic environment as robotic soccer. In fact, the on-line adaptability of the EA-EMT and its computational efficiency will be particularly useful.

Finally, we also would like to investigate the possibility of altering the weight adaptation to include *forgetting* (inherent tendency of weights to equalise over time) and *update extrapolation* (simultaneous weight modification of actions with similar effects). In particular, forgetting and update extrapolation can serve well in combination with learning approaches. Specifically, we would like to consider the situation where a library of behaviour primitives (or experts) is dynamically composed (see e.g. MOSAIC [16]). In this case, the appearance of new control sub-systems can be handled better, if the expert mixture can be initialised, rather than learned over time, by means of *update extrapolation*. Similarly, older sub-systems can be phased out more effectively if *forgetting* is applied.

[1] W. T. Powers, Behavior: The control of perception, Aldine de Gruyter, 1973.

[2] S. Thrun, Bayesian landmark learning for mobile robot localization, Machine Learning 33 (1) (1998) 41–76.

[3] Notably, these methods, particularly Haruno et al. [16], naturally assume control signal metric, which we do not, therefore allowing for more abstract action spaces

[3] A. Lazanas, J. claude Latombe, Landmark-based robot navigation, in: Algorithmica, 1992, pp. 816–822.

[4] R. C. Arkin, Behavior-Based Robotics, MIT Press, 1998.

[5] M. M. Taylor, Editorial: Perceptual control theory and its application, International Journal of Human-Computer Studies 50 (6) (1999) 433–444.

[6] W. T. Bourbon, Perceptual control theory, in: H. L. Roitblat, J.-A. Meyer (Eds.), Comparative approaches to cognitive science, MIT Press, 1995.

[7] R. F. Stengel, Optimal Control and Estimation, Dover Publications, 1994.

[8] Z. Sun, S. S. Ge, Analysis and synsthesis of switched linear control systems, Automatica 41 (2005) 181–195.

[9] M. Morari, J. Lee, Model predictive control: Past, present and future, Computers and Chemical Engineering 23 (9) (1999) 667–682.

[10] J. Morningred, B. Paden, D. Seborg, D. Mellichamp, An adaptive nonlinear predictive controller, Chem. Eng. Sci 47 (4) (1992) 755–765.

[11] R. S. Sutton, Integrated architectures for learning, planning, and reacting based on approximating dynamic programming, in: Proceedings of the ICML, 1990, pp. 216–224.

[12] P. Poupart, N. Vlassis, Model-based bayesian reinforcement learning in partially observable domains, in: Proceedings of the ISAIM, 2008.

[13] R. Jaulmes, J. Pineau, D. Precup, A formal framework for robot learning and control under model uncertainty, in: IEEE ICRA, 2007.

[14] L. Giovanini, A. W. Ordys, M. J. Grimble, Adaptive predictive control using multiple models, switching and tuning, International Journal of Control, Automation, and Systems 4 (6) (2006) 669–681.

[15] G. Angelis, System analysis, modelling and control with polytopic linear models, Ph.D. thesis, University of Eindhoven (2001).

[16] M. Haruno, D. M. Wolpert, M. Kawato, MOSAIC model for sensorimotor learning and control, Neural Computation 13 (2001) 2201–2220.

[17] M. Karny, T. V. Guy, Fully probabilistic control design, Systems and Control Letters 55 (4) (2006) 259–265.

[18] A. Robertsson, On observer-based control of non-linear systems, Ph.D. thesis, Department of Automatic Control, Lund Institute of Technology (1999).

[19] M. R. James, S. Singh, M. L. Littman, Planning with predictive state representations, in: Proceedings of the ICMLA, 2004, pp. 304–311.

[20] Z. Rabinovich, J. S. Rosenschein, Extended Markov Tracking with an application to control, in: The 1st MOO Workshop, 2004, pp. 95–100.

[21] Z. Rabinovich, J. S. Rosenschein, Multiagent coordination by Extended Markov Tracking, in: The 4th AAMAS, 2005, pp. 431–438.

[22] A. Adam, Z. Rabinovich, J. S. Rosenschein, Dynamics based control with PSRs, in: 7th AAMAS, 2008, pp. 387–394.

[23] N. Cesa-Bianchi, G. Lugosi, Prediction, learning, and games, Cambridge University Press, 2006.

[24] S. Kullback, Probability densities with given marginals, The Annals of Mathematical Statistics 39 (4) (1968) 1236–1243.

[25] E. Cramer, Conditional iterative proportional fitting for Gaussian distributions, Journal of Multivarate Analysis 65 (2) (1998) 261–276.

[26] S.-C. Fang, J. R. Rajasekera, H. S. J. Tsao, Entropy Optimization and Mathematical Programming, Kluwer Academic Publishers, 1997.

[27] E. Even-Dar, S. M. Kakade, Y. Mansour, Experts in a Markov decision process, in: NIPS, 2004.

[28] B. Argall, B. Browning, M. Veloso, Learning to select state machines using expert advice on an autonomous robot, in: ICRA, 2007.

[29] H. Bui, A general model for online probabilistic plan recognition, in: 18th IJCAI, 2003, pp. 1309–1315.

[30] P. Riley, M. Veloso, Recognizing probabilistic opponent movement models, in: The 5th RoboCup Competitions and Conferences, 2002.

[31] D. V. Pynadath, M. P. Wellman, Probabilistic state-dependent grammars for plan recognition, in: 16th UAI, 2000, pp. 507–514.