Robot-Control Based on Extended Markov Tracking: Initial Experiments

Zinovi Rabinovich Jeffrey S. Rosenschein
School of Engineering and Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, Israel
{nomad, jeff}@cs.huji.ac.il

Abstract

Extended Markov Tracking (EMT) is a computationally tractable method for the online estimation of Markovian system dynamics. In this paper, we present our initial experimentation with EMT-based control applied to robotic motion. In our experiments, a robot uses a predetermined mapping of the world onto an abstract model, over which EMT Control is applied; this dictates the choice of an abstract action, which in turn is mapped back into actual robot operation. Simulations in which a robot was constrained to follow a target show that although the abstract model was (intentionally) only weakly coherent with the real dynamics of the robot's world, EMT Control was able to provide reasonable performance.

We also demonstrate that EMT-provided data provides sensible information for action model calibration. We do so by constructing a calibration scheme based on a training technique and simple data statistics. The scheme is then validated by carrying out additional robot motion control simulations, using the calibrated abstract model.

1 Introduction

Consider a sophisticated cleaning robot, moving along corridors, sweeping floors, and picking up crumpled pieces of paper. Although over time its wheels are not as steady as they once were, and its motors cannot spin as fast, the robot is adaptive, adjusts to its degraded performance, and works on. Suddenly, one of the wheels runs over some spilled coffee; for a moment the wheel begins to spin, but the robot quickly corrects for it.

The seemingly simple scenario above requires much more sophistication than may, at first, be apparent. Whatever the robot's initial world model, this model's coherence with the real, physical world will deteriorate over time — effectors and sensors are subject to wear and tear, and their performance will change. Another source of incoherence, though usually point-wise in time, are sudden, unpredictable changes in the environment. Although the robot's world model may still recognize the resulting situations, it cannot predict them in advance, thus momentarily becoming incoherent.

Each of these types of incoherence has a typical solution in the literature. Prolonged, systemic incoherence is generally solved by *model calibration* (e.g., [14, 13]), while sudden, irregular incoherence is the domain of contingency planning (e.g., [1, 15, 4]). These represent, in fact, two general approaches to the problem of maintaining coherence. First, one would like to learn from experience, to avoid being surprised in the future; second, one has to be able to deal with unlikely or surprising occurrences.

Although systemic incoherence occurs over time and has a persistent character, one may still choose to view it as a kind of unpleasant surprise. In that way, contingency planners or controllers can be applied to solve both prolonged and point-wise model/environment incoherence. Especially beneficial under these circumstances would be one kind of contingency planning, known as continual planning [5], where the planning algorithm has the ongoing ability to readjust its future actions. In the extreme case, this becomes a controller algorithm, and for our research on robot behavior modulation, we chose a controller based on Extended Markov Tracking (EMT) [11].

1.1 Overview of the Paper

The paper is organized as follows. We first present Extended Markov Tracking (EMT) (Section 2), then show how EMT would be applied to control, both in general and to the robot domain, specifically (Section 3). We then apply the EMT Controller to a robot control task with a relevant, but incoherent, world model, to simulate (for example) the effect of prolonged wear and tear on actuators or sensors (Section 4). Experiments show that EMT Control can in fact provide an adequate response to continuing inconsistency between predictions of an internal model and the actual world's response. We then proceed (Section 5) to apply EMT to statistical model calibration, and verify the resulting world model by repeating the robot control task. The data from these experiments demonstrate the effectiveness of EMT-based calibration.

2 Extended Markov Tracking (EMT)

Determining, from observations, the behavioral trends of an environment during interaction with that environment is known to be a hard problem. It is universally treated by complex models that require significant amounts of data and involve a large computational burden [12, 2, 10]. However, usually interaction is either short-lived and discrete in time or, although internally complex, exhibits simple external trends. Thus, a simple Markovian model will often be sufficient for our needs.

2.1 Model

A Markovian environment is described by a tuple
$< S, A, T, O, \Omega, s_0 >$, where:

- S is the set of all possible environment states;
- s_0 is the initial state of the environment (which can also be viewed as a distribution over S);
- A is the set of all possible actions applicable in the environment;
- T is the environment's probabilistic transition function: $T : S \times A \to \Pi(S)$. That is, $T(s'|a, s)$ 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. This is what the sensor input would look like for an outside observer;
- Ω is the observation probability function: $\Omega : S \times A \times S \to \Pi(O)$. That is, $\Omega(o|s', a, s)$ is the probability that one will observe o given that the environment has moved from state s to state s' under action a.

In many current approaches, including Markov Decision Problems (MDPs), "tracking" means a continual estimation of the system state as seen through the distortion of observations. We believe this approach is insufficient, since it actually disregards the *continual* nature of tracking. Rather than understanding *where* the system is moving, it is of greater importance to know *how* it is moving. In other words, instead of tracking the system state, one needs to track the way the system changes, i.e., the dynamics of the system.

Recently, a practical algorithm, Extended Markov Tracking (EMT) [11], was introduced to track system dynamics in an abstract Markov model of the environment, and utilized in a two-level control scheme, the EMT Controller. Below, we describe in detail the EMT algorithm, both because it is novel, and because its specific details are exploited in this work. Then, in Section 3, we describe the EMT Controller with an application to robot motion.

2.2 Tracking of Dynamics

EMT tracks the system dynamics by continually performing a conservative update of a system dynamics estimate $PD : S \to \Pi(S)$. After every development epoch of the system, the EMT algorithm searches for an explanation dynamics D that can account for the change in system state. An explanation that differs least from the old dynamics estimate then becomes the new estimate.

The distance between old and new system estimates is measured by EMT using the Kullback-Leibler (KL) divergence function [3]:

$$D_{KL}(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)}.$$

From the information theoretic point of view, this is the price one will have to pay for using distribution q to encode a source distributed by p. In our case, the old dynamics estimate is the encoding we use, while the new dynamics estimate stands in for the true source distribution. In a sense, KL divergence would measure the "regret" of using an old estimate in light of new evidence and a new estimate of the system dynamics. Conservative update dictates the minimization of this regret.

However, to complete the measure and system dynamics tracking, one has to maintain one auxiliary kind of information: beliefs about the current system state. Initial beliefs are given by the definition in the environment tuple. Assume that at some stage we believe $p \in \Pi(S)$, and we obtain an observation $o \in O$ after action a. Then we can update our beliefs using the Bayesian formula:

$$\tilde{p}(s) \propto \Omega(o|s,a) \sum_{s'} T(s|a,s')p(s').$$

Notice that, although action is present in computations of the state estimator, it serves as an input to the procedure, and not as an unknown parametric component.

Given that we keep track of our beliefs about the environment state, we can do the same with regard to the observed dynamics. Our prior beliefs in this case might obviously depend on the domain, but in the absence of any other preference, one commonly accepted prior belief is the uniform distribution. That is, we assume at the beginning that anything can happen in the environment with equal likelihood. The other popular alternative is a 'static' environment, that is, the assumption that the system state does not change.

Thus, we can formally define EMT update as follows. Denote old beliefs about the exhibited system dynamics by $PD(s'|s)$, and the new beliefs by $D(s'|s)$, that explain the change in system state that is $\tilde{p}(s') = \sum\limits_s D(s'|s)p(s)$, where $p(\cdot)$ and $\tilde{p}(\cdot)$ are, respectively, the old and the new estimations of the system state. D is the conservative update we seek if it exists:

$$\begin{gathered} D(s'|s) = \arg\min_{Q(s'|s)} \left\langle D_{KL}(Q(s'|s)\|PD(s'|s))\right\rangle_{p(s)} \\ s.t. \\ \forall s' \;\; \tilde{p}(s') = \sum_s Q(s'|s)p(s) \\ \forall s \;\; \sum_{s'} Q(s'|s) = 1 \end{gathered}$$

The optimization problem above is a convex one, and can be solved efficiently in time and space polynomial in the description of the system, e.g., numerically by gradient descent or alternatively by an internal point algorithm.

Notice that the Kullback-Leibler extension for Markovian dynamics, $\left\langle D_{KL}(Q(s'|s)\|PD(s'|s))\right\rangle_{p(s)}$, can also provide *failure detection* for controllers. This is used in a two-level control architecture into which EMT is incorporated, as described below.

3 Control with EMT

The EMT control framework we use consists of two continually interacting layers:

- The *strategic* layer is concerned with the following question: given a high level goal, what kind of system dynamics will suffice to achieve it;

- The *tactical* layer has no concern whatsoever with the high level goal. Rather, it attempts to ensure that the system dynamics of the changing state indeed match the one desired by the strategic layer.

Failure of the tactical layer is reported back to the strategic layer, forming a closed control loop.

Although the strategic/tactical paradigm is evident in many old and new planners and controllers [6, 1, 15, 9], the EMT Controller is among the pioneers in shifting responsibilities to the tactical layer, which typically is present only in a degenerate sense within existing approaches. The tactical layer is implemented in the EMT Controller by means of Extended Markov Tracking as presented above, and action selection is based directly on the system dynamics. Thus the strategic layer has only to dictate the ideal system dynamics $r : S \rightarrow \Pi(S)$ (or so-called tactical target), and the tactical layer will provide the complete means to achieve it.

Formally, the EMT-extended tactical solution, or EMT Controller, operates as follows. Denote by $H(\bar{p}, p, PD)$ the EMT procedure of obtaining the optimal explanation for transition between belief states p and $\bar{p}$ with respect to the dynamics-prior PD. Denote as p_t the belief about the system state at time t, and PD_t the beliefs about the exhibited system dynamics at time t. Also, let T_a be the environment transition function restricted to action a ($T_a p$ thus becomes a matrix applied on a vector). Then, the action of choice in the EMT Controller is:

$$a^* = \arg\min_a \left\langle D_{KL}\left(H(T_a p, p, PD) \| r\right)\right\rangle_p$$

While the overall EMT Controller algorithm may be written as follows:

0. Initialize estimators:

   - the system state estimator $p_0(s) = s_0 \in \Pi(S)$,
   - system dynamics estimator

   $$PD_0(\bar{s}|s) = prior(\bar{s}|s)$$

   Set time to $t = 0$.

1. Select action a^* to apply using the following computation:

   - For each action $a \in A$ predict the future state distribution $\bar{p}^a_{t+1} = T_a * p_t$;
   - For each action, compute

   $$D_a = H(\bar{p}^a_{t+1}, p_t, PD_t)$$

   - Select $a^* = \arg\min_a \left\langle D_{KL}\left(D_a \| r\right)\right\rangle_{p_t}$

2. Apply the selected action a^* and receive an observation $o \in O$.

3. Compute p_{t+1} due to the Bayesian update.

4. Compute $PD_{t+1} = H(p_{t+1}, p_t, PD_t)$.

5. Set $t := t + 1$, goto 1.

In essence, the tactical algorithm above utilizes extended Markov tracking both to guide its action selection, and to ensure that the exhibited behavior indeed concurs with the given preference $r : S \rightarrow \Pi(S)$.

3.1 Tactical Target Design

Designing a tactical target may potentially be a complicated matter. However, our experiments have suggested that its construction (so as to get particular behavior from an agent) is considerably more intuitive than, for example, construction of a reward scheme for Markov Decision Problems (MDPs).

MDPs use the same environment model as EMT. However, instead of using a tactical target, the model is complemented by a *reward scheme*. This is a function, $c : S \times A \times S \rightarrow \mathcal{R}$, that is defined so as to specify a reward for all system transitions. A task is established with regard to that reward — usually the task is to maximize the expected total of the (discounted) reward. The idea behind this is to encourage self-interested agents to exhibit required behavior; that behavior is what will optimize the agent's reward.

Now consider the following problem: a drunk man stumbles along a path between his home and nettle bushes, making random steps left (home) and right (away from home). Assume that we are given a set of actions that are capable of tilting the probability balance between left and right steps, and our goal is to keep the man (roughly) equidistant from both his home and the bushes.

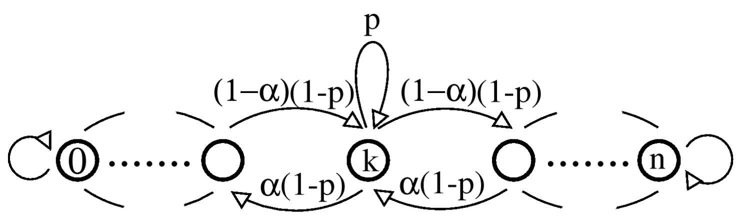

Figure 1: Drunk Man state transition diagram. α is subject to action effect.

Assume for a moment, that unlike in EMT's environment model, we do have complete knowledge of the Drunk Man's position. In this case, given a reward scheme, the problem can be easily solved via dynamic programming. But what would a suitable reward scheme be? A first intuitive design might be the following: since we would like the man to reach the middle of the path and stay there, let us reward 1 for being in the middle of the path at the next moment, and ϵ otherwise — for trying.

However, this reward scheme does not make the MDP solution behave as required. The Drunk Man moves randomly, and although motion trends can be affected, he is unlikely to stay put. This leads the MDP solver to a *controlled fall* solution: the Drunk Man will leave the middle of the path anyway (applying maximal force to make sure of where he ends up), then apply the inverse to make sure he comes back. An example of the distribution over positions along the path can be seen in Figure 2. It shows that the MDP solution forces the Drunk Man away from the vicinity of the path's middle for a large portion of time.

However, the intuition that failed for MDP works correctly for EMT control, even if the Drunk Man's position is known only partially. Normalizing the reward function, we easily obtain a tactical target, and EMT creates a distribution as we have intended (Figure 3).

3.2 Robot Motion with EMT Control

Computerized algorithmic control of a robot, unlike low-level reactive control, requires a discretization of sensory information received by the robot, and later the carrying out, by the robot's actuators, of a discrete action description. In effect, the actual robot serves as an interface between the real world and a discrete controller, transforming sensory information to fit the internal environment model used to make decisions.

Two adjustments are required, in order to apply the approach of EMT Control to robotic motion.

If the internal model is chosen to be a Markovian model $< S, A, T, O, \Omega, s_0 >$, then sensory discretization requires mapping real world sensations onto the observation set O. This makes the set of states S abstract — it is only indirectly

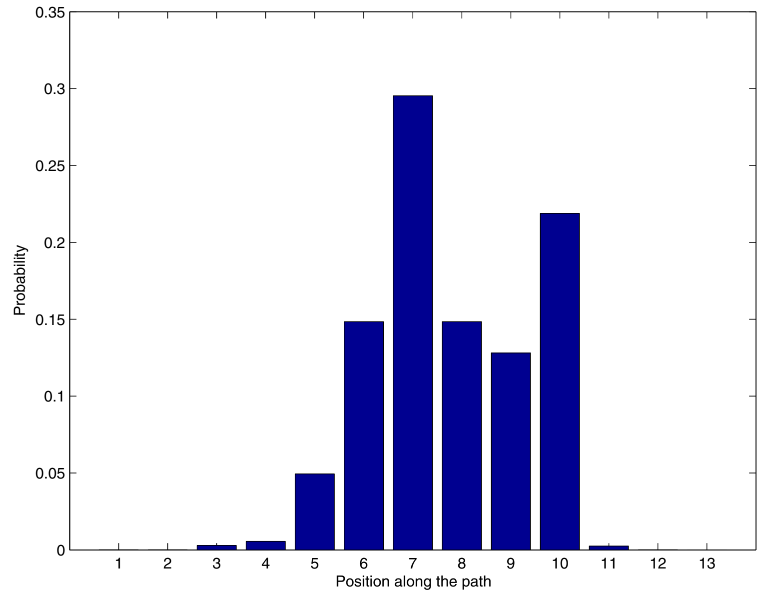

Figure 2: Drunk Man state distribution under MDP solution control

connected to the real world, and all $s \in S$ gain meaning through the observation probability function Ω.

Abstraction of S has several consequences. First, a control algorithm based on a Markovian model would usually require the transition function $T : S \times A \to \Pi(S)$, also called the "action model," to be strongly coherent with the environmental dynamics of the system state's *meanings*, determined by O and Ω. However, the EMT Controller manages to achieve reasonable, though not optimal, results even with an inaccurate action model (Section 4).

Second, if EMT Control is to be applied, its strategic layer has to create a tactical target by "reverse engineering" — a high-level target is described by limitations on *observations*, and the strategic layer uses Ω to convert it into system state dynamics, $r : S \to \Pi(S)$.

Other than the two adjustments above, the basic EMT Controller remains unchanged for robot control.

4 Target-Following Experiment

We designed a straightforward simulation experiment, within the Player/Stage simulation environment [7], to see whether EMT Control would be effective for robot control, and whether it would be able to cope with an inaccurate action model.

In the experiment, two independent EMT Controllers, $EMTC_1$ and $EMTC_2$, were applied to linear and rotation speed modulation of a single (simulated) Pioneer-2X robot, with the task of following another robot's motion. The sensory information was received through a *blob finder* — an on-robot camera with basic image analysis that makes possible the detection of color blobs within the picture. Camera information was approximately mapped onto the observation sets: color blob relative area and centering within the picture. Thus, the observation distributions provided state meanings of linear distance for $EMTC_1$, and angular distance for $EMTC_2$.

To follow (but not capture) an object, a robot must solve two balancing problems: stay within a certain distance from the object, and stay directed straight at it.

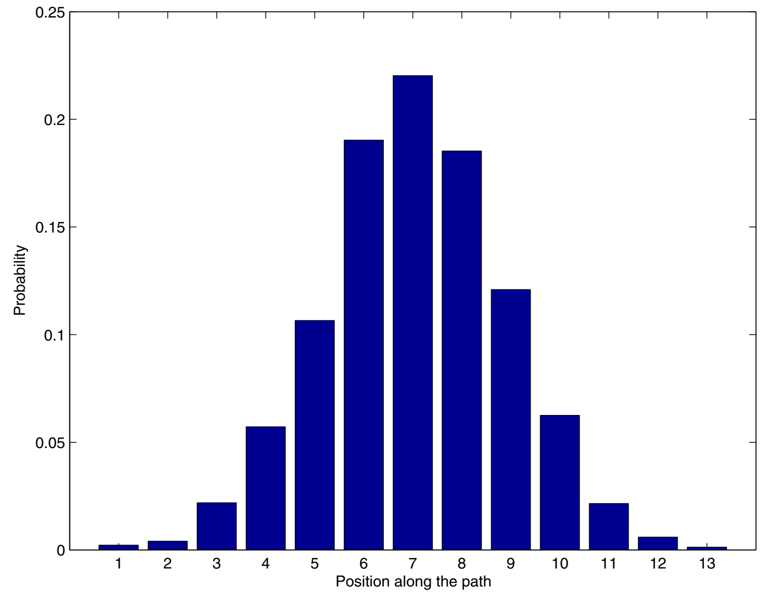

Figure 3: Drunk Man state distribution under EMT control

Since the previously described Drunk Man Walk (DMW) domain is also a balancing problem, and EMT has been successful at it, we adopted the DMW as the internal model for the $EMTC_1$ and $EMTC_2$ controllers. Their balancing acts translated into a tactical target in a natural way, essentially projecting all states in S_i onto $s = \frac{|S_i|}{2}$.

Notice, however, that the transition function of a DMW (our action model) is not entirely coherent with the real world's reactions, and with the meaning of system states. This incoherence serves our needs, since we want to verify the performance of EMT-control with respect to an imperfect model. For example, $EMTC_1$ uses system states S_1 to represent distance from a followed object, and since their meaning is based on the camera picture, they represent an irregular set of distances. This lack of regularity would dictate uneven transition probabilities from different states under the same action, but this is not the case in the DMW model, where all system states are symmetric. A similar problem exists within $EMTC_2$, because change in the visible angle of an object may depend on its linear speed as well as its angular speed. However, this is not accounted for by the DMW model within $EMTC_2$.

This incoherence indeed influences the EMT Controller's performance; however, it was still able to successfully perform the tracking task. A sample run of the EMT-controlled robot can be seen in Figure 4, depicting three positions of the robots at different times. In this run, the followed robot (which we shall call the *prey*) performed a constant loop, and the EMT-controlled robot that followed it (the *predator*) managed to capture this motion.[1] The predator traces a smaller loop, concentric with the one traced by the prey. However, due to action model incoherence, the predator did not perform optimally and reacted to the change in the prey's position with insufficient correcting actions. Effectively, it resulted in the predator being significantly further away from the prey than was required by the tactical target.

[1]Predator/Prey terminology is used here for naming convenience only, since there is no actual 'capture' intended.

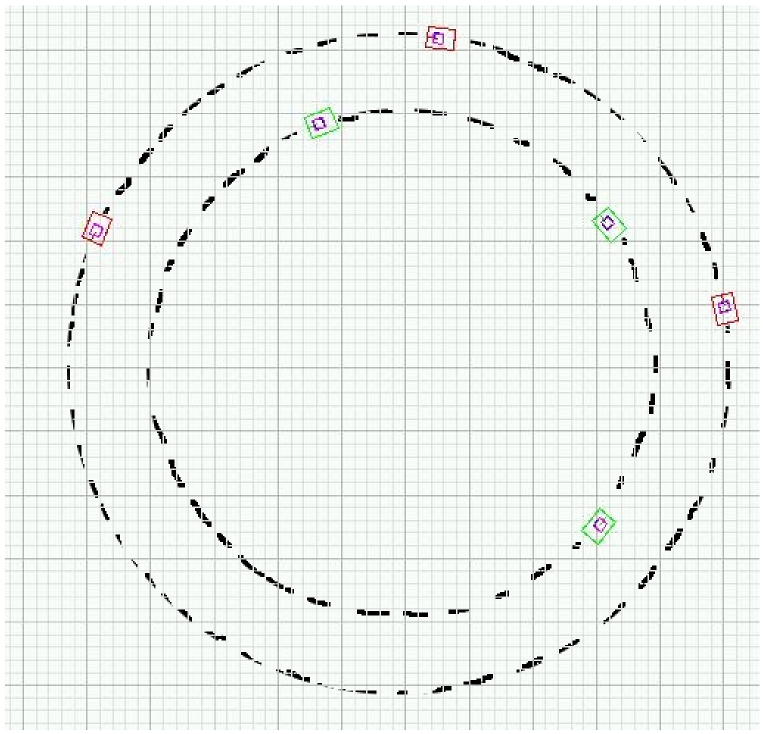

Figure 4: Target Following with a Weakly Coherent Model

5 EMT-Based Action Model Calibration

Although EMT robot control suffers from an inaccurate action model, EMT can also provide a remedy to the model — calibration. The simplest way to calibrate a model would be to accumulate data on state transitions and build an action model from statistics. However, this is impossible for us, since we do not have complete knowledge of the state, neither of the abstract nor of the real environment.

Although our state information is filled with uncertainty, it is only required to build the dynamics estimator, and has little importance on its own. This is exactly what Extended Markov Tracking was designed for: estimate dynamics, given only noisy state knowledge. Then, EMT-based action model calibration becomes a matter of data accumulation and statistics:

0. Assume a Markovian environment model $< S, A, T, O, \Omega, s_0 >$ to be calibrated. For each action $a \in A$ let:
   - $\bar{t}_a$ be the accumulator of the EMT dynamics estimators, initialized $\bar{t}_a = T_a$.
   - N_a the counter, initialized $N_a = 1$.

   Set time $t = 0$
1. Select and perform an action $a \in A$.
2. Assume that system state beliefs changed from $p \in \Pi(S)$ to $\bar{p} \in \Pi(S)$.
3. Using the EMT procedure, obtain an explanation $D = H(\bar{p}, p, Prior)$.

4. Let $\bar{t}_a := \bar{t}_a + D$, $N_a := N_a + 1$ and $t := t + 1$

5. If $t \geq t_{calibration}$

   - For all $a \in A$ let $T_a = \frac{1}{N_a}\bar{t}_a$

   else goto 1

This is, however, an incomplete scheme — there are still two questions to be handled: selection of $Prior$ in 3 and action selection in 1. To handle them, we draw upon the features of EMT and previous techniques for action model calibration [13, 14].

If the $Prior$ in the EMT update procedure is a uniform distribution, the procedure formally becomes a maximum entropy update, and overall calibration can be seen as the likelihood maximization procedure as in [13]. However, this would completely discard the previous abstract model. Instead, since EMT allows us to look directly into the dynamics of the transition, we prefer to use the accumulated estimator that is $Prior = \frac{1}{N_a}\bar{t}_a$.

To complete the action selection, we use a training procedure for the robot as was done in [14]. The robot continually spans the state space using random action selection, and its repeated application. However, since in our case the system model is discrete and finite, we can refine the process of action selection. For each possible observation we keep a count of actions performed, and set the probability of an action being selected inversely to the values of these counters. This allows the robot to experience the full range of observation/action pairs uniformly, and improves the calibration procedure.

5.1 Calibrated Target Following

To test the EMT-based action model calibration, we attempted to correct the Drunk Man Walk model that was used in the target tracking experiment above. The $EMTC_1$ controller, responsible for linear speed modulation, was calibrated for a *stationary* target, and the resulting environment model was then used in tracking a moving target. By using a stationary target for calibration, we attempted to avoid over-fitting to a particular type of motion. During the calibration, the robot alternately walked to and from the target, switching direction if the target became point-like (went too far away), or if the target effectively blocked the camera view (went too close). This procedure is essentially as in [14], with variation of stopping criteria.

Even without the calibration of the angular controller, the ability of the system to follow a moving object was greatly improved. This can be seen in the example (Figure 5) of the prey moving circularly. The predator has almost completely matched the speed and trajectory of the prey. In fact, the predator cannot simply choose to move at the same speed as the prey; that speed is not available to the predator. Instead, it exhibits more sophisticated behavior. The predator alternates appropriately between two speeds that bracket the prey's speed, matching the latter on average. The distance between the robots was still greater than required, but we hypothesize that this is explained by residual incoherence of the internal model (since calibration occurred for a stationary object).

6 Conclusions and Future Work

We have presented an application of Extended Markov Tracking (EMT) to robotic control. In our experiments, two independent EMT Controllers were applied to correlated motion parameters. These controllers enabled the robot to successfully perform an object-following task. In spite of a rather inaccurate action model, the EMT-controlled robot's performance was reasonable, although not optimal.

We then presented an action model calibration technique based on data provided by EMT. The approach draws heavily upon the features of EMT that allow system dynamics recovery from a partially-observed system state. EMT enabled the

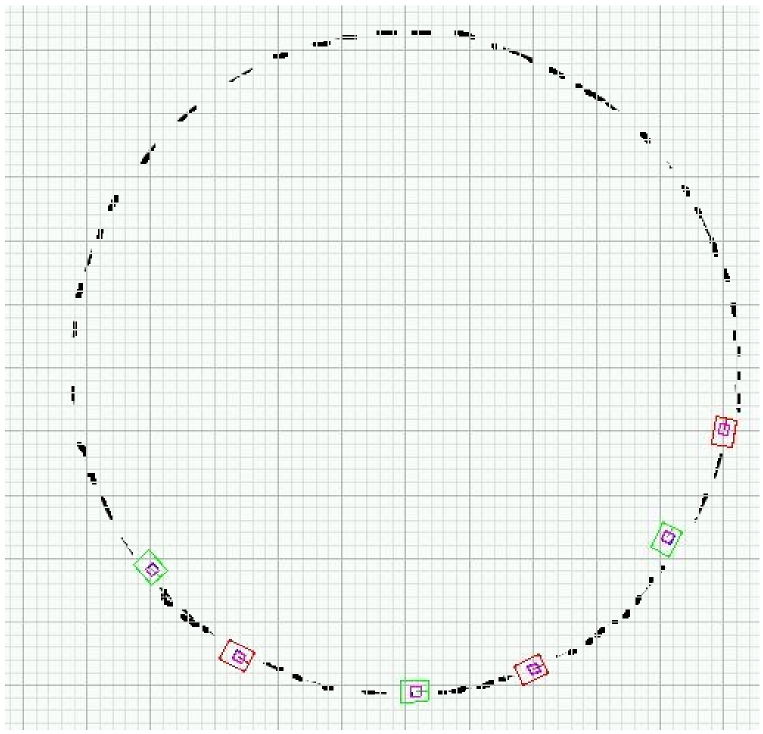

Figure 5: Target Following with a Calibrated Model

use of the simple statistical approach of data accumulation to calibrate the action model. Validity of our technique was demonstrated by calibrating the internal model of an EMT controller in the object-following experiment.

It is possible to pair EMT calibration of an action model with calibration of a sensor model as was done in [14]. We have tried several empirical methods, but surprisingly they only hindered performance. However, theoretical analysis points to the existence of an appropriate sensor model calibration method that would complement EMT action model calibration, and we intend to proceed in this direction. We also believe it will be possible to modify EMT for a parameterized environment model. This would allow us to generalize the calibration update between different actions (or observations), and create on-line calibration of the complete model.

Although initial experiments show the validity of EMT data for model calibration, the calibration algorithm itself is still fairly simple. We intend to draw upon existing machine learning techniques [8] to improve and extend our calibration mechanism.

References

[1] Jim Blythe. *Planning under uncertainty in dynamic domains*. PhD thesis, CMU, CS Department, 1998.

[2] Hung H. Bui. A general model for online probabilistic plan recognition. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1309–1315, Acapulco, Mexico.

[3] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley, 1991.

[4] T. L. Dean and M. P. Wellman. *Planning and Control*. Morgan Kaufmann, 1991.

[5] M. E. desJardins, E. H. Durfee, C. L. Ortiz Jr., and Michael J. Wolverton. A survey of research in distributed, continual planning. *AI Magazine*, 4:13–22, 1999.

[6] R. E. Fikes and N. J. Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[7] Brian Gerkey, Richard T. Vaughan, and Andrew Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th International Conference on Advanced Robotics (ICAR-03)*, pages 317–323, 2003.

[8] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[9] D. S. Nau, S. J. J. Smith, and K. Erol. Control strategies in HTN planning: Theory versus practice. In *AAAI/IAAI*, pages 1127–1133, 1998.

[10] David V. Pynadath and Michael P. Wellman. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of the 16th UAI Conference*, pages 507–514, 2000.

[11] Zinovi Rabinovich and Jeffrey S. Rosenschein. The authors' names are removed for paper submission. In *The Workshop on Agent Tracking: Modeling Other Agents from Observations, at the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems*, July 2004.

[12] Patrick Riley and Manuela Veloso. Recognizing probabilistic opponent movement models. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup-2001: The Fifth RoboCup Competitions and Conferences*. Springer Verlag, 2002.

[13] Nicholas Roy and Sebastian Thrun. Online self-calibration for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2292–2297, 1998.

[14] Daniel Stronger and Peter Stone. Simultaneous calibration of action and sensor models in a mobile robot. In *Proceedings of the International Conference on Robotics and Automation*, 2005. To appear.

[15] M. Veloso, J. Carbonell, A. Prez, D. Borrajo, E. Fink, and J. Blythe. Integrating planning and learning: The PRODIGY architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 7(1):81–120, 1995.

[16] Richard Washington. Markov tracking for agent coordination. In *Agents-98: The Second International Conference on Autonomous Agents*, pages 70–77, 1998.