Dynamics Based Control with an Application to Area-Sweeping Problems

Zinovi Rabinovich
Engineering and Computer Science
Hebrew University of Jerusalem
Jerusalem, Israel
nomad@cs.huji.ac.il

Jeffrey S. Rosenschein
Engineering and Computer Science
Hebrew University of Jerusalem
Jerusalem, Israel
jeff@cs.huji.ac.il

Gal A. Kaminka
The MAVERICK Group
Department of Computer Science
Bar Ilan University, Israel
galk@cs.biu.ac.il

ABSTRACT

In this paper we introduce Dynamics Based Control (DBC), an approach to planning and control of an agent in stochastic environments. Unlike existing approaches, which seek to optimize expected rewards (e.g., in Partially Observable Markov Decision Problems (POMDPs)), DBC optimizes system behavior *towards specified system dynamics*. We show that a recently developed planning and control approach, Extended Markov Tracking (EMT) is an instantiation of DBC. EMT employs greedy action selection to provide an efficient control algorithm in Markovian environments. We exploit this efficiency in a set of experiments that applied multi-target EMT to a class of area-sweeping problems (searching for moving targets). We show that such problems can be naturally defined and efficiently solved using the DBC framework, and its EMT instantiation.

Categories and Subject Descriptors

I.2.8 [**Problem Solving, Control Methods, and Search**]: Control Theory; I.2.9 [**Robotics**]; I.2.11 [**Distributed Artificial Intelligence**]: Intelligent Agents

General Terms

Algorithms, Theory

Keywords

Control, Multi-Agent Systems, Robotics, Target Dynamics, Dynamics Based Control

1. INTRODUCTION

Planning and control constitutes a central research area in multiagent systems and artificial intelligence. In recent years, Partially Observable Markov Decision Processes (POMDPs) [12] have become a popular formal basis for planning in stochastic environments. In this framework, the planning and control problem is often addressed by imposing a reward function, and computing a policy (of choosing actions) that is optimal, in the sense that it will result in the highest expected utility. While theoretically attractive, the complexity of optimally solving a POMDP is prohibitive [8, 7].

We take an alternative view of planning in stochastic environments. We do not use a (state-based) reward function, but instead optimize over a different criterion, a transition-based specification of the desired system dynamics. The idea here is to view plan-execution as a process that compels a (stochastic) system to change, and a plan as a dynamic process that shapes that change according to desired criteria. We call this general planning framework *Dynamics Based Control* (DBC).

In DBC, the goal of a planning (or control) process becomes to ensure that the system will change in accordance with specific (potentially stochastic) target dynamics. As actual system behavior may deviate from that which is specified by target dynamics (due to the stochastic nature of the system), planning in such environments needs to be continual [4], in a manner similar to classical closed-loop controllers [16]. Here, optimality is measured in terms of probability of deviation magnitudes.

In this paper, we present the structure of Dynamics Based Control. We show that the recently developed Extended Markov Tracking (EMT) approach [13, 14, 15] is subsumed by DBC, with EMT employing greedy action selection, which is a specific parameterization among the options possible within DBC. EMT is an efficient instantiation of DBC.

To evaluate DBC, we carried out a set of experiments applying multi-target EMT to the Tag Game [11]; this is a variant on the area sweeping problem, where an agent is trying to "tag" a moving target (quarry) whose position is not known with certainty. Experimental data demonstrates that even with a simple model of the environment and a simple design of target dynamics, high success rates can be produced both in catching the quarry, and in surprising the quarry (as expressed by the observed entropy of the controlled agent's position).

The paper is organized as follows. In Section 2 we motivate DBC using area-sweeping problems, and discuss related work. Section 3 introduces the Dynamics Based Control (DBC) structure, and its specialization to Markovian environments. This is followed by a review of the Extended Markov Tracking (EMT) approach as a DBC-structured control regimen in Section 4. That section also discusses the limitations of EMT-based control relative to the general DBC framework. Experimental settings and results are then presented in Section 5. Section 6 provides a short discussion of the overall approach, and Section 7 gives some concluding remarks and directions for future work.

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

978-81-904262-7-5 (RPS)

2. MOTIVATION AND RELATED WORK

Many real-life scenarios naturally have a stochastic target dynamics specification, especially those domains where there exists no ultimate goal, but rather system behavior (with specific properties) that has to be continually supported. For example, security guards perform persistent sweeps of an area to detect any sign of intrusion. Cunning thieves will attempt to track these sweeps, and time their operation to key points of the guards' motion. It is thus advisable to make the guards' motion dynamics appear irregular and random.

Recent work by Paruchuri et al. [10] has addressed such randomization in the context of single-agent and distributed POMDPs. The goal in that work was to generate policies that provide a measure of action-selection randomization, while maintaining rewards within some acceptable levels. Our focus differs from this work in that DBC does not optimize expected rewards—indeed we do not consider rewards at all—but instead maintains desired dynamics (including, but not limited to, randomization).

The Game of Tag is another example of the applicability of the approach. It was introduced in the work by Pineau et al. [11]. There are two agents that can move about an area, which is divided into a grid. The grid may have blocked cells (holes) into which no agent can move. One agent (the hunter) seeks to move into a cell occupied by the other (the quarry), such that they are co-located (this is a "successful tag"). The quarry seeks to avoid the hunter agent, and is always aware of the hunter's position, but does not know how the hunter will behave, which opens up the possibility for a hunter to surprise the prey. The hunter knows the quarry's probabilistic law of motion, but does not know its current location. Tag is an instance of a family of area-sweeping (pursuit-evasion) problems.

In [11], the hunter modeled the problem using a POMDP. A reward function was defined, to reflect the desire to tag the quarry, and an action policy was computed to optimize the reward collected over time. Due to the intractable complexity of determining the optimal policy, the action policy computed in that paper was essentially an approximation.

In this paper, instead of formulating a reward function, we use EMT to solve the problem, by directly specifying the target dynamics. In fact, any search problem with randomized motion, the so-called class of *area sweeping* problems, can be described through specification of such target system dynamics. Dynamics Based Control provides a natural approach to solving these problems.

3. DYNAMICS BASED CONTROL

The specification of Dynamics Based Control (DBC) can be broken into three interacting levels: Environment Design Level, User Level, and Agent Level.

- **Environment Design Level** is concerned with the formal specification and modeling of the environment. For example, this level would specify the laws of physics within the system, and set its parameters, such as the gravitation constant.
- **User Level** in turn relies on the environment model produced by Environment Design to specify the target system dynamics it wishes to observe. The User Level also specifies the estimation or learning procedure for system dynamics, and the measure of deviation. In the museum guard scenario above, these would correspond to a stochastic sweep schedule, and a measure of relative surprise between the specified and actual sweeping.
- **Agent Level** in turn combines the environment model from the Environment Design level, the dynamics estimation procedure, the deviation measure and the target dynamics specification from User Level, to produce a sequence of actions that create system dynamics as close as possible to the targeted specification.

As we are interested in the continual development of a stochastic system, such as happens in classical control theory [16] and continual planning [4], as well as in our example of museum sweeps, the question becomes how the Agent Level is to treat the deviation measurements over time. To this end, we use a probability threshold—that is, we would like the Agent Level to maximize the probability that the deviation measure will remain below a certain threshold.

Specific action selection then depends on system formalization. One possibility would be to create a mixture of available system trends, much like that which happens in Behavior-Based Robotic architectures [1]. The other alternative would be to rely on the estimation procedure provided by the User Level—to utilize the Environment Design Level model of the environment to choose actions, so as to manipulate the dynamics estimator into believing that a certain dynamics has been achieved. Notice that this manipulation is not direct, but via the environment. Thus, for strong enough estimator algorithms, successful manipulation would mean a successful simulation of the specified target dynamics (i.e., beyond discerning via the available sensory input).

DBC levels can also have a back-flow of information (see Figure 1). For instance, the Agent Level could provide data about target dynamics feasibility, allowing the User Level to modify the requirement, perhaps focusing on attainable features of system behavior. Data would also be available about the system response to different actions performed; combined with a dynamics estimator defined by the User Level, this can provide an important tool for the environment model calibration at the Environment Design Level.

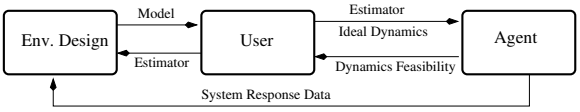

Figure 1: Data flow of the DBC framework

Extending upon the idea of Actor-Critic algorithms [5], DBC data flow can provide a good basis for the design of a learning algorithm. For example, the User Level can operate as an exploratory device for a learning algorithm, inferring an ideal dynamics target from the environment model at hand that would expose and verify most critical features of system behavior. In this case, feasibility and system response data from the Agent Level would provide key information for an environment model update. In fact, the combination of feasibility and response data can provide a basis for the application of strong learning algorithms such as EM [2, 9].

3.1 DBC for Markovian Environments

For a Partially Observable Markovian Environment, DBC can be specified in a more rigorous manner. Notice how DBC discards rewards, and replaces it by another optimality criterion (structural differences are summarized in Table 1):

- **Environment Design** level is to specify 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 probability 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.

- **User Level**, in the case of Markovian environment, operates on the set of system dynamics described by a family of conditional probabilities $\mathcal{F} = \{\tau : S \times A \to \Pi(S)\}$. Thus specification of target dynamics can be expressed by $q \in \mathcal{F}$, and the learning or tracking algorithm can be represented as a function $L : O \times (A \times O)^* \to \mathcal{F}$; that is, it maps sequences of observations and actions performed so far into an estimate $\tau \in \mathcal{F}$ of system dynamics.

  There are many possible variations available at the User Level to define divergence between system dynamics; several of them are:

  - *Trace distance* or L_1 distance between two distributions p and q defined by
$$D(p(\cdot), q(\cdot)) = \frac{1}{2} \sum_x |p(x) - q(x)|$$
  - *Fidelity* measure of distance
$$F(p(\cdot), q(\cdot)) = \sum_x \sqrt{p(x)q(x)}$$
  - *Kullback-Leibler divergence*
$$D_{KL}(p(\cdot)\|q(\cdot)) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

  Notice that the latter two are not actually metrics over the space of possible distributions, but nevertheless have meaningful and important interpretations. For instance, Kullback-Leibler divergence is an important tool of information theory [3] that allows one to measure the "price" of encoding an information source governed by q, while assuming that it is governed by p.

  The User Level also defines the threshold of dynamics deviation probability θ.

- **Agent Level** is then faced with a problem of selecting a control signal function a^* to satisfy a minimization problem as follows:
$$a^* = \arg\min_a Pr(d(\tau_a, q) > \theta)$$
where $d(\tau_a, q)$ is a random variable describing deviation of the dynamics estimate τ_a, created by L under control signal a, from the ideal dynamics q. Implicit in this minimization problem is that L is manipulated via the environment, based on the environment model produced by the Environment Design Level.

3.2 DBC View of the State Space

It is important to note the complementary view that DBC and POMDPs take on the state space of the environment. POMDPs regard state as a stationary snap-shot of the environment; whatever attributes of state *sequencing* one seeks are reached through properties of the control process, in this case reward accumulation. This can be viewed as if the sequencing of states and the attributes of that sequencing are only introduced by and for the controlling mechanism—the POMDP policy.

DBC concentrates on the underlying principle of state sequencing, the system dynamics. DBC's target dynamics specification can use the environment's state space as a means to describe, discern, and preserve changes that occur within the system. As a result, DBC has a greater ability to express state sequencing properties, which are grounded in the environment model and its state space definition.

For example, consider the task of moving through rough terrain towards a goal and reaching it as fast as possible. POMDPs would encode terrain as state space points, while speed would be ensured by negative reward for every step taken without reaching the goal—accumulating higher reward can be reached only by faster motion. Alternatively, the state space could directly include the notion of speed. For POMDPs, this would mean that the same concept is encoded twice, in some sense: directly in the state space, and indirectly within reward accumulation. Now, even if the reward function would encode more, and finer, details of the properties of motion, the POMDP solution will have to search in a much larger space of policies, while still being guided by the implicit concept of the reward accumulation procedure.

On the other hand, the tactical target expression of variations and correlations between position and speed of motion is now grounded in the state space representation. In this situation, any further constraints, e.g., smoothness of motion, speed limits in different locations, or speed reductions during sharp turns, are explicitly and uniformly expressed by the tactical target, and can result in faster and more effective action selection by a DBC algorithm.

4. EMT-BASED CONTROL AS A DBC

Recently, a control algorithm was introduced called *EMT-based Control* [13], which instantiates the DBC framework. Although it provides an approximate greedy solution in the DBC sense, initial experiments using EMT-based control have been encouraging [14, 15]. EMT-based control is based on the Markovian environment definition, as in the case with POMDPs, but its User and Agent Levels are of the Markovian DBC type of optimality.

- **User Level** of EMT-based control defines a limited-case target system dynamics independent of action:
$$q_{EMT} : S \to \Pi(S).$$
It then utilizes the Kullback-Leibler divergence measure to compose a momentary system dynamics estimator—the Extended Markov Tracking (EMT) algorithm. The algorithm keeps a system dynamics estimate τ^t_{EMT} that is capable of explaining recent change in an auxiliary Bayesian system state estimator from p_{t-1} to p_t, and updates it conservatively using Kullback-Leibler divergence. Since τ^t_{EMT} and $p_{t-1,t}$ are respectively the conditional and marginal probabilities over the system's state space, "explanation" simply means that
$$p_t(s') = \sum_s \tau^t_{EMT}(s'|s) p_{t-1}(s),$$
and the dynamics estimate update is performed by solving a

Table 1: Structure of POMDP vs. Dynamics-Based Control in Markovian Environment

Level	Approach	
	MDP	**Markovian DBC**
Environment Design	$< S, A, T, O, \Omega >$,where S — set of states A — set of actions $T : S \times A \to \Pi(S)$ — transition O — observation set $\Omega : S \times A \times S \to \Pi(O)$	
User	$r : S \times A \times S \to \mathcal{R}$ $F(\pi^*) \to r$ r — reward function F — reward remodeling	$q : S \times A \to \Pi(S)$ $L(o_1, ..., o_t) \to \tau$ q — ideal dynamics L — dynamics estimator θ — threshold
Agent	$\pi^* = \arg\max_\pi E[\sum \gamma^t r_t]$	$\pi^* = \arg\min_\pi Prob(d(\tau \| q) > \theta)$

minimization problem:

$$\begin{aligned} \tau^t_{EMT} &= H[p_t, p_{t-1}, \tau^{t-1}_{EMT}] \\ &= \arg\min_\tau D_{KL}(\tau \times p_{t-1} \| \tau^{t-1}_{EMT} \times p_{t-1}) \\ s.t. & \\ & p_t(s') = \sum_s (\tau \times p_{t-1})(s', s) \\ & p_{t-1}(s) = \sum_{s'} (\tau \times p_{t-1})(s', s) \end{aligned}$$

- **Agent Level** in EMT-based control is suboptimal with respect to DBC (though it remains within the DBC framework), performing greedy action selection based on prediction of EMT's reaction. The prediction is based on the environment model provided by the Environment Design level, so that if we denote by T_a the environment's transition function limited to action a, and p_{t-1} is the auxiliary Bayesian system state estimator, then the EMT-based control choice is described by

$$a^* = \arg\min_{a \in A} D_{KL}(H[T_a \times p_t, p_t, \tau^t_{EMT}] \| q_{EMT} \times p_{t-1})$$

Note that this follows the Markovian DBC framework precisely: the rewarding optimality of POMDPs is *discarded*, and in its place a dynamics estimator (EMT in this case) is manipulated via action effects on the environment to produce an estimate close to the specified target system dynamics. Yet as we mentioned, naive EMT-based control is suboptimal in the DBC sense, and has several additional limitations that do not exist in the general DBC framework (discussed in Section 4.2).

4.1 Multi-Target EMT

At times, there may exist several behavioral preferences. For example, in the case of museum guards, some art items are more heavily guarded, requiring that the guards stay more often in their close vicinity. On the other hand, no corner of the museum is to be left unchecked, which demands constant motion. Successful museum security would demand that the guards adhere to, and balance, both of these behaviors. For EMT-based control, this would mean facing several tactical targets $\{q_k\}_{k=1}^K$, and the question becomes how to merge and balance them. A balancing mechanism can be applied to resolve this issue.

Note that EMT-based control, while selecting an action, creates a preference vector over the set of actions based on their predicted performance with respect to a given target. If these preference vectors are normalized, they can be combined into a single unified preference. This requires replacement of standard EMT-based action selection by the algorithm below [15]:

- Given:
  - a set of target dynamics $\{q_k\}_{k=1}^K$,
  - vector of weights $w(k)$
- Select action as follows
  - For each action $a \in A$ predict the future state distribution $\bar{p}^a_{t+1} = T_a * p_t$;
  - For each action, compute
$$D_a = H(\bar{p}^a_{t+1}, p_t, PD_t)$$
  - For each $a \in A$ and q_k tactical target, denote
$$V(a,k) = \langle D_{KL}\left(D_a \| q_k\right) \rangle_{p_t}.$$
Let $V_k(a) = \frac{1}{Z_k} V(a,k)$, where $Z_k = \sum_{a \in A} V(a,k)$ is a normalization factor.
  - Select $a^* = \arg\min_a \sum_{k=1}^k w(k) V_k(a)$

The weights vector $\vec{w} = (w_1, ..., w_K)$ allows the additional "tuning of importance" among target dynamics without the need to redesign the targets themselves. This balancing method is also seamlessly integrated into the EMT-based control flow of operation.

4.2 EMT-based Control Limitations

EMT-based control is a sub-optimal (in the DBC sense) representative of the DBC structure. It limits the User by forcing EMT to be its dynamic tracking algorithm, and replaces Agent optimization by greedy action selection. This kind of combination, however, is common for on-line algorithms. Although further development of EMT-based controllers is necessary, evidence so far suggests that even the simplest form of the algorithm possesses a great deal of power, and displays trends that are optimal in the DBC sense of the word.

There are two further, EMT-specific, limitations to EMT-based control that are evident at this point. Both already have partial solutions and are subjects of ongoing research.

The first limitation is the problem of negative preference. In the POMDP framework for example, this is captured simply, through

the appearance of values with different signs within the reward structure. For EMT-based control, however, negative preference means that one would like to *avoid* a certain distribution over system development sequences; EMT-based control, however, concentrates on getting as *close* as possible to a distribution. Avoidance is thus unnatural in native EMT-based control.

The second limitation comes from the fact that standard environment modeling can create *pure sensory actions*—actions that do not change the state of the world, and differ only in the way observations are received and the quality of observations received. Since the world state does not change, EMT-based control would not be able to differentiate between different sensory actions.

Notice that both of these limitations of EMT-based control are absent from the general DBC framework, since it may have a tracking algorithm capable of considering pure sensory actions and, unlike Kullback-Leibler divergence, a distribution deviation measure that is capable of dealing with negative preference.

5. EMT PLAYING TAG

The Game of Tag was first introduced in [11]. It is a single agent problem of capturing a quarry, and belongs to the class of area sweeping problems. An example domain is shown in Figure 2.

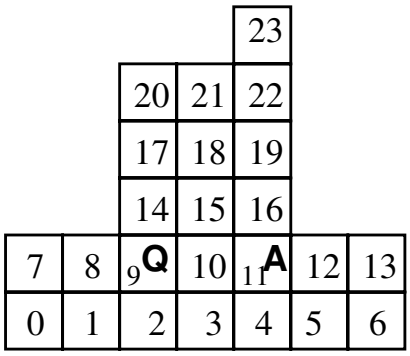

Figure 2: Tag domain; an agent (A) attempts to seek and capture a quarry (Q)

The Game of Tag extremely limits the agent's perception, so that the agent is able to detect the quarry only if they are co-located in the same cell of the grid world. In the classical version of the game, co-location leads to a special observation, and the 'Tag' action can be performed. We slightly modify this setting: the moment that both agents occupy the same cell, the game ends. As a result, both the agent and its quarry have the same motion capability, which allows them to move in four directions, North, South, East, and West. These form a formal space of actions within a Markovian environment.

The state space of the formal Markovian environment is described by the cross-product of the agent and quarry's positions. For Figure 2, it would be $S = \{s_0, ..., s_{23}\} \times \{s_0, ..., s_{23}\}$.

The effects of an action taken by the agent are deterministic, but the environment in general has a stochastic response due to the motion of the quarry. With probability q_0[1] it stays put, and with probability $1 - q_0$ it moves to an adjacent cell further away from the agent. So for the instance shown in Figure 2 and $q_0 = 0.1$:

$$P(Q = s_9|Q = s_9, A = s_{11}) = 0.1$$
$$P(Q = s_2|Q = s_9, A = s_{11}) = 0.3$$
$$P(Q = s_8|Q = s_9, A = s_{11}) = 0.3$$
$$P(Q = s_{14}|Q = s_9, A = s_{11}) = 0.3$$

Although the evasive behavior of the quarry is known to the agent, the quarry's position is not. The only sensory information available to the agent is its own location.

We use EMT and directly specify the target dynamics. For the Game of Tag, we can easily formulate three major trends: catching the quarry, staying mobile, and stalking the quarry. This results in the following three target dynamics:

$$T_{catch}(A_{t+1} = s_i|Q_t = s_j, A_t = s_a) \propto \begin{cases} 1 & s_i = s_j \\ 0 & otherwise \end{cases}$$

$$T_{mobile}(A_{t+1} = s_i|Q_t = s_o, A_t = s_j) \propto \begin{cases} 0 & s_i = s_j \\ 1 & otherwise \end{cases}$$

$$T_{stalk}(A_{t+1} = s_i|Q_t = s_o, A_t = s_j) \propto \frac{1}{dist(s_i, s_o)}$$

Note that none of the above targets are directly achievable; for instance, if $Q_t = s_9$ and $A_t = s_{11}$, there is no action that can move the agent to $A_{t+1} = s_9$ as required by the T_{catch} target dynamics.

We ran several experiments to evaluate EMT performance in the Tag Game. Three configurations of the domain shown in Figure 3 were used, each posing a different challenge to the agent due to partial observability. In each setting, a set of 1000 runs was performed with a time limit of 100 steps. In every run, the initial position of both the agent and its quarry was selected at random; this means that as far as the agent was concerned, the quarry's initial position was uniformly distributed over the entire domain cell space.

We also used two variations of the environment observability function. In the first version, observability function mapped all joint positions of hunter and quarry into the position of the hunter as an observation. In the second, only those joint positions in which hunter and quarry occupied different locations were mapped into the hunter's location. The second version in fact utilized and expressed the fact that once hunter and quarry occupy the same cell the game ends.

The results of these experiments are shown in Table 2. Balancing [15] the catch, move, and stalk target dynamics described in the previous section by the weight vector $[0.8, 0.1, 0.1]$, EMT produced stable performance in all three domains.

Although direct comparisons are difficult to make, the EMT performance displayed notable efficiency vis-à-vis the POMDP approach. In spite of a simple and inefficient Matlab implementation of the EMT algorithm, the decision time for any given step averaged significantly below 1 second in all experiments. For the irregular open arena domain, which proved to be the most difficult, 1000 experiment runs bounded by 100 steps each, a total of 42411 steps, were completed in slightly under 6 hours. That is, over 4×10^4 online steps took an order of magnitude less time than the offline computation of POMDP policy in [11]. The significance of this differential is made even more prominent by the fact that, should the environment model parameters change, the online nature of EMT would allow it to maintain its performance time, while the POMDP policy would need to be recomputed, requiring yet again a large overhead of computation.

We also tested the behavior cell frequency entropy, empirical measures from trial data. As Figure 4 and Figure 5 show, empir-

[1]In our experiments this was taken to be $q_0 = 0.2$.

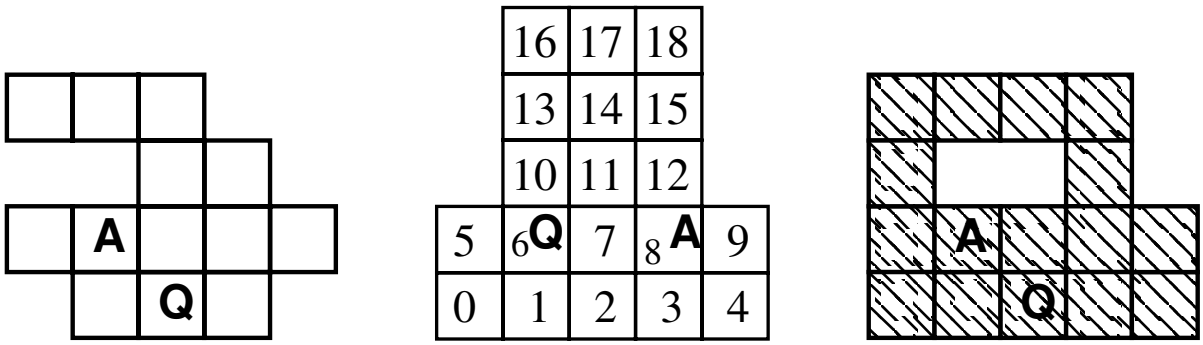

Figure 3: These configurations of the Tag Game space were used: a) multiple dead-end, b) irregular open arena, c) circular corridor

Table 2: Performance of the EMT-based solution in three Tag Game domains and two observability models: I) omniposition quarry, II) quarry is not at hunter's position

Model	Domain	Capture%	E(Steps)	Time/Step
I	Dead-ends	100	14.8	72(mSec)
	Arena	80.2	42.4	500(mSec)
	Circle	91.4	34.6	187(mSec)
II	Dead-ends	100	13.2	91(mSec)
	Arena	96.8	28.67	396(mSec)
	Circle	94.4	31.63	204(mSec)

ical entropy grows with the length of interaction. For runs where the quarry was not captured immediately, the entropy reaches between 0.85 and 0.95[2] for different runs and scenarios. As the agent actively seeks the quarry, the entropy never reaches its maximum.

One characteristic of the entropy graph for the open arena scenario particularly caught our attention in the case of the omniposition quarry observation model. Near the maximum limit of trial length (100 steps), entropy suddenly dropped. Further analysis of the data showed that under certain circumstances, a fluctuating behavior occurs in which the agent faces equally viable versions of quarry-following behavior. Since the EMT algorithm has greedy action selection, and the state space does not encode any form of commitment (not even speed or acceleration), the agent is locked within a small portion of cells. It is essentially attempting to simultaneously follow several courses of action, all of which are consistent with the target dynamics. This behavior did not occur in our second observation model, since it significantly reduced the set of eligible courses of action—essentially contributing to tie-breaking among them.

6. DISCUSSION

The design of the EMT solution for the Tag Game exposes the core difference in approach to planning and control between EMT or DBC, on the one hand, and the more familiar POMDP approach, on the other. POMDP defines a reward structure to optimize, and influences system dynamics indirectly through that optimization. EMT discards any reward scheme, and instead measures and influences system dynamics directly.

[2]Entropy was calculated using log base equal to the number of possible locations within the domain; this properly scales entropy expression into the range $[0, 1]$ for all domains.

Thus for the Tag Game, we did not search for a reward function that would encode and express our preference over the agent's behavior, but rather directly set three (heuristic) behavior preferences as the basis for target dynamics to be maintained. Experimental data shows that these targets need not be directly achievable via the agent's actions. However, the ratio between EMT performance and achievability of target dynamics remains to be explored.

The tag game experiment data also revealed the different emphasis DBC and POMDPs place on the formulation of an environment state space. POMDPs rely entirely on the mechanism of reward accumulation maximization, i.e., formation of the action selection procedure to achieve necessary state sequencing. DBC, on the other hand, has two sources of sequencing specification: through the properties of an action selection procedure, and through direct specification within the target dynamics. The importance of the second source was underlined by the Tag Game experiment data, in which the greedy EMT algorithm, applied to a POMDP-type state space specification, failed, since target description over such a state space was incapable of encoding the necessary behavior tendencies, e.g., tie-breaking and commitment to directed motion.

The structural differences between DBC (and EMT in particular), and POMDPs, prohibits direct performance comparison, and places them on complementary tracks, each within a suitable niche. For instance, POMDPs could be seen as a much more natural formulation of economic sequential decision-making problems, while EMT is a better fit for continual demand for stochastic change, as happens in many robotic or embodied-agent problems.

The complementary properties of POMDPs and EMT can be further exploited. There is recent interest in using POMDPs in hybrid solutions [17], in which the POMDPs can be used together with other control approaches to provide results not easily achievable with either approach by itself. DBC can be an effective partner in such a hybrid solution. For instance, POMDPs have prohibitively large off-line time requirements for policy computation, but can be readily used in simpler settings to expose beneficial behavioral trends; this can serve as a form of target dynamics that are provided to EMT in a larger domain for on-line operation.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a novel perspective on the process of planning and control in stochastic environments, in the form of the Dynamics Based Control (DBC) framework. DBC formulates the task of planning as support of a specified target system dynamics, which describes the necessary properties of change within the environment. Optimality of DBC plans of action are measured

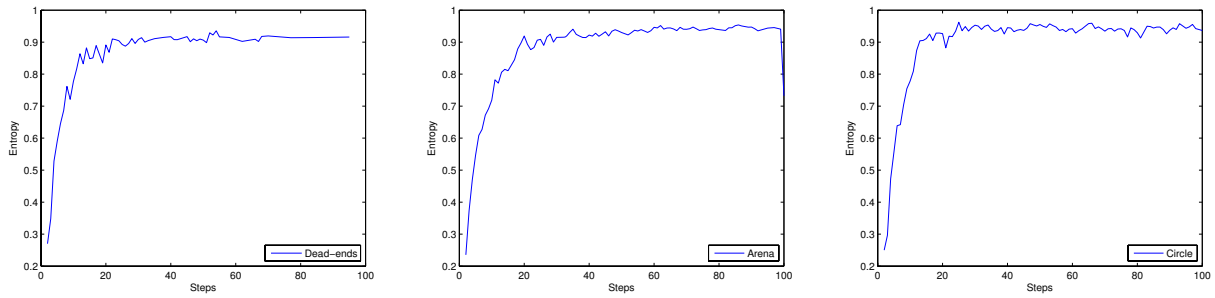

Figure 4: Observation Model I: Omniposition quarry. Entropy development with length of Tag Game for the three experiment scenarios: a) multiple dead-end, b) irregular open arena, c) circular corridor.

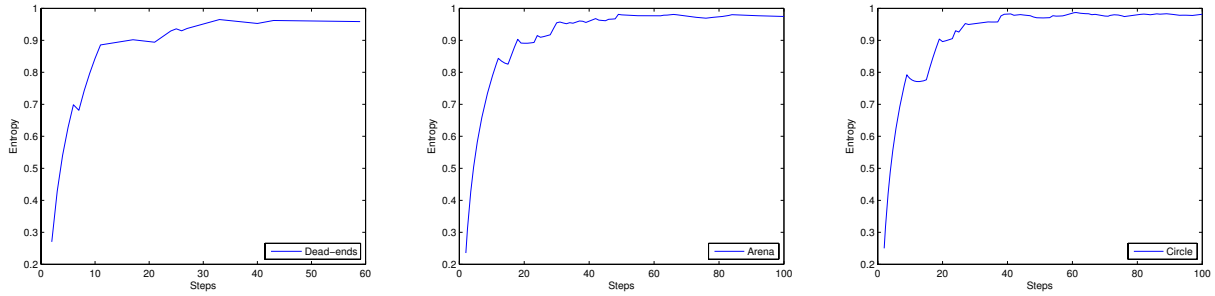

Figure 5: Observation Model II: quarry not observed at hunter's position. Entropy development with length of Tag Game for the three experiment scenarios: a) multiple dead-end, b) irregular open arena, c) circular corridor.

with respect to the deviation of actual system dynamics from the target dynamics.

We show that a recently developed technique of Extended Markov Tracking (EMT) [13] is an instantiation of DBC. In fact, EMT can be seen as a specific case of DBC parameterization, which employs a greedy action selection procedure.

Since EMT exhibits the key features of the general DBC framework, as well as polynomial time complexity, we used the multi-target version of EMT [15] to demonstrate that the class of area sweeping problems naturally lends itself to dynamics-based descriptions, as instantiated by our experiments in the Tag Game domain.

As enumerated in Section 4.2, EMT has a number of limitations, such as difficulty in dealing with negative dynamic preference. This prevents direct application of EMT to such problems as Rendezvous-Evasion Games (e.g., [6]). However, DBC in general has no such limitations, and readily enables the formulation of evasion games. In future work, we intend to proceed with the development of dynamics-based controllers for these problems.

8. ACKNOWLEDGMENT

The work of the first two authors was partially supported by Israel Science Foundation grant #898/05, and the third author was partially supported by a grant from Israel's Ministry of Science and Technology.

9. REFERENCES

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

[2] J. A. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and Hidden Markov Models. Technical Report TR-97-021, Department of Electrical Engeineering and Computer Science, University of California at Berkeley, 1998.

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

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

[5] V. R. Konda and J. N. Tsitsiklis. Actor-Critic algorithms. *SIAM Journal on Control and Optimization*, 42(4):1143–1166, 2003.

[6] W. S. Lim. A rendezvous-evasion game on discrete locations with joint randomization. *Advances in Applied Probability*, 29(4):1004–1017, December 1997.

[7] M. L. Littman, T. L. Dean, and L. P. Kaelbling. On the complexity of solving Markov decision problems. In *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 394–402, 1995.

[8] O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence Journal*, 147(1–2):5–34, July 2003.

[9] R. M. Neal and G. E. Hinton. A view of the EM algorithm

that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[10] P. Paruchuri, M. Tambe, F. Ordonez, and S. Kraus. Security in multiagent systems by policy randomization. In *Proceeding of AAMAS 2006*, 2006.

[11] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025–1032, August 2003.

[12] M. L. Puterman. *Markov Decision Processes*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics Section. Wiley-Interscience Publication, New York, 1994.

[13] Z. Rabinovich and J. S. Rosenschein. Extended Markov Tracking with an application to control. In *The Workshop on Agent Tracking: Modeling Other Agents from Observations, at the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 95–100, New-York, July 2004.

[14] Z. Rabinovich and J. S. Rosenschein. Multiagent coordination by Extended Markov Tracking. In *The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 431–438, Utrecht, The Netherlands, July 2005.

[15] Z. Rabinovich and J. S. Rosenschein. On the response of EMT-based control to interacting targets and models. In *The Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 465–470, Hakodate, Japan, May 2006.

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

[17] M. Tambe, E. Bowring, H. Jung, G. Kaminka, R. Maheswaran, J. Marecki, J. Modi, R. Nair, J. Pearce, P. Paruchuri, D. Pynadath, P. Scerri, N. Schurr, and P. Varakantham. Conflicts in teamwork: Hybrids to the rescue. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 3–10, Utrecht, The Netherlands, July 2005.