Dynamics Based Control Framework

by Zinovi Rabinovich

Critical MAS Research Group
(director: Prof. Jeffrey S. Rosenschein)
School of Engineering and Computer Science
The Hebrew University of Jerusalem
Agenda

- Building Intuition
- Dynamics Based Control (DBC)
  - Control and Planning Perspective
- DBC in Markovian Environment
  - Extended Markov Tracking (EMT)
  - Multi-agent EMT
  - Multi-target EMT
  - EMT with inexact model
- Conclusions
Everything flows, nothing stands still. Nothing endures, but change.

Heraclitus
535-475BC

The universe is change; our life is what our thoughts make it.

Marcus Aurelius
121AD-180AD
Building Intuition: Scenario 1 - Fun

- A “Foundation” is established under the pretense of preservation of the human knowledge. The Foundation has to survive the crumbling Galactic Empire, and eventually develop into a new one.

- The Foundation is compelled to develop under the pressure of circumstances, imminent or perceived.

- The Second Foundations is established to maintain and create artifacts that would hint The (First) Foundation onto a predetermined development course.
Seldon’s Problem

The Second Foundation knows:
- how the First Foundation operates,
- how artifacts create pressure,
- how the First Foundation should ideally develop.

“Seldon’s Problem”:
Which artifacts, and when, should be created to keep The Foundation development dynamics as ideal as possible?

“Seldon’s Perceptual Correction”:
The problem has to be formulated with perceived development dynamics.
Building Intuition: Scenario 2 - Science

- Human vision relies on change to discover a scene’s composition.
  - Not any particular “frame”, but the change between them.
  - Dynamics, i.e. rules of the change, are recovered.
  - The dynamics tag (or recognize) different parts of a scene.
Human vision relies on change to discover a scene’s composition.

Not any particular “frame”, but the change between them.

Dynamics, i.e. rules of the change, are recovered.

The dynamics tag (or recognize) different parts of a scene.

CG relies on this to build illusions.

The recognition process is essentially controlled into believing that a certain form of change occurs.
Building Intuition: Components

The following components can be identified:

- Dynamics tracking/recognition algorithm, \( L \).
- A controlled environment, \( S \), that produces inputs for \( L \).
- A target result, \( \tau^* \), of the recognition algorithm.

Control problem:

- Find a control method, applied to \( S \), so that the produced input would be perceived by \( L \) as \( \tau^* \), or a close alternative.
Building Intuition: Components

The following implicit components can be identified:
- Mathematical models of:
  - the environment, $S$,
  - the tracking/recognition algorithm, $L$,
  - the target $\tau^*$, and other possible outcomes of $L$,
  - a measure of proximity between different outcomes of $L$

Control problem:
- Find a control method, based on the mathematical model of $S$, so that the algorithm $L$ will recreate a dynamics model closest to $\tau^*$
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
- User (strategic) level
- Agent (tactical) level
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
  - Formal specs and modeling of the environment
- User (strategic) level
- Agent (tactical) level
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
- User (strategic) level
- Agent (tactical) level
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
- User (strategic) level
  - Ideal (target) dynamics and dynamics estimator specs, and dynamics divergence measure
- Agent (tactical) level
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
- User (strategic) level
- Agent (tactical) level
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
- User (strategic) level
- Agent (tactical) level
  - Tactical solution
  - Utilization of environment model and dynamics estimator to create target dynamics within environment.
Dynamics Based Control (DBC)

Formulated by three levels:
- Environment Design level
- User (strategic) level
- Agent (tactical) level

The data flow between the levels can be depicted as:
Control Perspective

Agent Level algorithm can be seen from the control theory perspective:

- Control perspective requires dynamics estimation to change “smoothly”
- Not the environment state, but rules of the change – system dynamics – are inferred.
Control Perspective: Model Following

The system is controlled as a function of the error between the ideal and actual responses.
Control Perspective: Model Following

The system is controlled as a function of the error between the ideal and actual responses.

\( \tau^* \) is the ideal dynamics. If we “model follow” it, would it not force \( L \) to \( \tau^* \)’s recognition?
The system is controlled as a function of the error between the ideal and actual responses.

\( \tau^* \) is the ideal dynamics. If we “model follow” it, would it not force \( L \) to \( \tau^* \)’s recognition?

Thus DBC Agent is trivially solved by Model Following.
Control Perspective: Model Following

The system is controlled as a function of the error between the ideal and actual responses.

\( \tau^* \) is the ideal dynamics. If we “model follow” it, would it not force \( \mathcal{L} \) to \( \tau^* \)’s recognition?

Deception of \( \mathcal{L} \) into recognizing \( \tau^* \) within the environment \( S \) can be easier than recreating \( \tau^* \).
The system is controlled as a function of the error between the ideal and actual responses.

*τ* is the ideal dynamics. If we “model follow” it, would it not force L to *τ*’s recognition?

Frequency analysis of the sequence *H, T, H, T, H, T, H, T, H, T* would state that a fair coin is used.
The system is controlled as a function of the error between the ideal and actual responses.

τ* is the ideal dynamics. If we “model follow” it, would it not force L to τ*’s recognition?

Noise and stochasticity of the environment S can lead L away from τ*.
The system is controlled as a function of the error between the ideal and actual responses.

$\tau^*$ is the ideal dynamics. If we “model follow” it, would it not force $L$ to $\tau^*$’s recognition?

Much harsher steps may be required than keeping in sink with $\tau^*$. E.g. tolerance in noisy prisoner’s dilemma.
The system is controlled as a function of the error between the ideal and actual responses.

DBC Agent is not trivially solved by Model Following.
The system is controlled as a function of the error between the ideal and actual responses.
The system is controlled as a function of the error between the ideal and actual responses.

Action is selected based on divergence of $L$’s response and $\tau^*$.
Control Perspective: Model Following II

The system is controlled as a function of the error between the ideal and actual responses.

Action is selected based on divergence of L’s response and $\tau^*$. Thus DBC Agent is a form of Model Following.
The system is controlled as a function of the error between the ideal and actual responses.

- Action is selected based on divergence of \( L \)'s response and \( \tau^* \).
- \( \tau^* \) is not a response of some ideal recognition algorithm, but a reference signal.
The system is controlled as a function of the error between the ideal and actual responses.

DBC Agent is not a form of Model Following.
Control Perspective: Perceptual Control

- Psychological theory of animal and human behavior
  - Mechanical: behavior is a function of perception,
  - Perceptual: behavior is means to control perceptions.
Control Perspective: Perceptual Control

- Psychological theory of animal and human behavior
  - Mechanical: behavior is a function of perception,
  - Perceptual: behavior is means to control perceptions.

- If $L$ is a part of a perception mechanism, then a behavior within $S$ (actions selection), is means to control the perceptions.
Psychological theory of animal and human behavior

- Mechanical: behavior is a function of perception,
- Perceptual: behavior is means to control perceptions.

If \( L \) is a part of a perception mechanism, then a behavior within \( S \) (actions selection), is means to control the perceptions.

- DBC Agent can indeed be seen as a perceptual controller
Planning Perspective: Continual

- Appears if dynamics estimate is based on a non-trivial sequence of actions, rather then corrected.

- Dynamics Estimator is akin to plan recognition

- Agent level deals with sets of plans, and chooses one whose remainder is to achieve the ideal dynamics.
DBC via Extended Markov Tracking

- DBC is very general
  - Multiple algorithmic solutions are possible
- When formulated for Markovian environments
  - Extended Markov Tracking (EMT) based control
    - A greedy approximate solution
For Markovian Environment it is possible to specify DBC in a more explicit form.

**Environment Design:** Markovian environment

\[ < S, A, T, O, \Omega, s_0 > \]

**User:** \( L : O \times (A \times O)^* \rightarrow \mathcal{F}, \tau^* \in \mathcal{F}, d : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R}. \)

**Agent:** \( a^* = \arg \min_a Pr(d(\tau_a, \tau^*) > \theta) \)
DBC for Markovian Environment

For Markovian Environment it is possible to specify DBC in a more explicit form.

Environment Design: Markovian environment
\(< S, A, T, O, \Omega, s_0 >\), where
- \(S\) - set of possible environment states
- \(s_0 \in \Pi(S)\) - the initial state (distribution)
- \(A\) - set of possible actions applicable
- \(T : S \times A \rightarrow \Pi(S)\) is the stochastic transition function
- \(O\) - set of (partial) observations
- \(\Omega : S \times A \times S \rightarrow \Pi(O)\) - stochastic observability function

User: \(L : O \times (A \times O)^* \rightarrow \mathcal{F}, \tau^* \in \mathcal{F}, d : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R}\).

Agent: \(a^* = \arg \min_a Pr(d(\tau_a, \tau^*) > \theta)\)
For Markovian Environment it is possible to specify DBC in a more explicit form.

Environment Design: Markovian environment

\(< S, A, T, O, \Omega, s_0 >\)

User: \(L : O \times (A \times O)^* \rightarrow \mathcal{F}, \tau^* \in \mathcal{F}, d : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R}.\)

Agent: \(a^* = \arg \min_a P_r(d(\tau_a, \tau^*) > \theta)\)
DBC for Markovian Environment

For Markovian Environment it is possible to specify DBC in a more explicit form.

Environment Design: Markovian environment
\[< S, A, T, O, \Omega, s_0 >\]

User: \( L : O \times (A \times O)^* \rightarrow \mathcal{F}, \tau^* \in \mathcal{F}, d : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R} \).
\[ \mathcal{F} = \{ \tau : S \times A \rightarrow \Pi(S) \} \] all possible dynamics
\( L \) is dynamics estimator
\( \tau^* \) is the ideal dynamics (tactical target)
\( d \) is the dynamics divergence measure

Agent: \( a^* = \arg \min_a Pr(d(\tau_a, \tau^*) > \theta) \)
DBC for Markovian Environment

For Markovian Environment it is possible to specify DBC in a more explicit form.

Environment Design: Markovian environment

< S, A, T, O, Ω, s₀ >

User: \( L : O \times (A \times O)^* \rightarrow \mathcal{F}, \tau^* \in \mathcal{F}, d : \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R} \).

Agent: \( a^* = \arg \min_a Pr(d(\tau_a, \tau^*) > \theta) \)
For Markovian Environment it is possible to specify DBC in a more explicit form.

**Environment Design:** Markovian environment

$< S, A, T, O, \Omega, s_0 >$

**User:** $L: O \times (A \times O)^* \rightarrow \mathcal{F}, \tau^* \in \mathcal{F}, d: \mathcal{F} \times \mathcal{F} \rightarrow \mathcal{R}.$

**Agent:** $a^* = \arg \min_a Pr(d(\tau_a, \tau^*) > \theta)$

with $\theta$ coming from User level as well, or algorithm specific

alternatively $a^* = \arg \min_a d(d(\tau_a, \tau^*), \delta(0))$
EMT-based control as DBC

- Environment Design
  - Markovian Environment $< S, s_0, A, T, O, \Omega >$

- User level
  - Dynamics divergence measure is Kullback-Leibler
  - Estimator is Extended Markov Tracking (EMT)

- Agent Level
  - **Greedy** action selection based on EMT predicted response
EMT: Intuition

- EMT assumes that the observed process is a non-controlled Markov chain.
- Space of all possible dynamics has the form $D : S \rightarrow \Pi(S)$.
- That is, the form of an explanation to a change in beliefs about the system state:

$$p_{t+1}(s') = D(s' | s)p_t(s)$$
But there are many such $D$'s for any pair of consecutive beliefs — how do we choose one?
EMT: Dynamics Deviation Distance

But there are many such $D$’s for any pair of consecutive beliefs — how do we choose one?

Can we measure the cost of choosing the wrong system dynamics?

... and wrong with respect to what?
EMT: Dynamics Deviation Distance

But there are many such $D$’s for any pair of consecutive beliefs — how do we choose one?

Using a conservative update from previous system dynamics estimate $PD,$

Discriminative Kullback-Leibler Distance:

$$D_{KL}(p||q) = \sum_x p(x) \log \left( \frac{p(x)}{q(x)} \right)$$
EMT: Tracking

- Given system state beliefs change from $p_t$ to $p_{t+1}$

$$p_{t+1}(s) \propto p(o | s, a) \sum_{s'} \mathcal{T}(s | a, s') p_t(s')$$

- Given previous system dynamics estimate $PD_t$, then the update of this belief is the solution to:

$$PD_{t+1} = \arg \min_Q E_{p_t(s)} [D_{KL}(Q(\cdot | s) || PD_t(\cdot | s))]$$

s.t.

$$p_{t+1} = Q \cdot p_t$$

- Denote $PD_{t+1} = H[p_{t+1}, p_t, PD_t]$. 
EMT-based Tactical Solution

Now that we have a means of understanding how the system develops, we can attempt to fix it to our liking, the liking of tactical target $\tau^* : S \rightarrow \Pi(S)$. 
EMT-based Tactical Solution

- Now that we have a means of understanding how the system develops, we can attempt to fix it to our liking, the liking of tactical target $\tau^*: \mathcal{S} \rightarrow \Pi(S)$.

- Tactical solution performs continual loop of
  - EMT estimation of system development
  - Action choice

$$a^* = \arg\min_a D_{KL}(H[T_{a^*} p_t, p_t, PD_t] \parallel \tau^*)$$

- Application of $a^*$. 
Example: Landing Angles

- **Fast Flash**: 6.0 Degrees
- **Glow Flash**: 4.0 Degrees
- **Steady**: 0.75 Degrees
- **Steady**: 0.50 Degrees
- **Steady**: 0.75 Degrees
- **Glow Flash**: 4.0 Degrees
- **Green**: On Centerline to Touchdown
- **Red**:
Example: Environment Design

One can easily see the aircraft descent development as a random walk over a linear graph — Drunk Man Walk:

The set $S$ of nodes can represent different angular sectors of approach.
Example: Environment Design

One can easily see the aircraft descent development as a random walk over a linear graph — Drunk Man Walk:

Random variations of the angle due to deck pitch and air turbulence can be represented by a probabilistic transition function $T : S \rightarrow \Pi(S)$.

Parameterizing $T : S \times A \rightarrow \Pi(S)$, we can model pilot actions by a set of parameter values $A$. 
Example: Environment Design

One can easily see the aircraft descent development as a random walk over a linear graph — Drunk Man Walk:

We can even model visibility conditions by introducing an observation set $O$.

And an observation distribution

$\Omega : S \times A \times S \rightarrow \Pi(O)$
Example: User level

- Estimator is set to be EMT
- Dynamics divergence metric: Kullbach-Leibler
- Optimal Dynamics
  - Let $\tau^*(s'|s) = \frac{1}{Z} \iff s' = \text{ideal.}$
  - Let $\tau^*(s'|s) = \frac{\epsilon}{Z}$ – positive tolerance
Applying EMT-based tactical solution we have:
Multiagent Formal Model

- Multiagent POMEM $< S, s_0, A, T, \{O_i\}_{i=1}^n, \{\Omega_i\}_{i=1}^n >$
  - $S$ - the set of the system states, $s_0 \in S$ the initial system state
  - $A = A_1 \times \cdots \times A_n$ - where $A_i$ is the set of actions applicable by the agent $i$.
  - $T : S \times A_1 \times \cdots A_n \rightarrow \Pi(S)$ - the system transition function
  - $O_i$ - the set of possible observations for agent $i$
  - $\Omega_i : S \times A \times S \rightarrow \Pi(O_i)$ - the observation probability distribution for agent $i$.

- Each agent can express the basic beliefs about the system state at time $t$ by a probability vector $\vec{\pi}_t \in \Pi(S)$. 
Multiagent EMT Control

- EMT has to be modified only slightly to admit the multiagent scenario. The agent has to consider the complete Cartesian product of actions $A = A_1 \times \cdots \times A_n$, but perform only it’s part.

- Multiagent EMT loop:
  - EMT estimation of system development due to local (agent specific) data.
  - Action choice

$$\vec{a}^* = \arg \min_{\vec{a}} D_{KL}(H[T_{a^*} p_t, p_t, PD_t] \parallel \tau^*)$$

- Application of $a_i^*$ from $a^* = (a_1^*, \ldots, a_n^*)$. 
Example: Springed bar problem

Consider a long bar resting its ends on two equal springs and two agents of equal mass are standing on the bar. Their task is to shift themselves around so that the bar would level.
Example: Springed bar problem

Consider a long bar resting it’s ends on two equal springs and two agents of equal mass are standing on the bar. Their task is to shift themselves around so that the bar would level.

Formally the system state is described by the positions of the two agent on the bar $S = [1 : d_{max}]^2$, where $d_{max}$ is the length of the bar in “steps”, and the initial state is a dis-balanced one $s_0 = (1, \frac{d_{max}}{2} + 1)$. The actions sets are $A_i = \{left, stay, right\}$, and the transition probability is built according to physics of motion.
Observations

1. $O_i = S = \{\text{all positions of the two agents}\}$, $\Omega_1 = \Omega_2$ and creates uniform noise over the immediate neighborhood of the real joint position of agents.

2. $O_i = [1 : d_{max}]$ and represents the position of the observing agent. $\Omega_i$ creates a uniform noise over the immediate neighborhood of the observing agent real position.
Results: First Observational scenario

In this first observation scenario, agents converge to a symmetric position around the ideal center of mass.
Results: First Observational scenario

In this first observation scenario, agents converge to a symmetric position around the ideal center of mass.
In the second observation scenario have found an equilibrium point, where each agent occupies the far end of the bar, thus balancing it.
Results: Second Observational scenario

In the second observation scenario have found an equilibrium point, where each agent occupies the far end of the bar, thus balancing it.
In the second observation scenario have found an equilibrium point, where each agent occupies the far end of the bar, thus balancing it.
Multiple Interacting Targets

- Overall behavior may be a complex composition of simpler interacting (and interfering) behavioral modules.
  - E.g. Behavior-Based Robotics
- EMT-based control resisted noise and model incoherence, creating reasonable performance.
- Can it also resist incoherence that stems from multiple targets?
  - Yes! – Idea: create a preference vector $V_k(a)$, based on actions performance with respect to a given target $\tau_k^*$. 
Multiple Interacting Targets: Algorithm

- For each action $a \in A$ predict the future state distribution $\bar{p}_{t+1}^a = T_a * p_t$;
- For each action, compute $D_a = H(\bar{p}_{t+1}^a, p_t, P D_t)$;
- For each $a \in A$ and $\tau_k^*$ tactical target, denote
  \[ V(a, k) = \langle D_{KL} (D_a \| \tau_k^*) \rangle_{p_t} . \]
  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 $a^* = \arg \min_a \sum_{k=1}^K w(k) V_k(a)$.
Example: Springed Bar

- Observation scenario
  - Noisy (independent) observations of joint position

- Task:
  - Target 1: Balance the bar
  - Target 2: Maintain predefined distance (4 in the experiment set)
Results

Under $(0.2, 0.8)$ balancing between the targets
Results

Under $(0.4, 0.6)$ balancing between the targets
Generally, target strength depends on transition probabilities within, thus
Example: Game of Tag

Agent $A$ tries to tag the quarry $Q$

$Q$ randomly chooses direction away from $A$ and takes a probabilistic step.

$A$ can take a (deterministic) step in either of four directions.
Game of Tag: Environment Design

- State space is a Cartesian product
  \[ S = \{c_0, \ldots, c_n\} \times \{c_0, \ldots, c_n\} \]

- Actions \( A = \{\text{North, South, West, East}\} \)

- Transition function \( T : S \times A \times S \) with accordance to random motion of the quarry \( Q \) and the step taken by agent \( A \).

- Observations are \( \{c_0, \ldots, c_n\} \) corresponding to the quarry location
  - Scenario I: Blind sweep: all observations are equiprobable
  - Scenario II: Shortsighted: all observations are equiprobable except the location of the agent which has zero probability.
Game of Tag: Targets

- Three targets:
  - **Catch the Quarry:**
    \[
    T_{catch}(A_{t+1} = s_i | Q_t = s_j, A_t = s_a) \propto \begin{cases} 
    1 & s_i = s_j \\
    0 & otherwise 
    \end{cases}
    \]
  - **Stay mobile:**
    \[
    T_{mobile}(A_{t+1} = s_i | Q_t = s_o, A_t = s_j) \propto \begin{cases} 
    0 & s_i = s_j \\
    1 & otherwise 
    \end{cases}
    \]
  - **Stalk the Quarry:**
    \[
    T_{stalk}(A_{t+1} = s_i | Q_t = s_o, A_t = s_j) \propto \frac{1}{\text{dist}(s_i, s_o)}
    \]
  - **Balancing** \([0.8, 0.1, 0.1] \)
Game of Tag: Domains

Dead-Ends

Arena

Circle
# Results

## Success Rate:

<table>
<thead>
<tr>
<th>Model</th>
<th>Domain</th>
<th>Capture%</th>
<th>( E(\text{Steps}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Dead-ends</td>
<td>100</td>
<td>14.8</td>
</tr>
<tr>
<td></td>
<td>Arena</td>
<td>80.2</td>
<td>42.4</td>
</tr>
<tr>
<td></td>
<td>Circle</td>
<td>91.4</td>
<td>34.6</td>
</tr>
<tr>
<td>II</td>
<td>Dead-ends</td>
<td>100</td>
<td>13.2</td>
</tr>
<tr>
<td></td>
<td>Arena</td>
<td>96.8</td>
<td>28.67</td>
</tr>
<tr>
<td></td>
<td>Circle</td>
<td>94.4</td>
<td>31.63</td>
</tr>
</tbody>
</table>
Results

Empirical agent location entropy:
Observation Scenario I: Omniposition Quarry: Dead-End
Results

Empirical agent location entropy:
Observation Scenario I: Omniposition Quarry:
Arena
Results

Empirical agent location entropy:
Observation Scenario I: Omniposition Quarry: Circle
Results

Empirical agent location entropy:
Observation Scenario II: Not at Agent Location: Dead-End

![Graph showing the entropy over steps]

Dynamics Based Control Framework – p.41/54
Results

Empirical agent location entropy:
Observation Scenario II: Not at Agent Location: Arena

![Graph showing entropy over steps]
Results

Empirical agent location entropy:
Observation Scenario II: Not at Agent Location: Circle

![Graph showing entropy over steps]

Steps

Entropy
Environment Design: Model coherence

- Environment Design level mapped the original problem onto a Markovian environment model.
- EMT has demonstrated resilience and intuitive design where the environment development concurs exactly with the Markovian model.
- But what would happen if, under the design level mapping, the real world development in response to action and the one prescribed by the model differ from each other?
- How EMT behaves under *action model incoherence*?
Robot Following Experiment

To test EMT performance with incoherent action model, we’ve designed a simulated robot following test, which was carried out using PlayerStage simulator.

The core of the test is the same as the aircraft landing task:

- The following robot has to keep within a certain distance range away from the followed robot.
- The following robot has to move in the direction of the followed robot.
Robot Following Experiment

Notice that the task involves two correlated controls:
- Linear speed control, to keep up with the followed robot;
- Rotation speed control, to keep up with the followed robot direction.

Both controls were subject to EMT control algorithm equipped with the linear graph random walk model.

Notice the action model incoherence due to:
- Correlation between the environment responses to the controls;
- Non-linearity of environment response, in contrast to the linear assumption of the controller’s random walk model.
EMT controllers were able to handle the task reasonably well.
Following with incoherent model

- EMT controllers were able to handle the task reasonably well.
- For example, in response to circularly moving target, resulting in persistent environment model failure, EMT created a consistent following of the target.
Following with incoherent model

- EMT controllers were able to handle the task reasonably well.
Following with incoherent model

- EMT controllers were able to handle the task reasonably well.
- However performance was not optimal.
  - Range was larger then required
  - Direction change response was delayed
Model Calibration

Model incoherence can be fixed, for example by letting controller to train prior to the task.
Model Calibration

- Model incoherence can be fixed, for example by letting controller to train prior to the task.

- But can EMT’s data, the captured momentary dynamics, be used for such training?
Model Calibration

- Model incoherence can be fixed, for example by letting controller to train prior to the task.
- But can EMT’s data, the captured momentary dynamics, be used for such training?
- To answer this question we devised a simple training procedure, which loops through:
  - Selecting an action in a way that scans the action/state space.
  - Performing statistics over the dynamics, captured by EMT, which result from that action.
0. Assume a POMEM $< S, A, T, O, \Omega, s_0 >$ 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
Following task II

EMT data was effective for model calibration.
Following task II

- EMT data was effective for model calibration.
- Even with linear speed controller calibrated alone, overall performance greatly improved.
  - EMT controller alternated between the available motion speeds, (with average) completely matching that of the target robot.
  - Moreover, this was done in a timely fashion to maintain the required distance.
Following task II

- EMT data was effective for model calibration.
Theoretical Remarks

- Kullback-Leibler divergence is dual to likelihood
- EMT tends towards a more likely explanation.

- For constant dynamics, a sequence of EMT updates weakly converges to that dynamics,

- EMT-based control prefers actions that produce dynamics likely relatively to the target $\tau^*$.

- If a system is controllable, the sequence of EMT updates is forced towards the prescribed dynamics, and potentially converges.
Conclusions

- Dynamics Based Control (DBC) paradigm
  - DBC concentrates on achieving system dynamics, not state

- Extended Markov Tracking based control
  - Algorithmic approximation of DBC
  - Polynomial in controlled system description
  - Maintains polytime in multiagent case
    - In all parameters, except number of agents
  - Copes with incoherent models
  - Copes with interacting multiple targets
(Close) Future Work

- EMT variation for a continuous state/action space dynamic system
  - Strongly non-linear systems
- Extension to other environment models
  - Predictive State Representations (PSRs)
Future Work

- General case DBC algorithm
- DBC in hybrid controllers
- Theoretical foundations
  - Complexity, convergence, stability proofs
Credits

- Comparison to Foundation is due to Dr. Ranjit Nair.
- “Bird” and “Sphere” illusions are part of Dr. Michael’s Bach collection.
THANK YOU