Distributed Intelligent Systems – W7
Multi-Level Modeling: Calibration and Combination with Machine Learning
Outline

• Summary of the multi-level modeling recipe
• Calibration methods for multi-level models
  – Microscopic and macroscopic parameters
  – Approximations
• A challenging example: distributed seed assembly
• Combined modeling and machine-learning methods
  – Homogenous and heterogeneous learning
  – Diversity and specialization
Multi-Level Modeling Recipe
1. Target System & Task(s)

Perform basic design choices for the experimental set-up:

- Hardware and software for the robotic platform
- Environment in which robots operate
- Task(s) robots must accomplish
2. Submicroscopic Model

Implement faithfully your design choices in a submicroscopic model (in principle even running the same control code; libraries and APIs or middleware interfaces such as ROS are usually provided in standard open-source simulators)
3. Model Structure and Metrics

- Exploit controller blueprint at submicroscopic/physical level as structure for higher level of abstraction ("behavior = state"); use it for both microscopic and macroscopic levels.
- State granularity arbitrary but (non spatial) performance metrics must be computable explicitly at all modeling levels.

\[
\text{Performance} = f(#\text{robots}_{\text{search}}, #\text{robots}_{\text{grip}})
\]
4. Microscopic Model

- Aggregate local interactions and reduce intra-robot details
- Maintain state space’s structure as defined at Step 3
- Maintain individual representation (and exact discrete quantities) for each robotic node and environmental object of interest
5. Macroscopic Model

- Aggregate individual nodes into one or multiple representations (castes) at collective level
- Maintain state space’s structure as defined at Step 3
- Solve numerically or analytically the ODE system (mean field approach)
- Exploit conservation laws (e.g. # of robots in an enclosed arena) to simplify the representation of the dynamical system
6. Model Parameters

- **Number of parameters**: decreasing with increasing abstraction
- **Incremental calibration**: a given level can be calibrated on the underlying one using aggregation techniques (e.g., first moment for a distribution at lower abstraction level)
- **Explicit representation**: any system parameter of interest should be captured explicitly at a given level
- **Multiple calibration methods** for model parameters:
  - Ad hoc experiments (e.g., interaction time, sensor transfer functions)
  - System identification techniques (with constrained parameter fitting)
  - Statistical verification techniques (e.g., trajectory analysis)
- **Submicroscopic models**: large parameter space (e.g., individual sensor and actuator features).
- **Micro- and macroscopic models**, essentially two parameter types:
  - State durations
  - State transition probabilities
Model Calibration
State Durations & Discretization Interval

1. Measure all interaction times of interest in your system, i.e. those which might influence the system performance metrics. 
   
   **Note:** often “delay states” can just summarize all what you need without getting into the details of what’s going on within the state.

2. Consider only average values (we might consider also parametrized distributions in the future, the modeling methodology does not prevent to do so)

3. For time-discrete systems: choose the **time step** $T = \text{GCF of all the durations measured}$ (e.g., 3 s obstacle avoidance, 4 s object manipulation, $T = 1$ s) -> no rounding error. 
   
   **Note:** more accuracy in parameter measuring means in this case more computational cost when simulating
State Transition Probabilities

- Geometric considerations
- Ad hoc calibration experiments
- Ex. stick-pulling experiment

\[ p_s = \frac{A_s}{A_a} \]
\[ p_r = \frac{A_r}{A_a} \]
\[ p_R = p_r (N_0 - 1) \]
\[ p_w = \frac{A_w}{A_a} \]
\[ p_{g1} = p_s \]
\[ p_{g2} = R_g p_s \]

\( A_a = \text{surface of the whole arena} \)

Note: As defined in [Martinoli et al, IJRR 2004]!
Experimental Validation of Spatiality Assumption

Symmetry of Stick Distribution

# sticks

Default

# sticks 13
Experimental Validation of Spatiality Assumption

Nonembodied obstacles = detection surfaces

Numerical example (mean ± std dev, 3 locations, 100 h simulated time):

<table>
<thead>
<tr>
<th></th>
<th>Square</th>
<th>Rect.</th>
<th>Round</th>
<th>All shapes</th>
<th>Geometry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normalized detection surface</td>
<td>0.31 ± 0.04</td>
<td>0.3 ± 0.03</td>
<td>0.32 ± 0.02</td>
<td>0.31 ± 0.03</td>
<td>0.31</td>
</tr>
</tbody>
</table>
Geometric Probabilities $g_i$

[Correll & Martinoli, ISER 2004]

- $g_s$, $g_w$, ... are function of sensor range, behavior, robot’s and object’s size, ... : interaction characterization!
- Geometric probabilities can be considered normalized detection areas (normalized over the total area of the experiment).

**Example: stick**

$$g_s = \frac{A_s}{A_{arena}}$$
Encountering Probabilities

[Correll & Martinoli, ISER 2004]

1. Measure geometric probabilities of detection $g_i$
2. Calculate the **encountering rate** $r_i$ [s$^{-1}$] for the object $i$ from the geometric probabilities $g_i$:

   $$r_i = \frac{v W_s}{A_s} g_i$$

   $A_s =$ detection area of the smallest object
   $v =$ mean robot speed
   $W_s =$ robot’s detection width for the smallest object (center-to-center)

3. For time-discrete models, calculate the **encountering probabilities** $p_i$ (per time step) from the encountering rates:

   $$p_i = r_i T$$

**Note:** slightly different from [Martinoli et al., IJRR04] (decoupled time and space)!
Model Calibration - Practice

- Assumptions (well-mixed, linear overlap of areas) might be only partially fulfilled
- We do not capture distributions in the model parameters, only deterministic average values; distributions might more faithfully capture:
  - Controller type (e.g., distal vs. proximal)
  - Active vs. passive objects (e.g., robot vs. wall)
  - Embodiment vs. non embodiment (e.g., area vs. real obstacle)
  - Way of measuring your metrics (e.g., egocentric, allocentric)
  - Impact on the considered swarm performance metric through error propagation (clear decoupling between parameters and structure inaccuracies of the model)
Model Calibration - Practice

Bin distribution of interaction time $T_a$ (mean $T_a = 25 \times 50\text{ ms} = 1.25\text{ s}$)

- Micro/macro, deterministic delay
- Micro/macro, prob. delay
- Sub-microscopic, distal controller
- Submicroscopic, proximal contr.
Geometric probability $g$: example of transition in space from search to obstacle avoidance (1 moving robot, 1 dummy robot, Webots measurements, egocentric)
A Challenging Example:
Distributed Seed Assembly
Robot behavior

• Reactive, non-communicating, non-adaptive behavior

• Qualitative stigmergy important: 2 rules in interaction with cluster:
  – Avoid if interaction with the cluster body
  – Manipulate if interaction with cluster tips

• Quantitative stigmergy minimal:
  – the bigger, the more stable the cluster
  – big cluster (>2) = number of manipulation sites as cluster of 2 seeds
  – almost no difference between cluster incrementing and decrementing probabilities

• 1 robot state: loaded, free
Robot Controller

Start → Search

Object? (N) → Search

Seed? (N) → Loaded?

Seed picking-up

Seed? (Y) → Robot?

Seed picking-up

Robot? (N) → Interference

Interference (Y) → Obstacle Avoidance

Obstacle Avoidance (N) → Search

Seed dropping

Interference (N) → Seed picking-up

Loaded? (N) → Seed picking-up

Seed dropping

Robot? (Y) → Interference

Interference (N) → Search

Obstacle Avoidance (Y) → Seed picking-up

Seed dropping

Seed picking-up

Seed dropping
State Granularity Choices

• Idea: split the search state in search (loaded) and search (free) and keep dedicated avoidance/interferences states for each of the split states

• Motivation:
  • Chosen metrics: assembly progress based on seeds on ground → seeds need to be tracked also when they are in the robot grippers (conservation law of seeds)
  • Status loaded (carrying a seed) or free (not carrying a seed) can only change in a deterministic fashion (e.g., cannot be changed by an avoidance operation) and only by going to one of the seed dropping or picking states → can be explicitly represented with a larger state space and deterministic transitions
  • Such enlarged state space facilitates the writing of the ODEs
Robot Controller – Restructured

Start

Search (free)

Object? N

Seed? N

Seed picking-up

Search (loaded)

Object? N

Seed? Y

Obstacle Avoidance

Interference Y

Robot? N

Obstacle Avoidance Y

Seed? Y

Seed dropping
Robot Controller

From Agassounon et al, 2004 (restructured representation):
Micro-Macroscopic Models

From Agassounon et al, 2004 (restructured representation):
Parameter Calibration

Geometric Estimations

- Incrementing probabilities
  
[Diagram showing different geometric estimations with incrementing probabilities]

- Decremiting probabilities
  
[Diagram showing different geometric estimations with decrementing probabilities]

Resulting Probabilities

Perimeters are relevant for computing the cluster modifying probabilities: robot turns on the spot for object distinction before approaching the cluster!
Models: Explanations and Predictions

Single cluster?  All models predicted yes and in roughly how much time!

Number of clusters (inter-distance between seeds < 1 seed) monotonically decreases if:

• Probability to create a NEW cluster of 1 seed in the middle of the arena is equal to zero
• No hard partitioning of the arena (robot homogeneously mix clusters)
• Cluster are not broken in two parts by removing one seed in the middle
Long Seed-Assembling Experiments

Submicroscopic Model
(Webots)

Real robots
(Khepera I)

• 10% white noise on all sensor and actuators
• Perfectly homogeneous team
• Kinematic mode

• Electrical floor: continuous power supply in any position and orientation
• Heterogeneities among teammates and components
• Inaccuracies in acting and sensing
• Dynamics (e.g., friction) plays a role
Results – Until a Single Cluster

- 3 robots
- real robots (5 runs), submicroscopic (10 runs), microscopic model (100 runs)
- [Martinoli, Ijspeert, Mondada, 1999]

- Mean size of clusters
- Size of the biggest cluster
- Number of clusters
Examples of Assembled Structures

Noise in S&A and poor navigation capabilities do not allow for precise, controllable structure building.

Submicroscopic

Real robots
Macroscopic Model: Distributed Building Dynamics

- \( d_i(k) = \text{decr}_i \text{ geom}_i \text{ probability}_i * p \_\text{find}_i(k) \)
- \( c_i(k) = \text{incr}_i \text{ geom}_i \text{ probability}_i * p \_\text{find}_i(k) \)
- \( p \_\text{find}_i(k) = \text{finding probability of all the cluster of size } i \)
- If \( n = \text{number of seeds} \rightarrow \text{macroscopic model of environment with } n \text{ nonlinearly coupled ODE (n for each possible cluster size)} + \text{robot states} \)

Some Results from Agassounon et al., 2004 (1, 5, 10 robots always active)

**Metric**: average cluster size (20 seeds)

Saturation phase: all seeds in a single cluster or in the robots’ grippers

1 and 5 robots

10 robots
Journal Publications using the Same Modeling Framework

**Stick Pulling**
- [Lerman, Galstyan, Martinoli, Ijspeert, *Artificial Life*, 2001]

**Object Aggregation**

**Robot Aggregation and Swarming** – explore no arena bounds
- [Winfield, Liu, Nembrini, Martinoli, *Swarm Intelligence J.*, 2008]

**Coverage** – use spatial models
Combined Modeling and Machine-Learning Methods
Rationale for Combined Methods (1)

• Any level of modeling (submicro, micro, or macro) allow us to consider certain parameters and leave others; models, as expression of reality abstraction, can be considered as more or less coarse “filters” of the reality

• Combined modeling/machine-learning techniques can be used at any of the abstraction levels; machine-learning techniques will explore the design parameters explicitly represented at a given level of abstraction

• Depending on the features of the hyperspace to be searched (size, continuity, noise, etc.), appropriate machine-learning techniques should be used (e.g., hill-climbing vs. population-based

• One particular optimization problem is system identification: the performance to optimize is the matching with the reality (or with a lower abstraction level). See model calibration in [Correll & Martinoli, DARS 2006].
Rationale for Combined Methods (2)

Macroscopic + ML? Most of the time not needed since very fast + continuous; homogeneous systems mainly; standard numerical optimization techniques/systematic search can be used

Microscopic + ML (see this lecture’s examples); for instance, diversity and specialization can be studied

Submicroscopic + ML (see Week 8 and 9 examples using PSO); for instance low-level design parameters can be learned

Target system + ML = adaptation with HW in the loop (on-board or off-board)
In-Line Adaptive Learning
In-Line Adaptive Learning
(Li, Martinoli, Abu-Mostafa, 2001)

- **GTP**: Gripping Time Parameter
- **Δd**: learning step
- **d**: direction
- Underlying low-pass filter for measuring the performance

Randomly pick $d$ from $\{+,-\}$

FIRST TRY

GTP $\leftarrow$ GTP + $\Delta_d$

worse $\quad$ better

Enlarge $\Delta_d$

SECOND TRY

GTP $\leftarrow$ GTP + $\Delta_d$

worse $\quad$ better

SWITCH DIR

$d \leftarrow -d$

GTP $\leftarrow$ GTP + $\Delta_d$

worse $\quad$ better
Algorithm Parameters

From Li et al., *Adaptive Behavior*, 2004

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_m$</td>
<td>2400</td>
<td>Averaging period for reinforcement signal (sec)</td>
</tr>
<tr>
<td>$E$</td>
<td>1.9</td>
<td>GTP offset enlarge factor</td>
</tr>
<tr>
<td>$F$</td>
<td>0.3</td>
<td>GTP factor enlarge ratio</td>
</tr>
<tr>
<td>$U$</td>
<td>2</td>
<td>GTP offset shrink divider</td>
</tr>
<tr>
<td>$V$</td>
<td>0.5</td>
<td>GTP factor shrink ratio</td>
</tr>
</tbody>
</table>

Low-pass filter

Adapting rules for the learning step
In-Line Adaptive Learning

**Differences with gradient descent methods:**
- Fixed rules for calculating step increase/decrease → limited descent speed → no gradient computation → more conservative but more stable
- Randomness for getting out from local minima (no momentum)
- Underlying low-pass filter is part of the algorithm

**Differences with Reinforcement Learning:**
- No learning history considered (only previous step)

**Differences with basic In-Line Learning:**
- Step adaptive → faster and more stability at convergence
Co-Learning in a Collaborative Framework
Sample Results – Homogeneous Learning

Short averaging window
(filter cut-off $f_{\text{high}}$)

Long averaging window
(filter cut-off $f_{\text{low}}$)

- Systematic (mean only)
- Learned (mean + std dev)

---

Note: 1 parameter for the whole group!
**Heterogeneous Learning**

Key question: does team diversity enhance performance? I.e., can individual members become specialized?

Performance **ratio** between 2 caste and homogeneous system (submicro/micro models, systematic search)
Heterogeneous vs. Homogenous Learning

Performance ratio between heterogeneous (full and 2-castes) and homogeneous groups AFTER learning

Notes:
- large $T_m$ (long averaging window)
- only private strategies
- global = group
  local = individual

[Li et al., *Adaptive Behavior*, 2004]
Measuring Diversity and Specialization
Diversity Metrics  
(Balch 1998)

Entropy-based diversity measure introduced in AB-04 could be used for analyzing threshold distributions

Simple entropy:  \[ H(\mathcal{R}) = -\sum_{i=1}^{m} p_i \log p_i. \]

Social entropy:  \[ D(\mathcal{R}) = \int_{0}^{\infty} H(\mathcal{R}, h) \, dh. \]

\( p_i \) = portion of the agents in cluster \( i \); \( m \) cluster in total; \( h = \) taxonomic level parameter

---

Input: a swarm system \( \mathcal{R} = \{r_1, r_2, \ldots, r_n\} \) of size \( n \); a difference measure \( d \).

For different level \( h \), the \( C_u \) clustering algorithm does:

1. Initialize \( n \) clusters with cluster \( c_i = \{r_i\} \);
2. For each \( c_i \): for each \( r_j \): If \( d(r_j, r_k) \leq h \) for all \( r_k \) in \( c_i \), add \( r_j \) to cluster \( c_i \);
3. Discard redundant clusters;
4. Calculate \( p_i \) and the entropy \( H(\mathcal{R}, h) \). Note that when \( r_j \) belongs to \( s \) clusters including \( c_i \), its contribution to \( p_i \) is \( 1/sn \).

Return \( \int_{0}^{\infty} H(\mathcal{R}, h) \, dh \) as the hierarchic social entropy.
Example – Simple Entropy

- $R = \{r_1, r_2, r_3\}$
- $n = 3$ (three swarm points)
- bi-dimensional space
- define a distance: Euclidian distance
- $h =$ taxonomic level parameter
- $m =$ number of clusters

$$H(R) = -\sum_{i=1}^{3} p_i \log p_i = H\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right) =$$

$$= -3 \frac{1}{3} \log \frac{1}{3} = 0.477$$

$$H(R) = -\sum_{i=1}^{2} p_i \log p_i = H\left(\frac{1}{3}, \frac{2}{3}\right) =$$

$$= -\frac{1}{3} \log \frac{1}{3} - \frac{2}{3} \log \frac{2}{3} = 0.159 + 0.117 = 0.276$$
Example – Simple Entropy

\[ H(R) = -\sum_{i=1}^{2} p_i \log p_i = H\left(\frac{1}{3} + \frac{1}{3} + \frac{1}{3}\right) = H\left(\frac{1}{2}, \frac{1}{2}\right) \]

\[ H(R) = -\sum_{i=1}^{1} p_i \log p_i = H\left(\frac{3}{3}\right) = -\log 1 = 0 \]

\[ \frac{1}{2} \log \frac{1}{2} + \frac{1}{2} \log \frac{1}{2} = 0.301 \]

Check \( \sum_{i=1}^{m} p_i = 1 \) with overlapping clusters!
Example – Social Entropy

\[ D(R) = \int_0^\infty H(R, h)dh = 3 \times H\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right) + 1 \times H\left(\frac{1}{3}, \frac{2}{3}\right) + 2 \times H\left(\frac{1}{2}, \frac{1}{2}\right) + 0 = 2.309 \]

Note: In contrast to simple entropy \( \geq 1 \)

Contrast with \( R = \{r_1, r_2, r_3\} \) and \( r_1 = r_2 = r_3 \) (homogeneous swarm), for any \( h \geq 0 \) \( \rightarrow \) single cluster \( \rightarrow \) \( D(R) = 0! \)
Differences with Plain Euclidian Diversity Measure

\[ d(a, b) = \sqrt{\sum_i (a_i - b_i)^2} \]

\[ D_{eu} = \frac{1}{N(N - 1)} \sum_a \left[ \sum_{b \neq a} d(a, b) \right] \]

- Underlying distance measure in the solution space might be the same (e.g. Euclidian distance)
- Social entropy is looking for possible clustering of the vectors (looking for possible castes) while Euclidian diversity is just looking how spread out/diverse in general are the vectors

Components in all dimensions

All points from any other point
Specialization Metric

Specialization metric introduced in AB-04:

\[ S = \text{corrcoef}(D; R) \times D. \]

S = specialization; D = diversity (e.g., social entropy); R = swarm performance

Notes

• Idea: “weighting diversity with performance”
• This is useful when the number of tasks to be solved is not well-defined or it is difficult to assess the task granularity a priori. In such cases the mapping between task granularity and caste granularity might not trivial (see the limited performance of a caste-based solution in the stick-pulling experiment)
• Could be used for analyzing specialization arising from a variable-threshold division of labor algorithm (see lecture Week 10)
Sample Results in the Standard Sticks

- 2 serial grips needed to get the sticks out
- 4 sticks, 2-6 robots, 80 cm arena

Relative Performance
- Specialists more important for small teams
- Local $p >$ global $p$
- Enforced caste: pay the price for odd team sizes

Diversity
- Measured using social entropy
- Flat curves, difficult to tell whether diversity brings performance

Specialization
- Specialization higher with global when needed, drop more quickly when not needed
- Enforcing caste: “low-pass filter” effect
Conclusion
Take Home Messages

• The multi-level modeling methodology is a framework that has been successfully used in multiple case studies.

• Models’ parameter calibration is difficult and still an open challenge.

• An additional case study has illustrated how to capture time-varying parameters depending on the environmental modifications introduced by the robots and how to choose an appropriate state granularity.

• Different modeling levels can be combined with machine-learning for design and optimization purposes.

• Microscopic models allows for efficiently studying diversity and specialization issues.

• Specialization is the part of diversity that improves performance.

• The diversity and specialization level of a heterogeneous swarm can be quantitatively measured.
Additional Literature – Week 7

Papers


