Distributed Intelligent Systems – W8
Multi-Level Modeling Methods
Applied to Distributed Robotic Systems
Outline

• Multi-Level Modeling Methodology
  – Motivation and rationale
  – Theoretical background
  – Methodological framework

• Examples
  – Obstacle avoidance (linear)
  – Collaborative stick pulling (nonlinear)
Modeling Rationale, Choices, and Framework Overview
Motivation for Modeling

“Modeling of systems has usually three objectives: abstraction, simplification, and formalization.” (p. 96)

“The main objective of modeling in swarm robotics is dimension reduction.” (p. 97)
Motivating Examples - Manipulation

Puck Aggregation
(5 robots – 25 cm)

[Beckers et al., SAB 1994]
[Martinoli et al., ECAL 1999]

Seed-Chain Assembling
(10 Khepera I – 5.5 cm)

[Martinoli et al., Robotic and Autonomous Systems, 1999]
[Agassounon et al., Autonomous Robots, 2004]
Motivating Examples - Sensing

Wireless-Based Swarming
(7 Linuxbots, 24 cm)
[Nembrini et al., SAB 2002]
[Winfield et al., Swarm Intelligence, 2008]

Wireless-Based Swarming
(40 e-pucks, 7 cm)
[Pereira et al., IROS 2013]
Another Key Motivation -
The Reverse Engineering Problem

e-puck robots, an EPFL-FIFO project by ASL/LSRO-DISAL-LIS, 2007
Motivation for Modeling

- Understanding the interplay of the various elements of the system (e.g., robot features, robot numbers, environment, noise level)
- Formally analyzing system properties
- Having additional tools for designing and optimizing the swarm robotic system
- Delivering performance predictions for the ensemble in shorter time or before doing actual experiments
- Investigating experimental conditions difficult or impossible to reproduce in reality
Modeling Choices

- **Gray-box approach**: to easily incorporate a priori information (e.g., # of agents, technological and environmental features)

- **Probabilistic**: to capture noisy interactions, noisy robotic components, stochastic control policies, and enable aggregation schemes towards abstraction

- **Multi-level**: to represent explicitly different design choices, trade off computational speed and faithfulness to reality, bridge mathematically tractable models and reality in an incremental way

- **Bottom-up**: start from the physical reality and increase the abstraction level until the highest abstraction level
Multi-Level Modeling Methodology

Macroscopic: representation of the whole swarm (typically a mathematical model)

Microscopic: multi-agent models, only relevant robot features captured, 1 agent = 1 robot

Submicroscopic: intra-robot (e.g., S&A, transceiver) and environment (e.g., physics) details reproduced faithfully

Target system (physical reality): information on controller, S&A, communication, morphology and environmental features
Multi-Level Implementation
Choices for this Course

• **Submicroscopic**: Webots

• **Microscopic**: non spatial, state = behavior, exact model in terms of quantities (e.g., agent/state)

• **Macroscopic**: non spatial, mean field approach, Ordinary Differential Equation (ODE) approximation applies (e.g., average number agents/state)
Experimental Invariant Features and Modeling Assumptions
Invariant Experimental Features

- **Short-range** (typically 1 robot diameter), **crude** (noisy, a few discrimination levels) proximity sensing
- **Local communication and teammate sensing** carried out with potentially longer range communication channels
- **Full mobility but basic navigation** (no planning, no absolute localization)
- **Reactive, behavior-based control**, with a few internal states, designed from a local perspective
- **Not overcrowded arenas**
- **Multiple runs** (typically 5+) for the same experimental parameters; **randomized robot poses** at the beginning
Modeling Assumptions:
Semi-Markovian Properties

- Description for environment and multi-robot system using states
- The system future state is a function of the current state (and possibly of the amount of time spent in it)

Submicroscopic
(pose, S&A state, etc.)

Microscopic/Macroscopic
(transition probabilities, state duration)
Modeling Assumptions: Spatiality

- **nonspatial metrics** for collective performance
- **well-mixed system** because of simple navigation, multiple randomized interactions in a convex environment, multiple runs with randomized initial conditions, no overcrowding (sparseness)

Submicroscopic: **spatial**

Micro/macroscopic: **nonspatial**

Free space
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>
Experimental Validation of Spatiality Assumption

Symmetry of Stick Distribution

Default

# sticks 17
Methodological Framework: Theoretical Background
Microscopic Level

\[ p(n,t) = \text{probability of an agent to be in the state } n \text{ at time } t \]

If Markov properties fulfilled:

\[ \Delta p(n,t) = p(n,t + \Delta t) - p(n,t) \]

\[ = \sum_{n'} p(n,t + \Delta t \mid n',t) p(n',t) - \sum_{n'} p(n',t + \Delta t \mid n,t) p(n,t) \]

*Transition probability*

*Sum over all possible states* \( n' \) *the agent can be in*

*Probability the agent was in a given state* \( n' \)

*Inflow*

*Outflow*
Macroscopic Level – Time-Continuous

Left and right side of the equation: averaging over the total number of agents, dividing by $\Delta t$, limit $\Delta t \to 0$; neglect distributions of the stochastic variables and assume homogeneous agents (mean field approach):

$$\frac{dN_n(t)}{dt} = \sum_{n'} W(n \mid n', t) N_{n'}(t) - \sum_{n'} W(n' \mid n, t) N_n(t)$$

Rate Equation (time-continuous)

n, n’ = states of the agents (all possible states at each instant)

$N_n = \text{average fraction (or mean number) of agents in state } n \text{ at time } t$

$$W(n \mid n', t) = \lim_{\Delta t \to 0} \frac{p(n, t + \Delta t \mid n', t)}{\Delta t}$$

Transition rate
Macroscopic Level – Time-Discrete

Rate Equation (time-discrete):

\[ N_n((k+1)T) = N_n(kT) + \sum_{n'} TW(n | n', kT) N_{n'}(kT) - \sum_{n'} TW(n' | n, kT) N_n(kT) \]

\text{inflow} \hspace{1cm} \text{outflow}

k = iteration index
T = time step, sampling interval
TW = transition probability per time step

Notation often simplified to:

\[ N_n(k+1) = N_n(k) + \sum_{n'} P(n | n', k) N_{n'}(k) - \sum_{n'} P(n' | n, k) N_n(k) \]

T is specified in the text once of all, P is calculated from T*W or other calibration methods
Time Discretization: The Engineering Recipe

Time-discrete vs. time-continuous models:

1. Assess what’s the time resolution needed for your system performance metrics (if time step chosen appropriately small, no impact on prediction accuracy in the type of experiments presented)

2. Choose whenever possible the most computationally efficient model: time-discrete less computationally expensive than emulation of continuity (e.g., Runge-Kutta, etc.)

3. Advantage of time-discrete models: a single common sampling rate can be defined among different modeling levels
Methodological Framework: An Incremental Bottom-Up 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. Metric(s) and State Space

- Define system performance metric(s)
- Define state space (number of states, granularity)
- State granularity arbitrary but (non spatial) performance metrics must be computable explicitly at all modeling levels
- Exploit controller blueprint at submicroscopic/physical level as structure for higher level of abstraction (“behavior = state”)

Performance = \( f(#\text{robots}_{\text{search}}, #\text{robots}_{\text{grip}}) \)
3. Submicroscopic Model

Implement faithfully your design choices in a submicroscopic model (in principle even running the same control code; libraries and APIs are usually provided in standard commercial or open-source simulators)
4. Microscopic Model

- Aggregate local interactions and reduce intra-robot details
- Maintain state space’s structure as defined at Step 2
- 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 2
- 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. Parameter Calibration

- Number of parameters is decreasing with the abstraction level
- Calibrate a given level based on the underlying one (e.g., submicroscopic with physical system; microscopic with submicroscopic, macroscopic with microscopic)
- Parametric (e.g., mean only, mean and variance) or non parametric (actual distribution recorded at the lower level) assumptions
- Various methods available
  - Ad hoc experiments [Correll & Martinoli, ISER 2004]
  - System identification techniques (e.g., constrained parameter fitting) [Correll & Martinoli, DARS 2006]
  - Statistical verification techniques (e.g., trajectory analysis) [Roduit et al., IROS 2007]
- Parameter example for micro- and macroscopic models:
  - State durations
  - State transition probabilities

\[ p_{\text{in}} \rightarrow T_{\text{state}} \rightarrow p_{\text{out}} \]
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 parameter 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

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

\[ A_a \] = surface of the whole arena
Linear Example: Obstacle Avoidance
A Simple Linear Model

Example: search (moving forwards) and obstacle avoidance

© Nikolaus Correll 2006
A Simple Example

Deterministic & microscopic characterization

PFSM

Probabilistic agent’s flowchart

Nonspatiality

robot’s flowchart
Linear Model – Probabilistic Delay

- **Ta** = mean obstacle avoidance duration
- **pa** = probability of moving to obstacle avoidance
- **ps** = probability of resuming search
- **Ns** = average # robots in search
- **Na** = average # robots in obstacle avoidance
- **N0** = # robots used in the experiment
- **k** = iteration index

\[ N_s(k+1) = N_s(k) - p_a N_s(k) + p_s N_a(k) \]

\[ N_a(k+1) = N_0 - N_s(k+1) \]

- **Ns(0) = N0 ; Na(0) = 0**

- **Ts =** mean obstacle avoidance duration
- **pa =** probability of moving to obstacle avoidance
- **ps =** probability of resuming search
- **Ns =** average # robots in search
- **Na =** average # robots in obstacle avoidance
- **N0 =** # robots used in the experiment
- **k = 0,1, ...** (iteration index)
Linear Model – Deterministic Delay

\[ N_s(k+1) = N_s(k) - p_a N_s(k) + p_a N_s(k-T_a) \]

\[ N_a(k+1) = N_0 - N_s(k+1) \]

\[ Ta = \text{mean obstacle avoidance duration} \]
\[ p_a = \text{probability moving to obstacle avoidance} \]
\[ N_s = \text{average } \# \text{ robots in search} \]
\[ N_a = \text{average } \# \text{ robots in obstacle avoidance} \]
\[ N_0 = \# \text{ robots used in the experiment} \]
\[ k = 0, 1, \ldots \text{ (iteration index)} \]
Linear Model – Sample Results

\[ \frac{N_a^*}{N_0} \]

**Submicro to micro comparison**
(different controllers, steady state comparison)

**Micro to macro comparison**
(same robot density but wall surface become smaller with bigger arenas)
Steady State Analysis

- \( N_n(k+1) = N_n(k) \) for all states \( n \) of the system → \( N_n^* \)
- Note 1: equivalent to differential equation of \( \frac{dN_n}{dt} = 0 \)
- Note 2: for time-delayed equations easier to perform the steady-state analysis in the Z-space but in t-space also ok (see IJRR-04)
- For our linear example (deterministic delay option):

\[
N_s^* = \frac{N_0}{1 + p_a T_a} \quad N_a^* = \frac{N_0 p_a T_a}{1 + p_a T_a}
\]

Ex.: normalized mean number of robots in search mode at steady state as a function of time for obstacle avoidance.
Nonlinear Example – Collaborative Stick Pulling
The Stick-Pulling Case Study

Physical Set-Up

- 2-6 robots
- 4 sticks
- 40 cm radius arena

Collaboration via indirect communication

- IR reflective band
- Proximity sensors
- Arm elevation sensor
Systematic Experiments

Real robots

• [Martinoli and Mondada, ISER, 1995]
• [Ijspeert et al., AR, 2001]

Submicroscopic model
Results of Experiments and Submicroscopic Modeling

- Real robots (3 runs) and submicroscopic model (10 runs)
- System bifurcation as a function of #robots/#sticks

\[ N_{\text{robots}} > N_{\text{sticks}} \]

\[ N_{\text{robots}} \leq N_{\text{sticks}} \]
State Transition Probabilities

\[ A_a = \text{surface of the whole arena} \]

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

Deterministic robot’s flowchart

Nonspatiality & microscopic characterization

PFSM Probabilistic agent’s flowchart
Full Macroscopic Model

For instance, for the average number of robots in searching mode:

\[
N_s(k+1) = N_s(k) - \left[ \Delta g_1(k) + \Delta g_2(k) + p_w + p_R \right] N_s(k) - \Delta g_1(k - T_{cga}) \Gamma(k; T_a) N_s(k - T_{cga}) + \\
+ \Delta g_2(k - T_{ca}) N_s(k - T_{ca}) + \Delta g_2(k - T_{cda}) N_s(k - T_{cda}) + p_w N_s(k - T_a) + p_R N_s(k - T_{ia})
\]

with time-varying coefficients (nonlinear coupling):

\[
\Delta g_1(k) = p_{g1} [M_0 - N_g(k) - N_d(k)] \\
\Delta g_2(k) = p_{g2} N_g(k) \\
\Gamma(k; T_{SL}) = \prod_{j=k-T_g-T_{SL}}^{k-T_{SL}} [1 - p_{g2} N_s(j)]
\]

- 6 states: 5 DE + 1 cons. EQ
- \( T_i, T_a, T_d, T_c \neq 0; T_{xyz} = T_x + T_y + T_z \)
- \( T_{SL} \) = Shift Left duration
- [Martinoli et al., *IJRR*, 2004]
**Swarm Performance Metric**

**Collaboration rate: # of sticks per time unit**

\[
C(k) = p_{g2} N_s(k-T_{ca}) N_g(k-T_{ca})
\]

: mean # of collaborations at iteration k

\[
\sum_{k=0}^{T_e} C(k)
\]

\[
C_t(k) = \frac{k=0}{T_e}
\]

: mean collaboration rate over \(T_e\)
Results (Standard Arena)

- Submicro (10 runs)
- Micro (100 runs)
- Macro (1 run)

Discrepancies because of ODE approximation (nonlinearities + discrete exact vs. average quantities)
Results: 4 x #Sticks, #Robots and Arena Area

- Submicro (10 runs)
- Micro (100 runs)
- Macro (1 run)
Reducing the Macroscopic Model

Goal: reach mathematical tractability

\[ T_i, T_a, T_d, T_c \ll T_g \rightarrow T_i = T_a = T_d = T_c = 0 \]
**Reduced Macroscopic Model**

- **Nonlinear coupling!**

**Search** → **Grip**

\[ N_s(k+1) = N_s(k) - p_{g1}[M_0 - N_g(k)]N_s(k) + p_{g2}N_g(k)N_s(k) + p_{g1}[M_0 - N_g(k-T_g)]\Gamma(k;0)N_s(k-T_g) \]

\[ N_g(k+1) = N_0 - N_s(k+1) \]

Initial conditions and causality

\[ N_s(0) = N_0, \ N_g(0) = 0 \]

\[ N_s(k) = N_g(k) = 0 \ \text{for all} \ k < 0 \]

- **successful**
- **unsuccessful**

\[ \Gamma(k;0) = \prod_{j=k-T_g}^{k}[1 - p_{g2}N_s(j)] \]

**N_s** = average # robots in searching mode

**N_g** = average # robots in gripping mode

**N_0** = # robots used in the experiment

**M_0** = # sticks used in the experiment

\( \Gamma \) = fraction of robots that abandon pulling

**T_e** = maximal number of iterations

\[ k = 0, 1, \ldots, T_e \] (iteration index)
Results Reduced Microscopic Model

- Microscopic (100 runs) and macroscopic models overlapped
- Only qualitatively agreement with submicroscopic/real robots results

- 4 robots, 4 sticks, $R_a = 40$ cm
- 16 robots, 16 sticks, $R_a = 80$ cm
Steady State Analysis (Reduced Macro Model)

- Steady-state analysis \( [N_n(k+1) = N_n(k)] \) → It can be demonstrated that:

\[
\exists \quad T_g^{opt} \quad for \quad \frac{N_0}{M_0} \leq \frac{2}{1 + R_g}
\]

with \( N_0 = \) number of robots and \( M_0 = \) number of sticks, \( R_g \propto \) approaching angle for collaboration.

- **Counterintuitive conclusion**: an optimal \( T_g \) can exist also in scenarios with more robots than sticks if the collaboration is very difficult (i.e. \( R_g \) very small)!
Example: \( \tilde{R}_g = \frac{1}{10} R_g \) (collaboration very difficult)
Optimal Gripping Time

- Steady-state analysis → $T_g^{opt}$ can be computed analytically in the simplified model (numerically approximated value):

$$T_g^{opt} = \frac{1}{\ln(1 - p_gR_g \frac{N_0}{2})} \ln \frac{1 - \frac{\beta}{2}(1 + R_g)}{1 - \frac{\beta}{2}} \quad \text{for} \quad \beta \leq \beta_c = \frac{2}{1 + R_g}$$

with $\beta = N_0/M_0 = \text{ratio robots-to-sticks}$

- $T_g^{opt}$ can be computed numerically by integrating the full model ODEs or solving the full model steady-state equations

Conclusion
Take Home Messages

- Three main levels of models: submicro, micro and macro
- Microscopic models use exact discrete quantities, macroscopic mean-field models use average quantities in terms of unit numbers
- Multi-level modeling allows for different approximations, accuracy/computation trade-offs
- If carefully designed, models allow also for system optimization and closing the loop between analysis and synthesis
- Methodological framework tested on multiple case studies (additional examples and open problems discussed next week)
Additional Literature – Week 8

Papers


