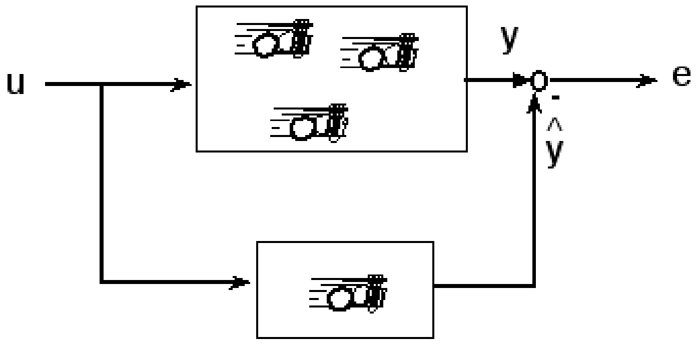
Distributed Intelligent Systems – W7

Multi-Level Modeling Methods for Swarm Robotic Systems

Outline

- Multi-level modeling framework
  - Motivation and rationale
  - Modeling assumptions
  - Methodology



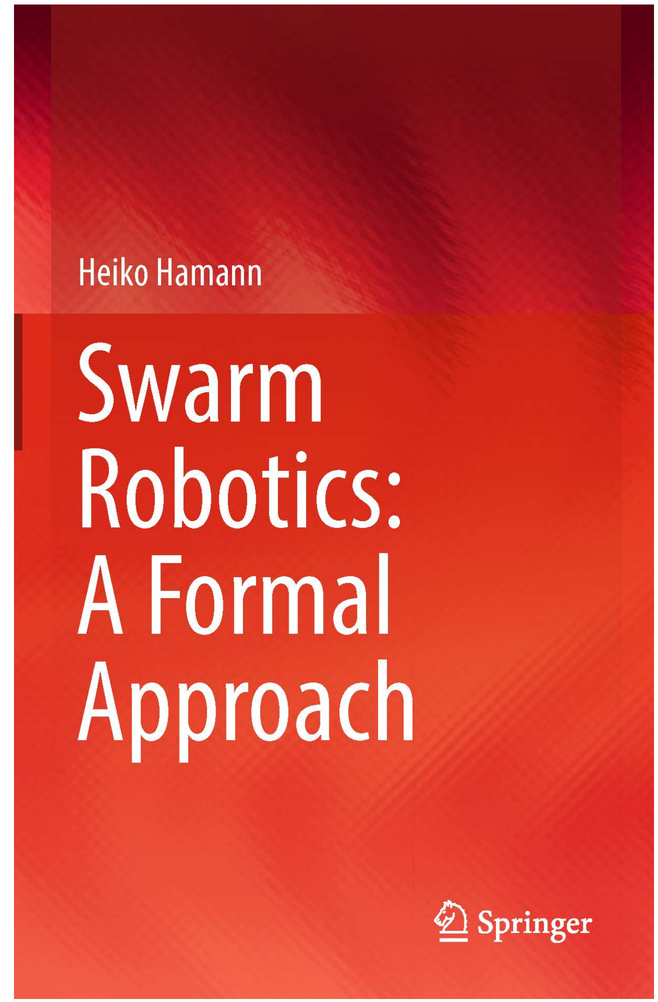
- A simple linear example
- Calibration methods for multi-level models
  - Microscopic and macroscopic parameters
  - Approximations

Modeling Rationale, Choices, and Framework Overview

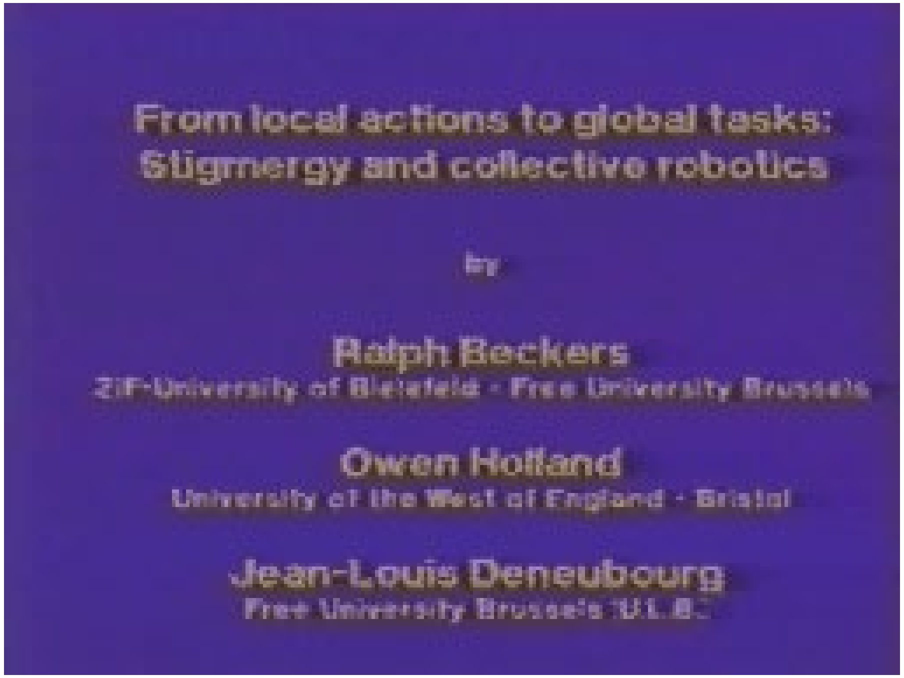
Motivation for Modeling

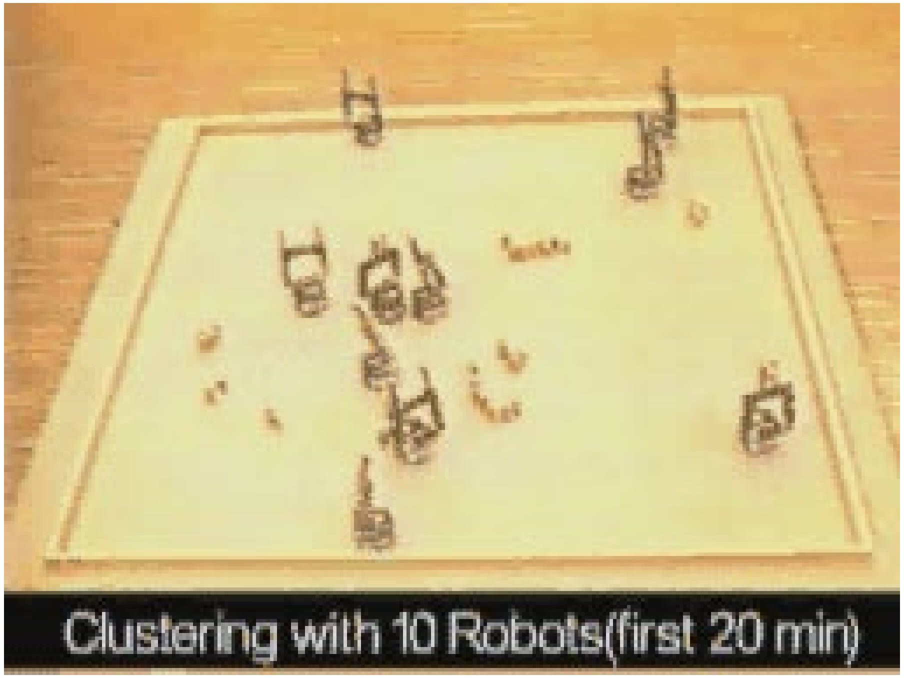
"Modeling of systems has usually three objectives: abstraction, simplification, and formalization." (p. 96)

"The main objective of modeling in swarm robotics is dimension reduction." (p. 97)



Motivating Examples - Manipulation





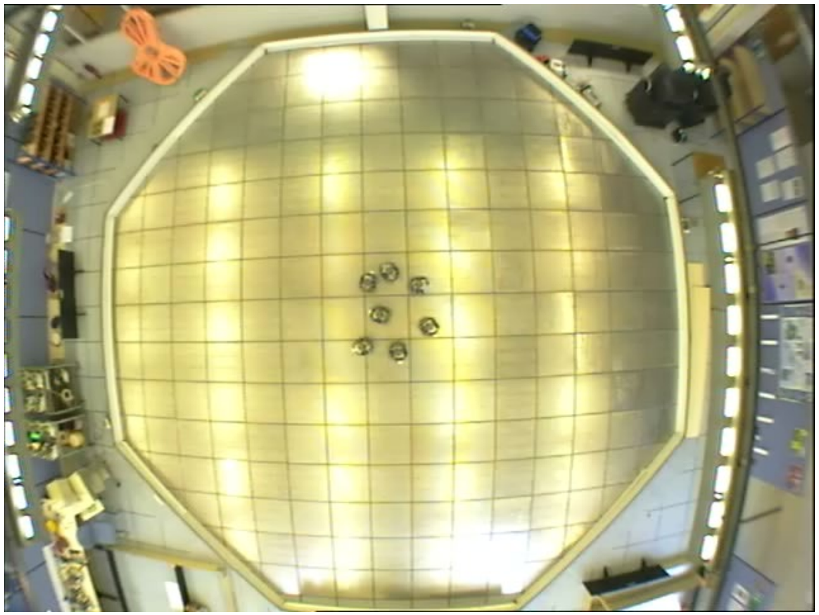
Puck Aggregation
(5 robots – 25 cm)

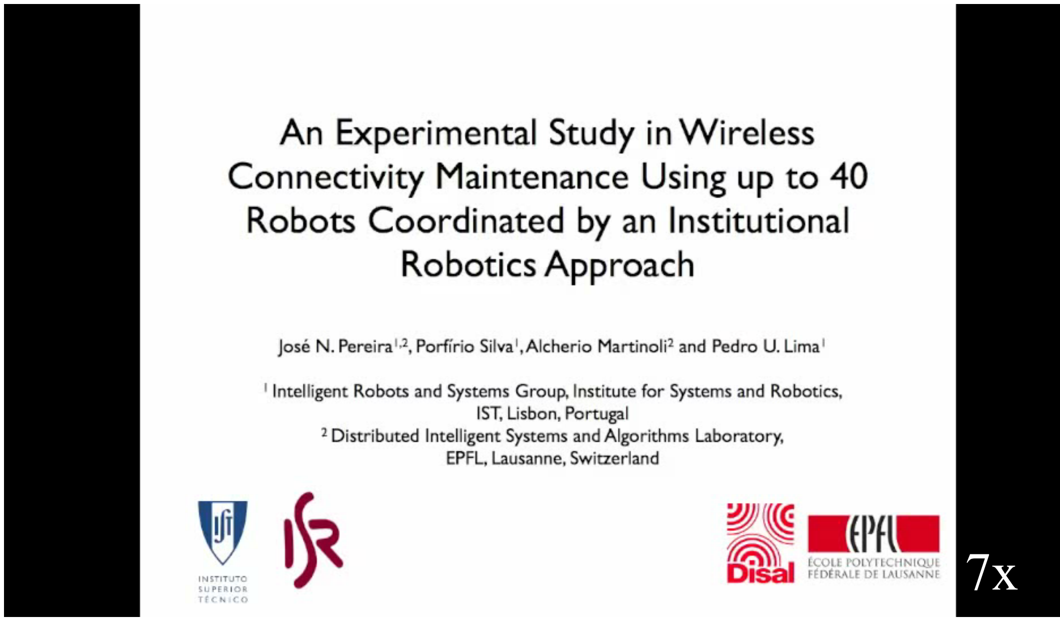
Seed-Chain Assembling
(10 Khepera I – 5.5 cm)

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

[Martinoli et al., *Robotic and Autonomous Systems,* 1999]
[Agassounon et al., *Autonomous Robots,* 2004]

Motivating Examples - Sensing





Wireless-Based Swarming
(7 Linuxbots, 24 cm)

Wireless-Based Swarming
(40 e-pucks, 7 cm)

[Nembrini et al., SAB 2002]
[Winfield et al., *Swarm Intelligence,* 2008]

[Pereira et al., IROS 2013]

Another Motivation - Solving a Key Inverse Problem

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

Motivation for Modeling

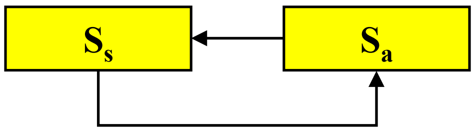
- Understanding the interplay of the various elements of the system (e.g., robot features, robot numbers, environment, noise level) ✓
- Formally analyzing system properties ✓
- Having additional tools for designing and optimizing the swarm robotic system +
- Delivering performance predictions for the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

EPFL

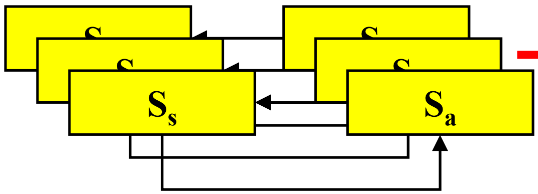
Modeling Choices

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

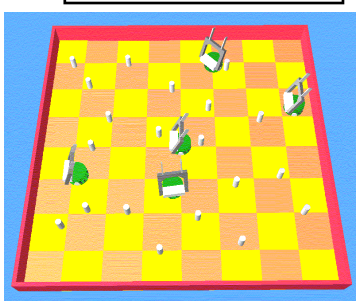
Multi-Level Modeling Methodology



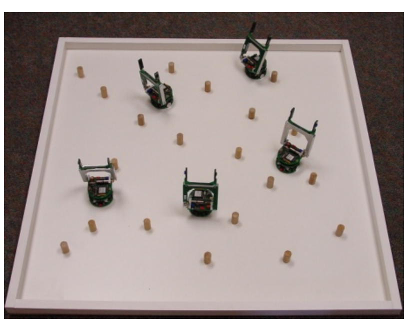
Macroscopic: representation of the whole swarm (typically a mathematical model)



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

Experimental time

Abstraction

Common metrics

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)

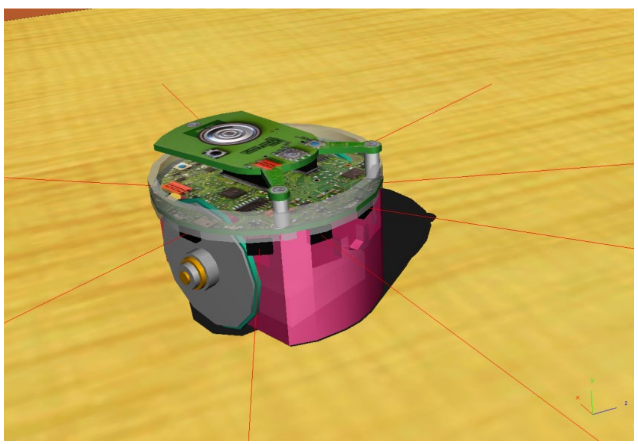
Modeling Assumptions

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)

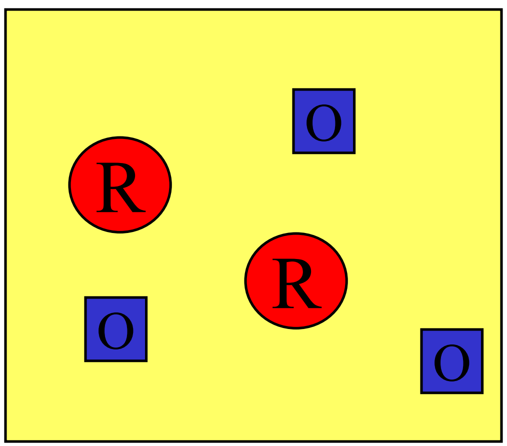


Submicroscopic
(pose, S&A state, etc.)

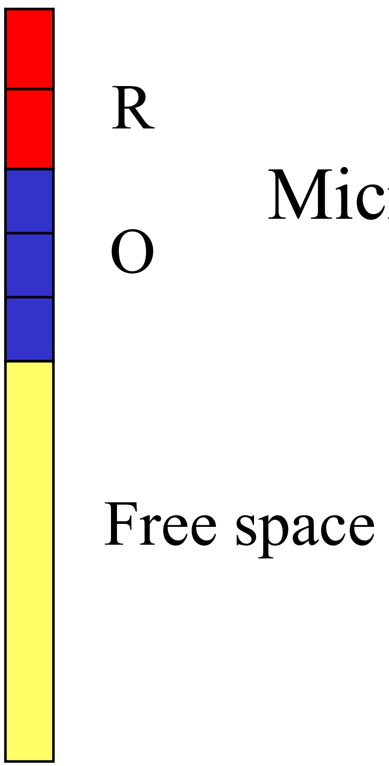
Microscopic/Macroscopic
(transition probabilities, state duration)

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

Modeling Framework

Microscopic Level

p(n,t) = probability of an agent to be in the state *n* at time *t*

If Markov properties fulfilled:

$$\Delta p(n,t) = p(n,t+\Delta t) - p(n,t)$$

$$= \underbrace{\sum_{n'} p(n,t+\Delta t \mid n',t)\, p(n',t)}_{\text{inflow}} - \underbrace{\sum_{n'} p(n',t+\Delta t \mid n,t)\, p(n,t)}_{\text{outflow}}$$

Transition probability

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

Probability the agent was in a given state n'

Macroscopic Level – Time-Continuous

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

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

Rate Equation
(time-continuous)

n, n' = states of the agents (all possible states at each instant)
N_n = average fraction (or mean number) of agents in state n 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):

$$N_n((k+1)T) = N_n(kT) + \sum_{n'} TW(n \mid n', kT) N_{n'}(kT) - \sum_{n'} TW(n' \mid n, kT) N_n(kT)$$

inflow

outflow

k = iteration index

T = time step, sampling interval

TW = transition probability per time step

Notation often simplified to:

$$N_n(k+1) = N_n(k) + \sum_{n'} P(n \mid n', k) N_{n'}(k) - \sum_{n'} P(n' \mid n, k) N_n(k)$$

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:

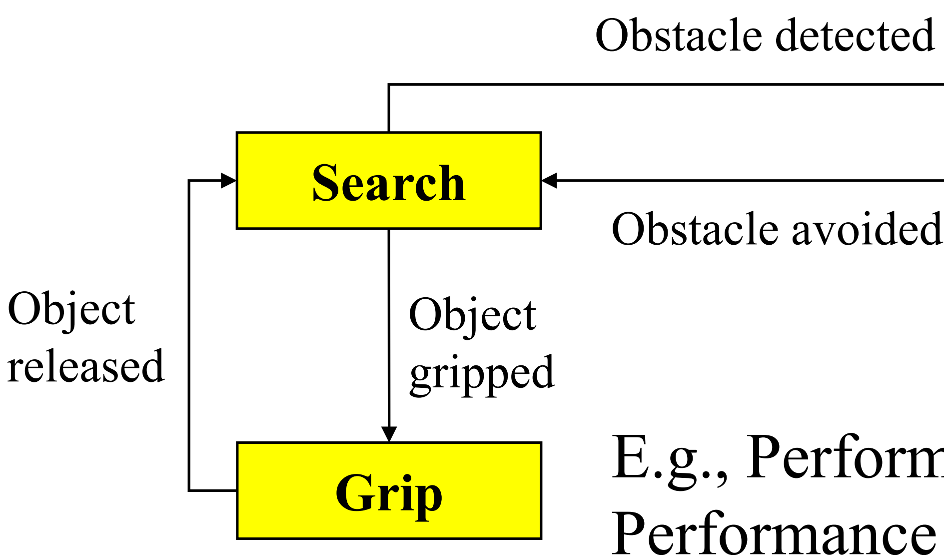
1. Assess what's the time resolution needed for your system performance metrics (if time step chosen appropriately small, no impact on prediction accuracy in the type of experiments presented)

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

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

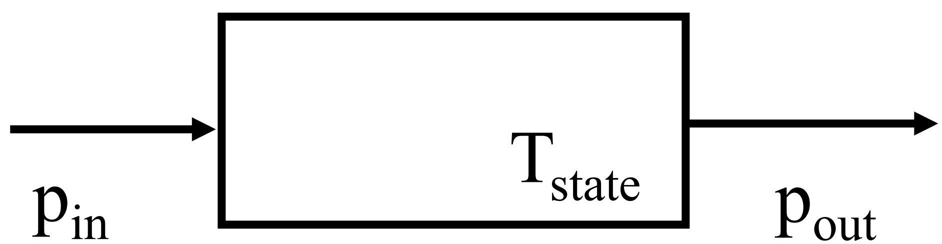


E.g., Performance = f(#robots$_{search}$,#robots$_{grip}$)

Performance = f(#robots$_{search}$,#robots$_{avoidance}$)

Model Parameters

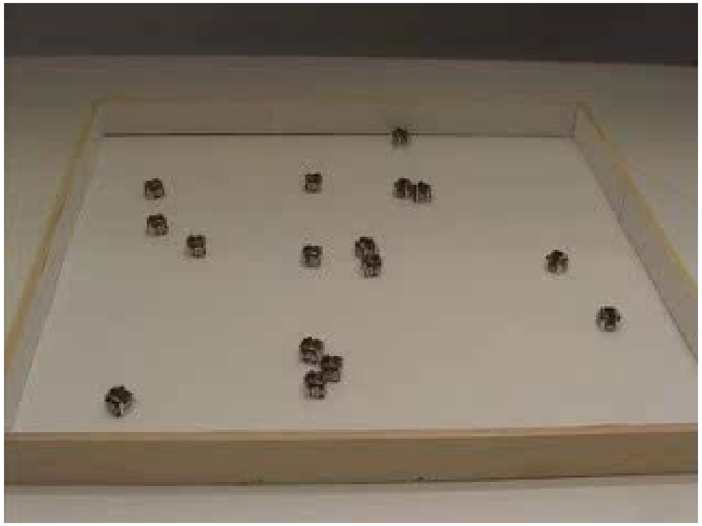
- Number of parameters: decreasing with increasing abstraction
- Incremental calibration: a given level can be calibrated on the underlying one using aggregation techniques (e.g., first moment for a distribution at lower abstraction level)
- Explicit representation: any system parameter of interest should 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

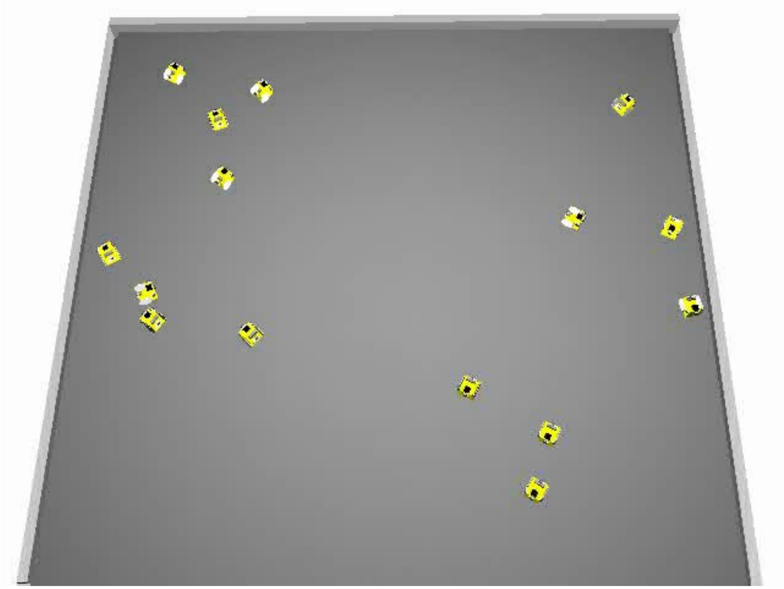


Linear Example: Obstacle Avoidance

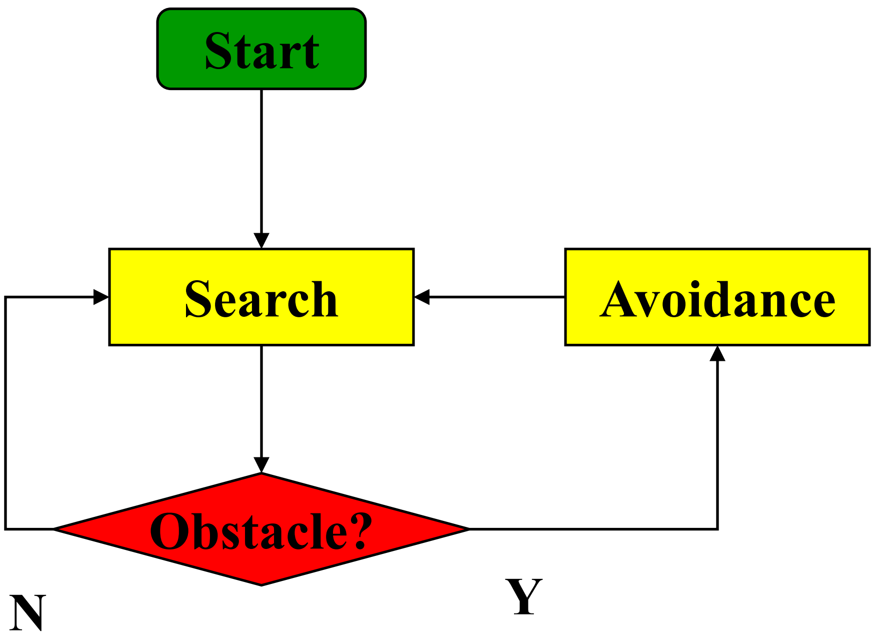
A Simple Linear Model

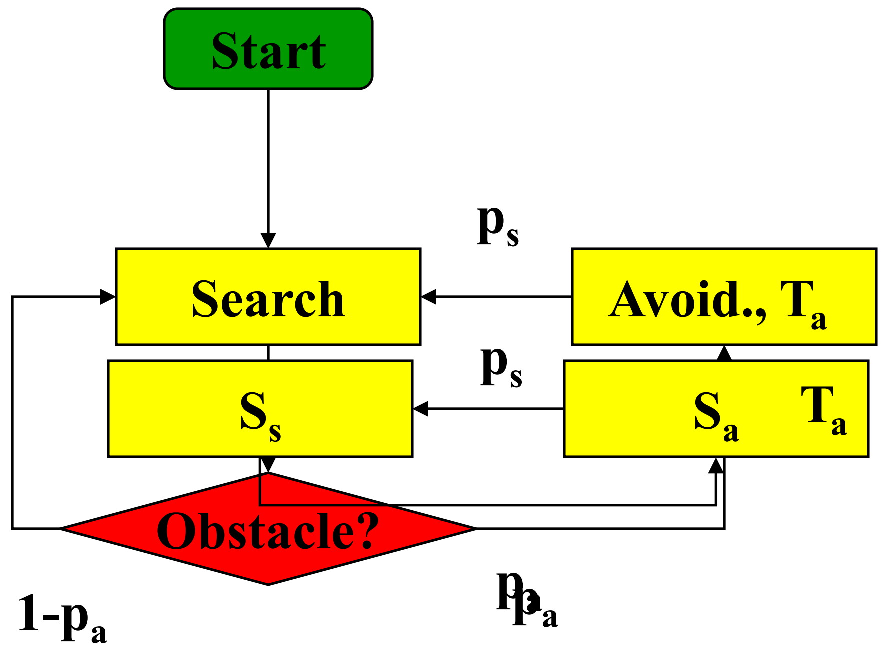
Example: search (moving forwards) and obstacle avoidance





A Simple Example





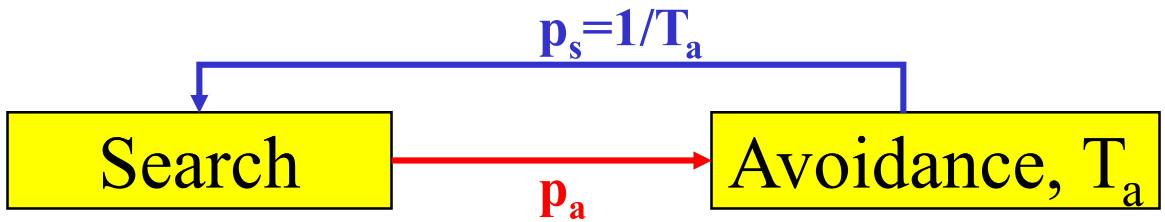
Nonspatiality & microscopic characterization

PFSM

Deterministic robot's flowchart

Probabilistic agent's flowchart

Linear Model – Probabilistic Delay



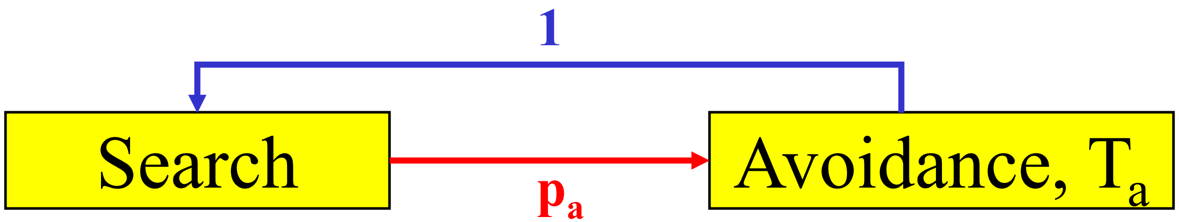
$$N_s(k+1) = N_s(k) - {\color{red}p_a N_s(k)} + {\color{blue}p_s N_a(k)}$$

$$N_a(k+1) = N_0 - N_s(k+1)$$

$$\boxed{N_s(0) = N_0 \ ; \ N_a(0) = 0}$$

T_a = mean obstacle avoidance duration
p_a = probability of moving to obstacle av.
p_s = probability of resuming search
N_s = average # robots in search
N_a= average # robots in obstacle avoidance
N_0 = # robots used in the experiment
k = 0,1, … (iteration index)

Linear Model – Deterministic Delay



$$N_s(k+1) = N_s(k) - p_a N_s(k) + p_a N_s(k-T_a)$$

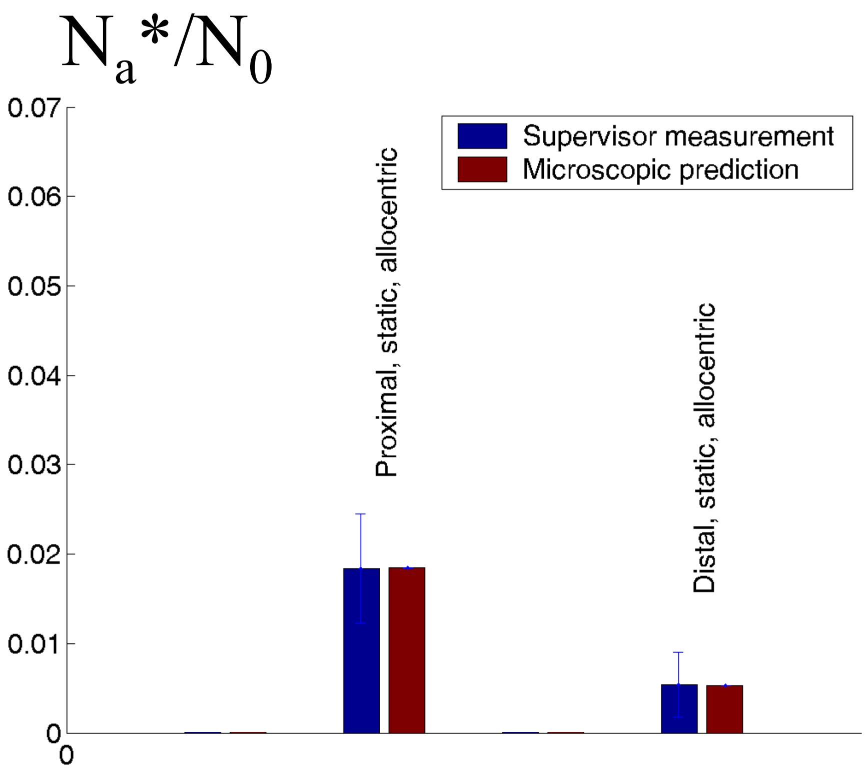
$$N_a(k+1) = N_0 - N_s(k+1)$$

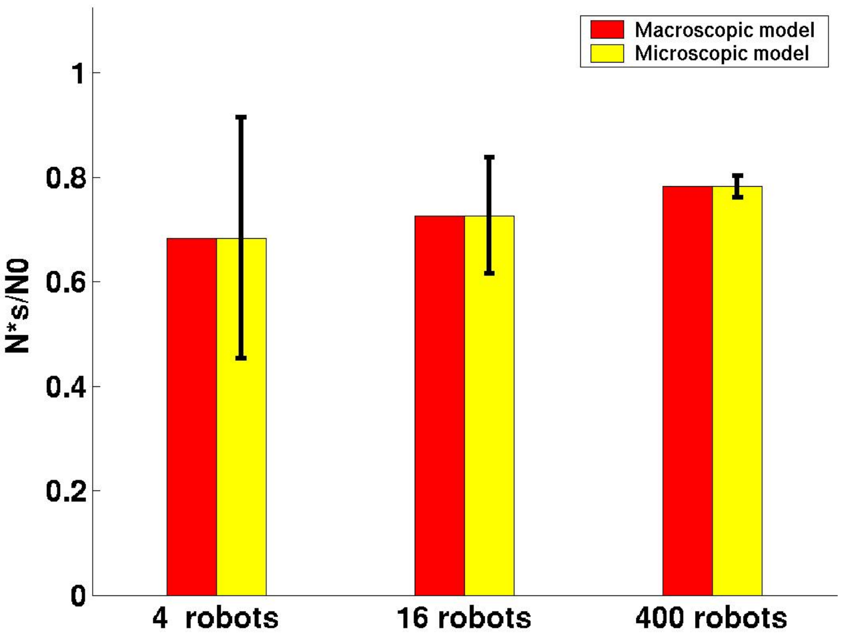
! $N_s(k) = N_a(k) = 0$ for all $k<0$!

$N_s(0) = N_0$; $N_a(0) = 0$

T_a = mean obstacle avoidance duration
p_a = probability moving to obstacle avoidance
N_s = average # robots in search
N_a = average # robots in obstacle avoidance
N_0 = # robots used in the experiment
k = 0,1, … (iteration index)

Linear Model – Sample Results



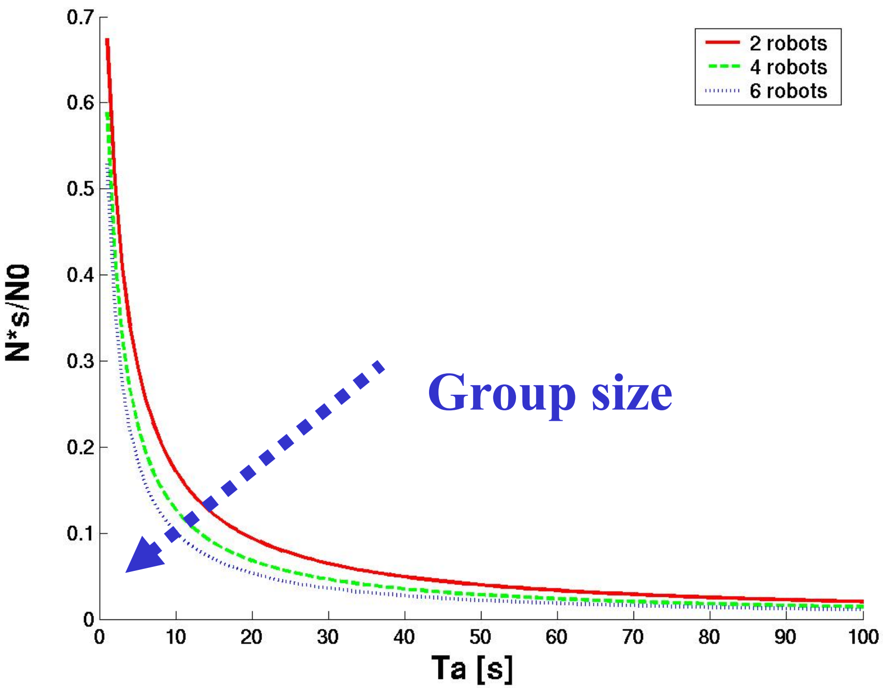


Submicro to micro comparison
(different controllers, steady state comparison)

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 $\rightarrow N_n^*$
- Note 1: equivalent to differential equation of $dN_n/dt = 0$
- 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):



$$N_s^* = \frac{N_0}{1 + p_a T_a} \qquad N_a^* = \frac{N_0 p_a T_a}{1 + p_a T_a}$$

Ex.: normalized mean number of robots in search mode at steady state as a function of time for obstacle avoidance

Model Calibration

State Durations & Discretization Interval

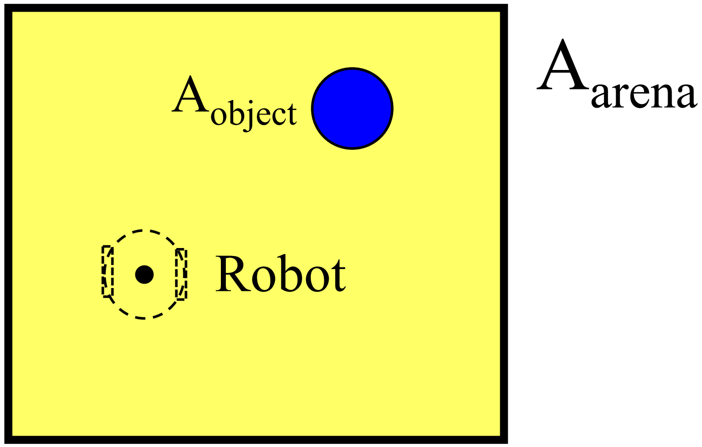
1. Measure all interaction times of interest in your system, i.e. those which might influence the system performance metrics.
   Note: often "delay states" can just summarize all what you need without getting into the details of what's going on within the state.

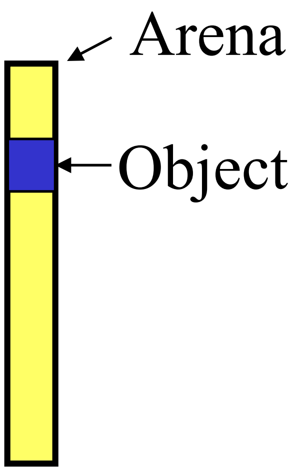
2. Consider only average values (we might consider also parametrized distributions in the future, the modeling methodology does not prevent to do so)

3. For time-discrete systems: choose the **time step T** = 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

- Assumptions:
  - non spatial metrics
  - well-mixed system
  - finite enclosed arena of area A_{arena}
  - single object of area A_{object}
- Non-spatial model: bodiless robot randomly hopping around
- Idea: probability encountering object $\propto A_{object} / A_{arena}$





1D probability space

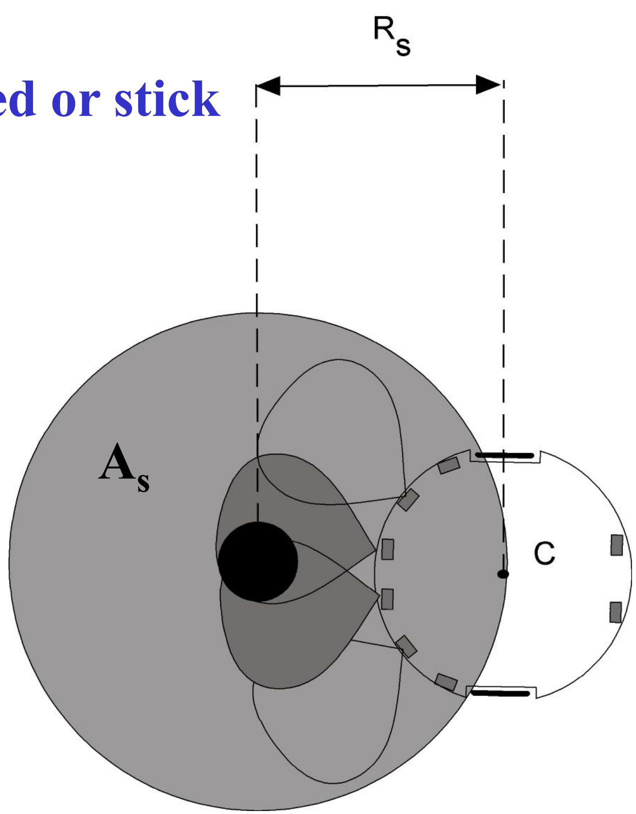
Geometric Probabilities g_i

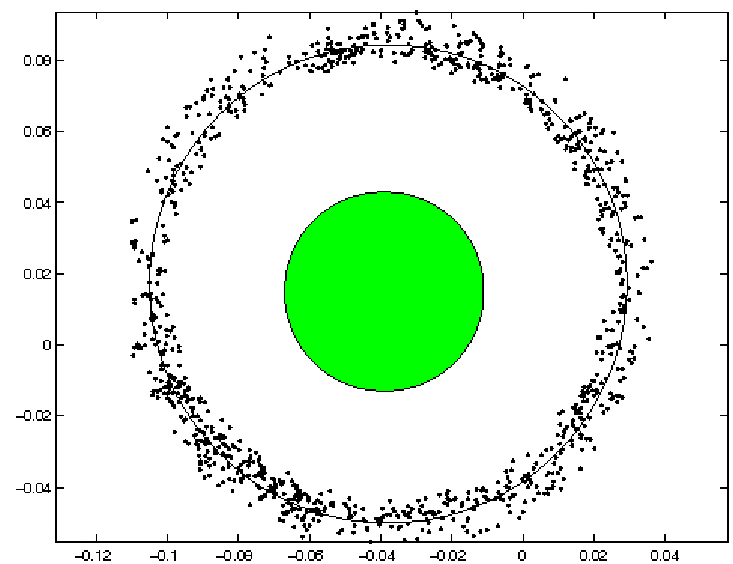
[Correll & Martinoli, ISER 2004]

- 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: seed or stick

$g_s = A_s / A_{arena}$





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** :

$$r_i = \frac{vW_s}{A_s} g_i$$

A_s = detection area of the smallest object

v = mean robot speed

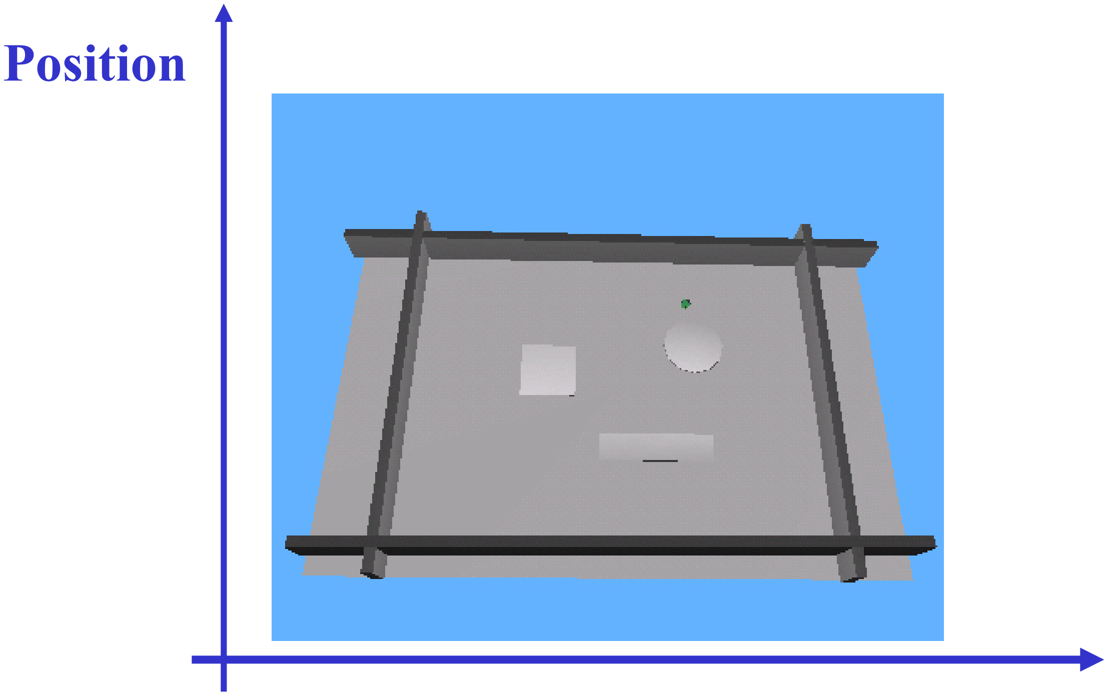
W_s = robot's detection width for the smallest object (center-to-center)

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) !

Experimental Validation of Spatiality Assumptions



Nonembodied obstacles = detection surfaces

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

	Square	Rect.	Round	All shapes	Geometry
Normalized detection surface	0.31 ± 0.04	0.3 ± 0.03	0.32 ± 0.02	0.31 ± 0.03	0.31

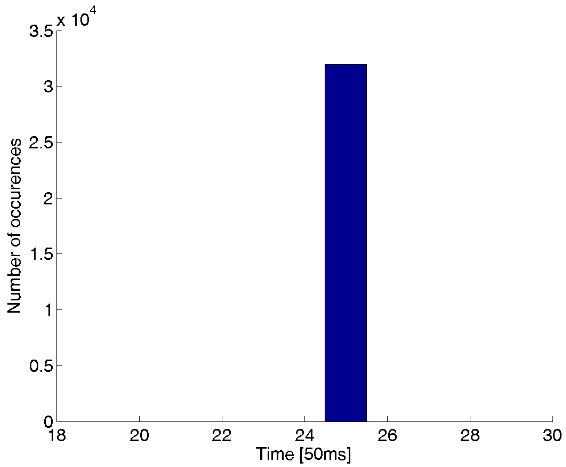
Model Calibration - Practice

- 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)

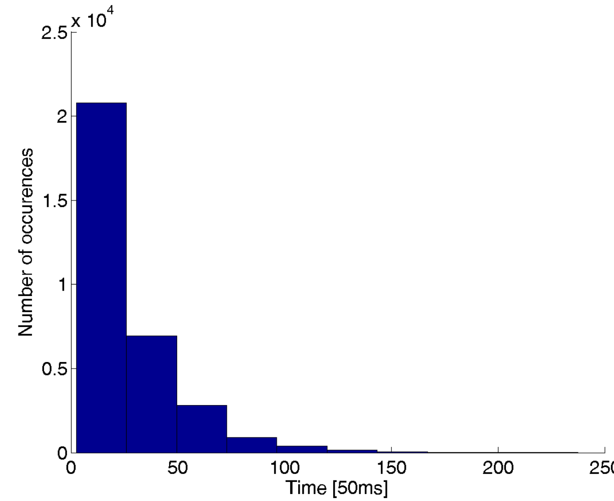
Model Calibration - Practice

Bin distribution of interaction time T_a (mean T_a= 25 *50 ms = 1.25 s)

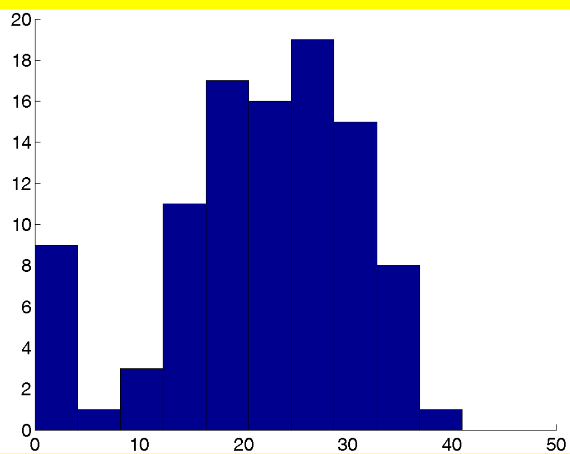
# of collisions



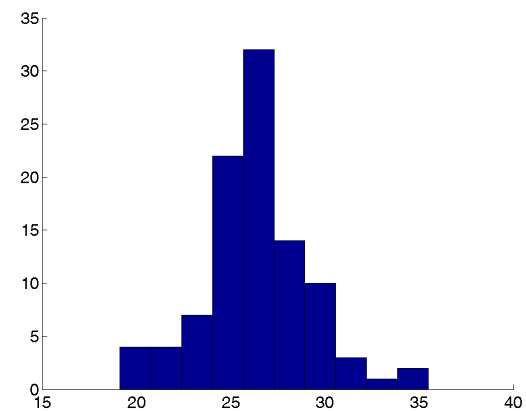
Micro/macro, deterministic delay



Micro/macro, prob. delay



Sub-microscopic, distal controller

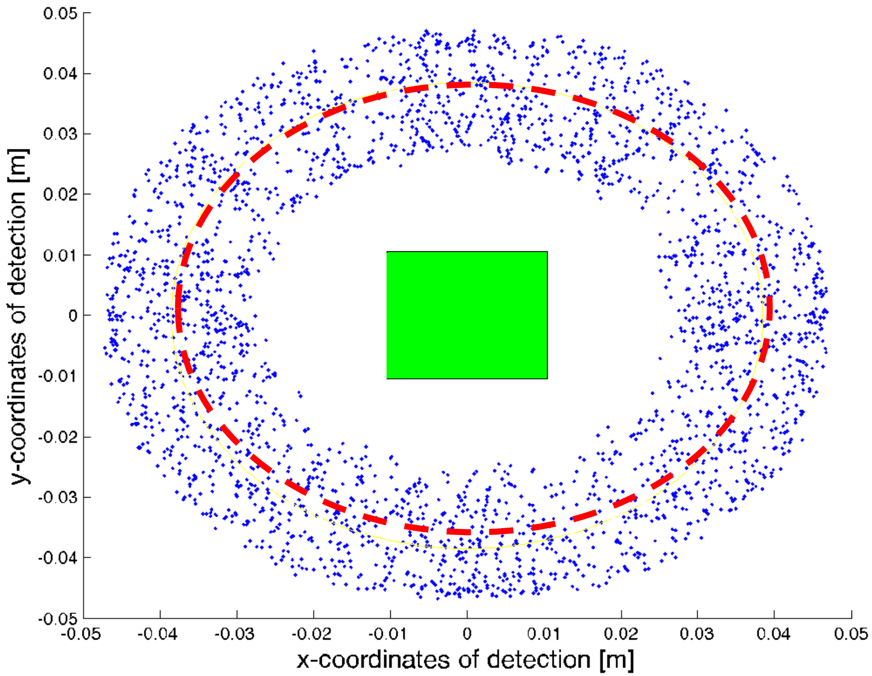


Submicroscopic, proximal contr.

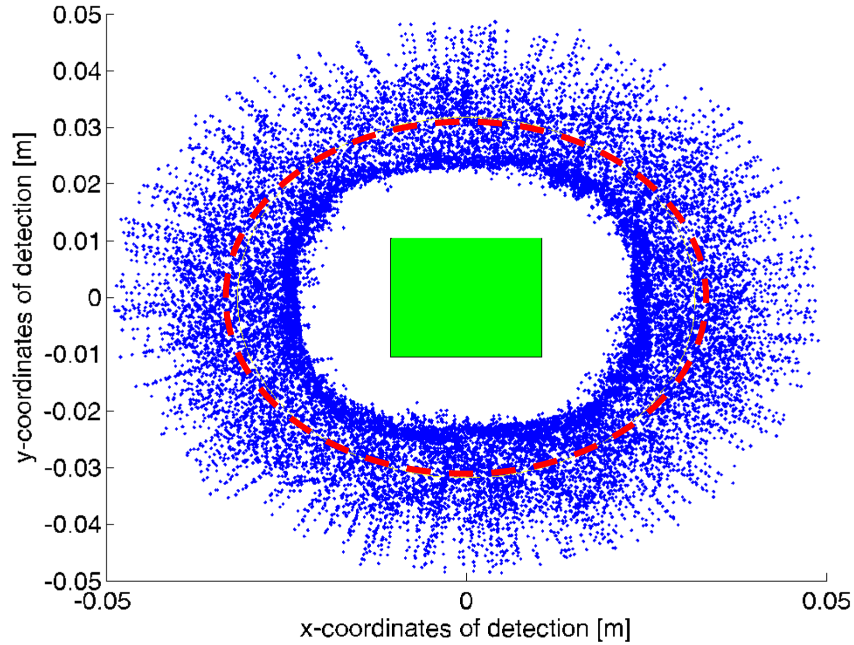
Collision time

Model Calibration - Practice

Geometric probability g: example of transition in space from search to obstacle avoidance (1 moving robot, 1 dummy robot, Webots measurements, egocentric)



Distal controller (rule-based)



Proximal controller (Braitenberg, linear)

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
- Models' parameter calibration is difficult and still an open challenge
- Methodological framework tested on multiple case studies (additional examples discussed next week)

EPFL

Additional Literature – Week 7

DISAL

Papers

- Prorok A., Correll N., and Martinoli A., “Multi-level Spatial Modeling for Stochastic Distributed Robotic Systems”. *Int. Journal of Robotics Research,* **30**(5): 574-589, 2011.
- Di Mario E., Mermoud G., Mastrangeli M., and Martinoli A. “A Trajectory-based Calibration Method for Stochastic Motion Models”. *Proc. of the 2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, September 2011, San Francisco, U.S.A., pp. 4341-4347.
- Roduit P., Martinoli A., and Jacot J., “A Quantitative Method for Comparing Trajectories of Mobile Robots Using Point Distribution Models”. *Proc. of the 2007 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems,* October-November 2007, San Diego, USA, pp. 2441-2448.
- Ijspeert A. J., Martinoli A., Billard A., and Gambardella L.M., “Collaboration through the Exploitation of Local Interactions in Autonomous Collective Robotics: The Stick Pulling Experiment”. *Autonomous Robots*, **11**(2):149–171, 2001.
- Lerman, K. and Galstyan, A. “Mathematical model of foraging in a group of robots: Effect of interference”. *Autonomous Robots*, **13**(2):127–141, 2002.
- S. Berman, A. Halasz, M. A.Hsieh, and V. Kumar. “Optimal Stochastic Policies for Task Allocation in Swarms of Robots”, *Trans. on Robotics*, **25**(4): 927–937, 2009.
- M. A. Hsieh, A. Halasz, S. Berman, and V. Kumar. "Biologically Inspired Redistribution of a Swarm of Robots Among Multiple Sites”. *Swarm Intelligence*, **2** (2-4): 121–141, 2008.
- T. W. Mather and M. A. Hsieh. "Analysis of Stochastic Deployment Policies with Time Delays for Robot Ensembles”. *Int. Journal of Robotics Research, ,* **30**(5): 590–600, 2011