ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

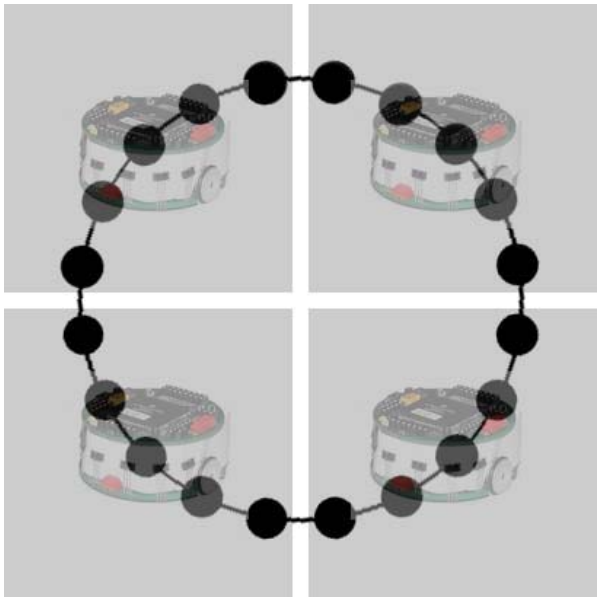
Disal

Distributed Intelligent Systems – W11

Machine-Learning Methods Applied to Distributed Robotic Systems

Outline

- Revisiting expensive optimization problems
  - Additional experimental evidence
  - Noise-resistant algorithms in single robot scenarios
- Challenges in multi-robot scenarios
  - Credit assignment problems
  - Co-adaptation strategies
  - Noise-resistance
- Co-adaptation examples in multi-robot obstacle avoidance



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Expensive Optimization and Noise Resistance

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Expensive Optimization Problems

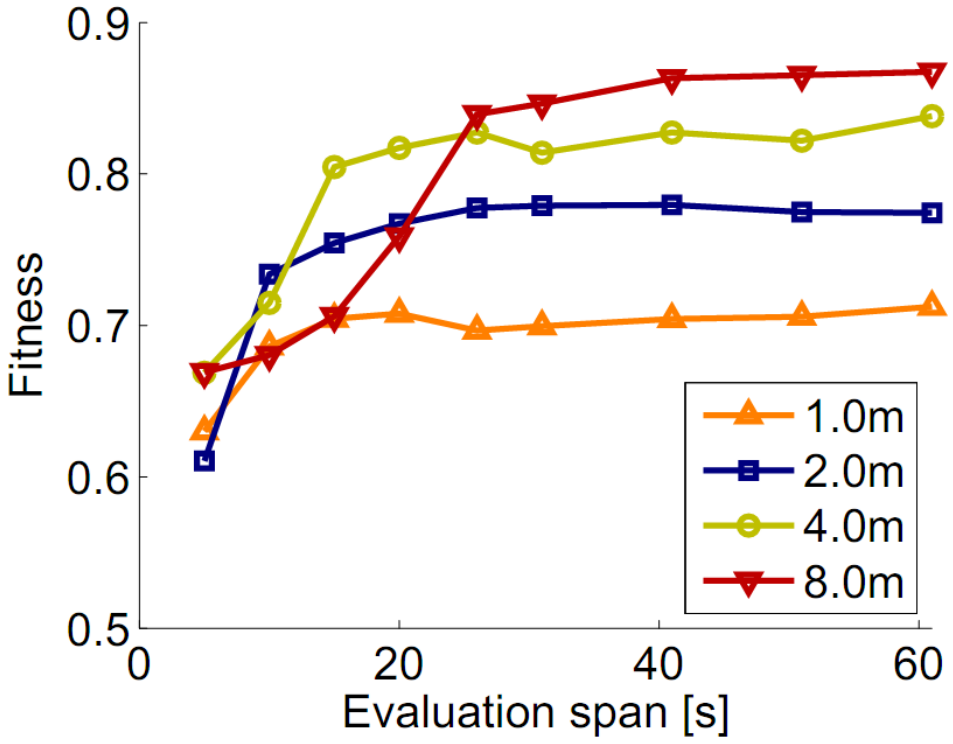
Two fundamental reasons making robot control design and optimization expensive in terms of time:

1. Time for evaluation of candidate solutions (e.g., tens of seconds) >> time for application of metaheuristic operators (e.g., milliseconds)
2. Noisy performance evaluations disrupt the adaptation process and require multiple evaluations for actual performance

Expensive Optimization Problems

1. Time for evaluation of candidate >> time for application of metaheuristic operators

- Example: obstacle avoidance, usual fitness function
- A single robot need to encounter obstacles to learn to avoid them
- Evaluation span 20-60 s depending on size of the arena
- Current processors can execute several billions of instructions in that time (e.g. ARM Cortex-A9 ~5000 MIPS)

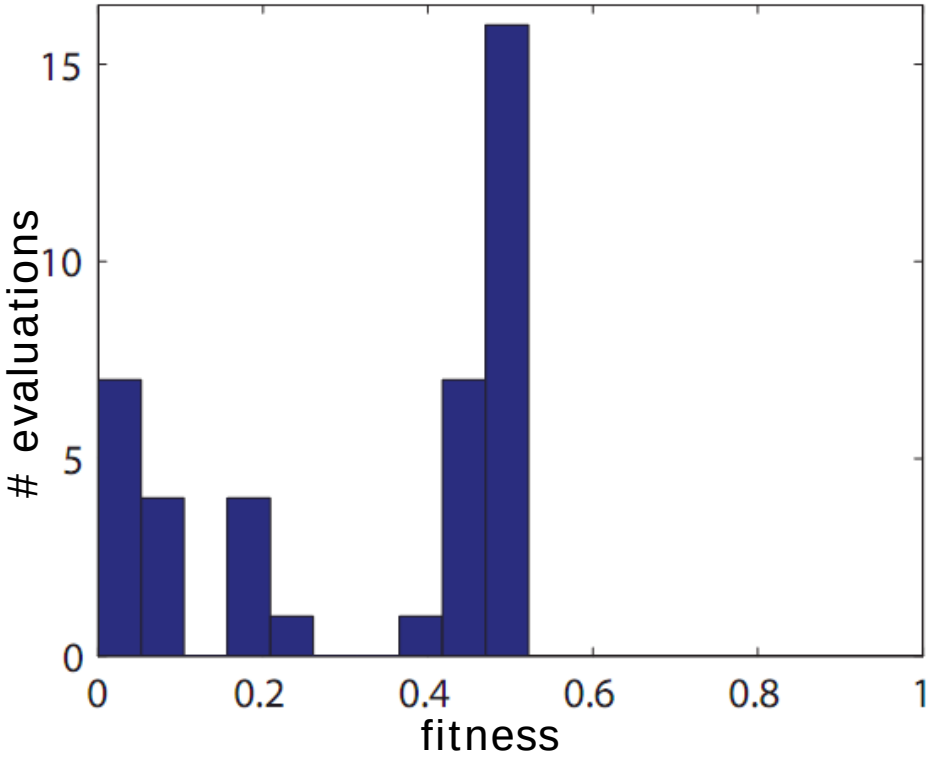


[Di Mario and Martinoli, *Robotica*, 2014]

Expensive Optimization Problems

2. Noisy performance evaluations disrupt the adaptation process and require multiple evaluations for actual performance

- Multiple evaluations at the same point in the search space yield different results
- Example: fitness distribution for obstacle avoidance
- Noise from: sensors, actuators, initial conditions, other robots
- Noise causes decreased convergence speed and residual error

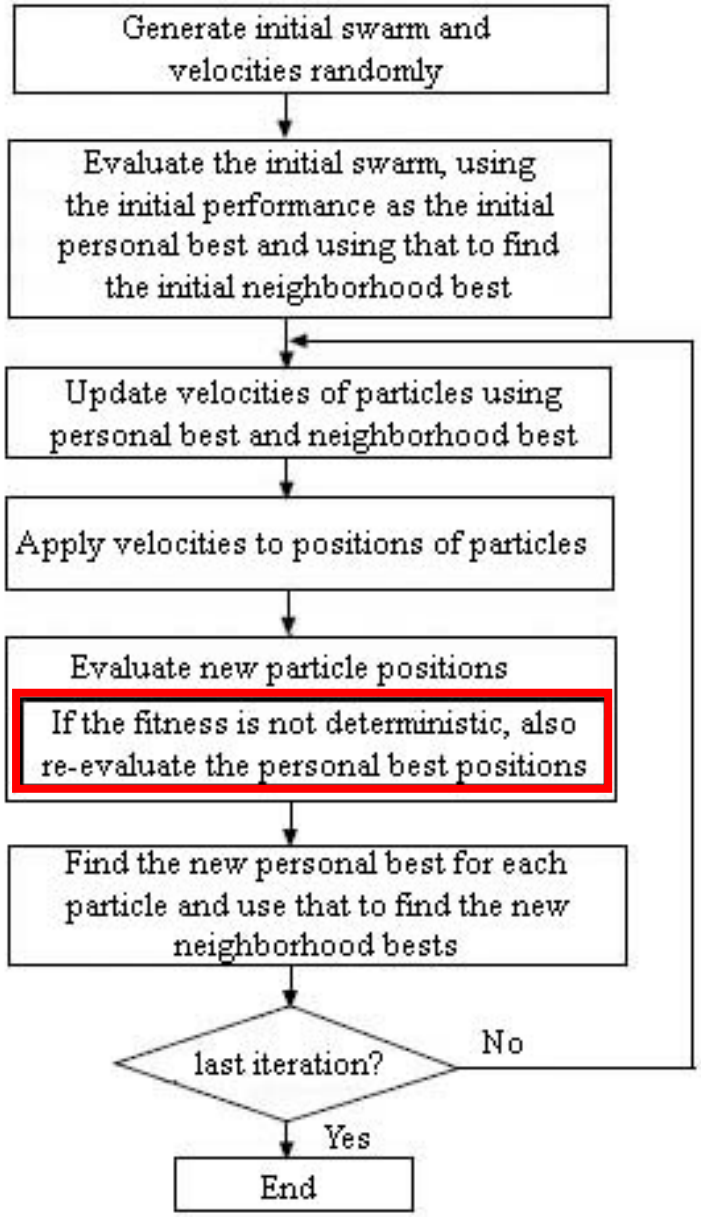


[Di Mario and Martinoli, *Robotica*, 2014]

Pbest-based Noise-Resistant PSO

Remarks:

- Better assessment of actual performance of a candidate solution through re-evaluation and aggregation of pbest performances over iterations
- Evaluations per particle and per iteration:
  - Noise-resistant: 2
  - Regular: 1
- Fair comparison with regular PSO using the same total number of evaluations



Testing Noise-Resistant on Benchmarks

- Benchmark 1 : Sphere and Generalized Rosenbrock functions
  - 30 real parameters [Pugh et al., SIS 2005], W10 (biased results)
  - 24 real parameters [Di Mario et al., CEC 2014] today
  - Minimize objective function
  - Expensive only because of noise

- Benchmark 2: obstacle avoidance on a robot
  - 24 real parameters
  - Maximize objective function
  - Expensive because of noise and evaluation time

Benchmark 1: Functions

[Di Mario et al., CEC 2014]

- Sphere

$$f_1(\mathbf{x}) = \sum_{i=1}^{D} x_i^2$$

- Rosenbrock

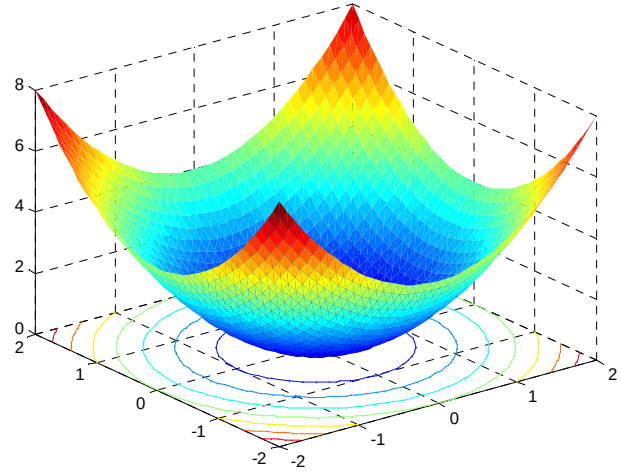
$$f_2(\mathbf{x}) = \sum_{i=1}^{D-1} [(1 - x_i^2) + 100(x_{i+1} - x_i^2)^2]$$

- Normalized and bounded to [0, 1]
- Gaussian noise model

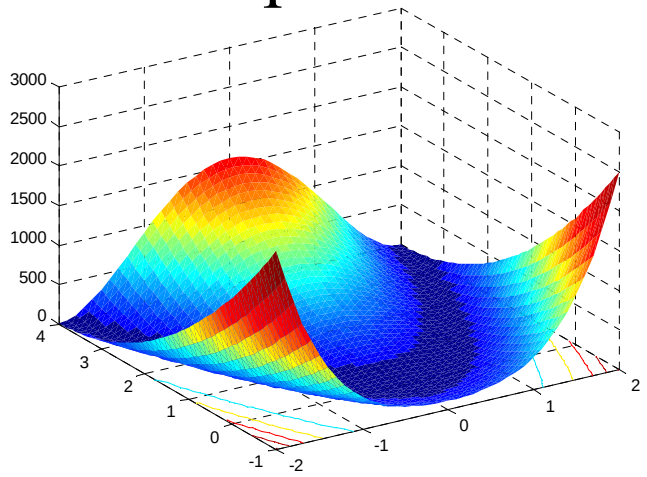
$$f_i^g(\mathbf{x}) = \frac{f_i(\mathbf{x})}{\max f_i} + \mathcal{N}(0, \sigma)$$

- Bernoulli noise model

$$f_i^b(\mathbf{x}) = \frac{f_i(\mathbf{x})}{\max f_i} + A \cdot \mathcal{B}(p)$$



Sphere

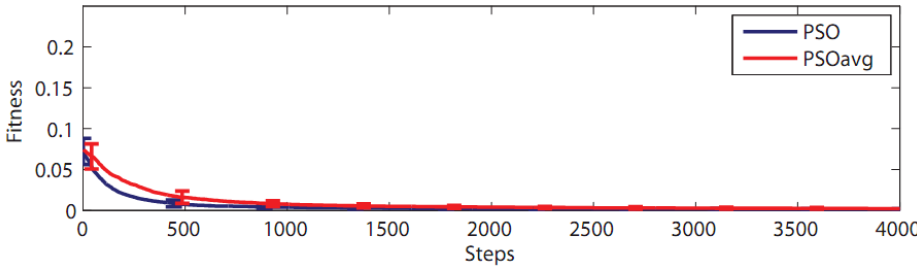


Rosenbrock

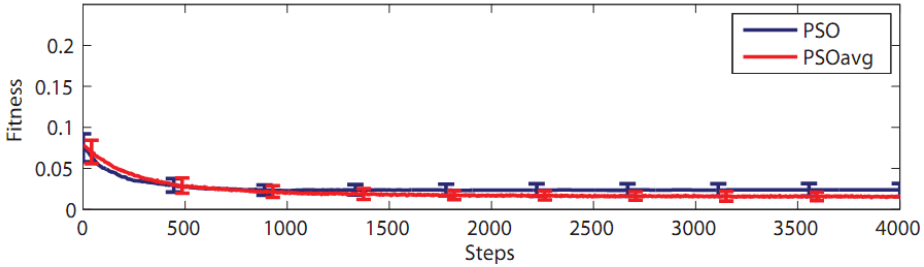
Rosenbrock with Gaussian Noise: Increasing σ

[Di Mario et al., CEC 2014]

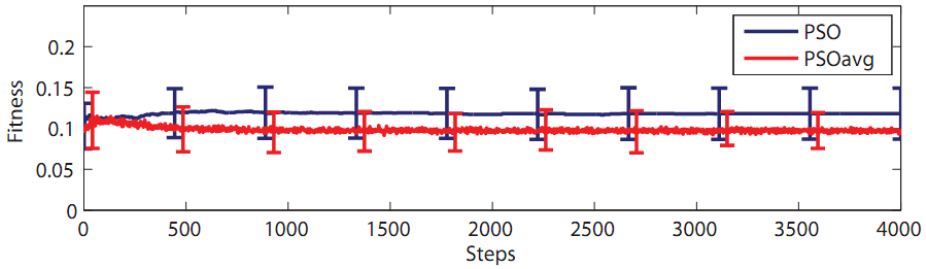
σ = 0



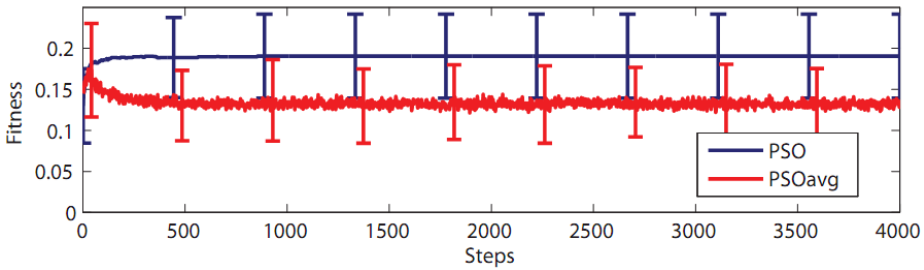
σ = 0.01



σ = 0.05

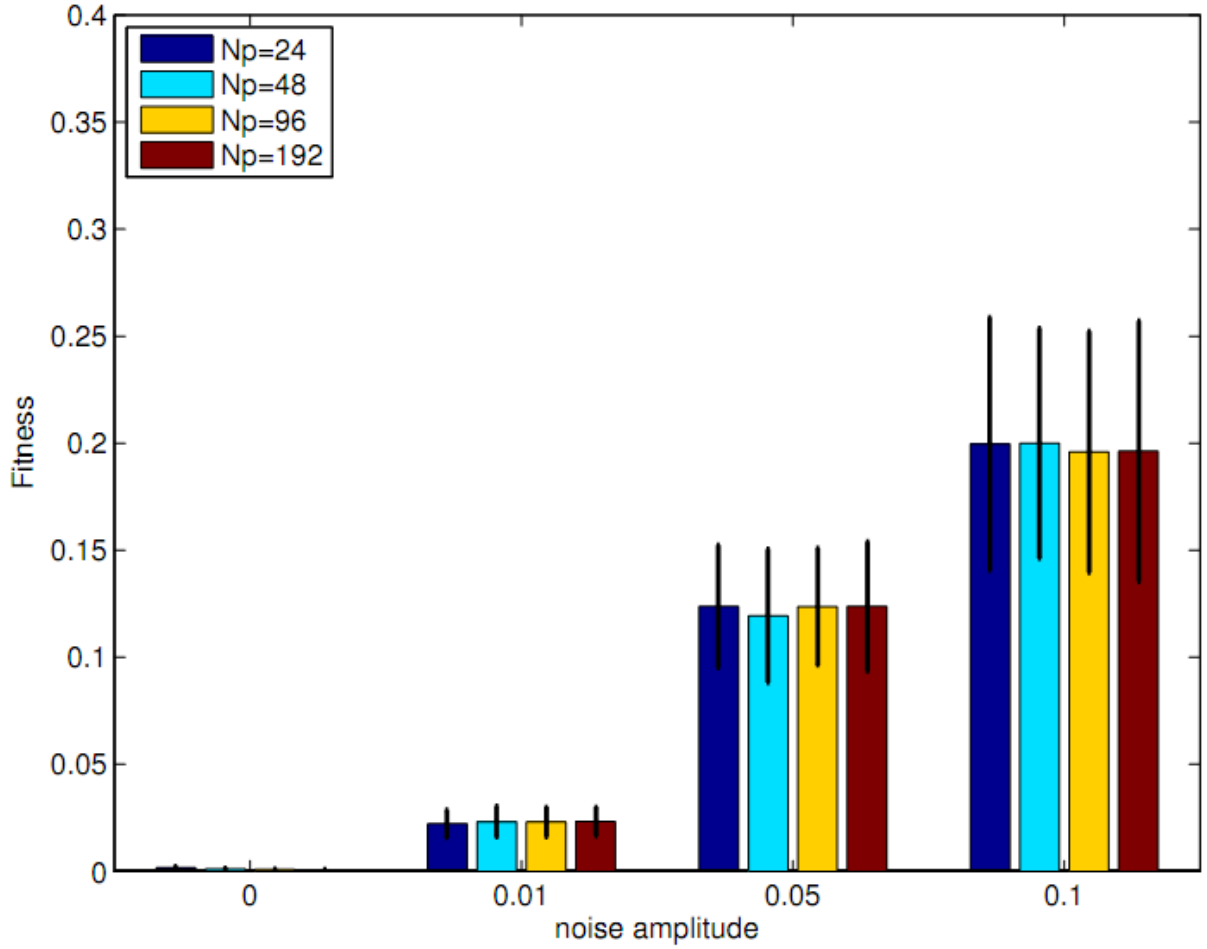


σ = 0.1



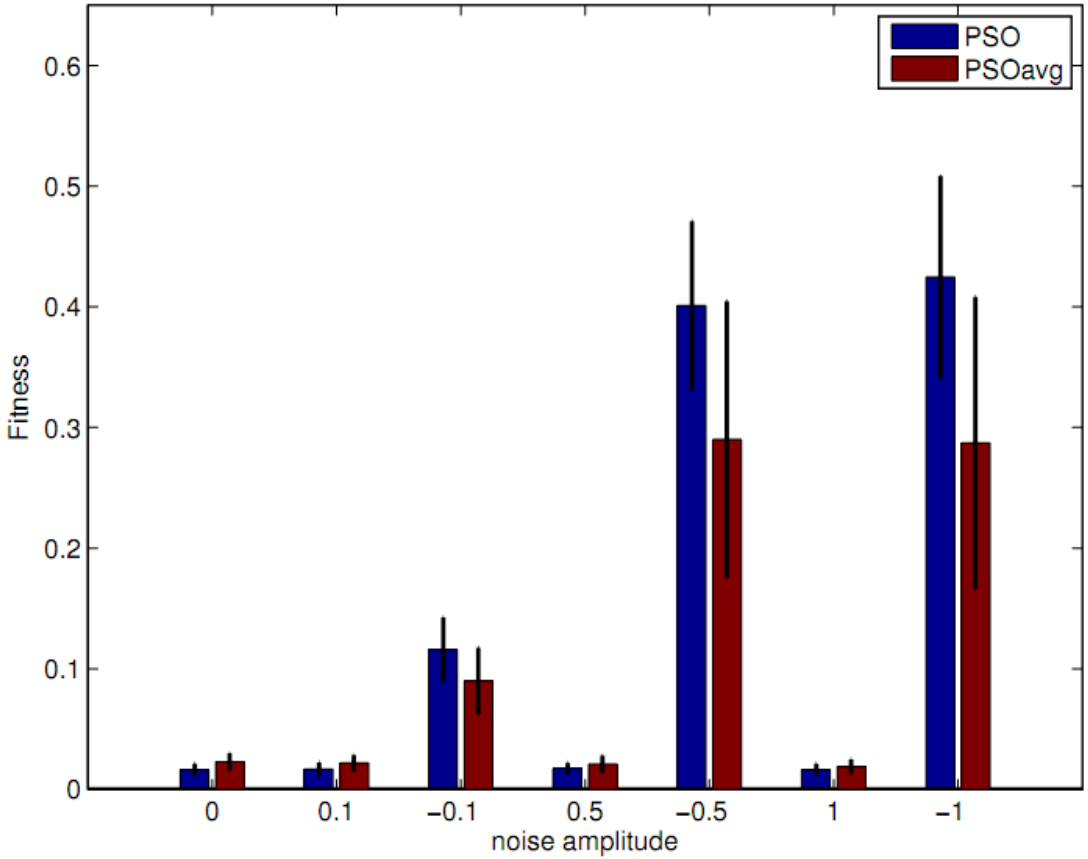
Increasing Population Size Does Not Help

[Di Mario et al., CEC 2014]



- Results in contrast with best practices in GA
- Other parameters constant (i.e. more evaluations for larger populations)

Bernoulli Noise: Positive and Negative Amplitudes



- Negative amplitudes (i.e. good spurious evaluation of bad solutions) have a more significant disruptive impact of positive amplitudes (bad spurious evaluation of good solutions)
- Noise-resistant algorithms can cope with this large negative outliers

$$f_i^b(\mathbf{x}) = \frac{f_i(\mathbf{x})}{\max f_i} + A \cdot \mathcal{B}(p)$$

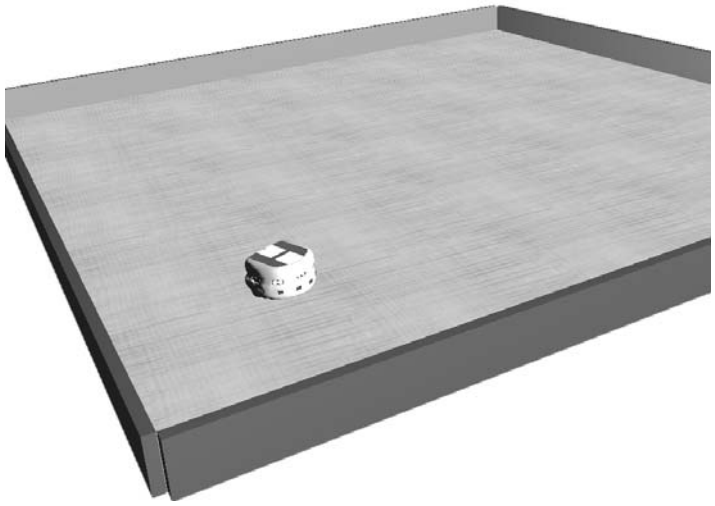
Benchmark 2: Obstacle Avoidance on a Mobile Robot

- Similar to [Floreano and Mondada 1996]
  - Discrete-time, single-layer, artificial recurrent neural network controller
  - Shaping of neural weights and biases (24 real parameters)
  - fitness function: rewards speed, straight movement, avoiding obstacles
- Different from [Floreano and Mondada, 1996]
  - Environment: bounded open-space of 2x2 m instead of a maze

$$F = V \cdot (1 - \sqrt{\Delta v}) \cdot (1 - i)$$

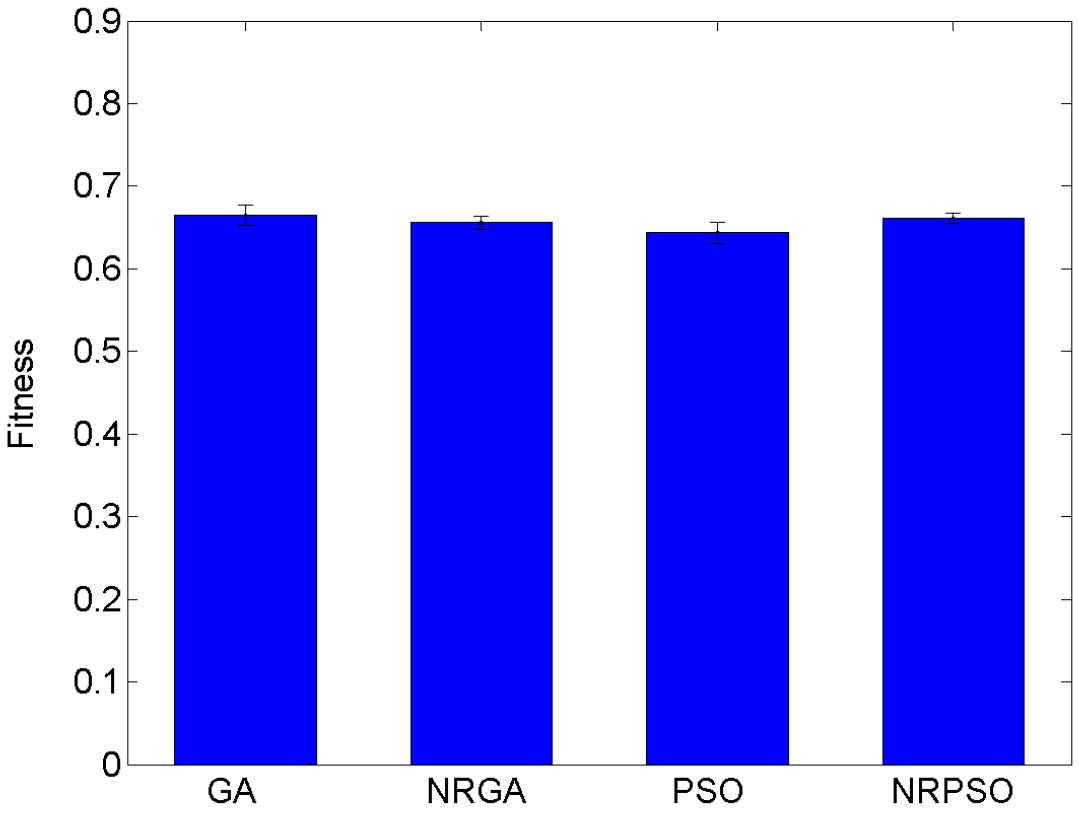
$$0 \leq V \leq 1, \quad 0 \leq \Delta v \leq 1, \quad 0 \leq i \leq 1$$

V = average wheel speed, Δv = difference between wheel speeds, i = value of most active proximity sensor



[Pugh J., EPFL PhD Thesis No. 4256, 2008]

Baseline Experiment: Extended-Time Adaptation



[Pugh J., EPFL PhD Thesis No. 4256, 2008]

- Compare the basic algorithms with their corresponding noise-resistant version
- Population size 100, 100 iterations, evaluation span 300 seconds (150 s for noise-resistant algorithms)→ 34.7 days
- Fair test: same total evaluation time for all the algorithms
- Realistic simulation (Webots)
- Best evolved/learned solutions averaged over 30 runs
- Best candidate solution in the final pool selected based on 5 runs of 30 s each; performance tested over 40 runs of 30s each
- Similar performance for all algorithms

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

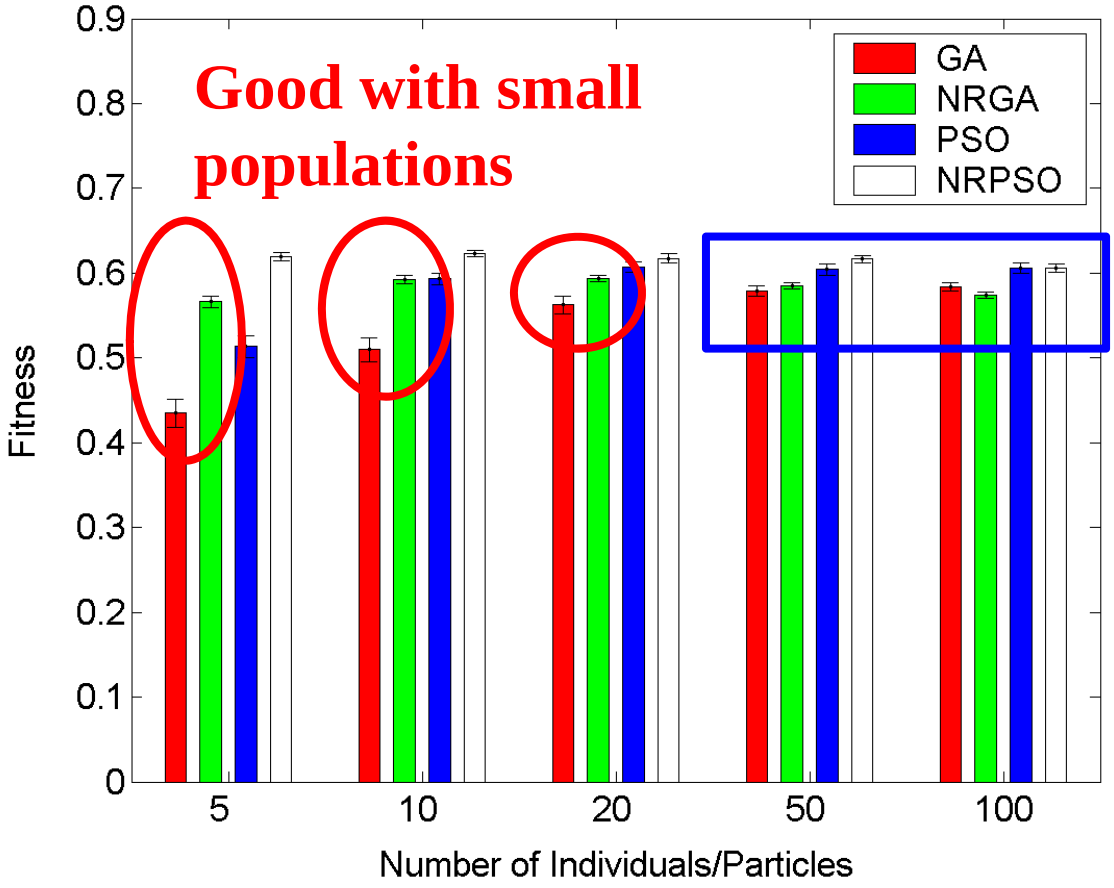
Where Can Noise-Resistant Algorithms Make the Difference?

- Limited adaptation time
- Hybrid adaptation (simulation/hardware in the loop)
- Large amount of noise (see Pugh et al., SIS 2005 and later in multi-robot systems)

Notes:

- all examples from shaping obstacle avoidance behavior
- best learned/evolved solution averaged over multiple runs
- fair tests: same total amount of evaluation time for all the different algorithms (standard and noise-resistant)

Limited-Time Adaptation Trade-Offs



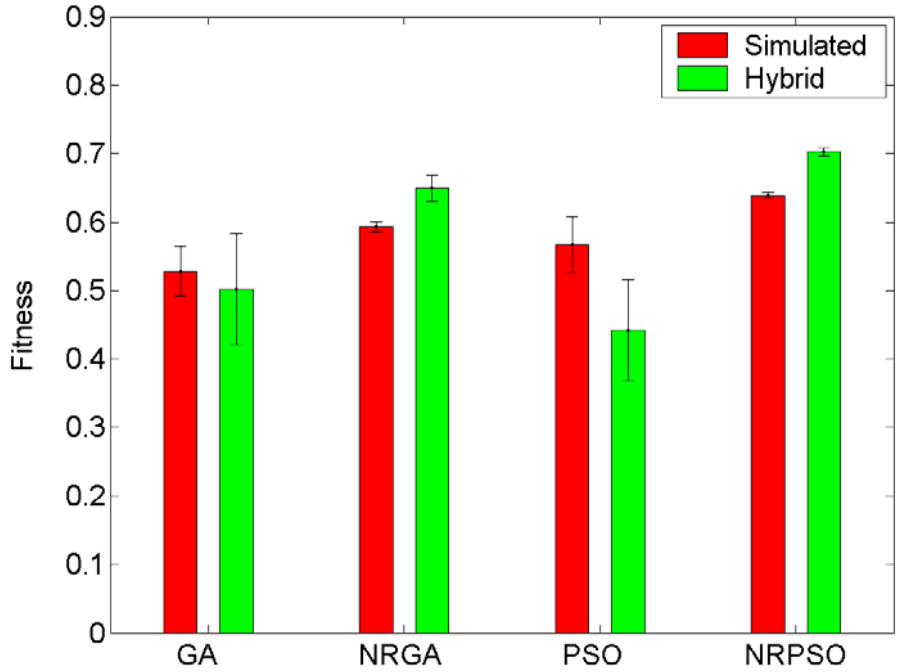
Varying population size vs. number of iterations

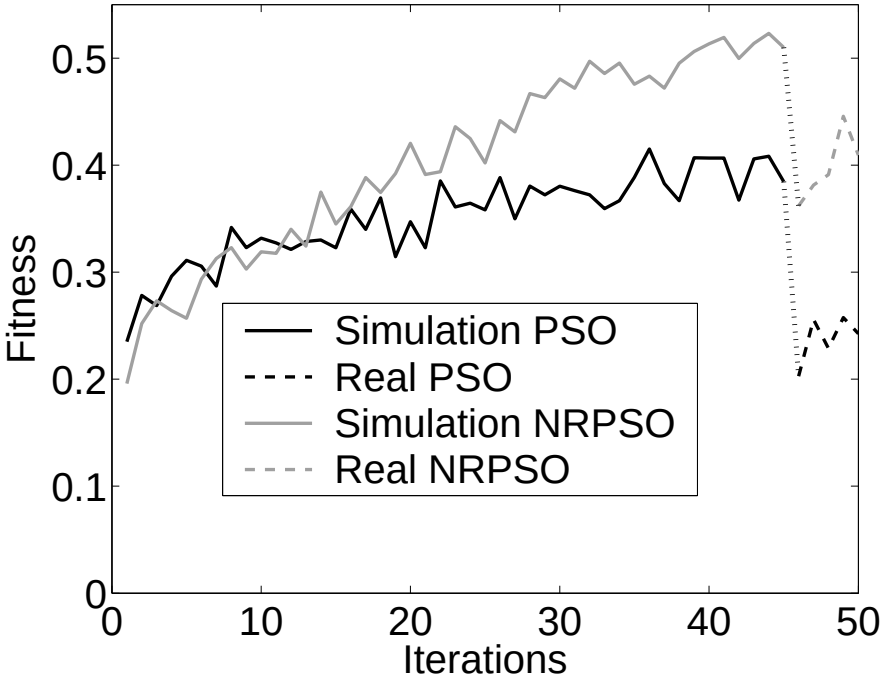
- 1 robot, 24 parameters
- Total adaptation time = 8.3 hours (1/100 of previous learning time)
- Trade-offs: population size, number of iterations, evaluation span
- Realistic simulation (Webots)

[Pugh J., EPFL PhD Thesis No. 4256, 2008]

Hybrid Adaptation with Real Robots

- Move from high-fidelity simulation (Webots) to real robots after 90% of the total number of iterations
- Compromise between time and accuracy
- Noise-resistance helps manage transition





[Pugh J., EPFL PhD Thesis No. 4256, 2008]

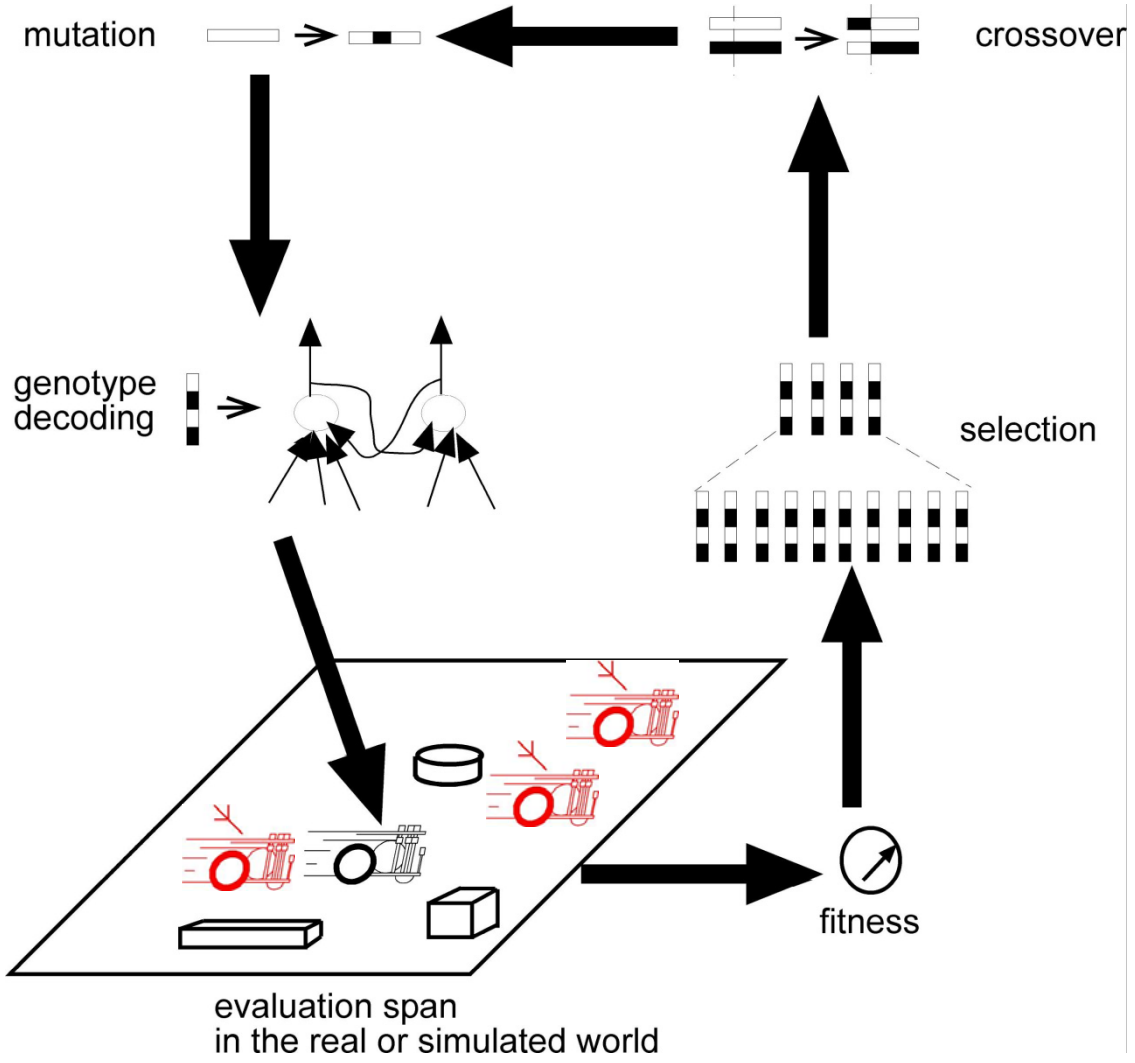
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

From Single to Multi-Unit Systems: Co-Adaptation in a Shared World

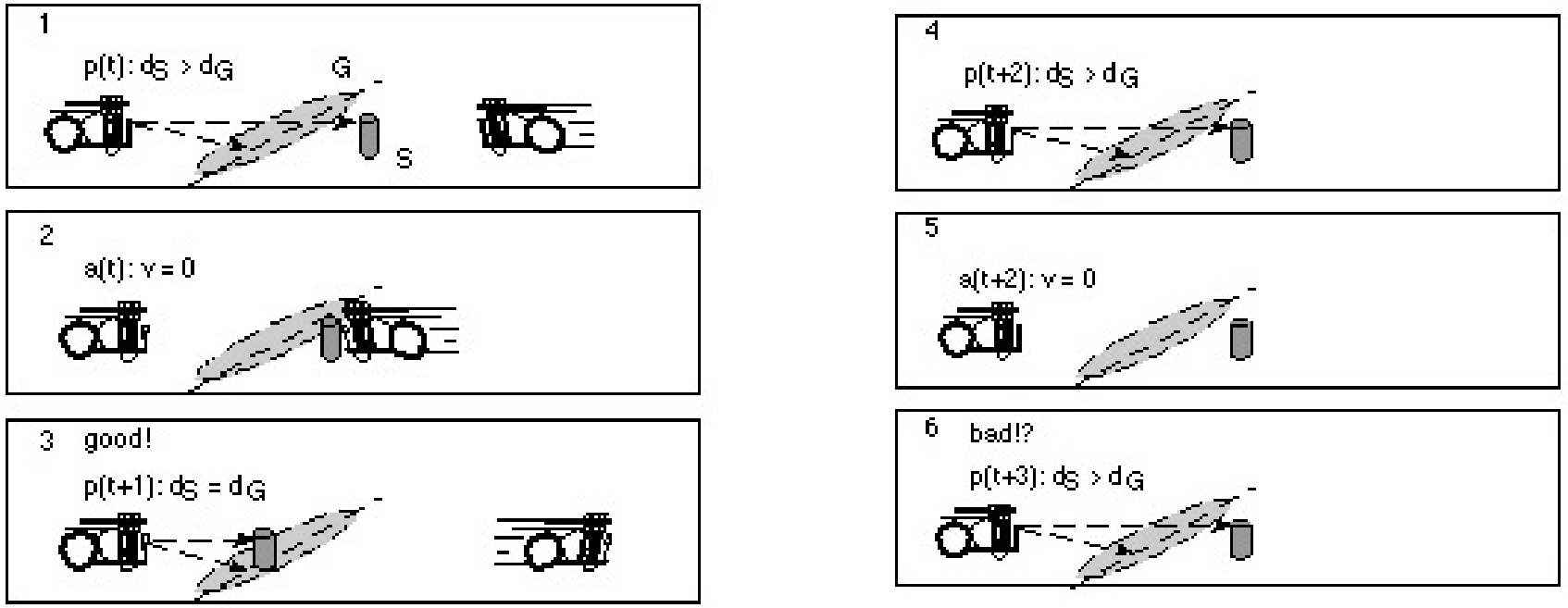
Adaptation in Multi-Robot Scenarios

- Collective: fitness become noisy due to partial perception, independent parallel actions



Credit Assignment Problem

With limited communication, no communication at all, or partial perception:



- A robot cannot distinguish between the environmental modifications caused by its own actions from those generated by others.
- Punishments and rewards are likely to be inconsistent.

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Co-Adaptation in Collaborative Multi-Robot Systems

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Axes for Co-Adaptation

Three orthogonal axes to consider (extremities and balanced solutions are possible):

1. Performance evaluation:
   individual vs. group fitness
2. Solution sharing:
   private vs. public policies
3. Team diversity:
   homogeneous (identical controller and hardware) vs. heterogeneous learning

Co-Adaptation Strategies

Do not make sense (inconsistent)

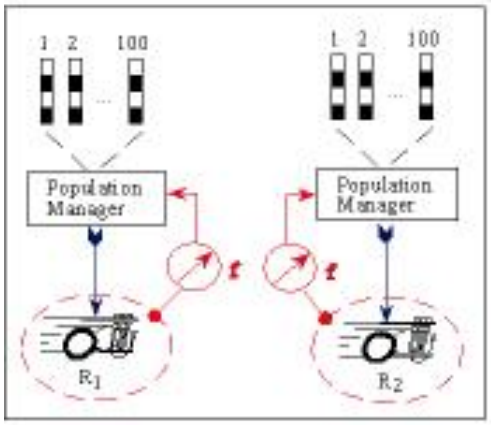
Possible but not scalable

Interesting (consistent)

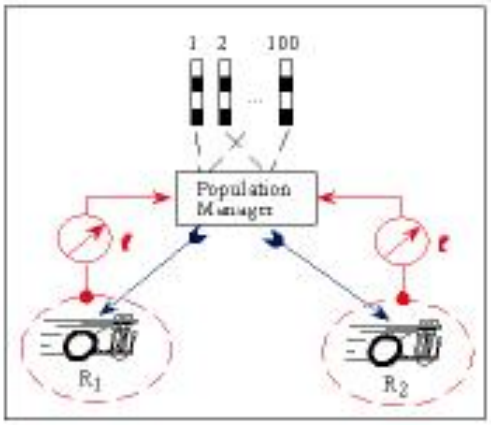
Policy	Performance	Sharing	Diversity
i-pr-he	individual	private	heterogeneous
i-pr-ho	individual	private	homogeneous
i-pu-he	individual	public	heterogeneous
i-pu-ho	individual	public	homogeneous
g-pr-he	group	private	heterogeneous
g-pr-ho	group	private	homogeneous
g-pu-he	group	public	heterogeneous
g-pu-ho	group	public	homogeneous

Population-Based Learning Strategies for Multi-Robot Systems

Private & Individual

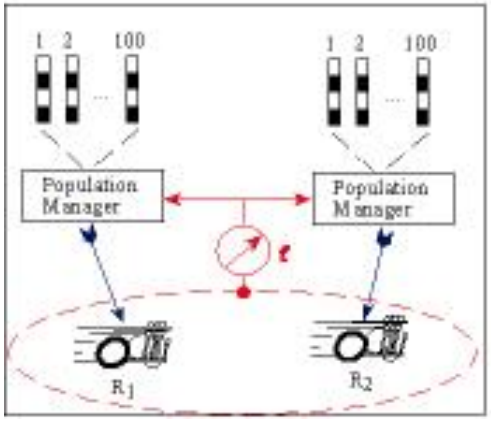


Public & Individual

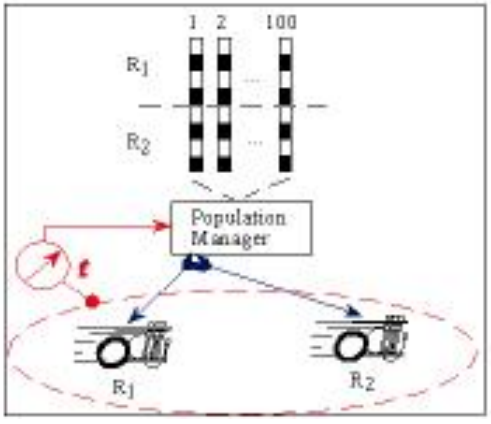


Example of collaborative co-learning with binary encoding of 100 candidate solutions and 2 robots

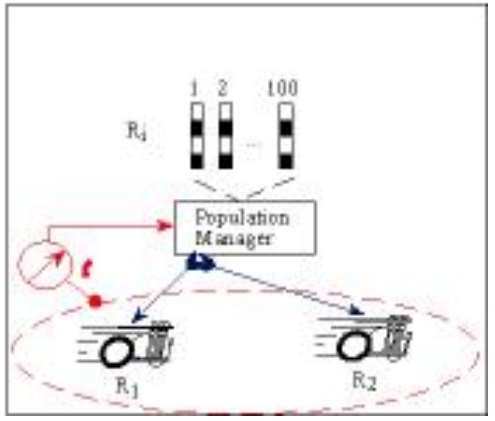
Private & Group



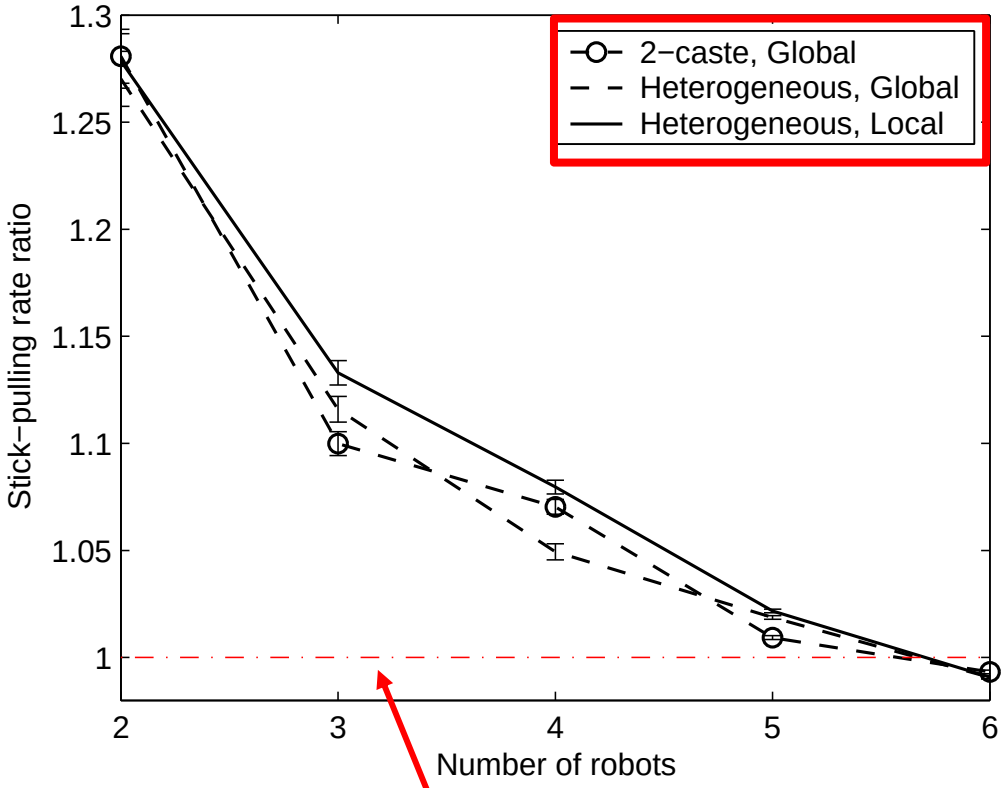
Public & Group (het)



Public & Group (hom)



From W9 (Stick-Pulling Case Study):
Heterogeneous vs. Homogenous Learning



[Li et al., *Adaptive Behavior*, 2004]

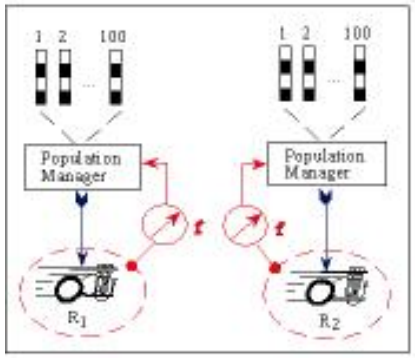
Notes:
- large T_m (long averaging window)
- only private strategies
- global = group
  local = individual

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

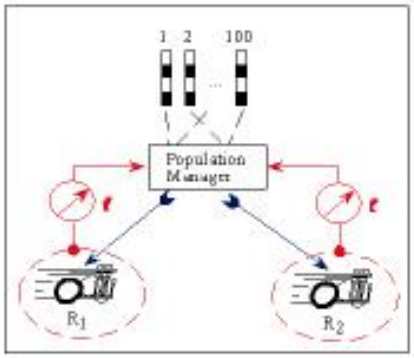
Stick-Pulling Case Study: Homogeneous Learning

- See W9 lecture
- Optimization of a single GTP for the whole team

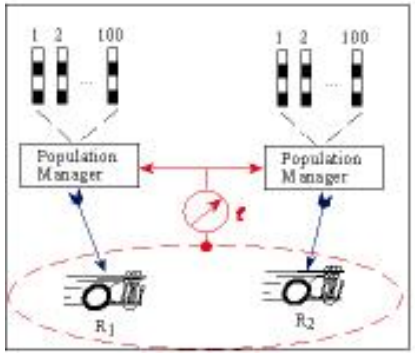
Private & Individual



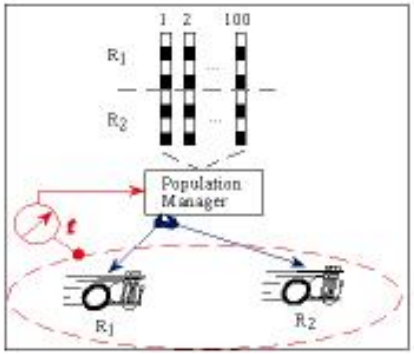
Public & Individual



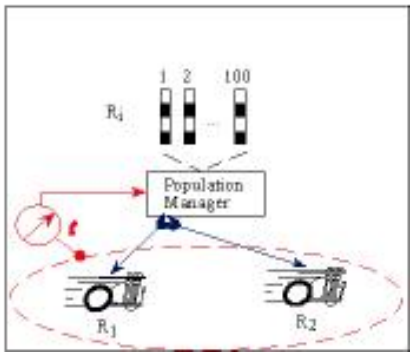
Private & Group



Public & Group (het)

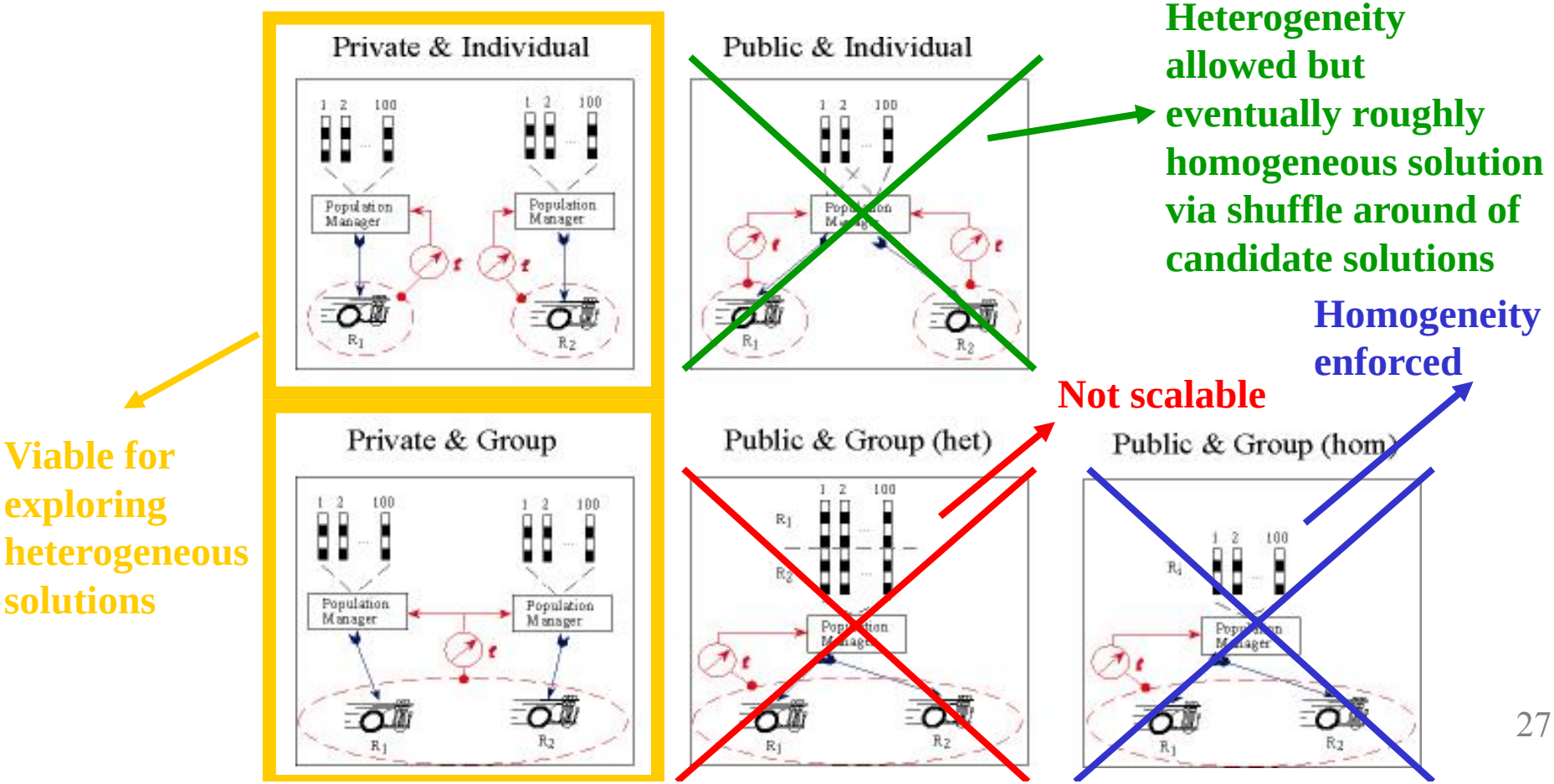


Public & Group (hom)



Stick-Pulling Case Study: Heterogeneous Learning

- See W9 lecture
- Learning to specialize the team members (multiple GTPs)



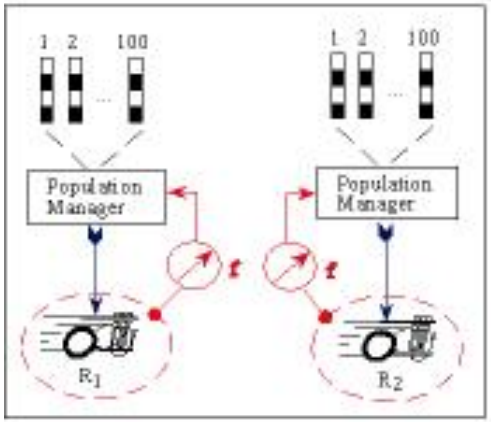
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

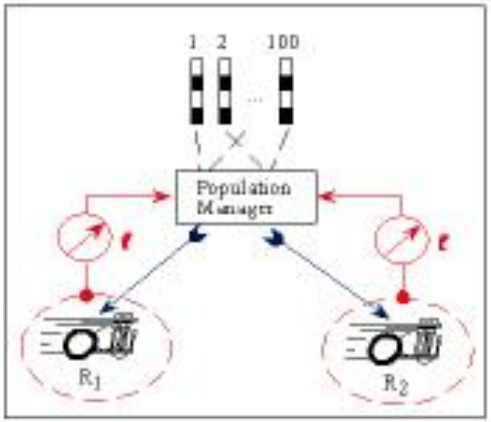
Co-Learning Obstacle Avoidance using PSO

Population-Based Learning Strategies for Multi-Robot Systems

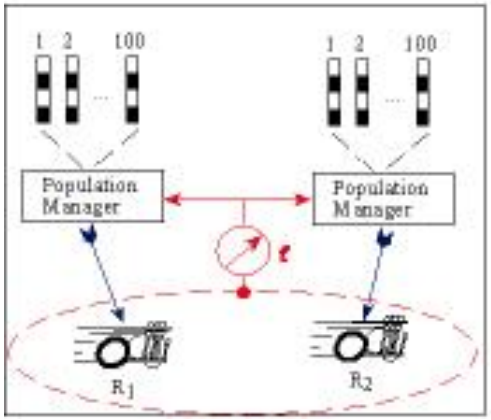
Private & Individual



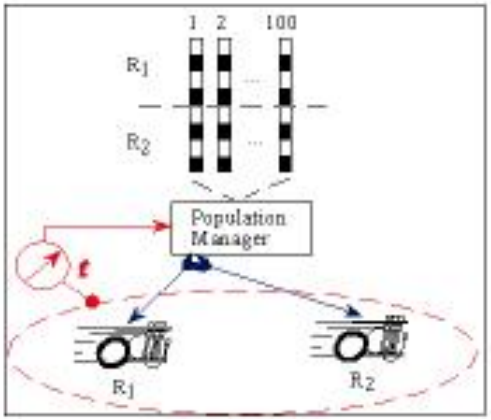
Public & Individual



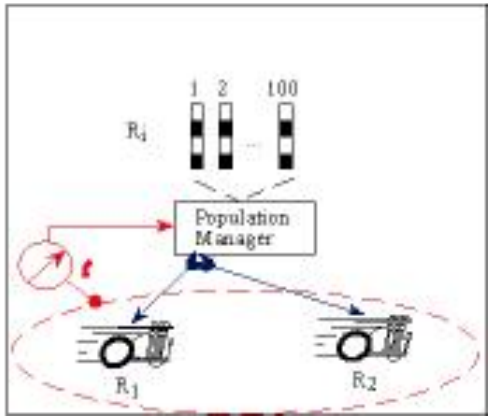
Private & Group



Public & Group (het)



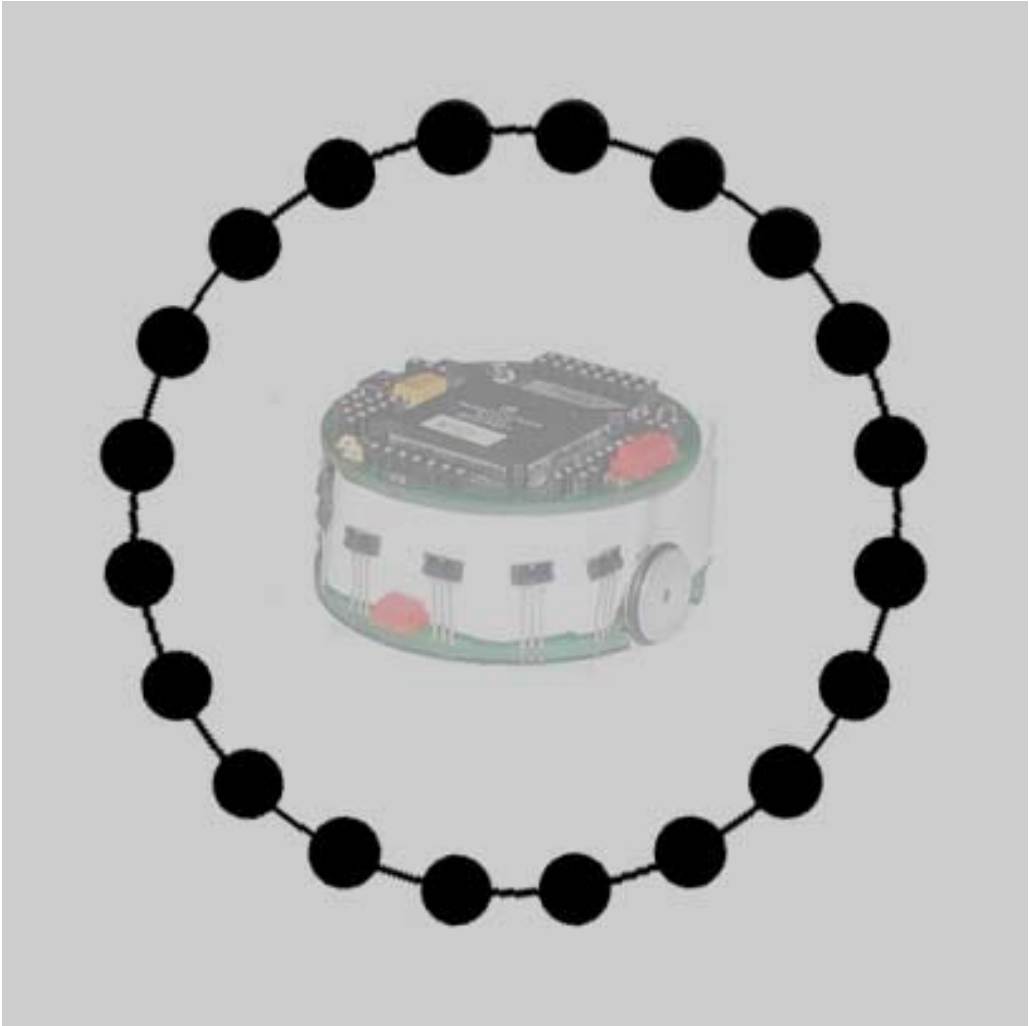
Public & Group (hom)



Distributed Learning using PSO

- Standard approach: evaluate candidate solutions on robots but centralize population manager (off-board)
- New approach: distributed also the population manager on the robots (on-board) and share candidate solutions within the neighborhood through communication channels
- Currently: synchronization at the end of an iteration/generation

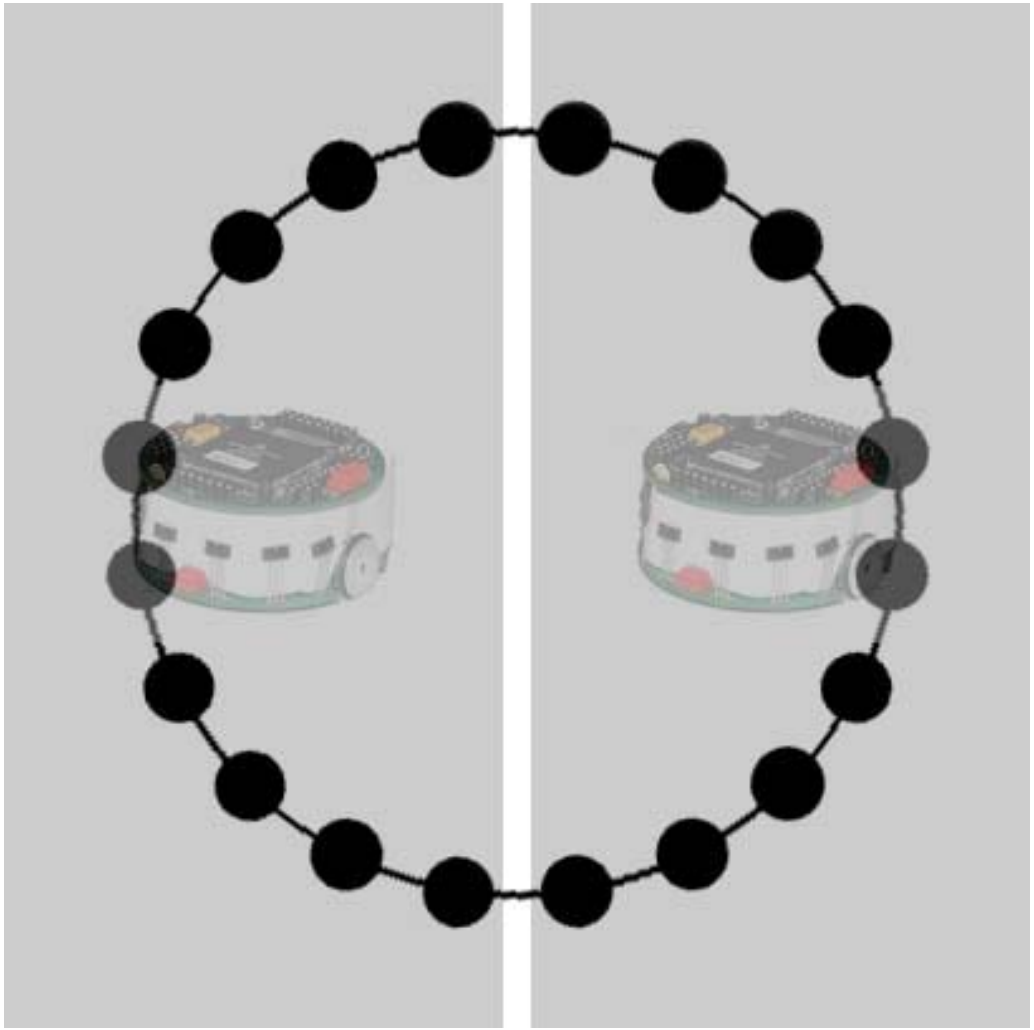
Varying the Robotic Group Size



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

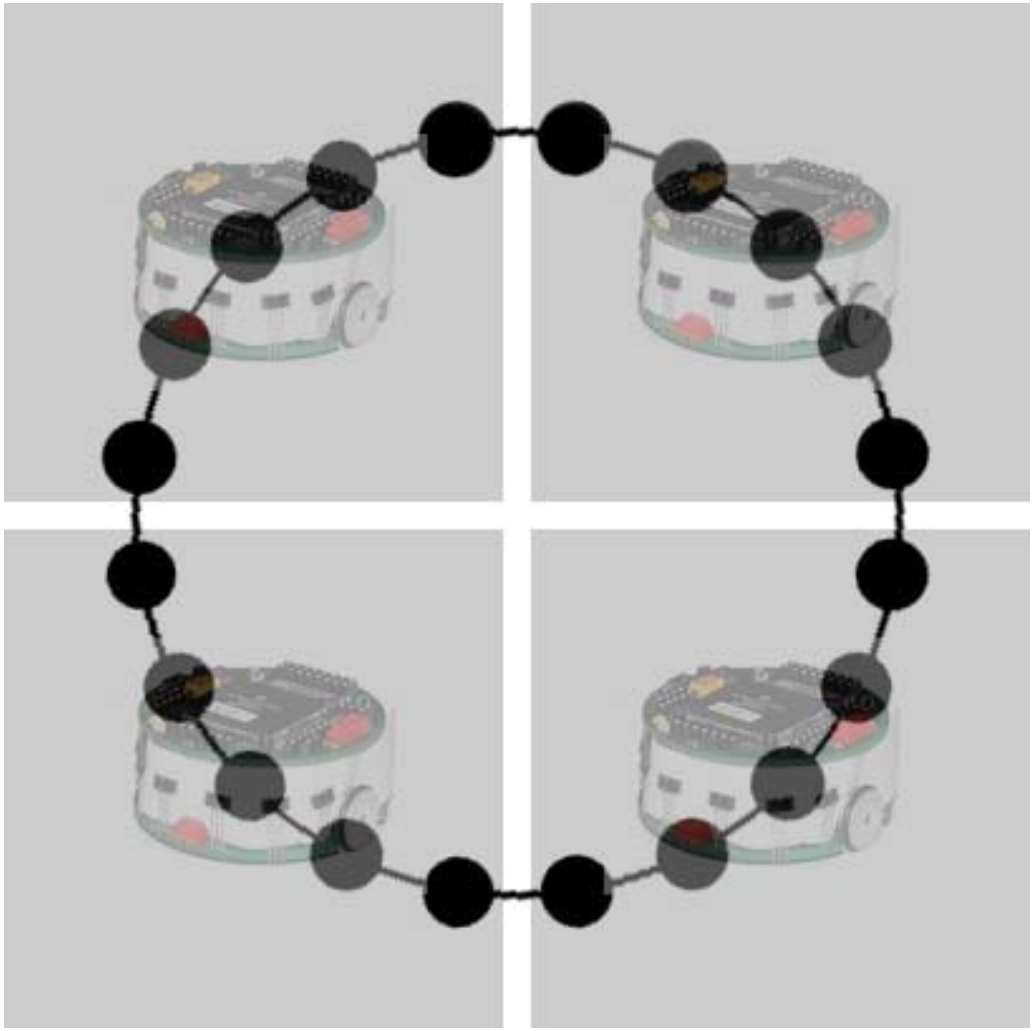
Varying the Robotic Group Size



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Varying the Robotic Group Size



Varying the Robotic Group Size

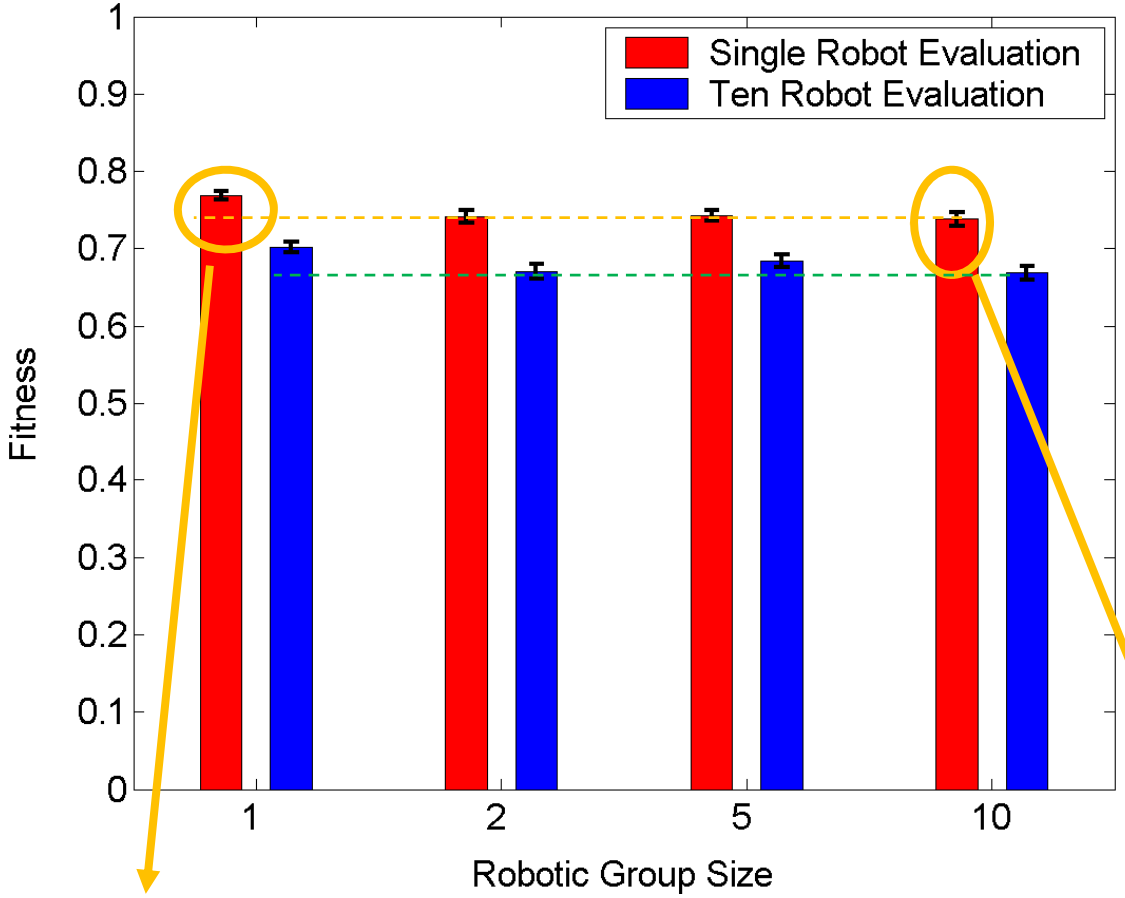
Varying the Robotic Group Size

- Same control architecture as [Floreano & Mondada, 1996] (ANN, 24 weights to tune, Khepera III has 9 proximity sensors)
- Same fitness function as [Floreano & Mondada, 1996]
- Similar Webots world as [Pugh et al., 2005] but 3x3 m
- Robot group size: 1, 2, 5, 10
- PSO parameters
  - Swarm size: 10
  - pw = nw = 2.0
  - w = 0.6

Works but bias the results as in [Pugh et al, 2005]

[Pugh and Martinoli, *Swarm Intelligence J.*, 2009]

Varying the Robotic Group Size – Learning vs. Testing Environment



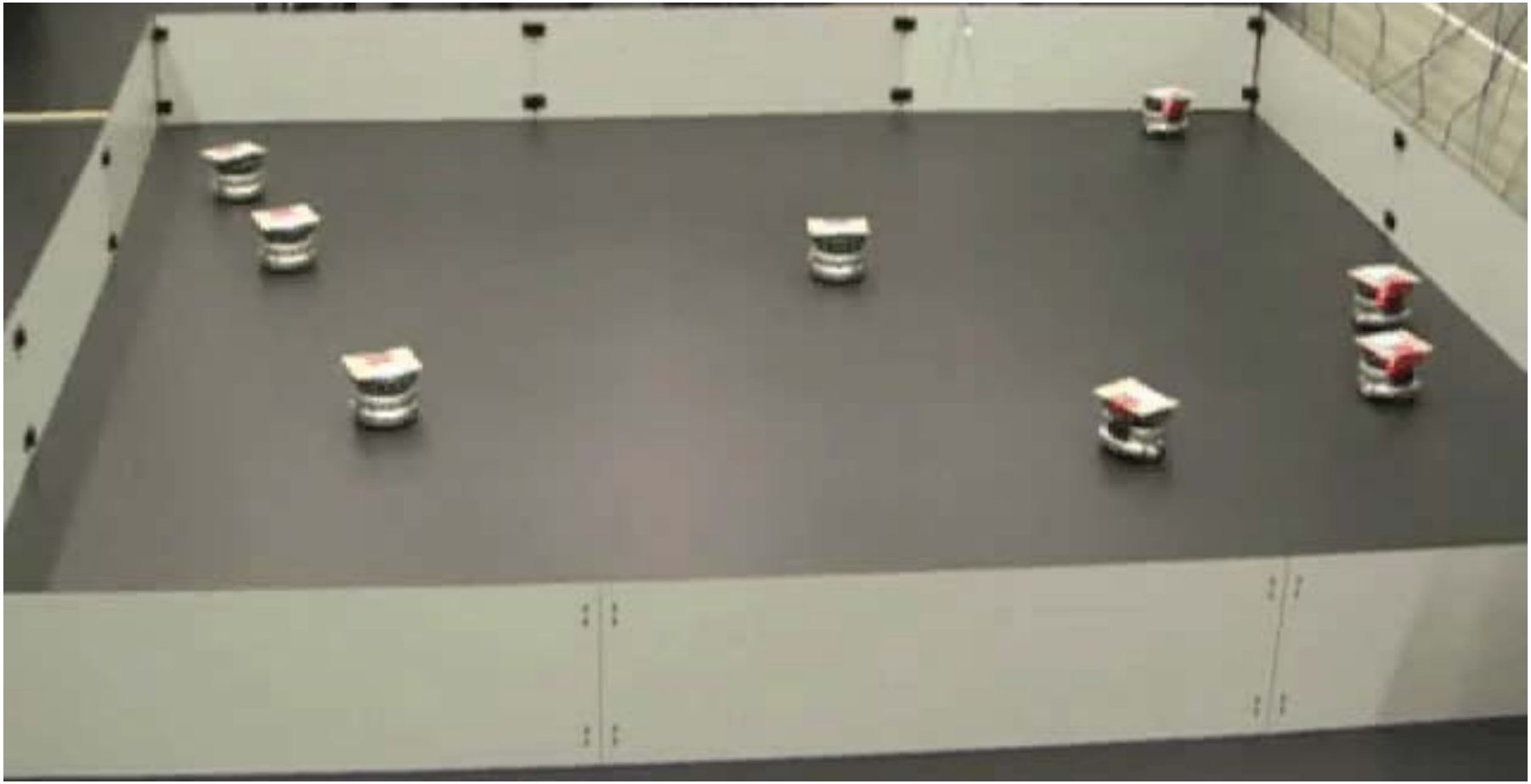
- Gradually increase number of robots on team
- Up to 10x faster learning with little performance loss
- Arena 3x3 m

Learned as single robot, final evaluation as single robot

Learned in a group of 10 robots (10x faster), final evaluation as single robot

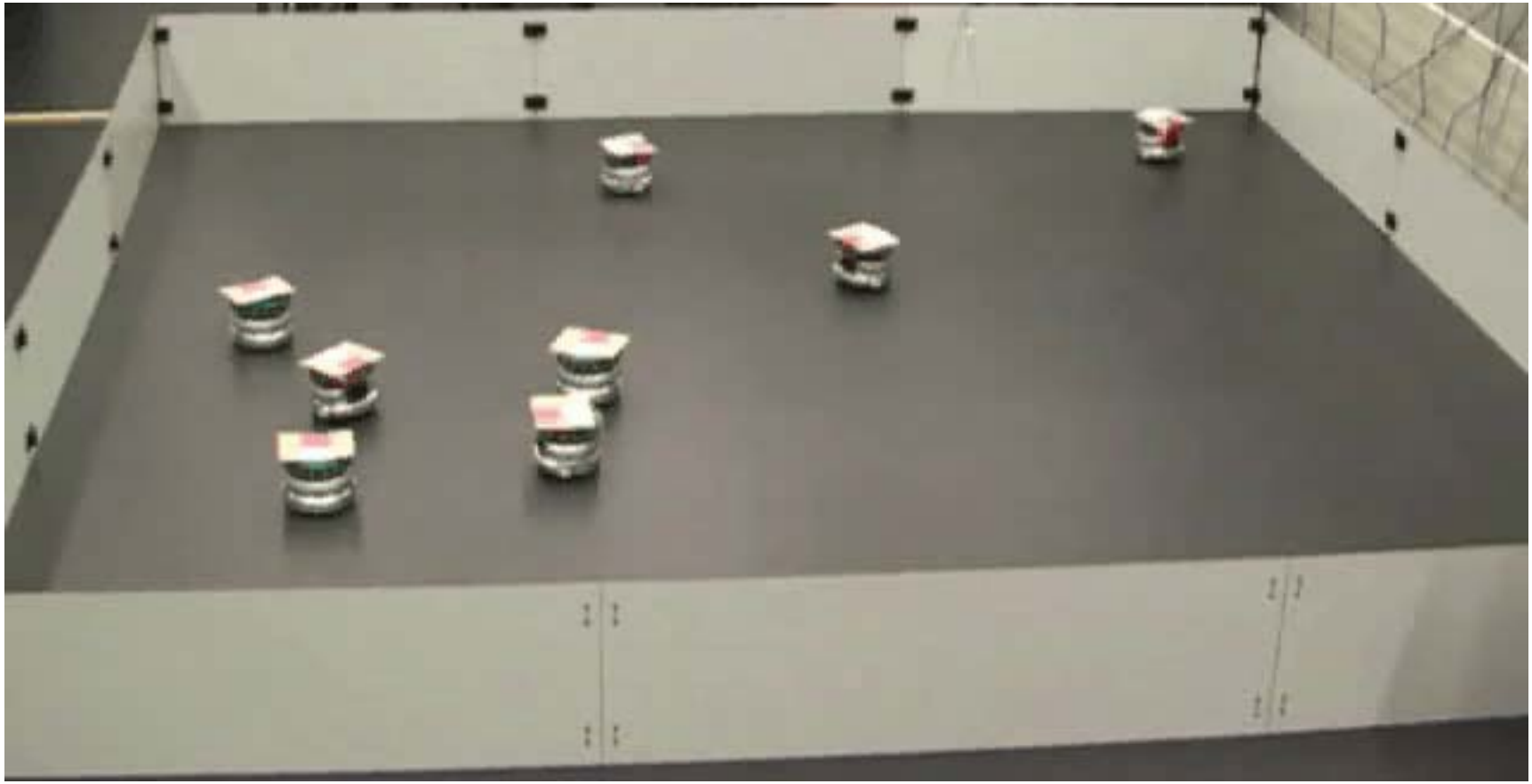
[Pugh and Martinoli, *Swarm Intelligence J.*, 2009]

Distributed Learning with Real Robots (Pugh, 2008)



Before learning (5x speed-up)

Distributed Learning with Real Robots (Pugh, 2008)



After learning (5x speed-up)

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Co-Learning Obstacle Avoidance – Communication-Based Neighborhoods

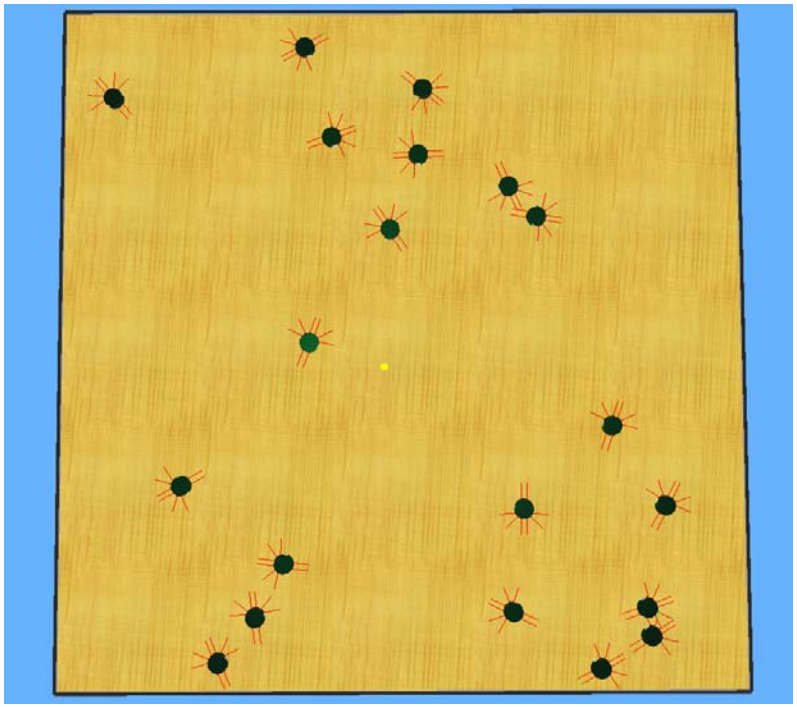
Standard Index-Based Neighborhood

- Default neighborhood: ring topology, 2 fixed index-based neighbors for each particle
- Problem for real robots: neighbors could be very far away

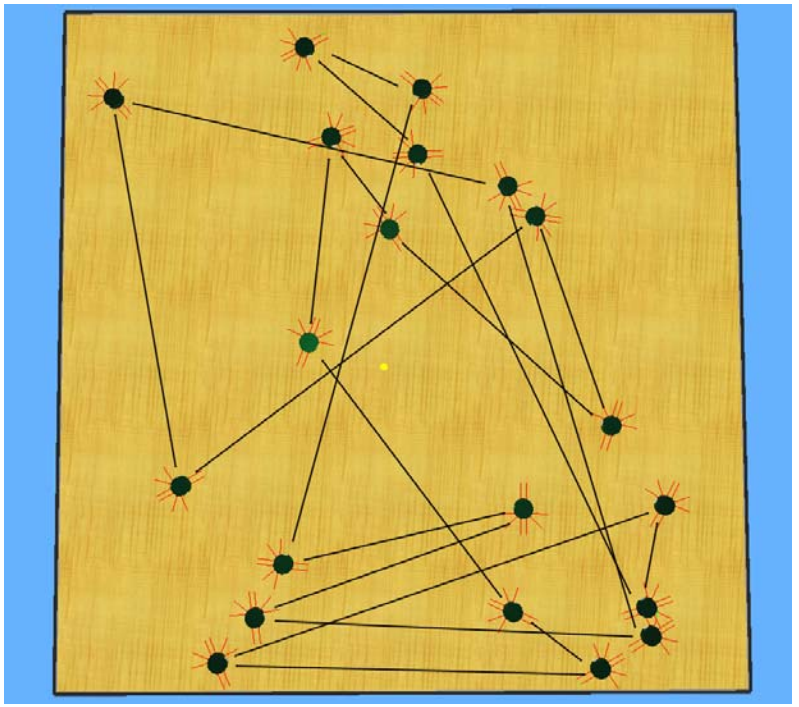
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Index-Based Neighborhood



A possible robot distribution



Ring Topology - Standard

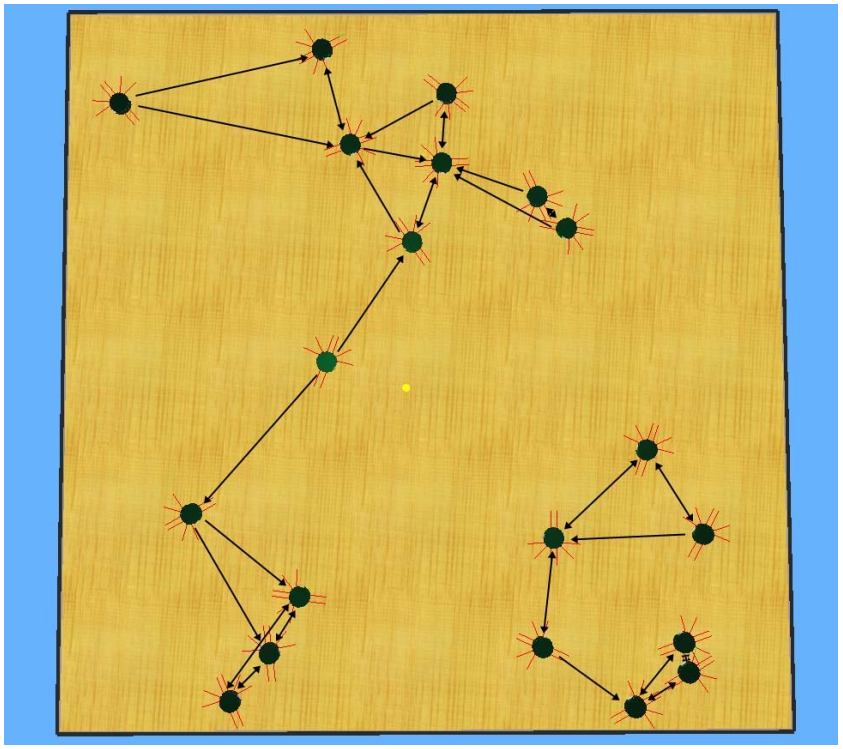
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

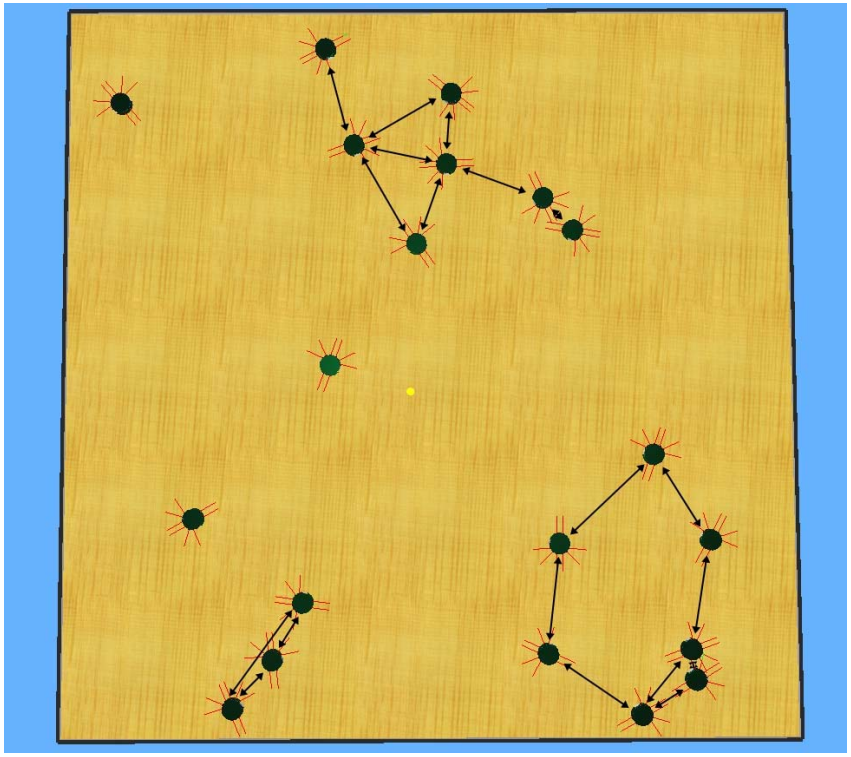
Communication-Based Neighborhoods

- Default neighborhood - ring topology, 2 fixed index-based neighbors for each particle
- Problem for real robots: neighbor could be very far away
- Possible solutions:
  - use two closest robots in the arena (capacity limitation)
  - use all robots within some radius r (range limitation)
- Reality is affected often by both capacity and range

Communication-Based Neighborhoods

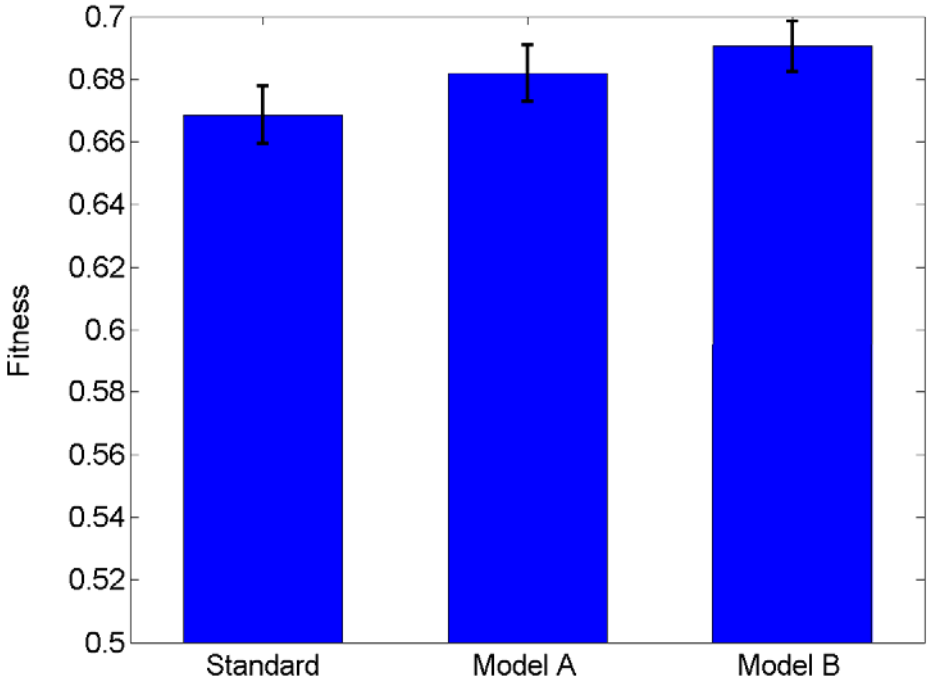


Model **A**: 2-closest robots (capacity limitation)

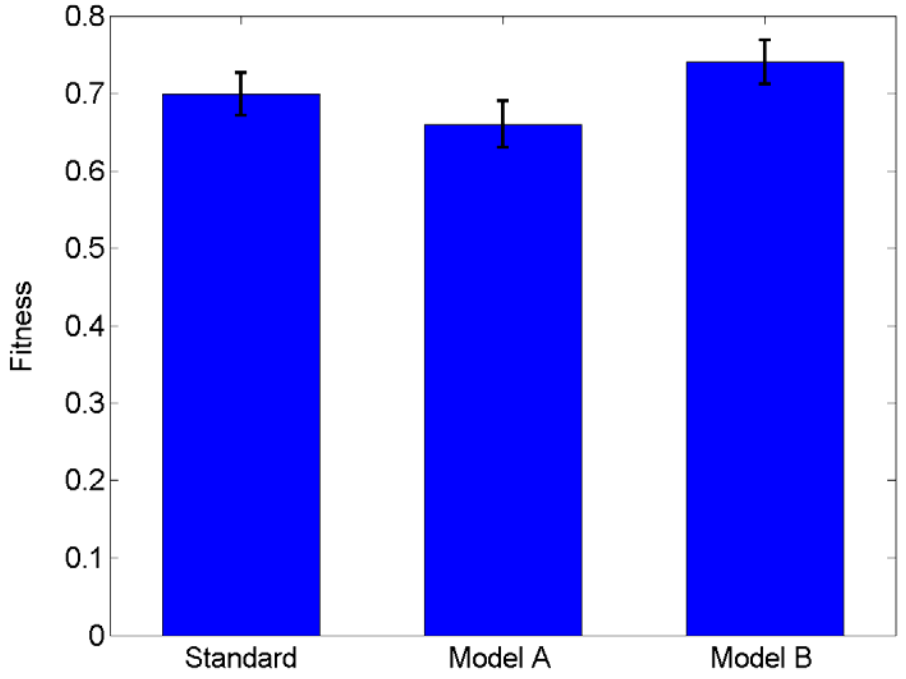


Model **B**: robots within range *r* (range limitation)

Performance Comparison using Different Neighborhoods for 10 Robots



Simulation (Webots)



Real robots

[Pugh and Martinoli, *Swarm Intelligence J.*, 2009]

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Re-Assessing Noise-Resistant Algorithms in Multi-Robot Systems

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

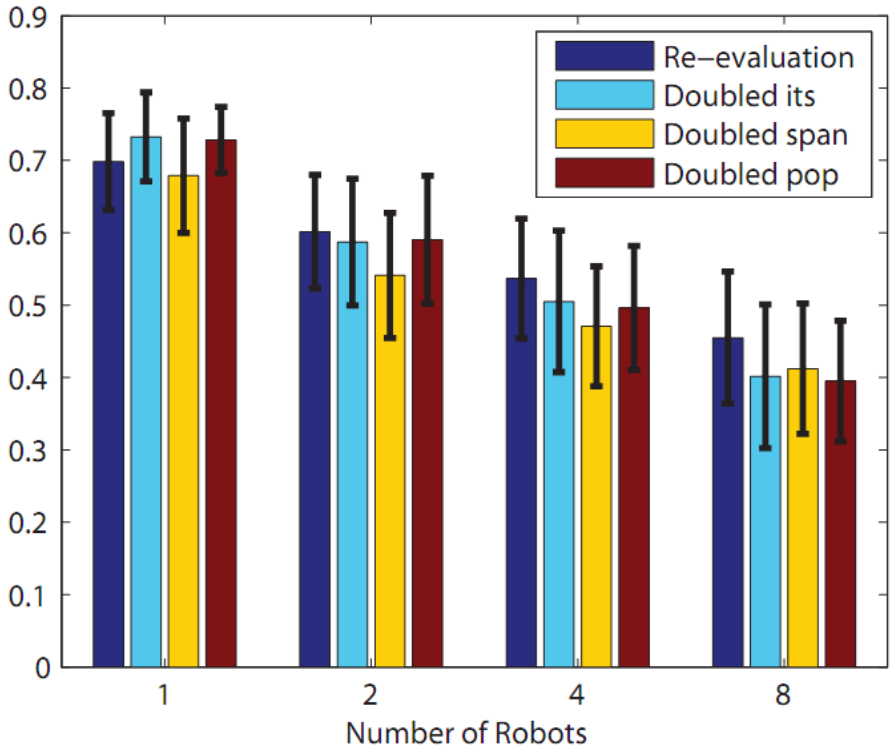
Where Can Noise-Resistant Algorithms Make the Difference?

- Large amount of noise (typically accentuated in multi-robot systems without centralized coordination)
- Limited adaptation time
- Hybrid adaptation (simulation/hardware in the loop)

Notes:

- all examples from shaping obstacle avoidance behavior
- best learned solution averaged over multiple runs
- fair tests: same total amount of evaluation time for all the different algorithms (standard and noise-resistant)

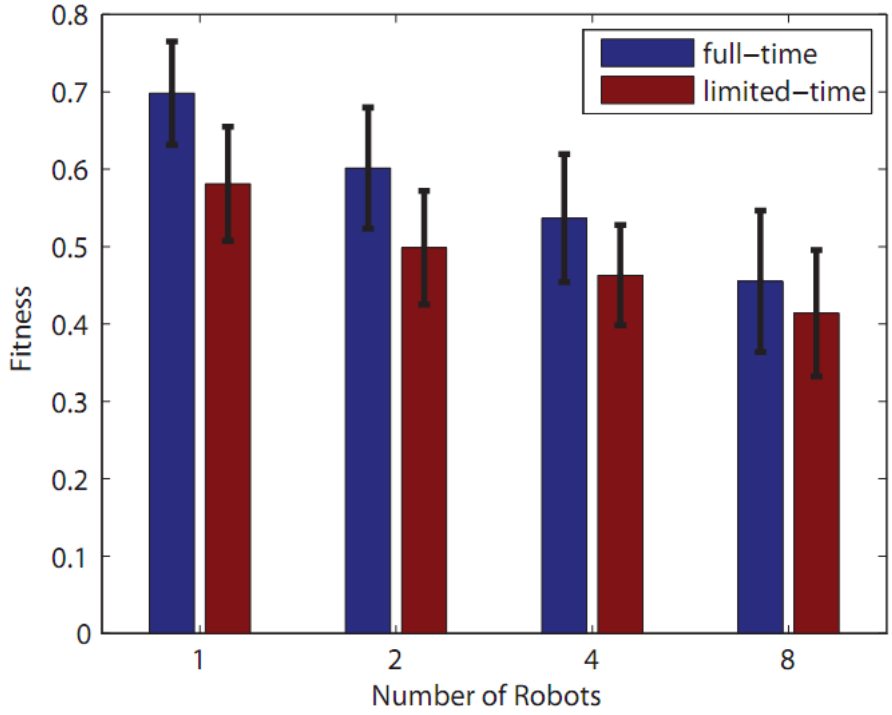
Increasing Number of Robots: Impact of Noise Resistance



- Webots experiments
- 1x1 m arena (high density!)
- Fair test: same amount of total evaluation time for each bar
- Performance decreases with number of robots (more difficult to avoid in overcrowded arenas)
- Noise-resistance make the difference in high density (i.e. noisier) scenarios

[Di Mario and Martinoli, *Robotica*, 2014]

Impact of Limited Time Adaptation



[Di Mario and Martinoli, *Robotica*, 2014]

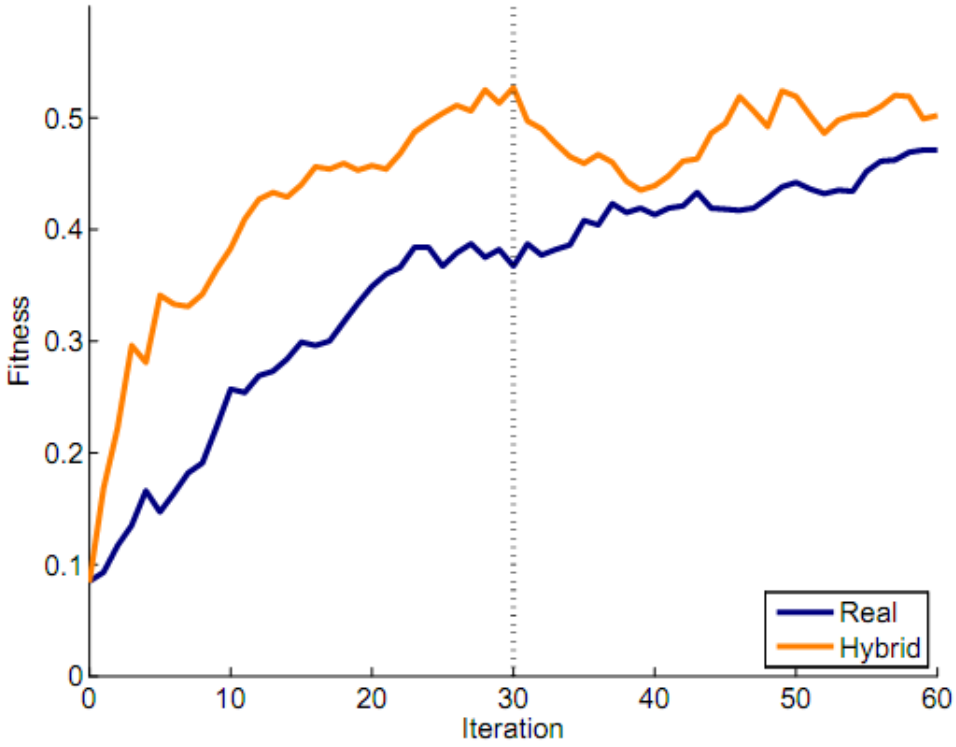
- Webots experiments
- 1x1 m arena (high density!)
- full-time adaptation: 417 h
- limited time adaptation: 8h
- 52 times smaller evaluation time, 17% max drop in performance
- same obstacle avoidance strategy

Recipe:

1. Evaluation span include at least 1 interaction
2. Swarm size = dimension of parameter space
3. Use noise-resistant algorithms
4. Dedicate max time budget to iterations

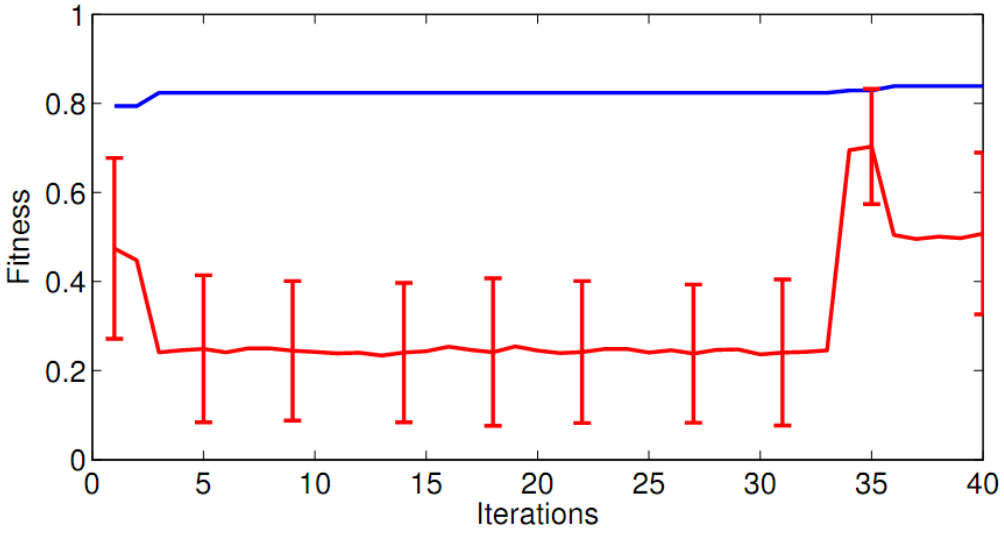
Hybrid Adaptation vs. Only Real Robots

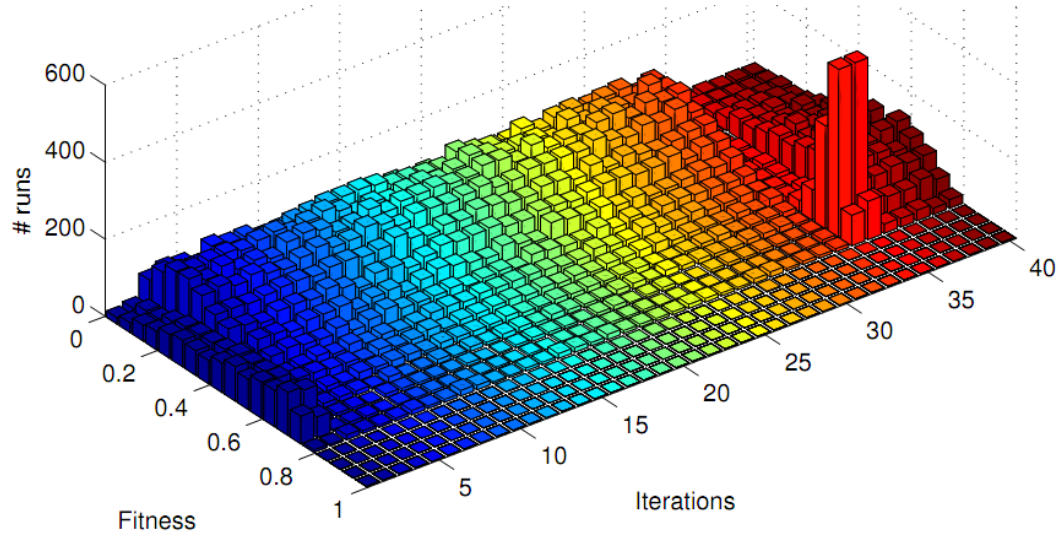
- Noise-resistant PSO
- 4 robots
- Hybrid: 30 iterations in simulation, then 30 iterations on real robots
- Achieves similar fitness as running 60 iterations on real robots
- Requires half the real robot evaluation time



[Di Mario and Martinoli, *Robotica*, 2014]

Why Noise-Resistant Algorithms Make the Difference?

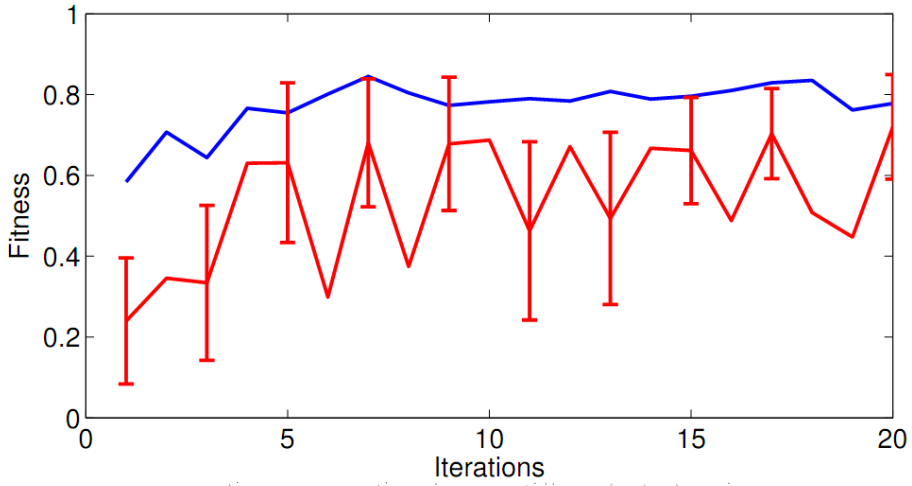


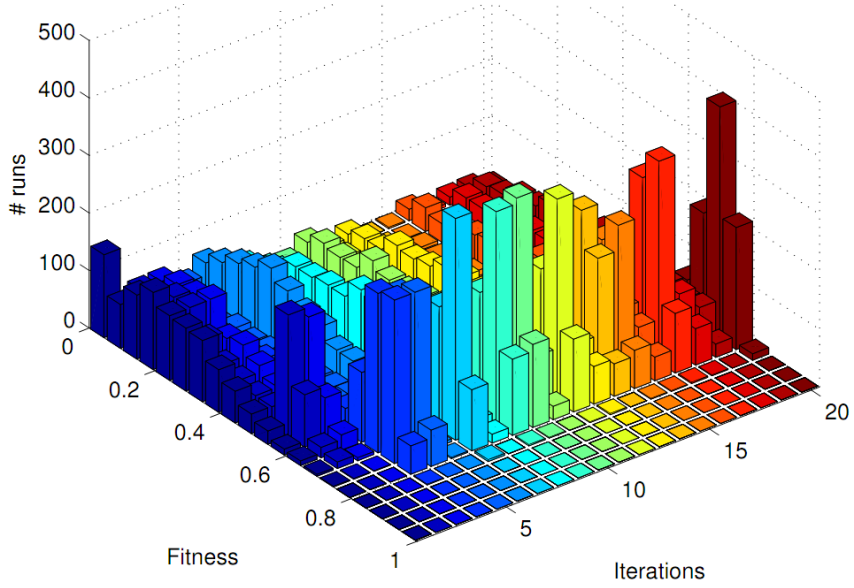


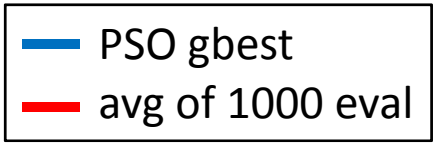
Standard PSO vs. A-Posteriori evaluations (Obstacle avoidance, 4 robots)

[Di Mario et al., CEC 2014]

Why Noise-Resistant Algorithms Make the Difference?







Noise-Resistant PSO vs. A-Posteriori evaluations

(Obstacle avoidance, 4 robots)

[Di Mario et al., CEC 2014]

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Noise-Resistant Algorithms in Multi-Robot Systems: From Pbest-based strategies to OCBA

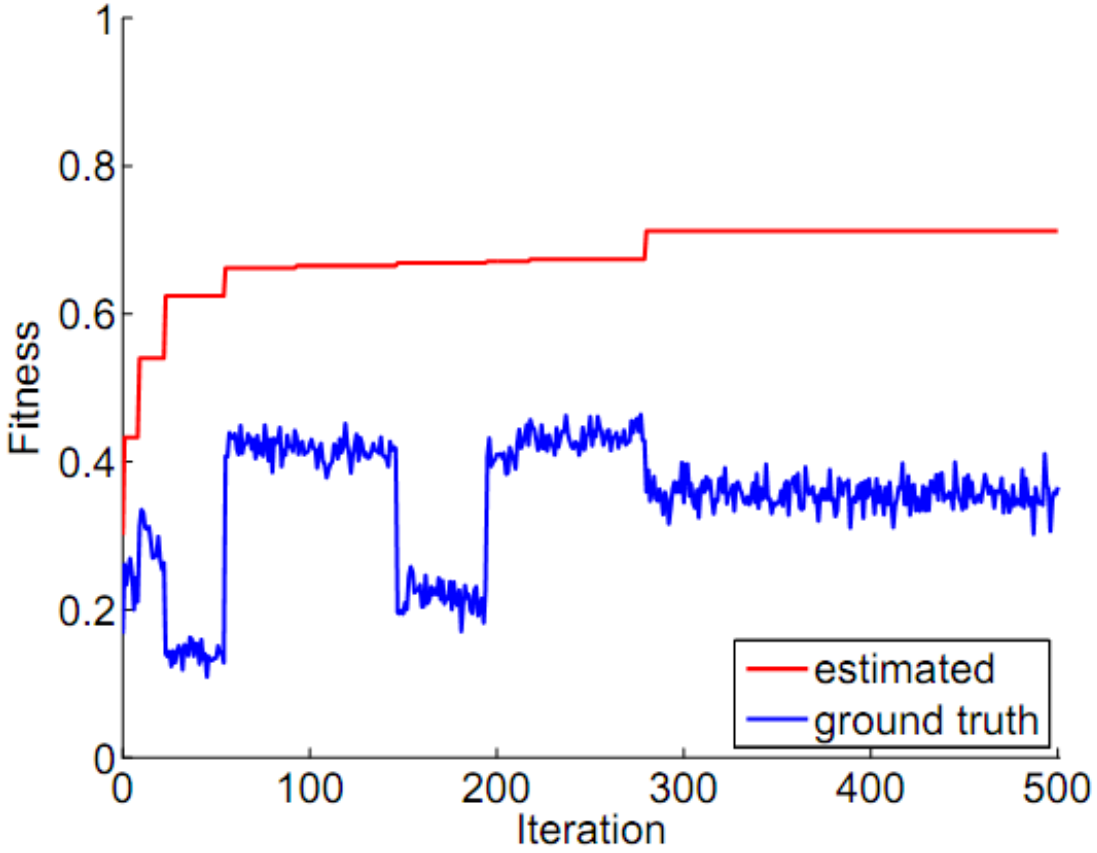
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Benchmark Task: Obstacle Avoidance

- 24 parameters of Artificial Neural Network (D= 24)
- Usual fitness function [Floreano and Mondada, 1996]
- [Di Mario et al., ICRA 2015 and CEC 2015]

Standard PSO: no re-evaluations



- D = 24
- 24 particles
- 500 iterations
  (24 eval./iteration)
- Over-estimation
- Stagnation
- Iterations wasted

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

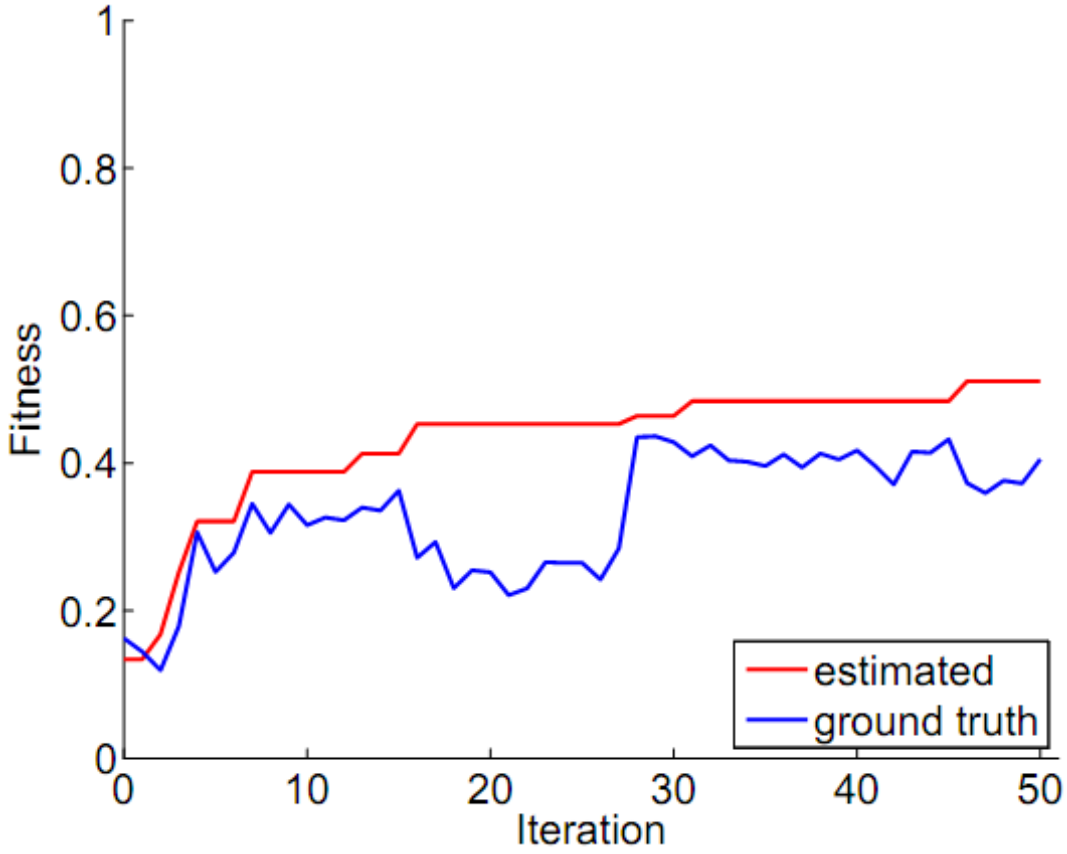
Disal

Approach 1: PSO rep

```
1: Initialize particles
2: for N_i iterations do
3:     for N_p particles do
4:         Evaluate particle position
5:         Update personal best
6:         Update neighborhood best
7:         Update particle position
8:     end for
9: end for
```

- Each function evaluation replaced by average of *k* evals
- PSO algorithm not changed
- If noise is Gaussian, std dev reduced by a factor of sqrt(*k*)

PSO rep10: 10 re-evaluations



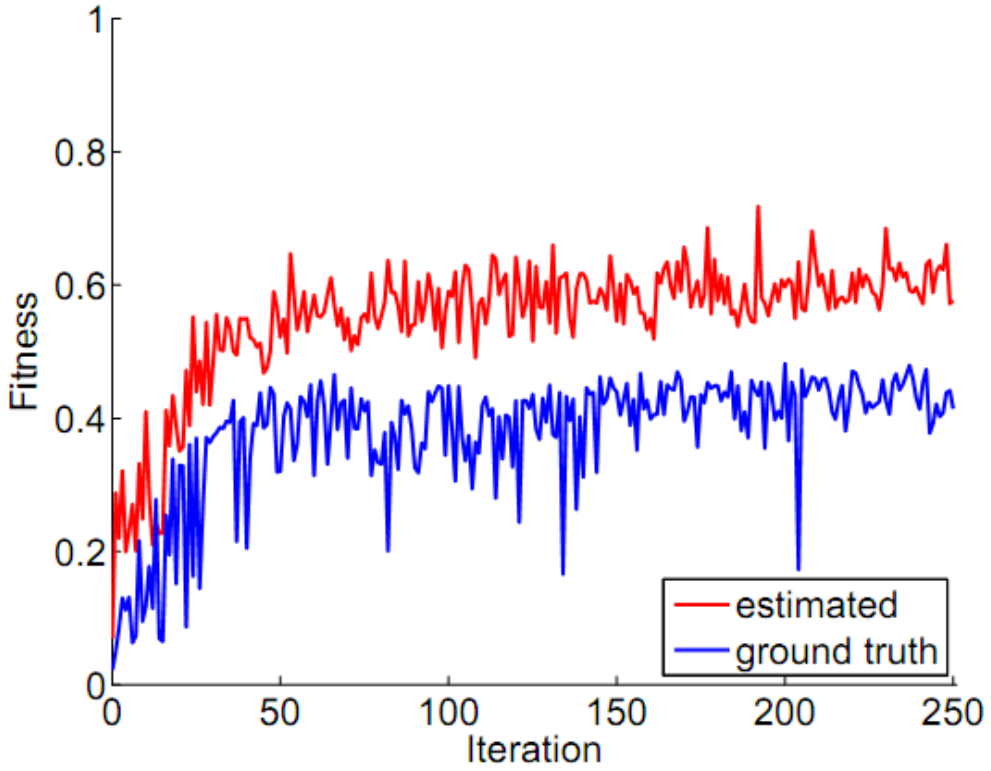
- 10x less iterations (24x10 = 240 evaluations/iteration)
- Less over-estimation
- Less stagnation

Approach 2: PSO *pbest*

1: Intialize particles
2: **for** N_i iterations **do**
3: **for** N_p particles **do**
4: Update particle position
5: Evaluate particle
6: Re-evaluate personal best
7: Aggregate with previous best
8: Share personal best
9: **end for**
10: **end for**

[Pugh and Martinoli, *Swarm Intelligence J.*, 2009]

PSO *pbest*



- 2x less iterations (2x24 = 48 evaluations/iteration)
- No stagnation
- Random drops: poor estimates of new candidates
- Still overestimation

Approach 3: OCBA

- Chen et al [1] : select the number of samples (evaluations) N_i per candidate i according to:

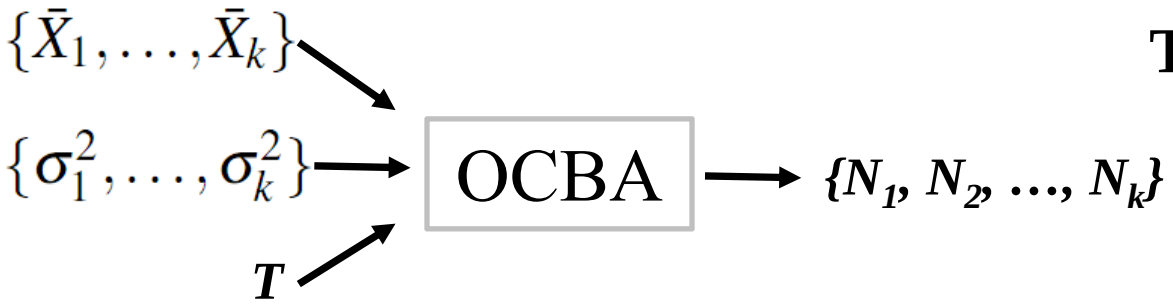
$$\frac{N_i}{N_j} = \left(\frac{\sigma_i/\delta_{b,i}}{\sigma_j/\delta_{b,j}}\right)^2, i \neq j \neq b \qquad N_b = \sigma_b \sqrt{\sum_{i=1, i\neq b}^{k} \frac{N_i^2}{\sigma_i^2}}$$

- Intuition: more samples for candidates with:
  - higher variance
  - mean closer to the best (low delta) $\delta_{i,j} = \bar{X}_i - \bar{X}_j$
- Proven that maximizes probability of correct selection of best candidate *b* for infinite total number of samples (evaluations) *T*
- But "works well in practice" for finite number of samples (evaluations) *T*

[1] C. Chen, J. Lin, E. Yücesan, and S. E. Chick, "Simulation Budget Allocation for Further Enhancing the Efficiency of Ordinal Optimization", 2000.

OCBA in Practice

k particles
T evaluations/iteration



- Use empirical means and std devs as estimates for OCBA
  1) Sample all candidates n_0 times
  2) Calculate initial empirical means and std devs
  3) While there is budget left:
     - –Allocate Δ additional samples using OCBA
     - –Evaluate the new samples
     - –Update means and std devs
     - –Reduce budget by Δ
- Parameter Δ controls the number of allocation steps

Approach 3: Centralized PSO OCBA

```
1: Initialize particles
2: for N_i iterations do
3:    for N_p particles do
4:       Evaluate new particle position n_0 times
5:    end for
6:    remaining budget := iteration budget - n_0 · N_p
7:    while remaining budget> 0 do
8:       Allocate Δ samples among current positions and
         personal bests using OCBA
9:       Evaluate allocated samples
10:      Recalculate mean and variance for new evaluations
11:      remaining budget := remaining budget - Δ
12:   end while
13:   for N_p particles do
14:      Update personal best
15:      Update neighborhood best
16:      Update particle position
17:   end for
18: end for
```

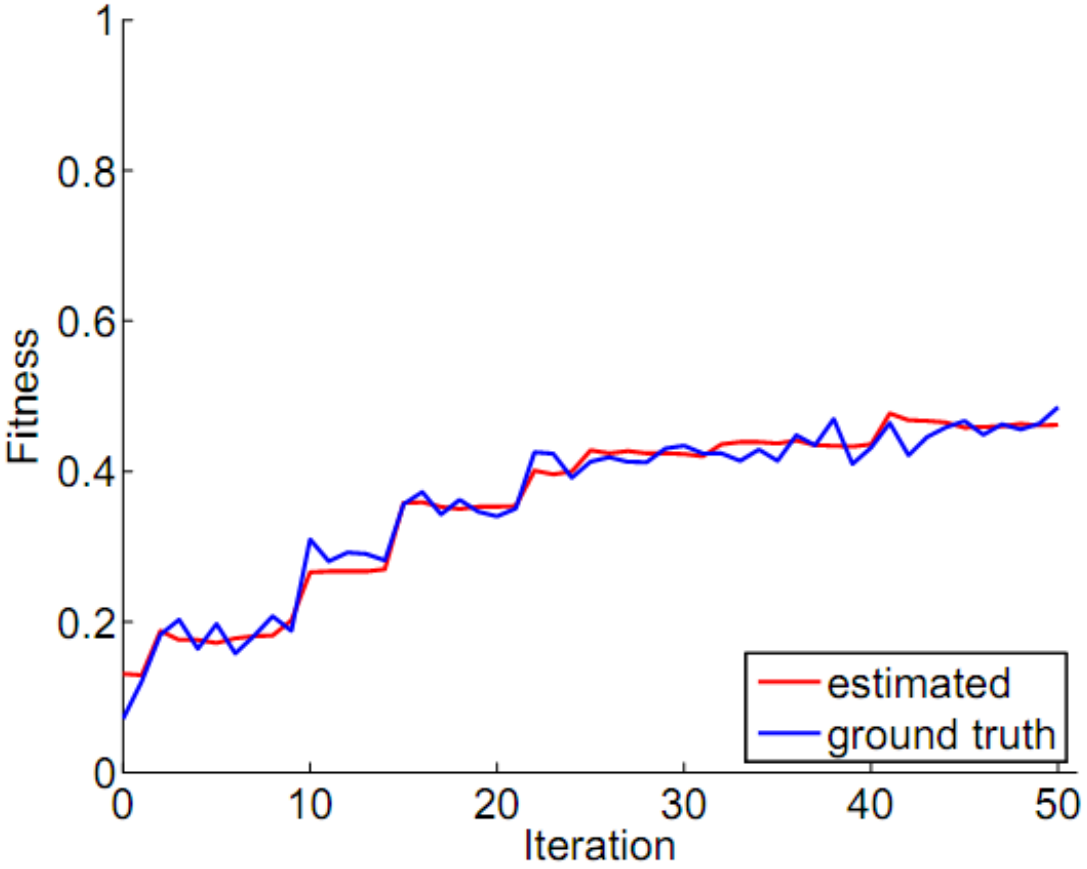
n_o = 2, 24 particles →
2x24 = 48 evaluations

Number of candidates is twice the total number of particles

Δ = 4 →
4x2x24 = 196 evaluations

48+196=240 eval./iteration

PSO OCBA C



- 10x less iterations (240 evaluations/iteration)
- No stagnation, no overestimation
- Can distribute on multi-robots with global networking

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Approach 4: Distributed PSO OCBA

- Each particle conducts its own OCBA allocation
- Candidates for OCBA are new positions and pbests in neighborhood
- N candidates = 2 * Neighborhood size
- Mean and standard deviation can be calculated online by storing only the previous values and the number of samples
- Memory and communication overhead is small and constant

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Distributed PSO OCBA

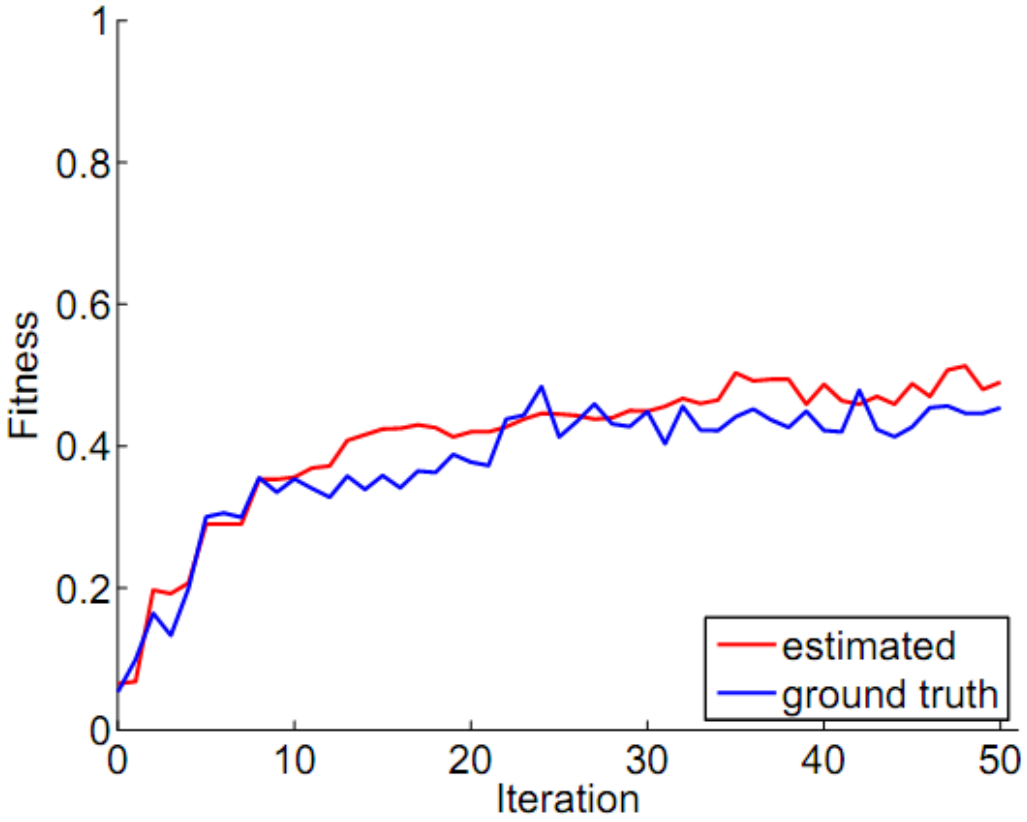
1: Initialize particle
2: **for** N_i iterations **do**
3: Evaluate new particle position n_0 times
4: Share evaluation results in neighborhood
5: Receive and store evaluation results from neighborhood
6: remaining budget := iteration budget - $n_0 \cdot N_p$
7: **while** remaining budget> 0 **do**
8: Allocate Δ samples among current positions and personal bests in neighborhood using OCBA
9: Evaluate allocated samples
10: Recalculate mean and variance for new evaluations
11: Share evaluation results in neighborhood
12: Receive and store evaluation results from neighborhood
13: remaining budget := remaining budget - Δ
14: **end while**
15: Update personal best
16: Update neighborhood best
17: Update particle position
18: **end for**

Pseudo code seen from a single particle

Number of candidates is twice the PSO neighborhood size

Iteration budget = 10 evaluations/particle → 10x24 = 240 evaluations/iteration

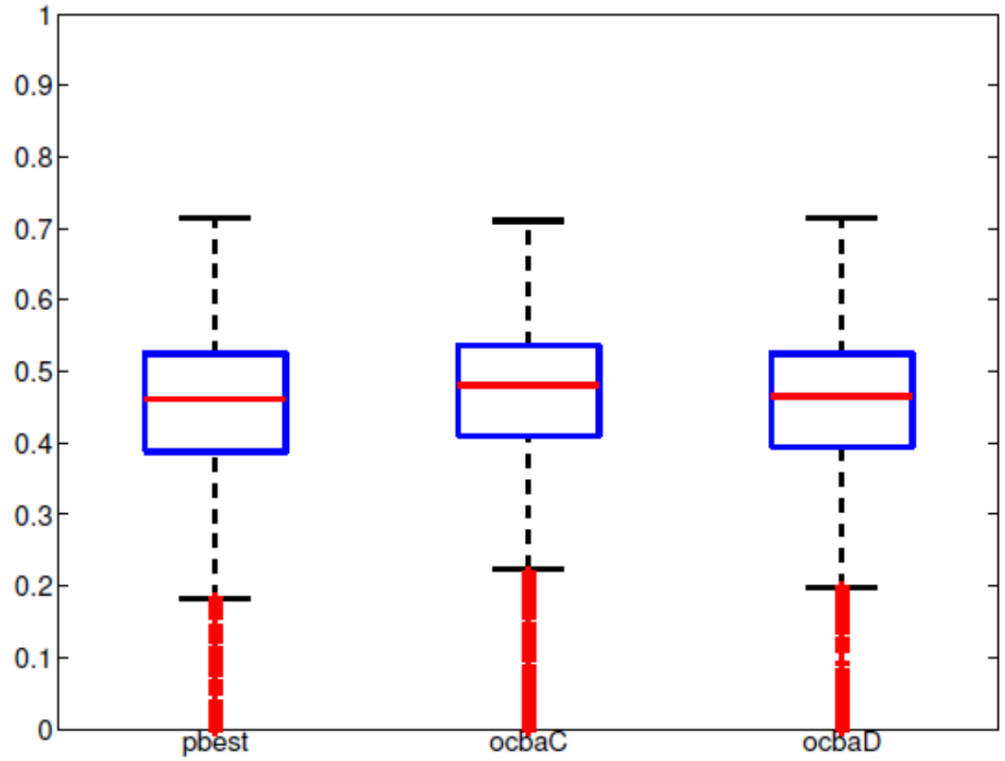
PSO OCBA D



- 10x less iterations (240 evaluations/iteration)
- No stagnation
- Very little overestimation, still higher than centralized OCBA
- Can distribute on multi-robot with local networks

Summary of Results

Despite a significant better estimation ("red" lines in the previous slides) of OCBA techniques , all noise-resistant algorithms lead in this scenario to only a slight increase of the absolute performance ("blue" lines in the previous slides)



[Di Mario et al., CEC 2015]

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Conclusion

Take Home Messages

- The cost of an optimization problem is heavily influenced by the amount of noise in the evaluation function, the time needed for evaluating a candidate solution, and the dimension of the parameter space
- Collaborative co-adaptation strategies can be differentiated along three axes: public/private solutions; homogeneous/heterogeneous system, individual/group performance
- Multi-robot platforms can be exploited for testing in parallel multiple candidate solutions
- One way to bypass the credit assignment problem in multi-robot contexts is to enforce homogeneity and reward group performance
- PSO appears to be well suited for fully distributed on-board operation and fairly robust to small pools of candidate solutions
- A series of noise-resistant techniques have been presented for dealing with noisy problems in multi-robot systems

Additional Literature – Week 11

Books

- T. Balch and L. E. Parker (Eds.), *Robot teams: From diversity to polymorphism.* Natick, MA: A K Peters.

Papers

- Zhang Y., Antonsson E. K., and Martinoli A., "Evolutionary Engineering Design Synthesis of On-Board Traffic Monitoring Sensors". *Research in Engineering Design*, 19(2-3): 113-125, 2008.
- Dorigo M., Trianni V., Sahin E., Groß R., Labella T., Nolfi S., Baldassare G., Deneubourg J.-L., Mondada F., Floreano D., and Gambardella L.. "Evolving Self-organising Behaviours for a Swarm-bot". *Autonomous Robots*, **17**:223–245, 2004
- Baldassarre G., Trianni V., Bonani M., Mondada F., Dorigo M., Nolfi S., "Self-Organised Coordinated Motion in Groups of Physically Connected Robots". *IEEE Transactions on Systems, Man, and Cybernetics*: *Part B*, **37**(1): 224-239, 2007.
- Murciano A. and Millán J. del R., "Specialization in Multi-Agent Systems Through Learning". *Biological Cybernetics*, **76**: 375-382, 1997.
- Mataric, M. J. "Learning in behavior-based multi-robot systems: Policies, models, and other agents". Special Issue on Multi-disciplinary studies of multi-agent learning, Ron Sun, editor, *Cognitive Systems Research*, **2**(1):81-93, 2001.
- I. Navarro, E. Di Mario, and A. Martinoli, "Distributed vs. Centralized Particle Swarm Optimization for Learning Flocking Behaviors," in *Proceedings of the European Conference on Artificial Life*, York, U.K., July 2015, pp. 302–309.