Particle Swarm Optimization for Multi-robot Systems

Distributed Intelligent Systems
25.11.2015

Zeynab Talebpour
Lab Structure

• Multi-robot PSO for obstacle avoidance:
  – Same fitness function as single-robot PSO
  – Differences in performance and evaluation time

• Budget allocation
  – Comparing standard PSO, PSO-pbest, PSO OCBA

• Multi-robot PSO for collaborative tasks:
  – Coordinated motion: move as far as possible while staying together
  – More difficult task than obstacle avoidance
  – New sources of uncertainties
Code Structure

Pso_sup.c

- Main()
  - Initialize world
  - Best=pso()
  - Evaluate best

- Calc_fitness()
  - Reposition robots randomly
  - Send candidate solutions to robots
  - Evaluate fitness
  - Return fitness

Pso.c

- Pso()
  - Initialize swarm
  - For each iteration
    - Move particles
    - Evaluate particles
  - Return best particle

Obs_con.c

- Main()
  - Initialize robot
  - Receive weights from supervisor
  - Run controller with weights
  - Send sensor data to supervisor
Noise-resistant PSO

• Setting NOISY=1 triggers two changes
  – Half the number of iterations
  – Revaluate performance for *lb*est (with flag EVOLVE_AVG)

• You need to implement the behavior for EVOLVE_AVG
  – Modified moving average (MMA) with *age* as the number of periods.
  – Remember to increase *age*. 
PSO-OCBA

- Optimal allocation of computation budget
- Very effective in the presence of noise
Notes and Clarifications

• Simulations take longer with complex tasks, read ahead and answer questions while the simulation runs.
• Performance evaluations have a high variance, you may need additional runs to establish clear trends
• You may also compare your results with your classmates

• Remember to fill the feedback form