

Distributed Intelligent Systems

Lab 7 Tutorial

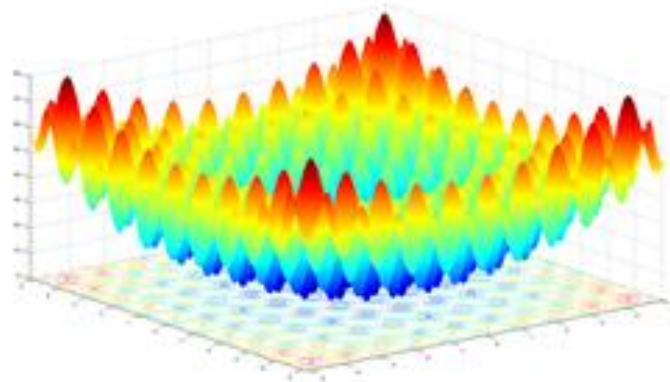
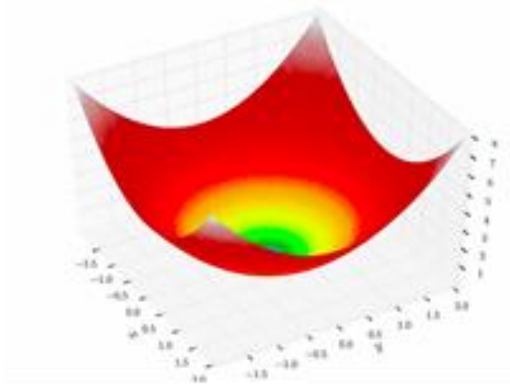
Faezeh Rahbar

Part 1: SwarmViz

- Software for PSO visualization
 - Implementation in c++
- Installation of Qt5
 - While waiting for installation, check out the code or the next part

Exploring PSO

- Run PSO on two benchmark functions (Sphere and Rastrigin functions) using SwarmViz

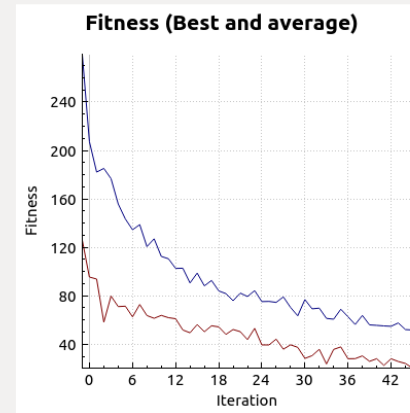
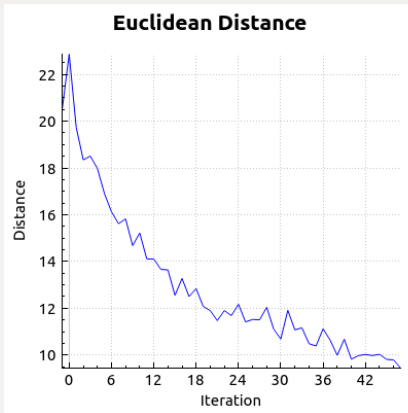
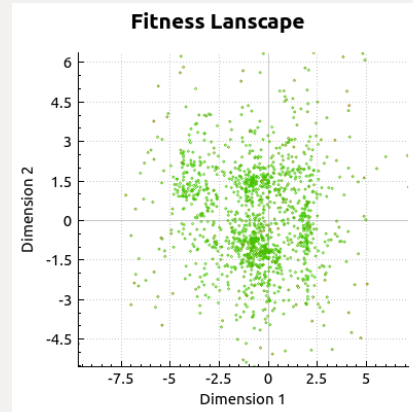


- Observe how swarm acts when varying parameters

SwarmViz

- Make sure you only have the indicated plots marked
- Fitness landscape plot
 - A history of all particles
 - Colors indicate fitness values
- Trajectory plots
 - Movement of particles
 - Previous positions can also be plotted

SwarmViz



Simulation Visualisation **Swarm** Files Plots

Benchmark function parameters

Fitness function: Sphere

Noise (sigma): 0.00

Dimension: 24

Swarm parameters

Particles: 30

Minimum: -5.12

Maximum: 5.11

Maximum velocity: 5.12

Inertia: 0.60

Max iterations: 1500

Local weight: 2.00

Neighbor weight: 2.00

Neighbor number: 2

PSO algorithm parameters

Noise resistance

Part 2 : PSO for Robotic Learning

- PSO with an Artificial Neural Network to do unsupervised robotic learning
- Design a fitness function for obstacle avoidance
 - Compare with the fitness proposed by Floreano and Mondada
- How is the performance affected by PSO parameter variations

Pso_sup.c

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

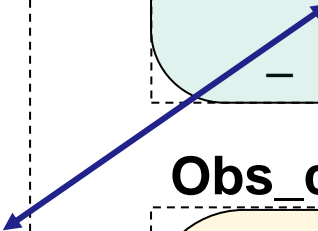
Pso.c

- 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

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



Notes

- The performances for robotic learning are printed in the console of Webots
- Please fill in the Feedback Forms on Moodle