Learning Machines Demo 1

Team Alpha: Mick IJzer, Leon Berberich, Justus Huebotter & Nikolay Polyanov

Agenda

- Problem definition
- Methodology
- Experimental setup
- Results
- Conclusions

Problem definition: Task 1

Obstacle avoidance using Robobo:

- **Crucial task** for a robot operating in any environment
- Robobo should be able to **detect objects** blocking it's path and **react** on the go
- Robobo also **learns** from it's experience
- Accomplished using **sensors** located in front and back of Robobo

Methodology

Different approaches tried:

- Hardcoded approach, using if statements (no learning)
- Q-learning (Q-table & TD3)
- Evolutionary algorithm (neural net & **Q-table**)

Problems faced

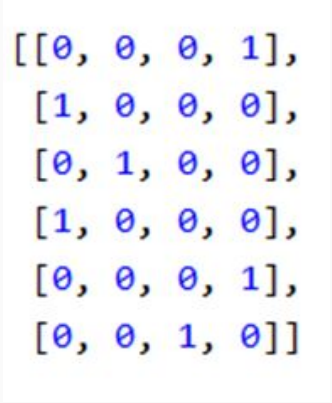
There were several challenges faced while working on the task:

- Python 2 & 3 confusion
- Q-Learning
- Evolutionary Algorithm with a Neural Net
- Experience scarcity

Methodology selected

Proceeded with a combination of an Evolutionary Algorithm and Q-Learning:

- Learning algorithm:
  - Evolutionary
  - Variation (Crossover & Mutation)
  - Selection (Parent & Survival Selection)
- Controller:
  - Q-table
  - Each row represents a state
  - Each column represents an action
  - Deterministic → Action *a* taken in some state *s* will always be the same for an individual



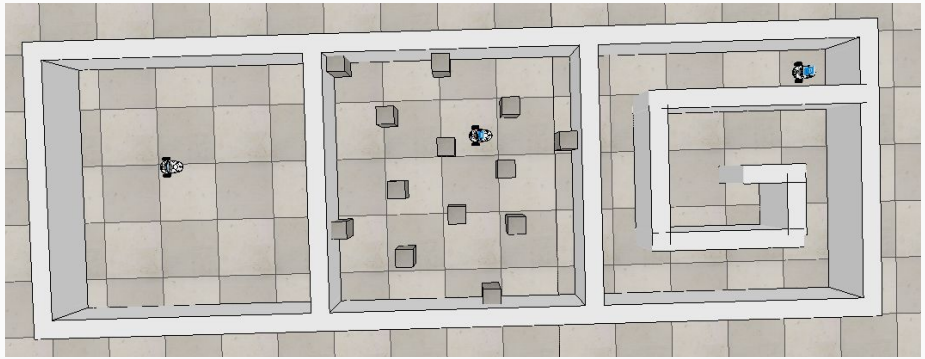
Evolution Parameters & Metrics

Phenotype	Behavior of the robot when in a state
Genotype	Q-Table (where every row is a state with q-values for the actions)
Mapping	Every state has 1 deterministic action
Fitness	-5 if collision, +5 moving forward, +0 turn (Accumulation over 50 timesteps)
Crossover	Uniform (Every row is a copy from either P1 or P2)
Mutation	Shuffle a row with probability mu (= 0.1)
General Settings	Parent selection: prob. Rank (2 parents) Survivor selection: prob. Rank (15 individuals) Initialization: random Termination: 10 generations or avg. fitness > 4

Experimental setup

3 different scenes used both in simulation and real arena:

- Empty scene
- **Obstacle scene**
- Maze scene



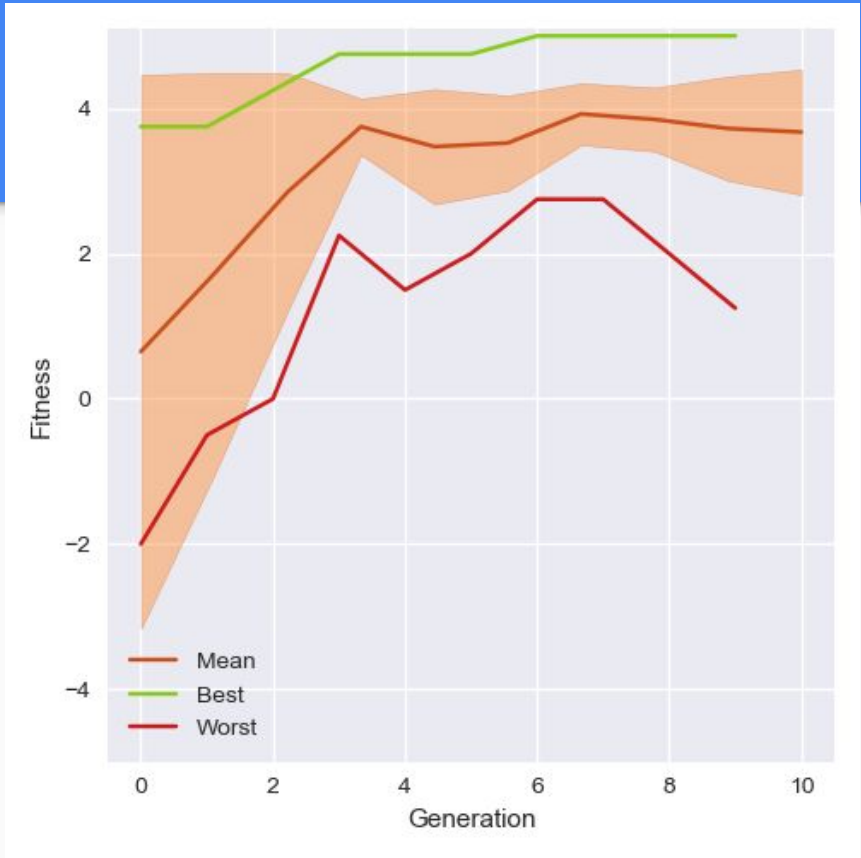
Results

Info:

- 10 generations
- 15 individuals
- -5 < fitness < 5

Plot shows:

- Mean fitness
- Variance of fitness
- Worst individual
- Best individual



Conclusions

- Evolutionary Approach gives a learning curve for the population
- Performance of best individual on the task is good
- Smoothness of the movements looks better (python 3)

Outlook:

- Extend q-table for more complicated tasks
- Try to add evolutionary parameters to the genotype

→ Don’t make it more complicated than it needs to be