Maastricht University

Department of Data Science and Knowledge Engineering

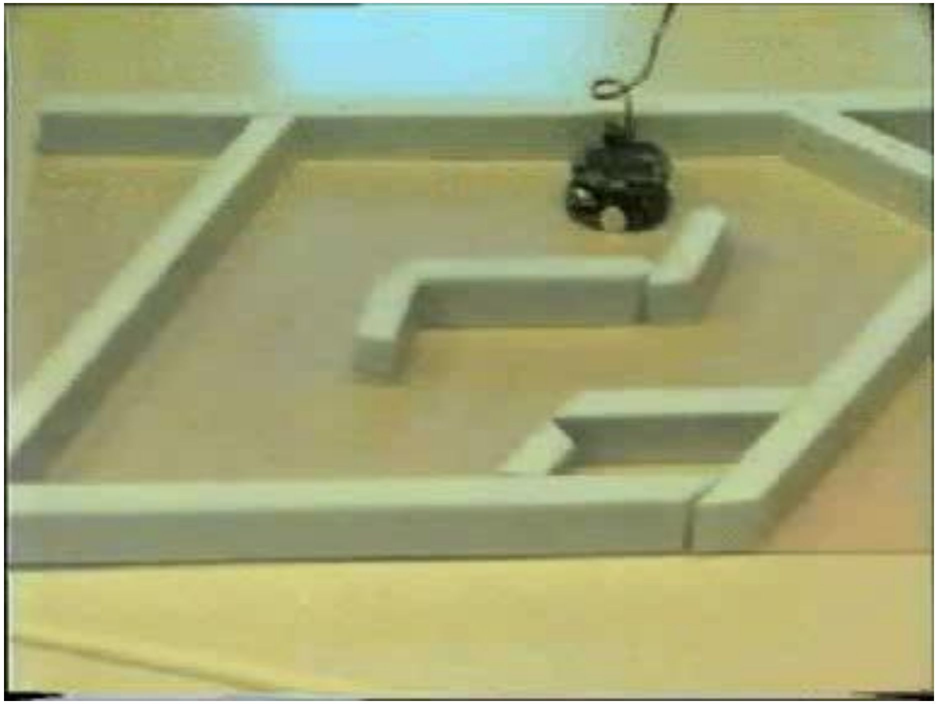
Autonomous Robotic Systems

Master Course

Assignment
Evolutionary Robotics

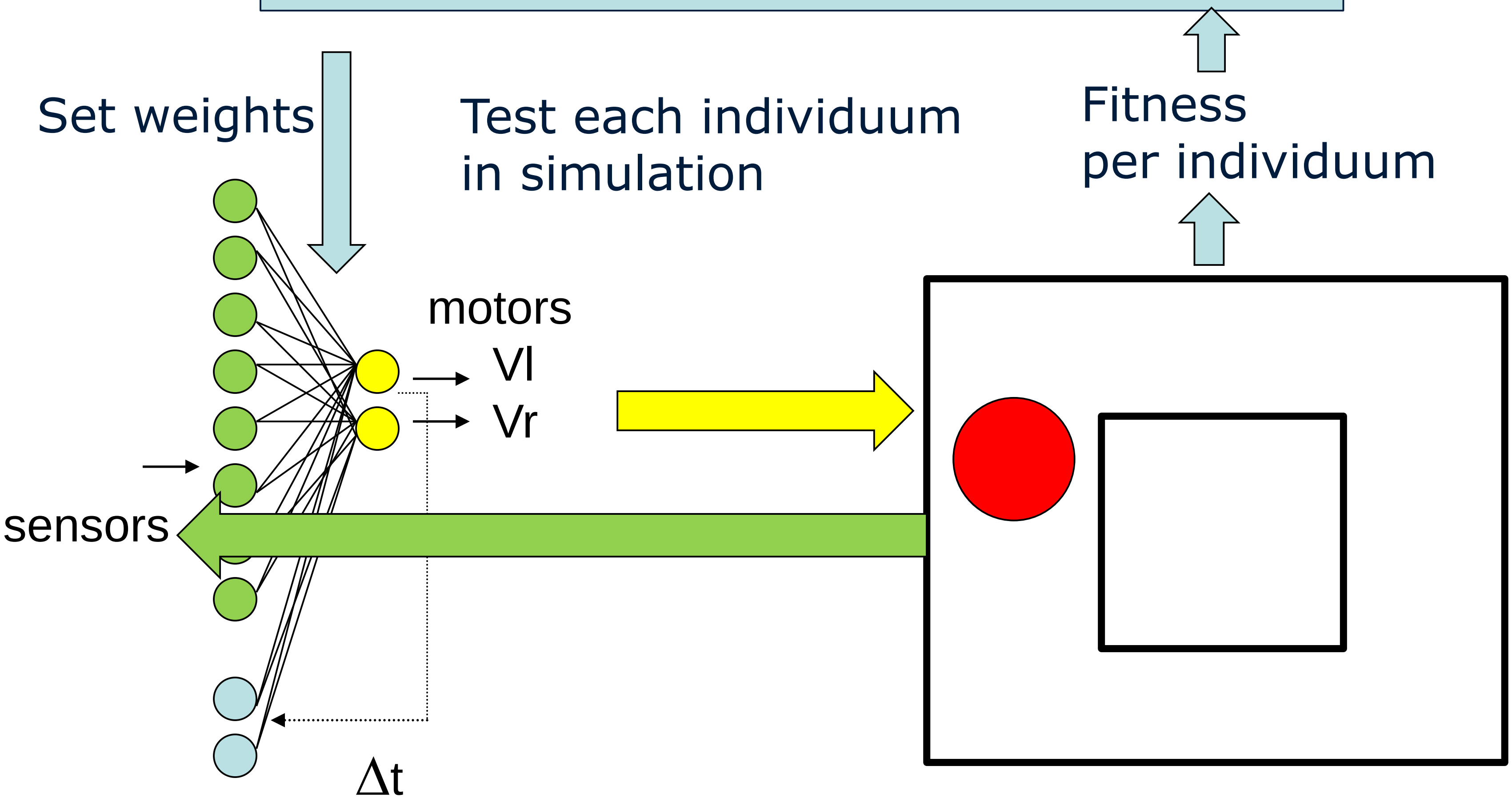
Goal: collision-free navigation of mobile robot

- Additional goal: cleaning robot: cover as much area as possible
- Use your mobile robot simulator
- Use ANN as controller
- Use EA to evolve weights of ANN (no back-propagation)



EA

Create new individuals
by selecting weights of ANN

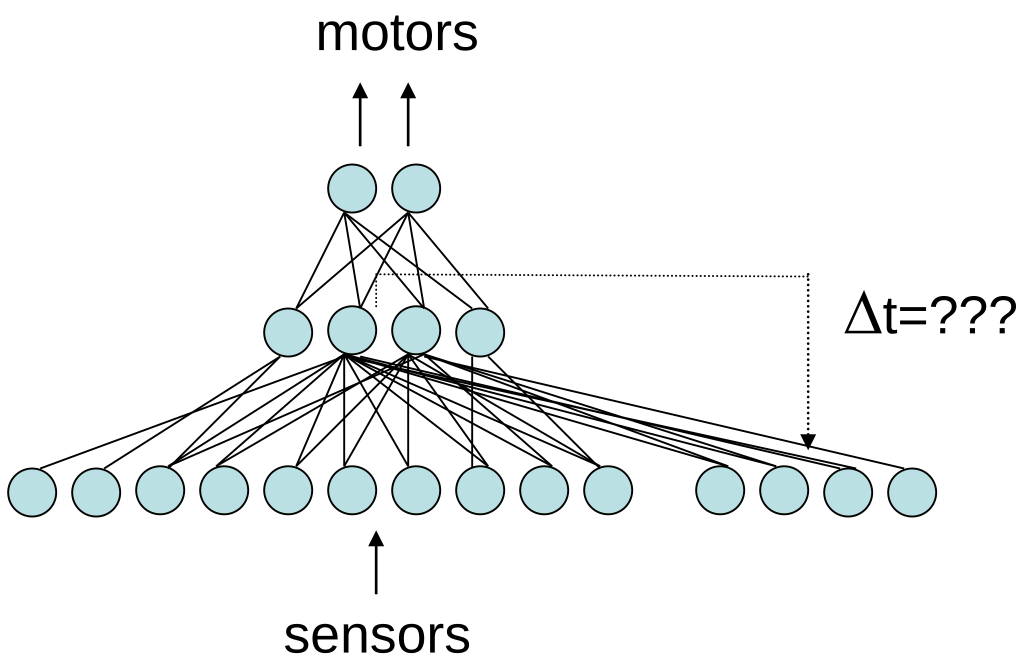


More instructions. Help your EA by shaping the fitness landscape

- Robot should clean as much area as fast as possible while avoiding collisions
- Design your own rooms
- Controller based on artificial neural network (use at least two layers with recurrent nodes)
- Robot with two wheels, ANN with two outputs – each output controls speed of one wheel
- 12 infrared distance sensors (30° distance) as input to ANN (it makes sense to use a max distance parameter)

ANN

- Need at least two layers.
- Use feedback to create memory
- Play with Δt (will depend on time step)



How to encode fitness criteria?

- Collision-free?
- Clean as much area as possible?
- How to weight both criteria?
- What about experiment?
  - Fix time for each experiment?
- Avoid that too many individuals receive zero fitness. Otherwise your EA gets stuck right from the beginning.

→ Requires some testing, but think first!!!

Make your life easier by shaping your fitness landscape/evaluation

- Cover as much area as possible: simulate dust, use removed dust as fitness

→Avoids robots that do not move enough

→Automatically covers speed criterion (if time for experiment is fixed)

→Motivates robots to move closer to walls (without collision)

Make your life easier by shaping your fitness landscape/evaluation

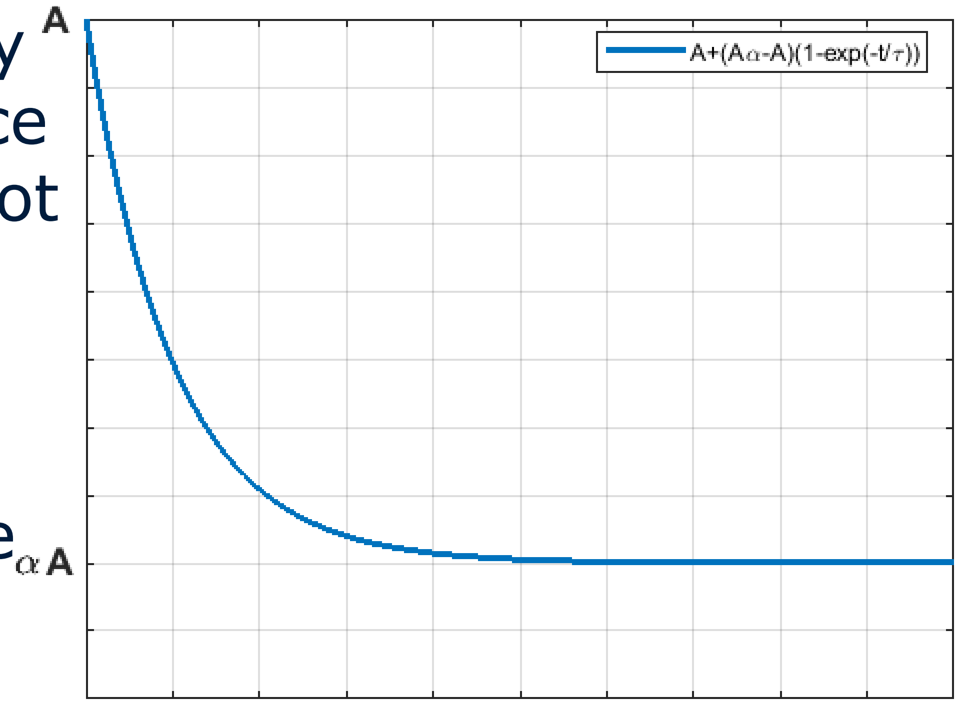
- Limit range of sensors

→ Makes sure that far away obstacles do not influence the behaviour of the robot

- Shape feedback of sensors:

→ Allows ANN to "understand" importance of distances

Sensor output



Actual distance

Advice on test rooms

- To achieve robustness you might want to evolve your robot using different training rooms
- Just because you think that a room is simple, it might not be for a robot (see narrow passages, open spaces etc.)
- Test your final robots on these training rooms plus on rooms that they have not seen before to allow for a strong discussion

Hand in

- Documented code (Python, C++, C, Java, Matlab)
  - Make sure that each group member codes something, add names to code (who did what?)
  - Upload plain files to Student portal or zip archive
- Final demonstration of evolved behaviour in class in three weeks (Carnival in between!!!)
- Intermediate demonstration EA on benchmark functions (same demos as for PSO, show updates of individuals) in two weeks (= after Carnival)
- Short research paper with abstract, methods (describe your EA & control), experiments, results, and discussion.

Bring printout of paper report to class!

Research paper

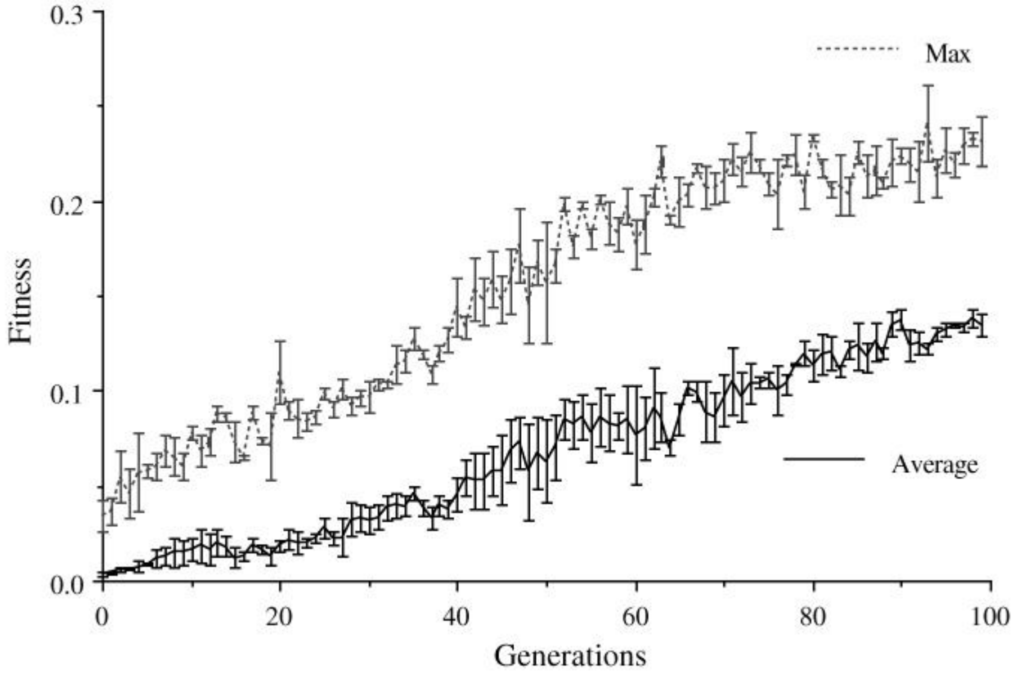
- Explains **all** steps of your EA so that someone could reimplement your code. Do not forget to explain experiments (concentrate on successful ones) → typical exam question
- Required graphs:
  - Show evolution of fitness (max + average)
  - Show evolution of diversity
  - Show evolved trajectory of robot
- Discuss results (**explain what is happening and why it is happening**)
- Do not forget to put all student names on report
- Put author names to each section

Report details (be precise! be consise!)

- Control (e.g. drawing of ANN)
- Morphology (here simulation)
- Genetic representation -> exact genome
- Population size -> exact number
- Initialization (how do you initialize genomes?) -> exact distributions and motivation
- Exact fitness function + motivation
- Evaluation procedure + motivation
  (experiments, how do you measure?)
- Selection and reproduction (be precise)
- Crossover & mutation (be precise)
- Stop criterion
- Data analysis (figures! describe results, explain, and discuss! Analyse possible causes for your results. This section might take more than a page of text)

Monitoring Performance

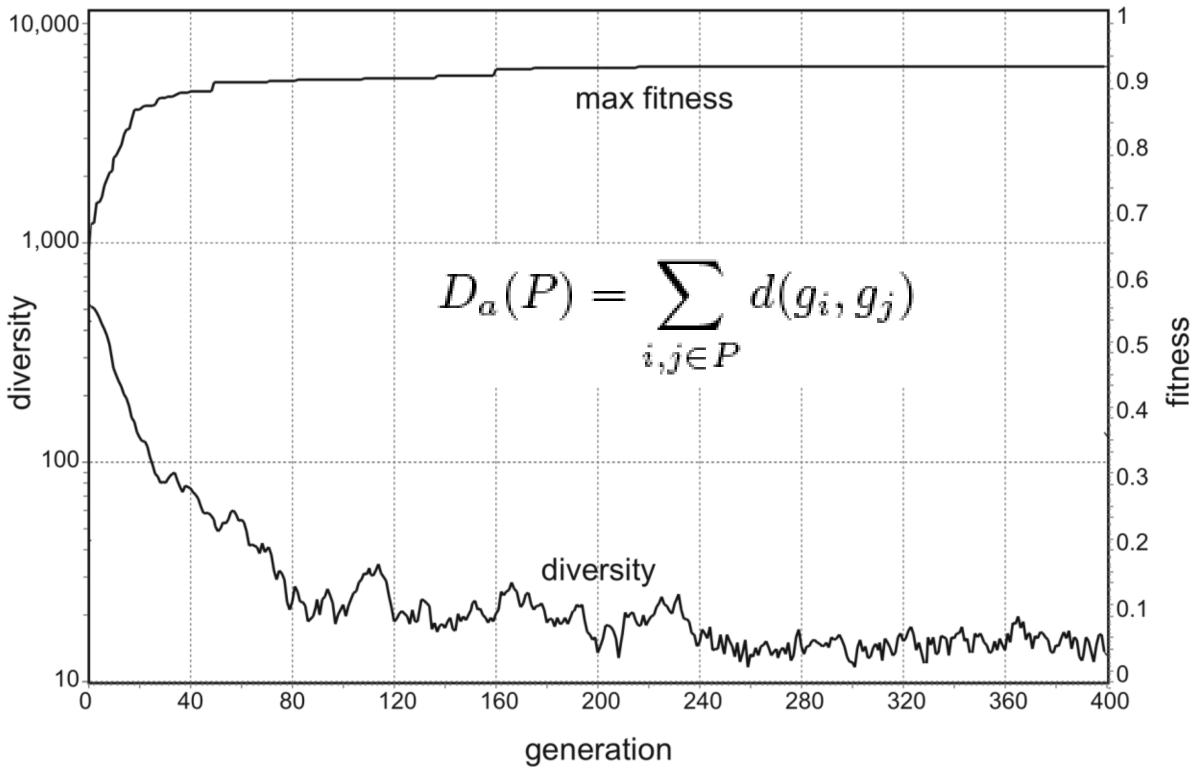
Track best and population average fitness of each generation
Multiple runs are necessary: plot average data and standard error



- Fitness graphs are meaningful only if the problem is stationary!
- Stagnation of fitness function may mean best solution found **or premature convergence**

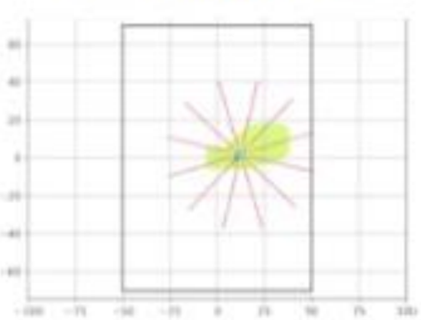
Measuring Diversity

Diversity tells whether the population has potential for further evolution
Measures of diversity depend on genetic representation
E.g., for binary and real valued, use sum of Euclidean or Hamming distances

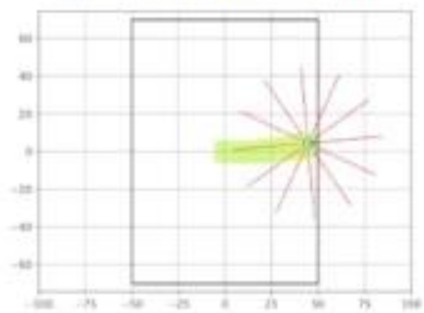


Possible final result

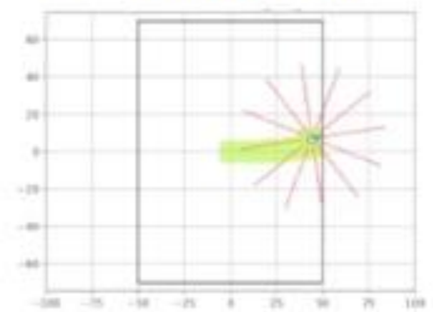
Generation 1



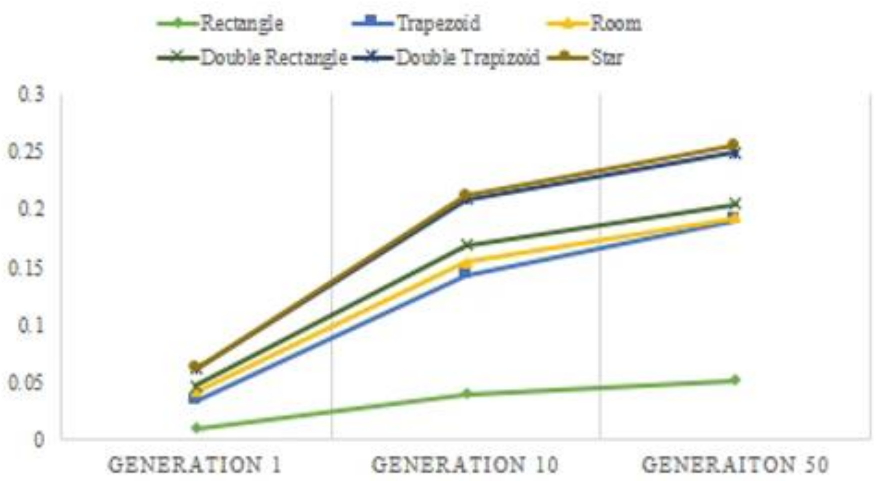
Generation 10

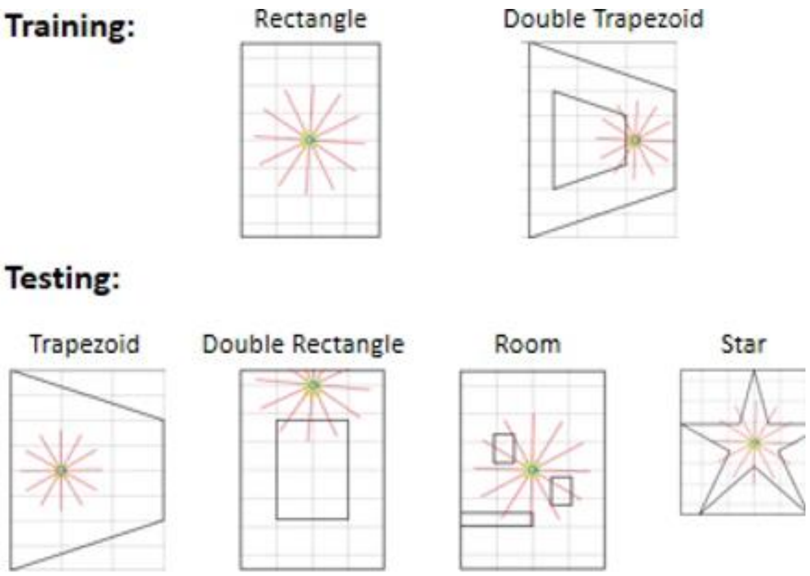


Generation 50



Possible final result





Groups

- Groups of ideally 3 students (not more than 3, not less than 2)
- Do not forget to register again to the student portal. Use same group number as before.
- If you wish to switch groups (for a good reason) tell me first

Plagiarism

- This is a group assignment
- Help other members of your group
- Do not copy and hand in code or reports from other groups (or other parties)
- **You must write code for EA, ANN, evaluations yourself (no external libraries allowed)!!!**
- You **can** use the simulator of another group with a proper motion/sensor/collision model
- Write your own software

Write simple software

- No need to use (a lot of) objects
- Use functions
- Do not distribute code over too many files
- Avoid complex constructions and data structures

Now

Do not run away!

- Register for new assignment on student portal to new group with same group number as last time
- Start working on assignment: make plans
- There is plenty of reason to discuss!
- Now we check simulators. **Have simulator running so we do not loose time!**

Do not run away!