Autonomous Intelligent Systems,
Institute for Computer Science VI, University of Bonn

Dr. N. Goerke
Friedrich-Hirzebruch-Allee 8, 53115 Bonn, Tel: (0228) 73-4167
E-Mail: goerke@ais.uni-bonn.de
www.ais.uni-bonn.de

Exercises for Artificial Life (MA-INF 4201), SS24

Exercises sheet 11, till: Mon 30. June, 2025

23.6.2025

Assignment 72 (2 Points)

Explain the idea and all main parts of the SMPA architecture (one sentence each) and draw a diagram to depict the SMPA structure.

Assignment 73 (4 Points)

A vehicle (two sensors and two motors) shall have the behavior to position itself facing an object in the working distance of d_w.
The working distance shall be maintained even in the two cases, that the object is moving away and that the object moves towards the vehicle.
Describe the necessary structure of the vehicle including the Sensor-Motor mapping characteristics in brief words and by using a sketch and a diagram.

Assignment 74 (2 Points)

Describe a situation where a vehicle with a type 3-a Braitenberg behavior is more favorable than a 2-b behavior.

Assignment 75 (3 Points)

Braitenberg type 3-b vehicles tend to have problems in corners.
Explain what the problems are, and describe in detail at least 2 stategies to circumvent this unwanted effect.

Assignment 76 (4 Points)

The inner control structure of a robot has been designed as a table $\mathbf{T}$, mapping from a set of $N = 8$ discrete sensory distance values $\mathbf{S} = (s_1, ..., s_8)$, to a set of 2 discrete motor values denoted with $\mathbf{M} = (m_l, m_r)$.
The set of discrete sensor values is $s_n = \{0.0, 2.0, 8.0, 64.0\}$, and the set of discrete motor values is $m_i = \{-0.5, -0.2, -0.1, 0.0, +0.1, +0.2, +0.5, +0.75, +1.0\}$.

Design and describe a genome and a fitness function for an EA to optimize the table $\mathbf{T}$ with respect to the task of obstacle avoiding movements.
Don't forget to define the objective and/or fitness function.

Programming Assignment: F (5 Points, due date **Mon 30.6.2025**)

Write a small robot simulator in Python.
The world of the robot is a rectangular world of 50x50 grid-cells. Each grid-cell is either 0 =*free space*, or 1 =*wall*. The grid-world shall be read in from an ASCII-file where the positions of all occupied cells can be found as one (X, Y)-position per line `PA-F-SS25-4201.txt`; the first line of this file contains a comment, and can be omitted.

You can visualize the world with the following command within *gnuplot*:

```
plot 'PA-F-SS25-4201.txt' using 1:2 with points
```

The robot has a size of 1 cell, it has a position (X, Y) within the world and an orientation a pointing to one of the eight surrounding cells. The robot can sense the situation of the eight surrounding cells, and in one simulation step can move one cell into the heading a .

Implement either a wall following or a Braitenberg type 3b behaviour.
Start your robot at position $(48, 17)$ heading south, and calculate the resulting movements for at least 200 steps.

It is not intended to programm a visual output.
To document the movement of the robot it is sufficient to print out the pose (X, Y, a) of the robot in every timestep into a file named `PA-F-robot.path`.

A *gnuplot* readable format would be; one pose per line, values separated by blanks, in the order X Y a
The following *gnuplot* command could then visualize the world and the journey of the robot:

```
plot 'PA-F-SS25-4201.txt' u 1:2 w p, 'PA-F-robot.path' u 1:2 w p
```

Other implementations for visualization are welcome but not requested.