University of Konstanz Winter Semester, 2025
Cyber-physical Systems Group Assignment 2
Group member name(s): Benedict Finsterwalder, Florian Shabani
Group member UID(s): 1564941, 1587539

INF-16580: Evolutionary Robotics

Introduction

In this problem we had to create a simulation of a simple obstacle avoidance behavior. We created a proximity sensor which checks our surroundings for obstacles using a ray-casting-like method. The sensor should be able to get the distance to the next obstacle (Task 2.1). Then we should implement the correct controller such that we avoid obstacles using the proximity sensor by turning before hitting a wall (Task 2.2). Using this sensor and controller, the robot should successfully avoid the walls. When it does not sense a wall (or the sensor value is below a threshold, the robot does a random walk.

Solution Method

We completed tasks 1.1 and 1.2. Our implementation can be seen on Github in the "assignment_2" branch.

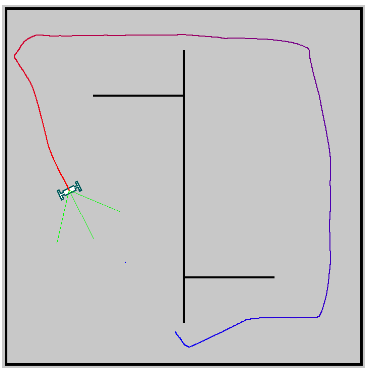
First we created the proximity sensor which is similar to the bumper sensor. For each direction (40°, 0° and -40°) we created a beam which we then test for collisions using the clip-function. Here one of the problems was that the id of the robot's body was not correct. Therefore we just used the objects from the static rect list and nothing else. If an object is hit, we calculate the distance and return the corresponding sensor values.
These sensor values are then used by the controller to implement the intended behavior. If we sense a wall, we want to avoid it. This behavior is implemented by testing four possibilities:

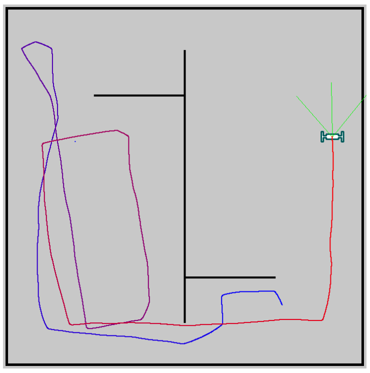
- All sensors return a value above a certain threshold (e.g. 0.6) or one sensor value is above a higher threshold (e.g. 0.8) $\rightarrow$ turn left
- The center sensor detects something $\rightarrow$ turn in the direction of the lower sensor value (left vs right) or random
- The left sensor is above a threshold $\rightarrow$ turn right
- The right sensor is above a threshold $\rightarrow$ turn left
- else we just do a random walk

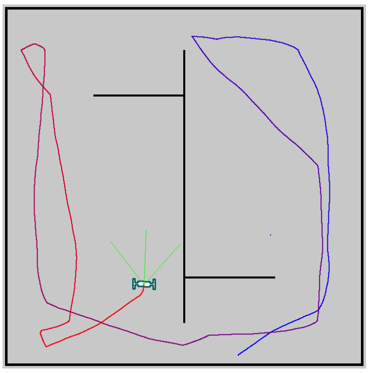
Results [1.75 pages]

We tested the robot for 2000 time steps (Picture 1) and for 4000 time steps (Pictures 2 and 3). The behavior of the robot was always the same (which is of no surprise).

Additionally we tested different thresholds for the sensor values. In this case there was one big difference. The space between the walls isn't always big enough for the robot to fit in-between i.e. if the threshold is set to 0.5 the robot will turn around when approaching the gap. This difference can be seen in Picture 1 (Threshold 0.6) and Picture 3 (Threshold 0.8).


Picture 1


Picture 2


Picture 3

Conclusion

We were able to effectively emulate the behavior of collision avoidance. With relatively simple programming and logic we have created an agent that seems to be able to avoid obstacles rather well.