Random Walk - Example solution

Vector-based reactive solution

For this exercise, we provide two example solutions. The file `obstacleAvoidance_vect.lua` contains an example, where the robot computes a vector from all observed obstacles and turns away from it. Since a closer object will result in a higher reading of the proximity sensor, we can use the value of the proximity sensor as the length of the vector pointing towards the obstacle. By summing up all vectors, the components of close obstacles (large proximity readings) are preserved, while farther obstacles (with small proximity readings) get averaged out.

If the length of the remaining vector is longer than a certain threshold, then the robot moves slowly forward, while turning away from the direction that the obstacle vector is pointing too.

This solutions is not without drawbacks, however. For example, the summation of all vectors means that two obstacles on opposite sides of the robot cancel each other out. Additionally, the robot also considers obstacles in the back and is trying to turn away from them, even if they are not in the direction of motion.

Sense-Think-Act solution

In the file `obstacleAvoidance_sta.lua`, we provide a second example to solve the exercise, based on the Sense-Think-Act pattern. The Sense-Think-Act paradigm is commonly used in robotics and refers to a way to structure control software. It is the general case of the Sense-Plan-Act paradigm often used in deliberative robotics, but also encompasses reactive control software.

In the *Sense* step, the robots reads its sensors and processes the sensor data. In the example, the robot checks the four sensors on the front left and the four sensors on the front right for any close obstacle.

In the *Think* step, the robots make decision of the action to take. In the example, this means that if a robot detected an obstacle, it chooses a random number of time steps and a random direction to turn to. It will then turn for the next control steps into the chosen direction, without considering anymore the new sensor readings.

In the *Act* step, the robots take the chosen action. In the example, the robot checks if it is currently avoiding an obstacle. If so, then the robot turns on the spot into the chosen direction. If the Think step did not decide to do obstacle avoidance, then the robot moves straight forward.