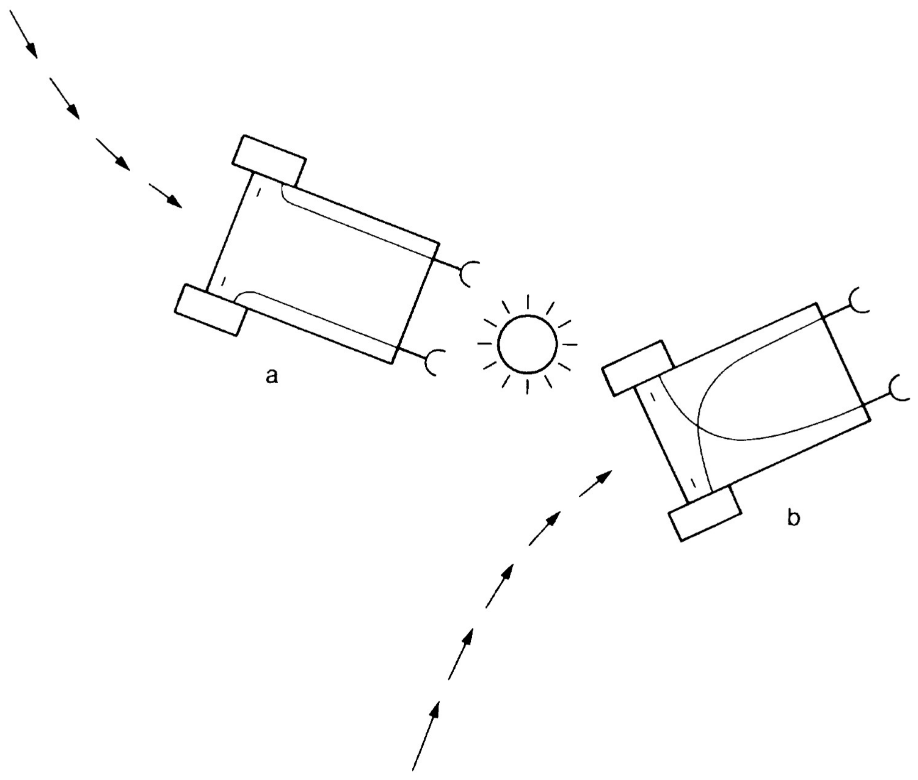
Gürtler Fabienne 12-935-508

Robotics Series 3

Exercise 1. Explorer and advanced lover

Theory: Braitenberg describes both the EXPLORER and the LOVE are vehicles that will slow down in the presence of a strong stimulus. Approaching the source vehicle *a* will orient towards it since the sensor nearer to the source will slow down the motor on the same side, producing a turn toward that side. This vehicle *loves* the source and will stay close to the source. We call this the LOVE behavior.



a: LOVE   b: EXPLORER

The vehicle with the crossed connections, vehicle *b*, will turn away from source for analogous reasons. This EXPLORER likes the source, but keeps an eye out for other sources.

The ADVANCED LOVER is a combination of both behaviors. It will behave like the LOVE vehicle until it comes to a certain proximity to the source. Then it changes its behavior and now acts like an EXPLORER. But once it is away from the source it will again try to go towards it. So the ADVANCED LOVER follows the source, but always keeps a certain distance.

Gürtler Fabienne 12-935-508

Aseba Implementation

Values: *proxRight = (8*prox[0]+4*prox[1]+2*prox[2])/14*
*proxLeft = (8*prox[7]+4*prox[6]+2*prox[5])/14*

These are the calculations for *proxRight/proxLeft* for the EXPLORER. Sensors in the front have more weight than those on the sides and those in the back are even neglected as a whole. Since e-puck always moves forward it is more important whats in front of it than on the side or back. So if the obstacle is right in front of the e-puck it will make quite a turn, while if the obstacle is on the side it will only swerve slightly to avoid it.

*proxRight = (6*prox[0]+4*prox[1]+3*prox[2])/13*
*proxLeft = (6*prox[7]+4*prox[6]+3*prox[5])/13*

The ADVANCED LOVER calculates *proxRight/proxLeft* very similarly, except the values at the side are given a little more weight than before. Since the e-puck follows the obstacle it is much more likely to encounter corners, so the sensors on the side are more important than with the EXPLORER.

Code: I generally followed the theory in the book, more than the given template. By the books logic the implementation of the behaviors would look something like this.

```
    EXPLORER
speed.left = S_INIT – ds_right
speed.right = S_INIT – ds_left
```

```
    ADVANCED LOVER
if proxSingle > P_THRESH_ADV then
    speed.left = S_INIT – ds_right
    speed.right = S_INIT – ds_left
else      # go to source
    speed.left = S_INIT – ds_left
    speed.right = S_INIT - ds_right
```

As explained in the theory if vehicle should turn away from source, the proximity sensors are coupled with the opposite motor. If e-puck is in turn supposed to go toward the source, a straight coupling of sensor to motor will accomplish this behavior.

Problems and Solutions

I've had to add in quite a bit of code to make sure e-puck's don't get stuck in either simulation or real arena.

```
if proxSingle > P_THRESH_ADV2 then
    speed.left = S_INIT
    speed.right = -S_INIT
```

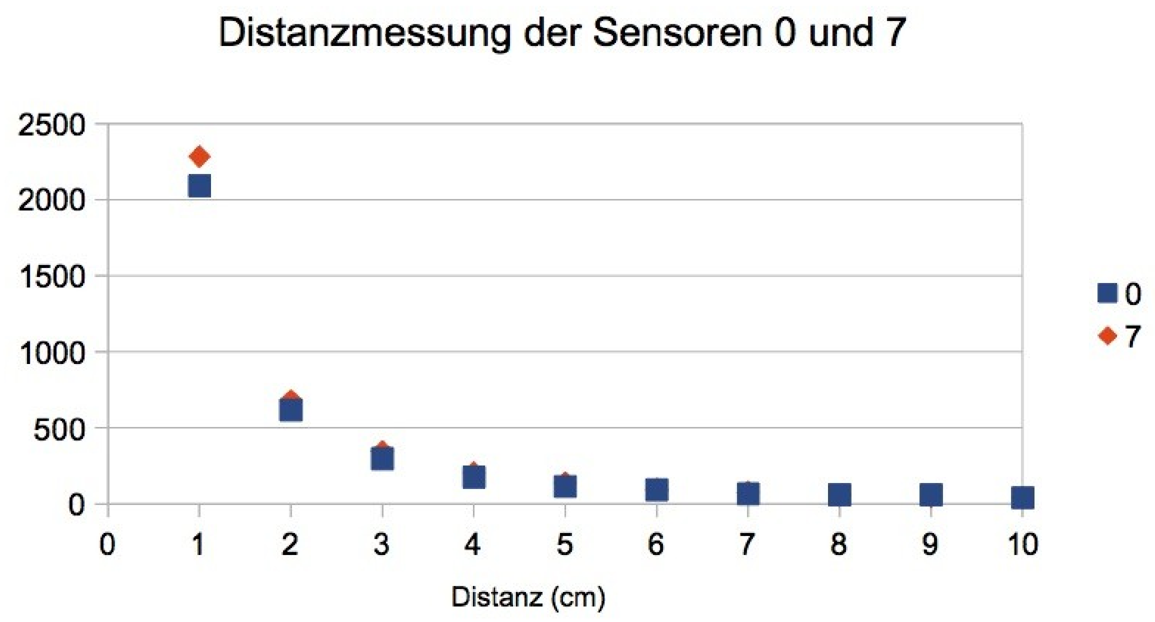
```
elseif proxSingle < 0 then
    speed.left = S_INIT
    speed.right = -S_INIT
```

This part was added for the case where e-puck gets itself stuck somewhere. It can mostly get stuck in corners, since then *proxLeft* and *proxRight* are almost identical. If both proximities are close to each other, it will hardly turn since both wheels are slowed down by almost the same value. So if the e-puck somehow gets itself in a situation like this it simply turns right by default.

For some reason I sometimes got negative proximity values, which should not be possible in theory. My guess is that if it is stuck somewhere and very close to obstacle the proximity value gets bigger than its range and becomes negative. If that happens, e-puck turns right.

Gürtler Fabienne 12-935-508

Exercise 2 Proximity Sensors Value



We put a obstacle in front of the e-puck and wrote down the values for proximity[0] and proximity[7] for each cm from 1 to 10. There are some differences between sensor 0 and 7 but the further away the source, the less it is noticeable.
The proximity sensors decrease potentially the further away the source is. Where at 1 cm the proximity sensor measures over 2000, at 2 cm it's already decreased to something over 500.
The function is roughly: $f(d) = 2000*x^{-1.7}$