Signals, instruments & systems

E-puck line following project

MIAZZA Raphaël & PIGNERES Marie

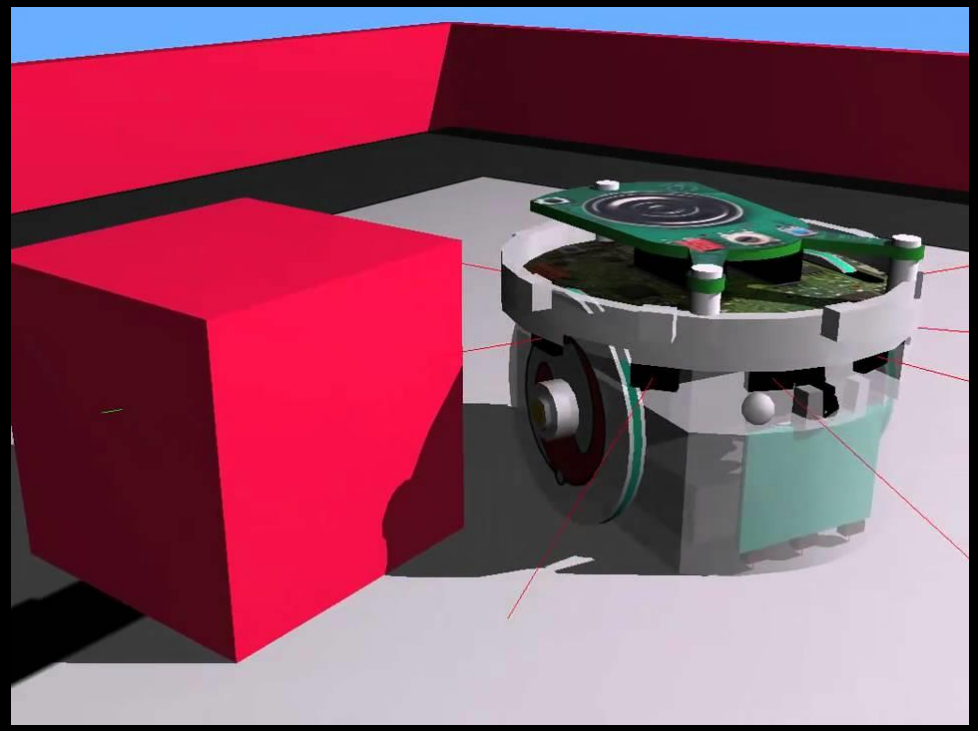
Introduction

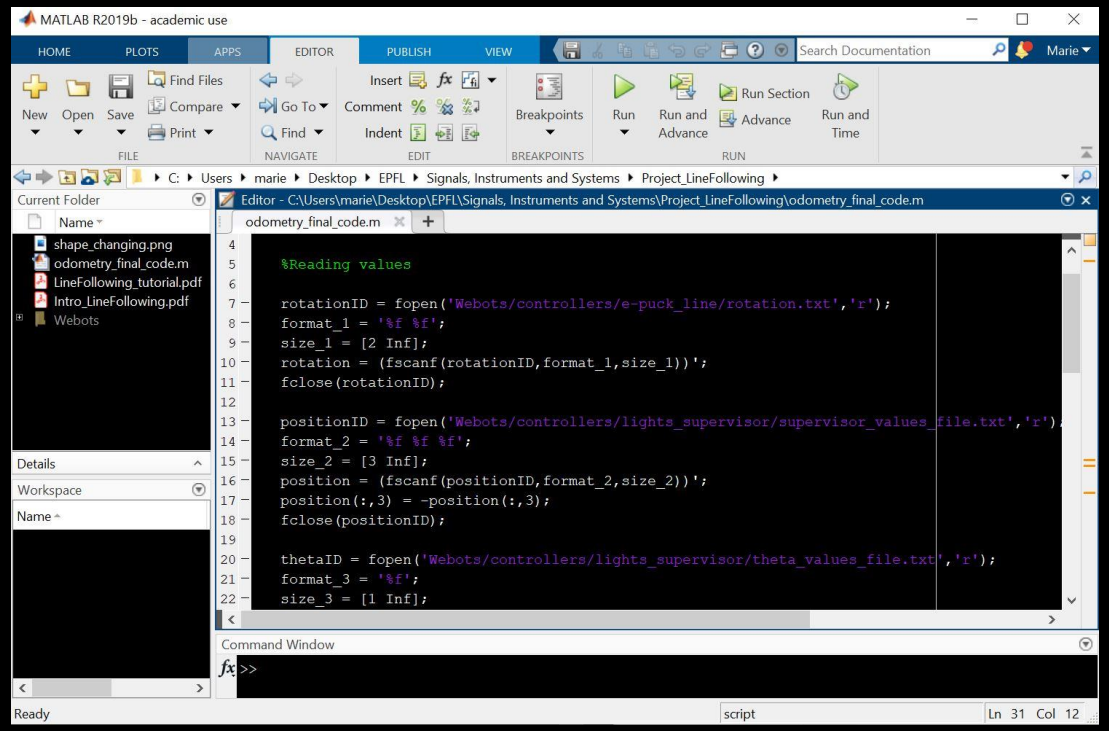
Goals of this project:

- Perform line following
- Reacquire line after losing it
- Avoid obstacle on top of line following
- Implement odometry

Introduction

Mediums : Webots & Matlab





Methods

E-puck devices used:

- Camera
  Color camera with 52x39 resolution
- Proximity sensors
  8 infra-red sensors measuring ambient light and proximity of obstacles in a 4 cm range
- Motors
  2 stepper motors with 20 steps per revolution and a 50:1 reduction gear

Methods

General structure of e-puck's behavior assignment

```
switch(obstacle)
   {case true: //obstacle detected
     obstacle avoidance code
    break;

    case false: //no obstacle detected
     if (line detected) {//line following
      line following code}
     else if(line lost, test_dead_end = 1
         and no far obstacle) {//turn around
         when meeting dead end
      turn around when dead end code}
     else { // e-puck is lost somewhere in
         the world
      keep going ;}
    break;}
```

Methods

Odometry

1. Export data from Webots into Matlab
2. Forward Euler's method -> rotational speed of each wheel
3. Forward kinematic model -> linear speed
4. Integration of speed -> position

Experiments

Line following and obstacle avoidance

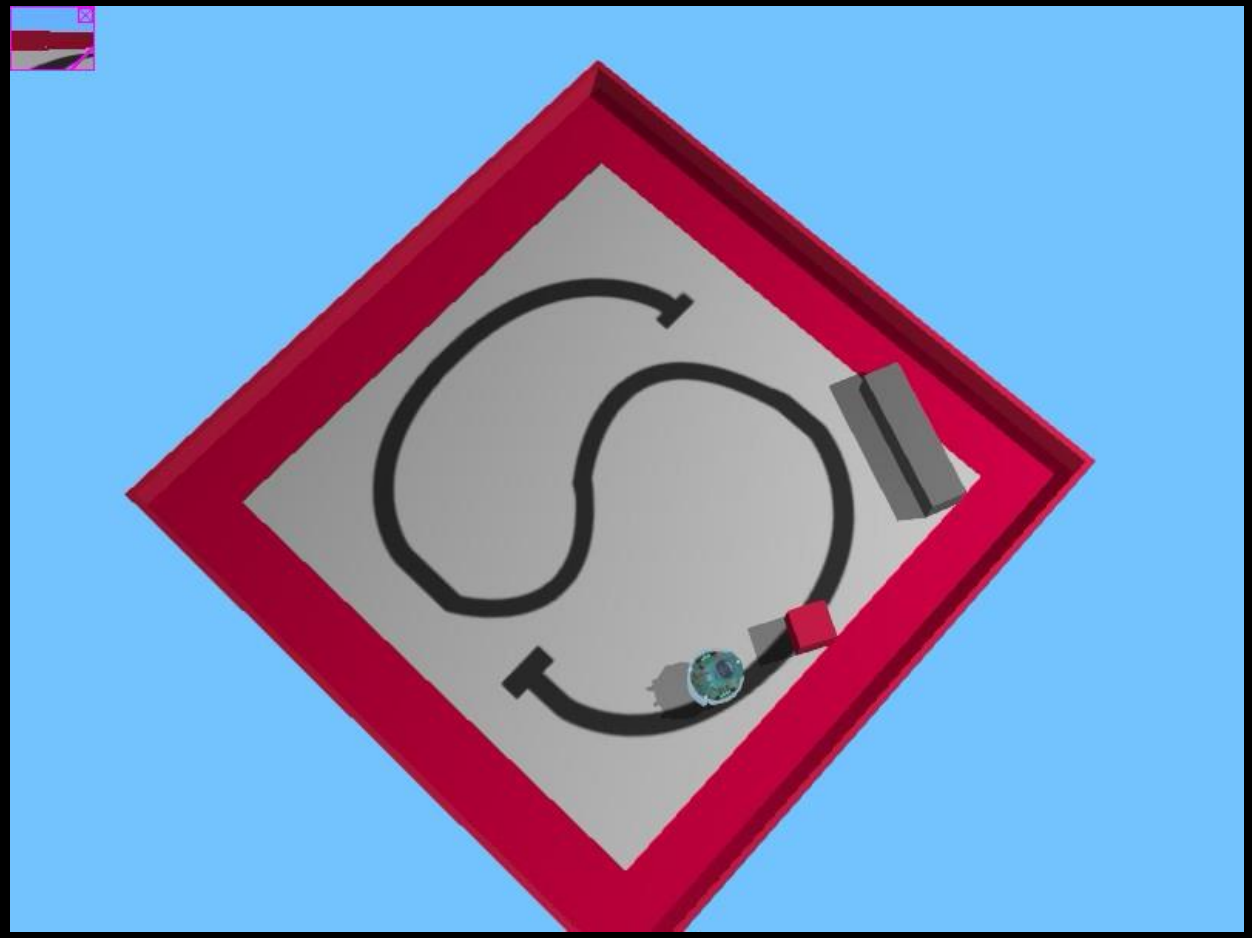
- Basic obstacle avoidance → Basic world
- Line following and obstacle avoidance → Oval line
- Dead end detection and final implemetation of the code → Funky line

Odometry

- While following the line → Oval line
- While performing U-turns and avoiding obstacles → Funky line

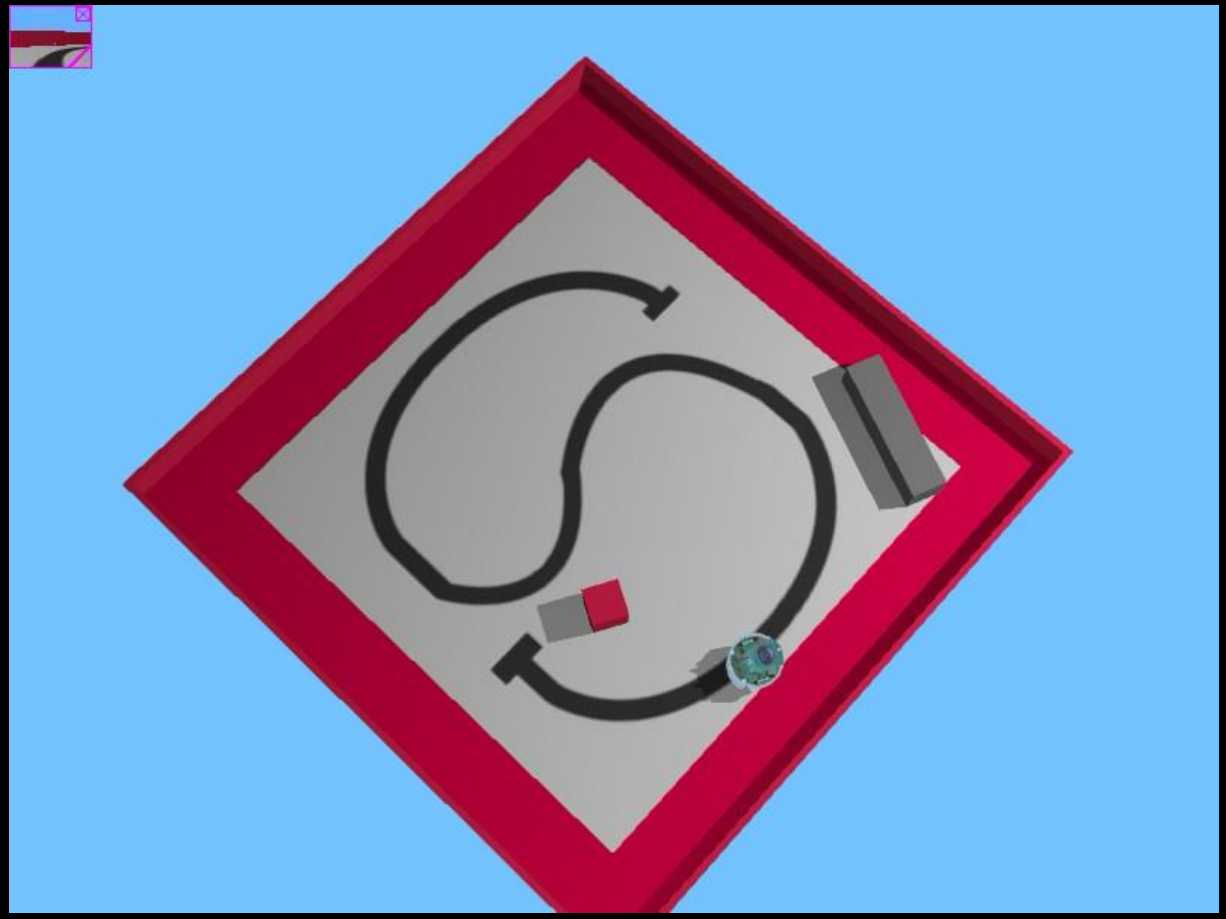
Results

Line following and obstacle avoidance



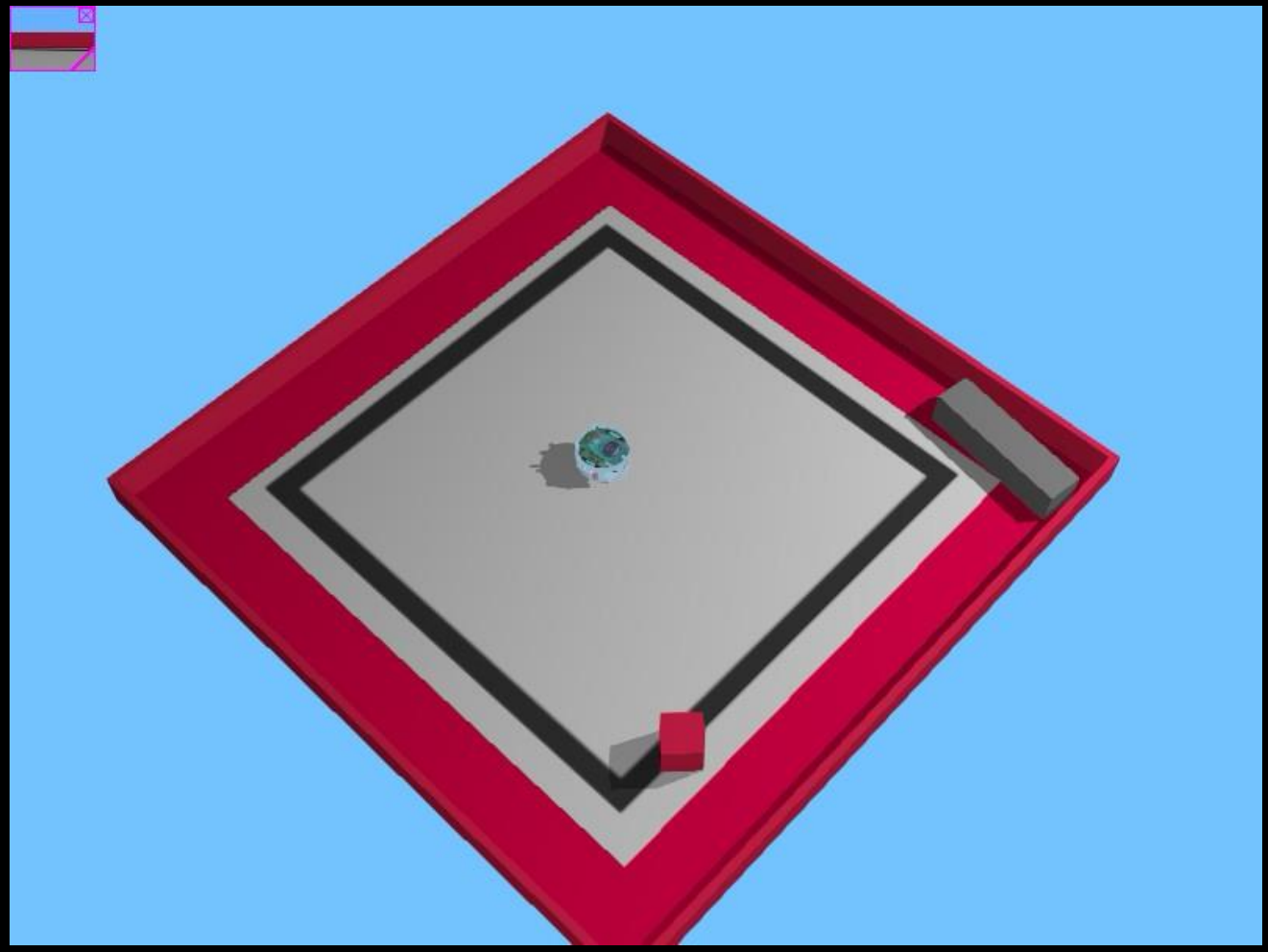
Results

Problems that may occur for dead end detection



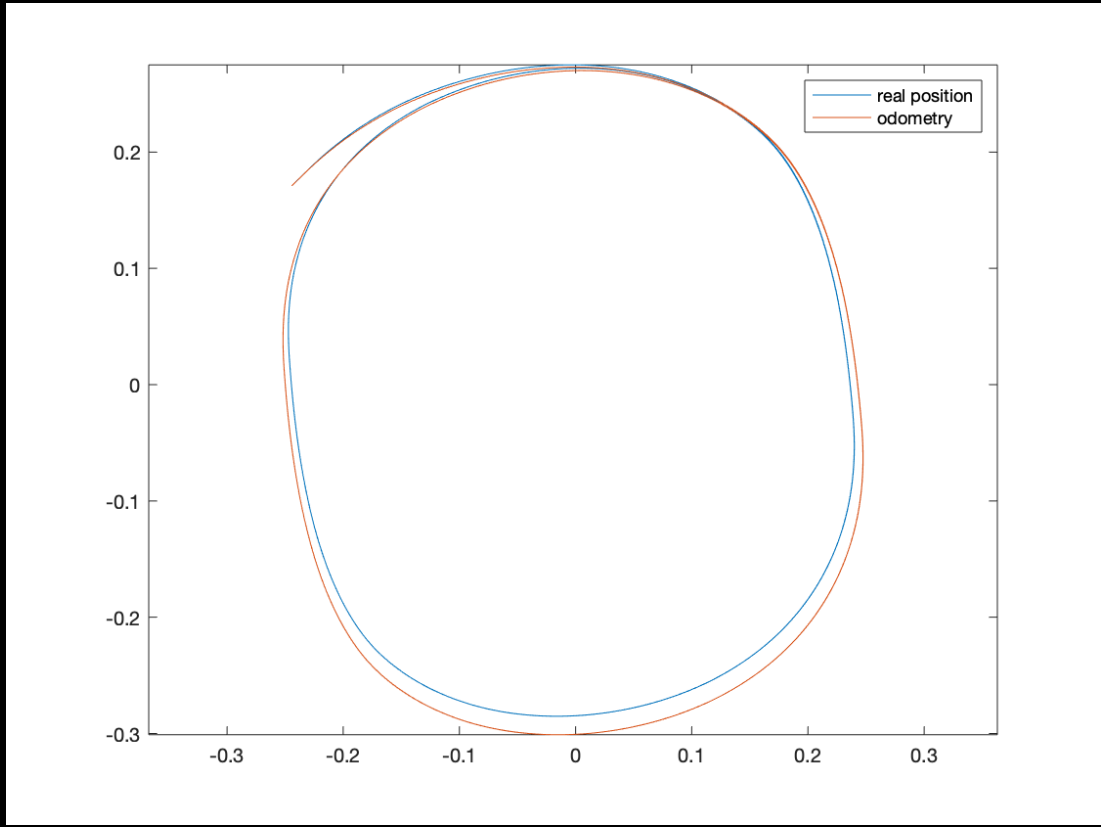
Results

Problems that may occur when crossing a line perpendicularly

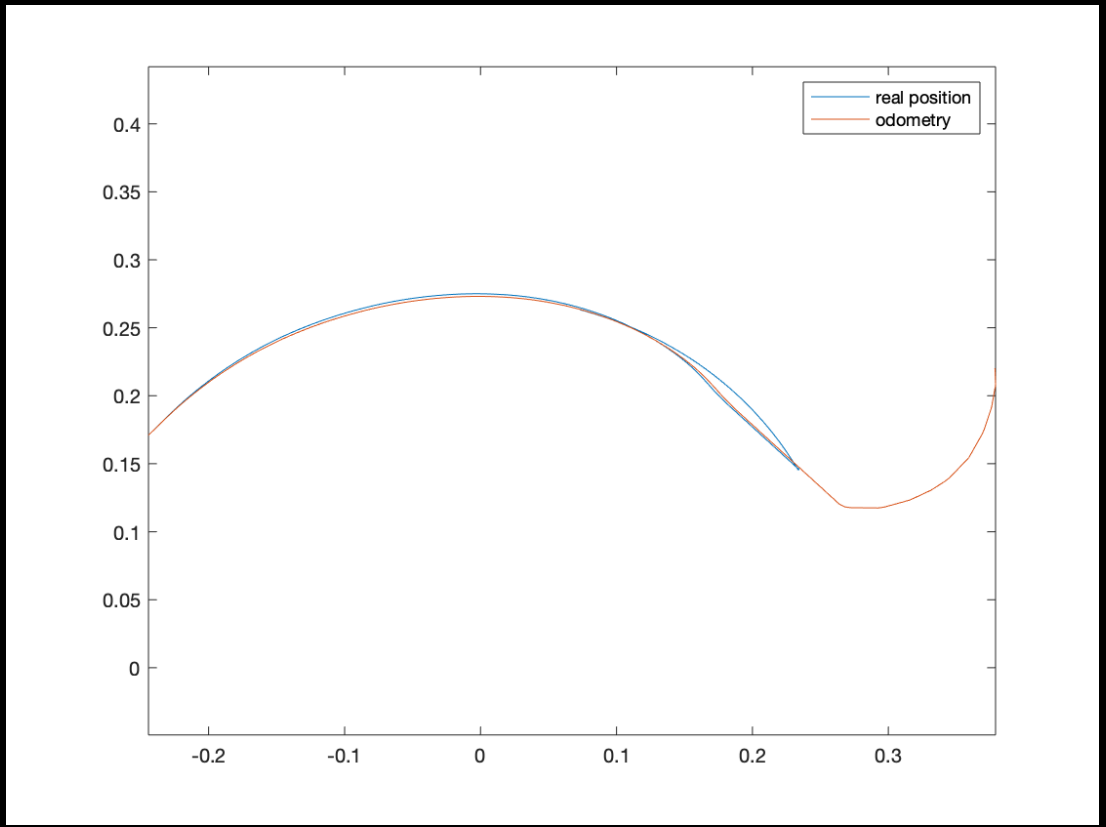


Results

Odometry



Trajectory comparaison while turning four times around the circuit



Trajectory comparaison while performing a U-turn

Conclusion

- Overall, each behavior worked pretty well, as well as the switching between each of them
- Odometry results weren't perfect but pretty close to the actual trajectory
- We got a deeper understanding of coding in the context of a robot's simulations