Line following using a camera with an e-puck robot

SIGNALS, INSTRUMENTS & SYSTEMS

Nouha Daghrir - Youssef Lafkihi - Mohamed Aymen Lamzouri

Overview

Introduction

Methods
- Sensors used
- Webots & real time simulations
- Main milestones & limitations

Experiments & results

Conclusion

Introduction

Objectives:

- For Webots simulation
  - Follow the line,
  - Find the line,
  - Avoid obstacles.
- For real time simulation, we have to make it work for a real e-puck robot

Methods

Sensors used

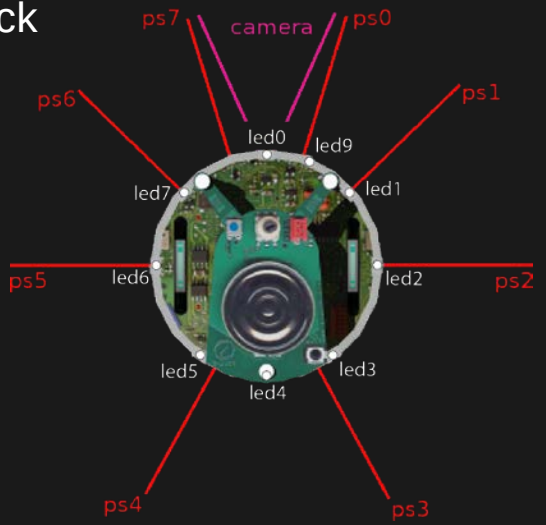
- Camera

  Resolution: 52x39 pixels in Webots simulation e-puck

  640x480 pixels in RGB color (3 bytes) in real e-puck

  ⚠ E-puck memory do not allow us enough storage

- 8 proximity IR sensors



Webots & real time simulation

General algorithm

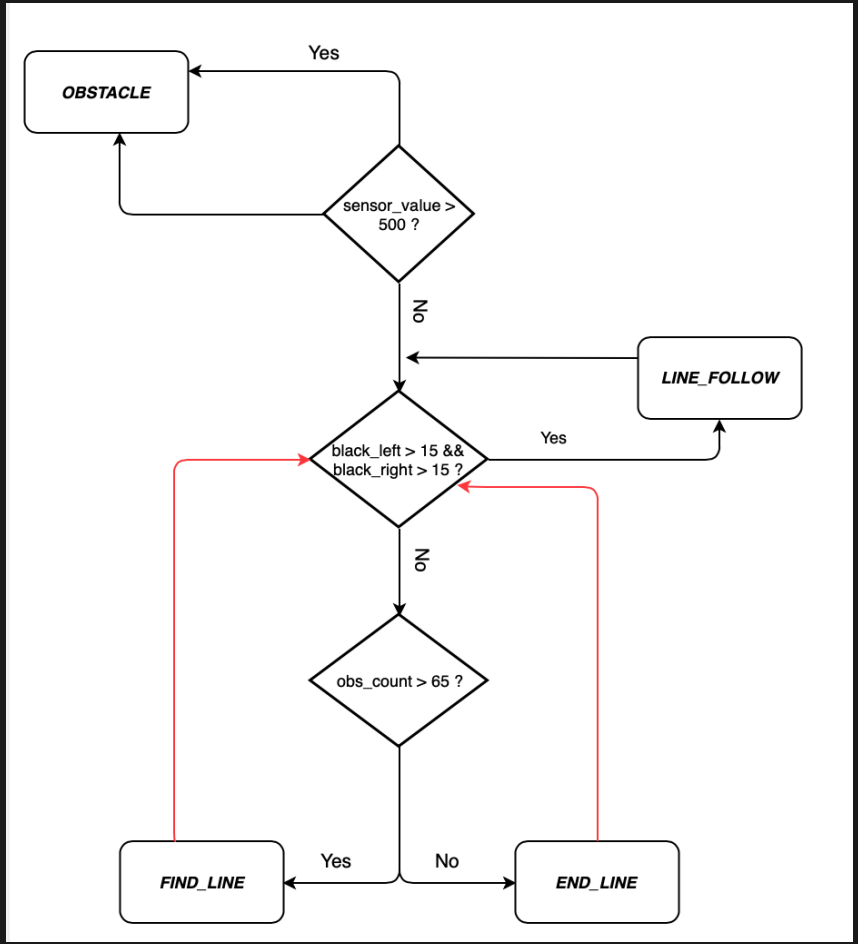
```
// line detected
     if(){ state = LINE_FOLLOW ; printf("\n \t State: LINE_FOLLOW\n") ; count_obs = 0; }
// obstacle detected
          else if(){ state=OBSTACLE ; printf("\n \t State: OBSTACLE!\n") ; }
// After avoiding the obstacle, try to find the line back
          else if (){ state=FIND_LINE ; }

  switch(state){
          case LINE_FOLLOW:                    // BASIC LINE FOLLOWING
                  ....   break;

          case OBSTACLE:                       // OBSTACLE AVOIDANCE
                  ....   break;

          case FIND_LINE:                      // LINE FINDING
                  ....   break;
```

Implementation scheme



Webots & real time simulation

- LINE_FOLLOWING state : Ø FFT, but with color differentiation (greyscale)
- OBSTACLE state : Braitenberg Coefficients
- FIND_LINE state : Braitenberg Coefficients

Limitations

Limitations: Webots simulation

- “Find Line” scheme : especially with perpendicular path, more difficulties for the virtual robot to get back to the line

- “End Line” algorithm : Detects the end of the line (robot doesn’t go straight away) but does not turn around to go back to its original path

Limitations: real time simulation

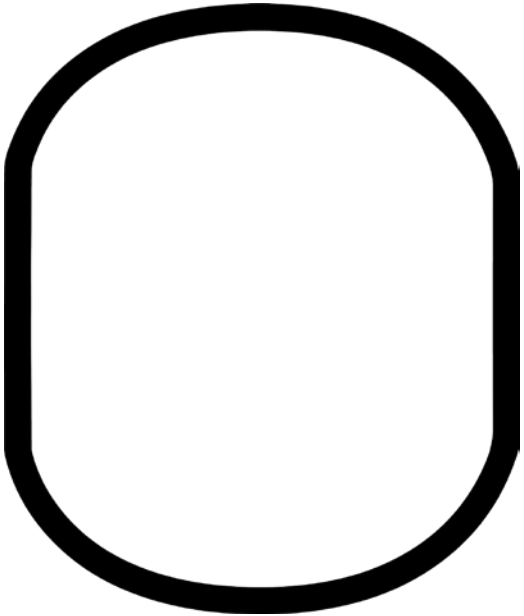
- After 2 or 2 and half loops, robot enters in a state of dead chuck : there may be a dead chunk somewhere in the code

- “Find Line” state : Detects something, works sometimes with smooth trajectories but has issues with lines perpendicular to its trajectory out of the path (when the robot is at right angle to the line, it detected but does not turn enough to get it, finding the line is not as aggressive as it should be)

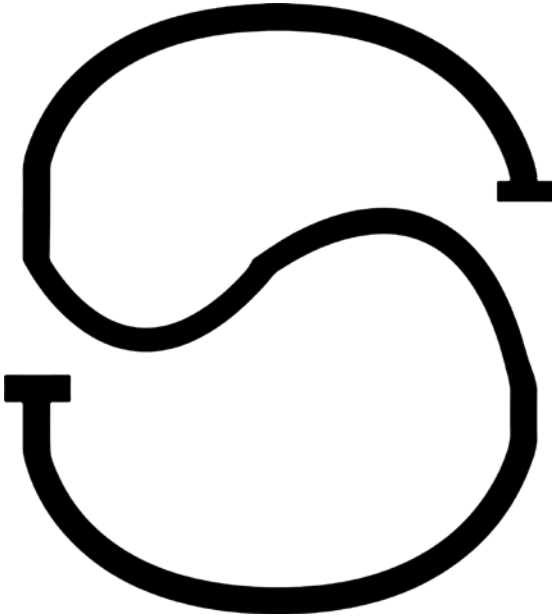
Experiments

Testing line following algorithm

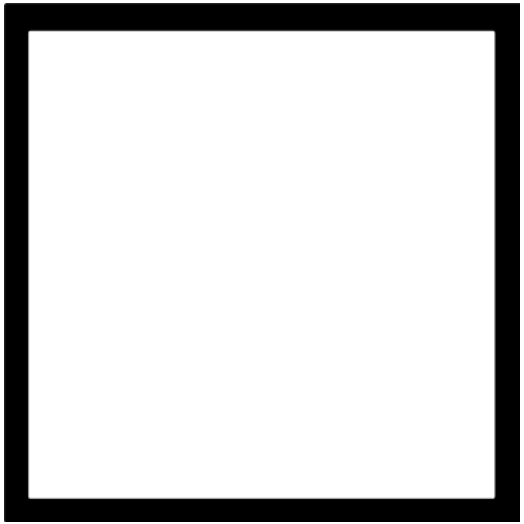
Oval



S-Shape

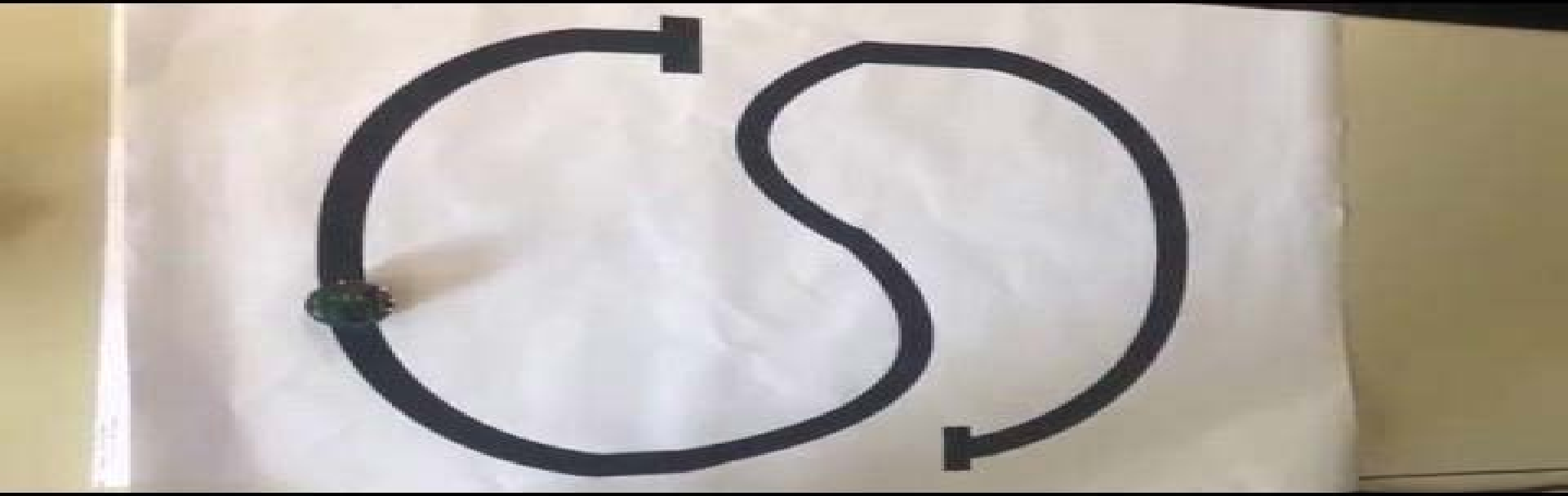


Square



Testing line following algorithm

Testing line following algorithm



Testing obstacle avoidance algorithm

Results

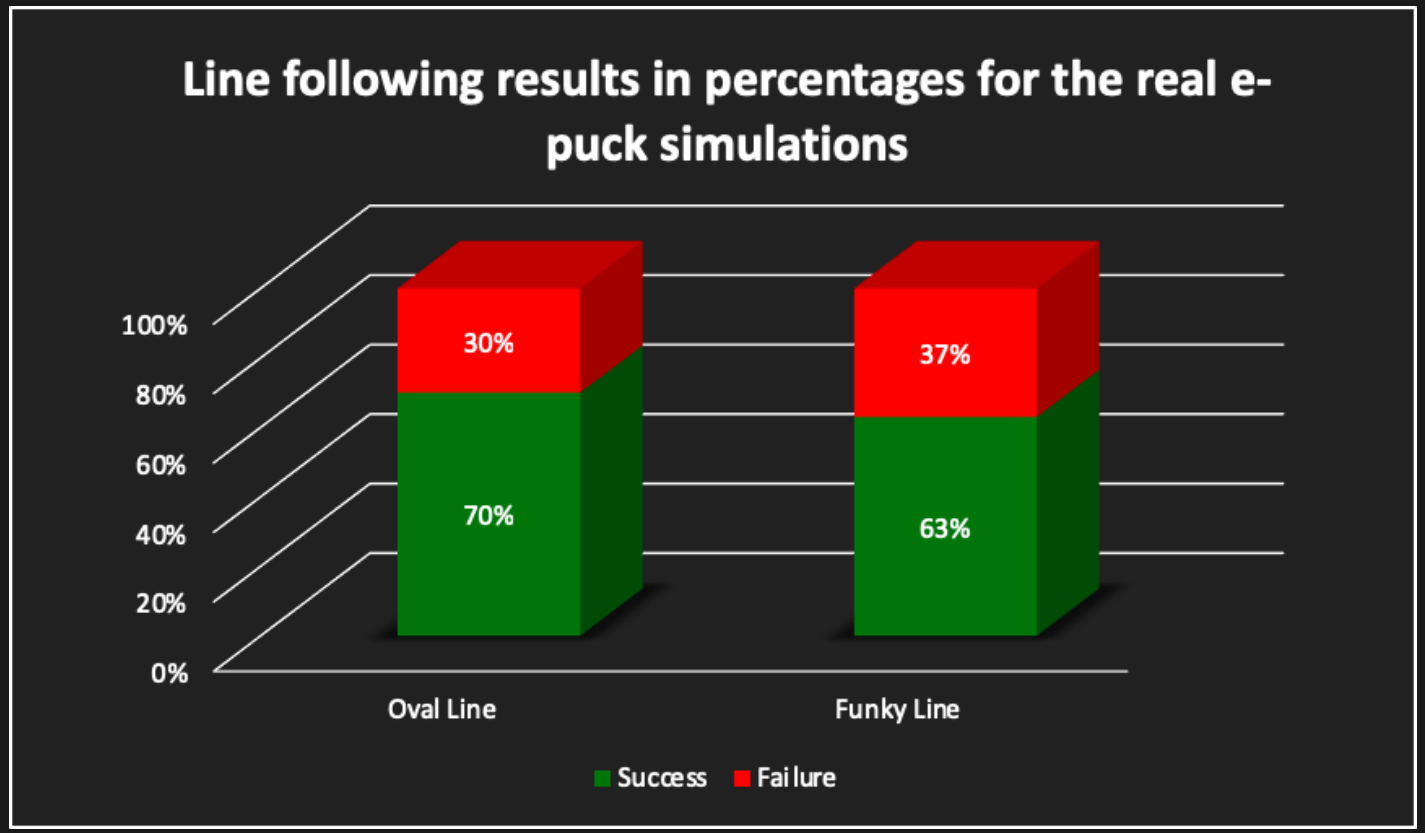
Webots simulation

- Line following with obstacle avoidance : working approximately well with all different paths (Oval, S-shape and square)
- All behaviours were almost perfect except for the “End Line” one : robot detects the end of the path but doesn’t make a turn to go back to its original path
- Sometimes, the “Find Line” scheme takes a lapse of time : due to the structure of the algorithm

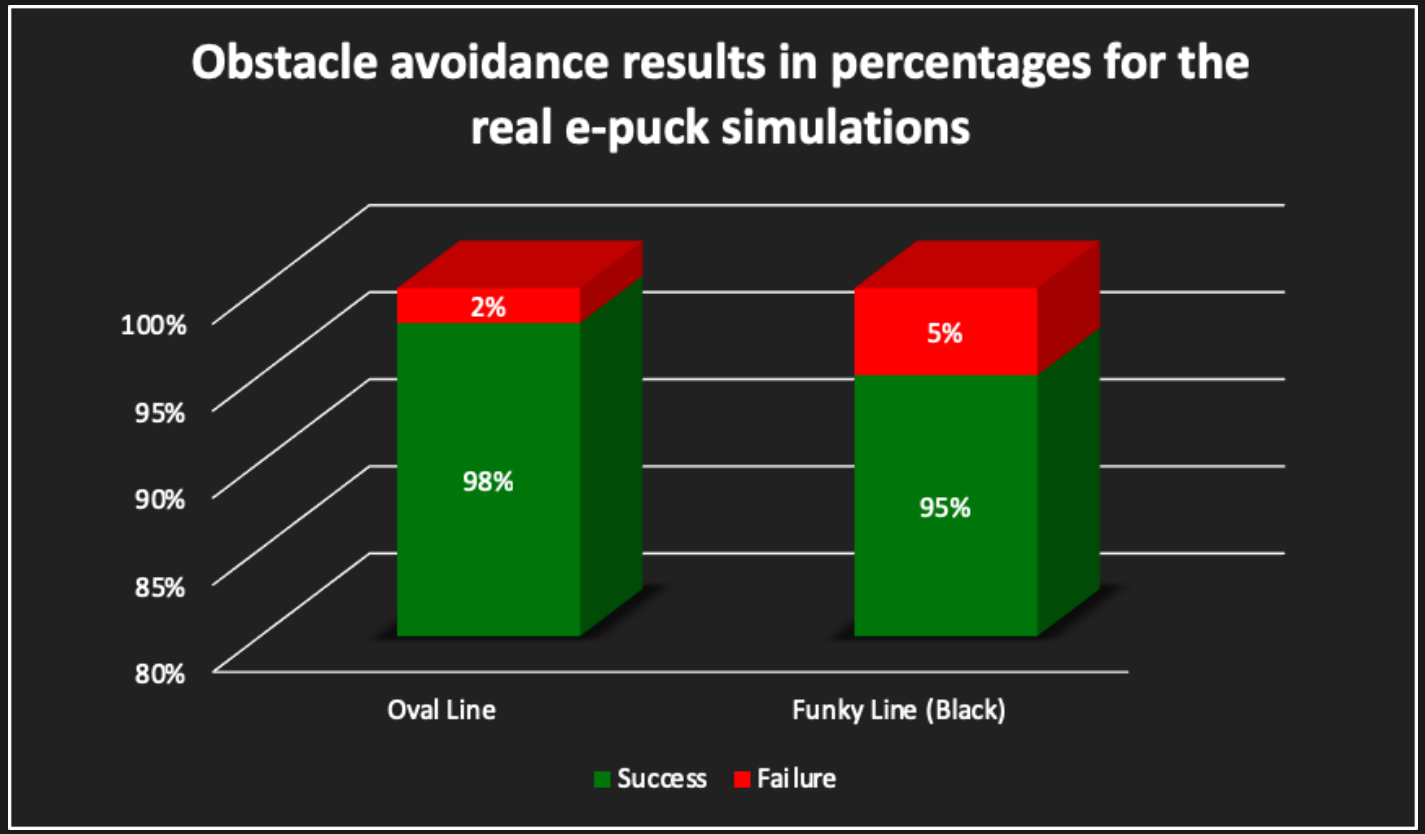
Real time simulation

- Line following + obstacle avoidance : works almost perfectly with different shapes (even if, for the obstacle avoidance, only the 6 front sensors are used)
- However : with a thinner path drawn on an A4 or A3 paper, we had a less efficient behaviour
- “Find Line” behaviour : sometimes, robot detects the presence of a path (slows its velocity and turns a little bit towards the path)
- “End Line” scheme : not done on the S-shape trajectory
- Robot’s behaviour doesn’t get disturbed by making shadow around it or multiplying the obstacles

Real time simulation : Line following tests (10 laps)



Real time simulation : Obstacle avoidance tests (10 laps)



Conclusion

- Main algorithms (line following and obstacle avoidance) worked pretty good
- Simulations were running quite well
- Real environment behaviours had more complex issues

References

[1] Siegwart R. and Nourbakhsh I. R., “Introduction to Autonomous, Mobile Robots”, MIT Press, 2004, Ch. 3 (pp. 47-53).

[2] Michel O., “Webots: Professional Mobile Robot Simulation”. Int. J. of Advanced Robotic Systems, 1: 39-42, 2004.

[3] Brooks R., “A robust layered control system for a mobile robot”, IEEE, J. of Robotics and Automation, 2(1): 14 – 23.

[4] Braitenberg V., “Vehicles: Experiments in Synthetic Psychology”, MIT Press, 1986.

[5] ”E-puck reference manual 1.0” generated by Doxygen 1.5.4 .

[6] SIS Course Slides 2019.

[7] Lab Assignments 8,9 and 10 + Solutions.

Thanks for your attention