Series 01 – Setup, Braitenberg vehicles & finite state machines

Robotics, BSc Course, 2nd Sem., Dr. Julien Nembrini, Marta Visetti

Handout on February 20th 2025 Due on March 9th 2025

Readings

— Lecture notes LN01 & LN02 available on Moodle
— "Robotics Cheat Sheet" available on Moodle
— "Vehicules", V. Braitenberg, 1984, Chap. 1-4 (pp 1-19) [1]

The reading material is available on Moodle

Sections to be completed in the Report 1 template

— Section 2.1 *Sensors* → *Proximity infra-red sensors*
— Section 3.1 *Behaviours* → *Braitenberg vehicle*

Installations

— Miniconda (Python 3.9 distribution) `https://www.anaconda.com/download/`

— Optional : Webots R2022a (see "Robotics Cheat Sheet")

Reminder : folder structure and naming convention

```
gg_nn_ff.zip
            /codes/lover.py
            /codes/explorer.py
            /codes/alt_lover.py
            /gg_nn_ff.pdf
```

gg = group number **00 if undefined**
nn = last name
ff = first name

1. Valentino Braitenberg : "Vehicules : Experiments in Synthetic Psychology", MIT Press, 1984

The exercises on this page should not be included in your submitted report. However, you are highly encouraged to do these steps before Lecture LN02 in order to be able to ask questions. When you have completed this part of the series, answer the questionnaire on moodle. **Completing this step before LN02 brings a bonus on this series' grade**.

1 Get started with the development environment

Install the python environment as explained in the "Robotics Cheat Sheet" document from Moodle. Follow the steps to connect with a robot and run your your first controllers.

2 E-puck first steps

This part is intended to guide you for your first steps with the E-puck robot. Its aim is to make you familiar with :

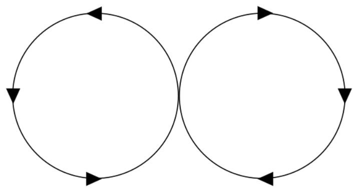
— the process of running code on the robot
— the dynamics of the robot

2.1 Forward/backward

Explore the behaviour of the robot when modifying the `S01_forward_backward.py` controller presented in class. Observe the kind of changes you made and find an explanation for their effects on the robot's behaviour.

A controller and a Webots world is available on moodle (see the "Robotics Cheat Sheet").

2.2 Figure 8



Modify the `S01_forward_backward.py` controller in order to make the robot move in a figure 8 (as depicted above). This should be possible by changing speed values and modifying counters. Observe the real robot behaviour.

What are your results ? How can you improve ? Explore the possibilities and do not hesitate to ask questions in the course or on the Moodle forum.

3 Proximity Sensor Values

Modify the provided controller `S01_IR_record.py` and use the python script `S01_plot_IR.py` (or your preferred plotting software) to implement a controller reconstructing an equivalent for the real robot of the webots infra-red lookup table graph [2] shown in Figure 1. Note that the values on the X axis may be different (steps instead of cm, for instance).

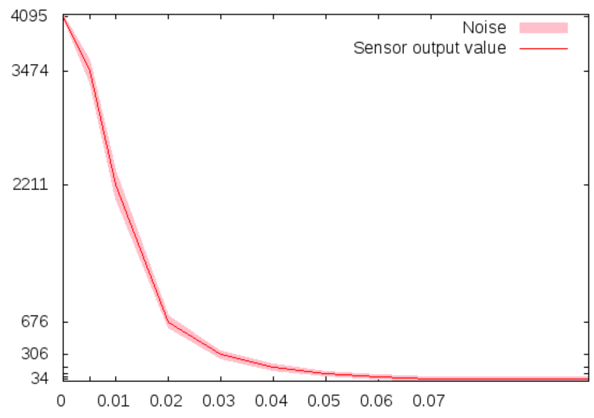


FIGURE 1 – Webots infra-red response graph : this defines the e-puck 2 IR sensor response for the simulation, with the band accounting for noise

Feel free to reuse code parts from the controller `S01_forward_backward.py`. Generate one graph with the calibrated values, using only the front proximity sensors (`PROX_RIGHT_FRONT` and `PROX_LEFT_FRONT`) of a **real robot**.

Analyze and explain potential differences between your graph and the Webots infra-red response graph (copy the graph above in your report). Describe in sufficient detail your experimental setup (do not forget to mention the robot number). Propose ways to treat the sensor values to improve the quality of the information they deliver.

Your analysis has to fill in the *Section 2.1 Sensors* → *Proximity infra-red sensor* section in the report template.

Hint : Place the robot near an obstacle and use the code `S01_forward_backward.py` to drive the robot further away, while recording sensor and counter values. Keep in mind that the calibration step needs the robot to stand away from obstacles at the beginning.

4 Explorer & Advanced Lover

Implement on a real robot both controllers for the explorer and advanced lover behaviour as discussed in class. For each controller :

— Describe the chosen formulas for the speed of the wheels,

— Choose two different sets of weights for the proximity formula and describe the influence on the robot's behaviour.

Expand the provided controller `S01_basic_lover` to include all proximity sensors.

2. (https://www.cyberbotics.com/doc/reference/distancesensor)

5 Random Lover

Using the controllers from the previous exercise, design, implement and test a behaviour that :

1) starts by running LOVER
2) if encountering an obstacle moves towards it
3) detects when reached the *equilibrium facing the obstacle*
4) switches to EXPLORER
5) detects when away from the obstacle
6) if encountering another obstacle, randomly decides to switch to LOVER and go to step 2), or stay in EXPLORER
7) detects when away from the obstacle
8) goes back to step 6).

Follow these steps :

a) Find a robust way to detect when the robot in LOVER behaviour is at equilibrium (e.g. almost stopped) facing the obstacle (e.g. in front of the obstacle). Discuss your solution, providing examples when it may fail and proposing solutions to this problem.
b) Find a robust way to detect when the robot in EXPLORER behaviour leaves the current obstacle
c) Use the python random module to randomly switch to LOVER
d) Using the states of your program (LOVER/EXPLORER/...) represent schematically the AFSM (using latex or your preferred drawing program). Pay special attention to have transitions only dependent on the current state and to consider all possible events.
e) Implement the behaviour and comment on your solution.
f) Record a video with LEDs visible to indicate the current state (rule : all LEDs on for the LOVER state, off for the EXPLORER state), upload it on your preferred sharing site, and include the link in your report. There is a smartphone stand in the lab to help you record videos.

Beware to have only one `go_on()` function in your code !

Describe and document your implementations. Both the above exercises *Explorer & Advanced Lover* and *Counting lover* have to fill in the *Section 3.1 Behaviours → Braitenberg vehicle* section of the report.

Do Not Forget !

Describe and document your implementations in your report by using *informative* and *short* code snippets. This is the way for the evaluators to assess your understanding of the code. In addition, all your controllers (in complete form) need to be put together with your report in PDF into a zip file **following the naming convention or your work will not be considered**.

All controllers, a Webots world, as well as a plotting script in python 3 for this series are available on moodle (see the "Robotics Cheat Sheet").