Lab 7

School of Architecture, Civil and Environmental Engineering

EPFL, SS 2017-2018

http://disal.epfl.ch/teaching/signals_instruments_systems/
What this lab is about

- Mini tutorial on Webots, to familiarize you with this tool
- Understanding sensors
  - examples of sensors on-board the e-puck
  - notion of ‘noise’
  - how Webots simulates sensors
- Understanding how to use sensor data
  - input for actuation (example: robot control)
  - manipulation of data (logging and plotting)
Reminder: Webots GUI

- **Scene tree**
- **World view**
- **Editor**

![Webots GUI Interface]

```c
#include <webots/robot.h>
#include <webots/differential_wheels.h>
#include <webots/forward_kinematics.h>
#include <webots/light_sensor.h>
#include <webots/camera.h>
#include <webots/accelerometer.h>

int main(int argc, const char *argv[]) {
    int i;
    WbDeviceTag ds[NB_SENSORS];
    WbDeviceTag ls[NB_SENSORS];
    WbDeviceTag camera;
    WbDeviceTag accel;

    wb_robot_init();

    // get an enable distance sensors
    char ds_name[] = "ps8";
    for (i = 0; i < NB_SENSORS; ++i) {
        ds[i] = wb_robot_get_device(ds_name);
        wb_distance_sensor_enable(ds[i], TIME_STEP);
    }

    // get an enable light sensors
    char ls_name[] = "l16";
    for (i = 0; i < NB_SENSORS; ++i) {
        ls[i] = wb_robot_get_device(ls_name);
        wb_light_sensor_enable(ls[i], TIME_STEP);
    }

    // get an enable camera
    camera = wb_robot_get_device("camera");
    wb_camera_enable(camera, TIME_STEP);

    // get an enable accelerometer
    accel = wb_robot_get_device("accelerometer");
    wb_accelerometer_enable(accel, TIME_STEP);

    // some code...
}
```

```
INFO: e-puck: Starting: "/home/mansolin/Desktop/SIS/Lab_hwk07/New/lab_05/controllers/e-puck/e-puck"
```
Reminder: Modeling sensors

- Capture **non-linearities** and **noise** of sensors.
- However, **calibration** is often approximative.
- Most often, sensor response is defined by a lookup table (here a proximity sensor):

  \[
  \text{lookupTable} = \begin{bmatrix}
  0 & 1000 & 0, \\
  0.1 & 1000 & 0.1, \\
  0.2 & 400 & 0.1, \\
  0.3 & 50 & 0.1, \\
  0.37 & 30 & 0 \\
  \end{bmatrix}
  \]

  **distance**  **value**  **noise**
Robot control – Obstacle avoidance
Robot control – Braitenberg