Lab 7

School of Architecture, Civil and Environmental Engineering

EPFL, SS 2018-2019

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
- Console

![Webots GUI Interface]

```
#include <websot/robot.h>
#include <websot/differential_wheels.h>
#include <websot/light_sensor.h>
#include <websot/camera.h>
#include <websot/accelerometer.h>
#include <stdio.h>

#define NB_SENSORS 8
#define TIME_STEP 16

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);
        +ds_name[2];
    }
    // get an enable light sensors
    char ls_name[] = "ls8";
    for (i = 0; i < NB_SENSORS; ++i) {
        ls[i] = wb_robot_get_device(ls_name);
        wb_light_sensor_enable(ls[i], TIME_STEP);
        +ls_name[2];
    }
    // 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);
    return 0;
}
```

```make
make -I/home/mansolin/Downloads/websots73/include/controller/c" -Wall -O3 -DLINUX -MM e-puck.c -MT build/release/e-puck.o > build/release/e-puck.d
gcc -c -Wall -O3 -DLINUX -I. -I/home/mansolin/Downloads/websots73/include/controller/c" e-puck.c -o build/release/e-puck.o
cp build/release/e-puck e-puck > /dev/null 2>$]
```
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):

```
lookupTable = [
    [0, 0, 0],
    [0.1, 1000, 0.1],
    [0.2, 400, 0.1],
    [0.3, 50, 0.1],
    [0.37, 30, 0]
]
```
Robot control – Obstacle avoidance
Robot control – Braitenberg