

CAI – Using C in Webots

Matthew Lewis
m.lewis4@herts.ac.uk

Part 1.2: Using C in Webots 7.4.3

1. You should have already followed the instructions in **Part 1: Practical Introduction to Webots** to assign a C controller to an e-puck robot.
2. Information about the C API can be found in the **Webots Reference Manual**. See in particular section 3.20: DifferentialWheels.
3. Information about the sensors and other capabilities of the e-puck robot is available in the **Webots UserGuide, section 8.1** and in the e-puck prototype file provided by Webots, located at:

C:\Program Files (x86)\Webots\resources\projects\robots\e-puck\protos\EPuck.proto

From this file we can see that the e-puck has 8 distance (proximity) sensors named “ps0”, “ps1”, ..., “ps7”; 8 light sensors, and 3 ground sensors, and 10 LEDs. In fact, the light sensors and the distance sensors use the same hardware for different purposes: infra-red detectors used as passive and active detectors respectively.

4. Since the simulated E-Puck model is derived from the DifferentialWheels model (which derives from the Robot model), you should **change the stub controller to use the DifferentialWheels class**:


```
#include <webots/differential_sheels.h>
```

You can now use the wb_differential_wheels functions to control the robot. Insert the following constant:

```
#define MAX_SPEED 1000
```

In the main “while (wb_robot_step())” loop, insert a function call to set the wheel speeds:

```
double left = MAX_SPEED/2;  
double right = MAX_SPEED/2;  
wb_differential_wheels_set_speed(left, right);
```

Now compile (cog icon ) and run your controller. The e-puck should move forward.

5. You should be able to change to above program to get the e-puck to move backwards, to turn on the spot, or to move in a curve. Try it to check that you understand. Ask if you are not sure.
6. To get/set LEDs on the robot you will need to use the `wb_led_set()` and `wb_led_get()` functions. Some initialisation will be needed, so it will be useful to add an initialisation function to the controller class.

```
#include <stdio.h>

#include <webots/led.h>

#define LED_COUNT 10

// Somewhere to store LED handles
static WbDeviceTag leds[LED_COUNT];

void epuck_init()
{
    char ledname[5] = "led0";
    int i;
    for (i=0; i<LED_COUNT; i++) {
        ledname[3] = '0'+i; // Name required to get LED
        printf("Initialising %s\n", ledname);
        leds[i] = wb_robot_get_device(ledsname);
    }

    ...

    // Add to main() before the loop:
    epuck_init();

    ...

    // Add to the main loop:
    wb_led_set(leds[0],1);
```

Now when you compile and run the simulation you should see the red ring LED light up. Also, you will see information that was printed in the `epuck_init()` function in the console (“Initialising led0” etc).