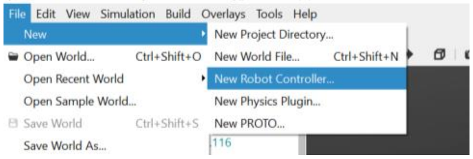
Feedback

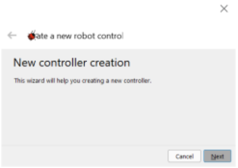
Content:

1. How to add your own controller program to the webots project
2. Use of the webots sample controllers
3. Use of e-puck three ground sensors proto
4. Robot motion coding tips
5. Left Hand on the Wall Algorithm

1. How to add your own controller program:

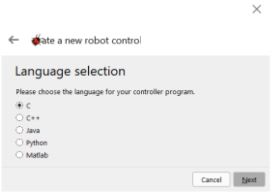
Follow the actions order below, row by row, from left to right





New controller creation

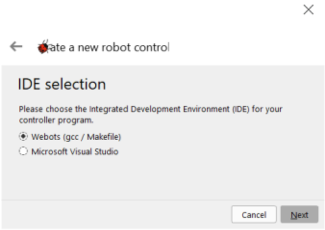
This wizard will help you creating a new controller.



Language selection

Please choose the language for your controller program.

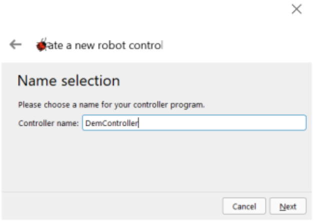
● C
○ C++
○ Java
○ Python
○ Matlab



IDE selection

Please choose the Integrated Development Environment (IDE) for your controller program.

● Webots (gcc / Makefile)
○ Microsoft Visual Studio

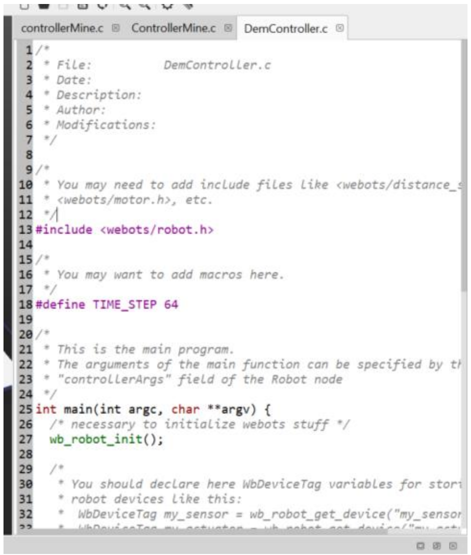


Name selection

Please choose a name for your controller program.

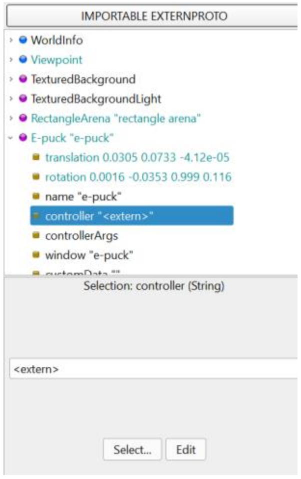
Controller name: DemController

A template code will be automatically added to your named controller program. The code is displayed under a dedicated named tab at the right side of the IDE, see below.



```c
/*
 * File:          DemController.c
 * Date:
 * Description:
 * Author:
 * Modifications:
 */

/*
 * You may need to add include files like <webots/distance_s
 * <webots/motor.h>, etc.
 */
#include <webots/robot.h>

/*
 * You may want to add macros here.
 */
#define TIME_STEP 64

/*
 * This is the main program.
 * The arguments of the main function can be specified by th
 * "controllerArgs" field of the Robot node
 */
int main(int argc, char **argv) {
  /* necessary to initialize webots stuff */
  wb_robot_init();

  /*
   * You should declare here WbDeviceTag variables for stori
   * robot devices like this:
   *  WbDeviceTag my_sensor = wb_robot_get_device("my_sensor
```

This controller program can now be added from the “Controller Choice” window. To do so, expand the e-puck/robot node from the left side of the panel and select “controller”



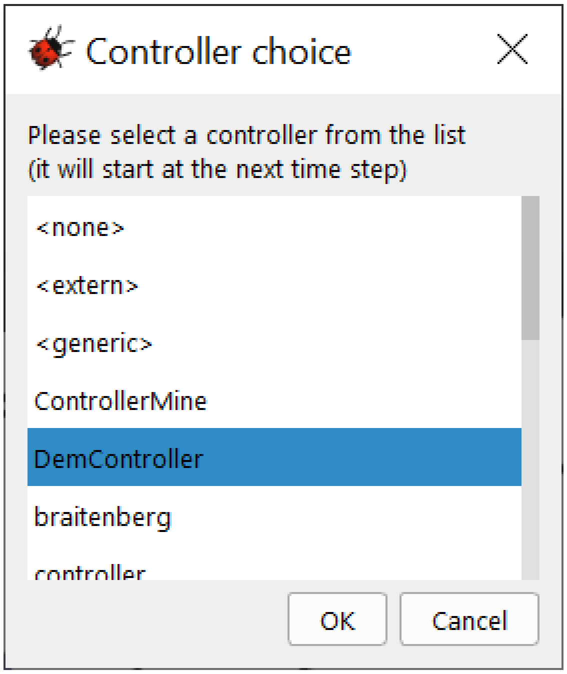
WorldInfo
Viewpoint
TexturedBackground
TexturedBackgroundLight
RectangleArena "rectangle arena"
E-puck "e-puck"
translation 0.0305 0.0733 -4.12e-05
rotation 0.0016 -0.0353 0.999 0.116
name "e-puck"
controller "<extern>"
controllerArgs
window "e-puck"

Selection: controller (String)

<extern>

Figure 1

Click on “Select” button from the below panel in figure 1. Your controller program (DemoController.c in this case) should be available in the list of controllers (scroll down if needed):



Controller choice

Please select a controller from the list
(it will start at the next time step)

If you select another sample controller program, click on “Edit” button in Figure 1; copy-paste the controller code in another editor (VScode, notepad, etc..) to save the program on your computer for easy future reference.

2. Use of webots sample controllers

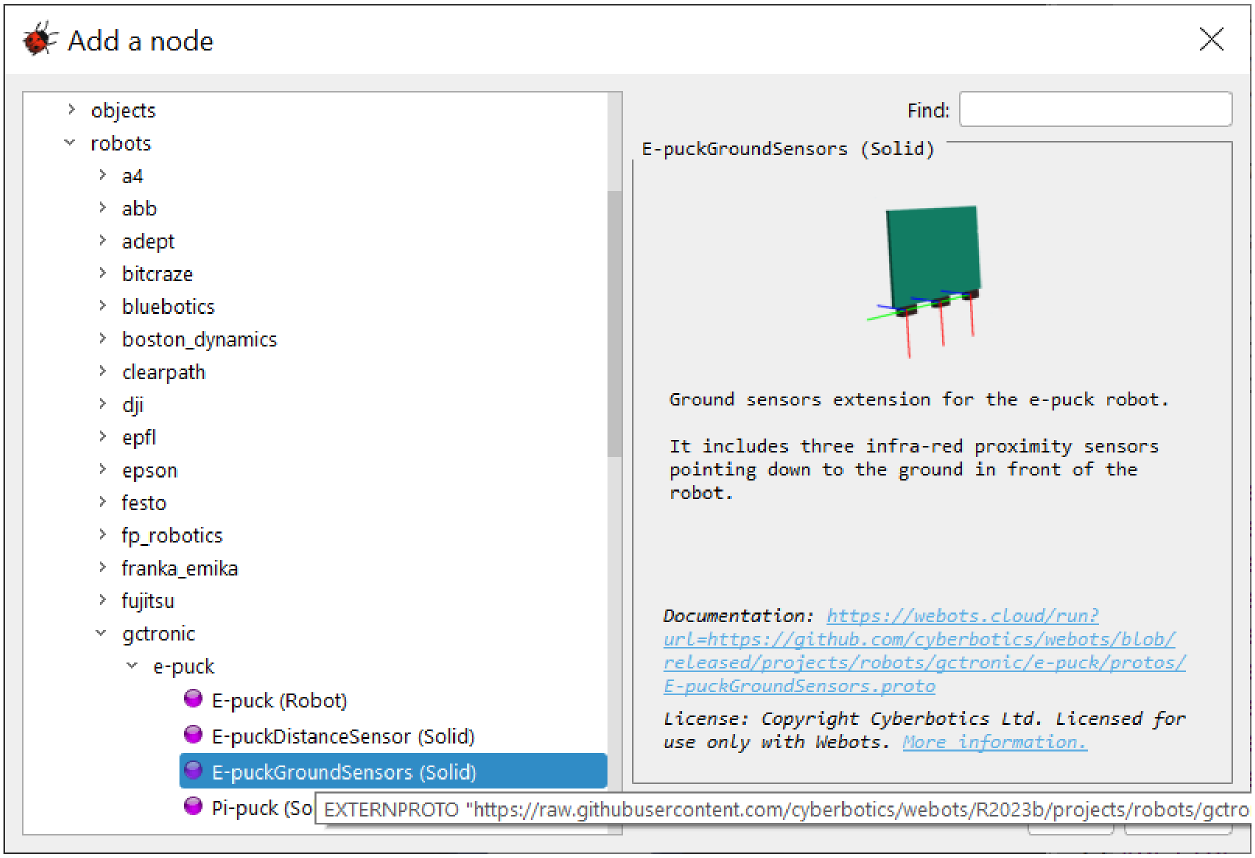
You can edit any of the webots sample controller; however, you have to acknowledge the source. Having said that, some of the sample controllers might have considerable irrelevant code to the project.

In the past weeks, I uploaded a much simpler controller program to an e-puck robot with one ground sensor. The controller prints the value of the ground sensor reading when the robot moves on white/black surface. The controller code is attached to this document although it is already part of the webots project available in Moodle which you should at least run.

3. Use of e-puck three ground sensors proto

We explained in SensorsConfiguration.docx how to add a ground sensor. However, as reported by a student, you can add a three ground sensors as follows.

Select “groundSensorsSlot” node from the e-puck/robot node in the left panel; right click and select “Add New”, expand “Proto nodes (webots project)”, then “robots”, then “gctornic” and select “E-puck GroundSensors(solid)”.



Ground sensors extension for the e-puck robot.

It includes three infra-red proximity sensors pointing down to the ground in front of the robot.

Documentation: *https://webots.cloud/run?url=https://github.com/cyberbotics/webots/blob/released/projects/robots/gctronic/e-puck/protos/E-puckGroundSensors.proto*

License: Copyright Cyberbotics Ltd. Licensed for use only with Webots. *More information.*

EXTERNPROTO "https://raw.githubusercontent.com/cyberbotics/webots/R2023b/projects/robots/gctro

However, you’ll highly likely have to move the positions of the ground sensors as per the need of the width of black lane in your assigned project track; see SensorsConfiguration.docx.

The C statements to control the motion of the robot are detailed in RobotControllerCoding.docx (it is the same file which was under the webotsController.zip file under the supportFolder). I would recommend to create four functions for forward, reverse, right, and left movements which you can call to drive the robot on the track.

4. Robot motion coding tips:

At least RobotControllerCoding.docx and webots sample controller programs.

5. Left Hand on the Wall Algorithm:

The project specification gives a high level but sufficient description of the algorithm with two additional useful references. With this algorithm, at any intersection a left movement is always prioritised.

The first Reference explores a number of possible intersections and the corresponding robot motion as per the algorithm; the second reference gives an example regarding the positions of the ground sensors to solve a particular track.

You need to consider carefully the minimum ground sensors you have to add to the e-puck robots and their positions in order for the robot to move on the track, well aligned, avoiding any obstacle until it reaches the end of the track.