UNIVERSITY OF NOTTINGHAM

Introduction to RobotC

Lab Sheet 1

NOTE:

If parts of this or other lab sheets during the module are unclear or the software does not behave as expected, please let us know so we can adjust the information.

G54ARS

Autonomous Robotic Systems

Introduction to RobotC

Table of Contents

1 Introduction

The LEGO MINDSTORM EV3 is a programmable microcontroller set in a LEGO environment (Figure 1).

FIGURE 1 EV3 BRICK

On the brick you have 6 buttons (4 directional buttons, mostly for browsing, an "Enter" button at the centre of 4 buttons, and an "Escape/Return" button below the screen.

EV3 bricks have 4 ports for motors (A, B, C, D), and 4 ports for connecting sensors (1, 2, 3, 4). A wide variety of sensors is available, such as ultrasonic, infrared and gyro sensors. Note that there is a built-in incremental encoder on each motor.

To switch the brick on, keep the centre button pressed for 1-2 seconds. A menu with 4 tabs will appear on the EV3 screen (Figure 2). Use left/right buttons to switch tab and up/down buttons to browse inside a tab.

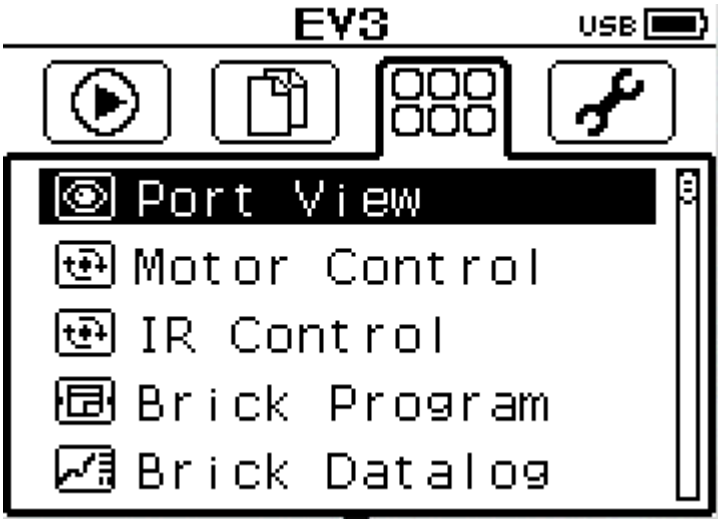

FIGURE 2: EV3 MENU

In the first tab, you will find the recently downloaded programs.

The second tab allows you to browse the EV3 file system. Your programs will be placed in the "rc" folder and datalog files will be generated in "rc-data" folder. Be careful, clicking on an opened folder will open a window which will ask you if you want to delete this folder. The folders "rc" and "rc-data" can be regenerated by downloading a program or creating a datalog, but previous code is erased from EV3 memory.

The third tab contains 3 important tools for sensor monitoring:

- Port View: this tool displays values from sensors and motors connected to the brick. LEGO sensors will be recognised by the brick and return numerical values such as a given distance or angle. These sensors have multiple modes (for example, angle and rate for the EV3 gyro sensor). Press the ENTER button to select the correct mode. Note: Third party sensors will not be correctly recognised, i.e. their values will often not have meaning with this tool.
- Motor Control: you can control the connected motors directly with the EV3 buttons.
- Brick Datalog: like the Port View, you can monitor your sensors, but this time graphically. The x-axis represents time and y-axis the value returned by the sensor. We won't generate datalogs with this tool, as it is only usable with official LEGO development software.

The fourth tab allows configuring your brick (volume, Bluetooth activation,…).

2 RobotC

Multiple platforms have been created to enable advanced development on the EV3 brick. We will use RobotC, an IDE originally developed at Carnegie Mellon University which allows programming the EV3 brick using a C based language. RobotC supports most standard C routines (loops, if-conditions...) and follows the C syntax.

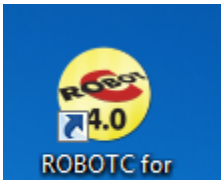

FIGURE 3: ROBOTC ICON

To launch the RobotC IDE, use the corresponding shortcut on the desktop (Figure 3) or click on Start Menu and find "ROBOTC for LEGO MINDSTORM 4.x".

2.1 Interface

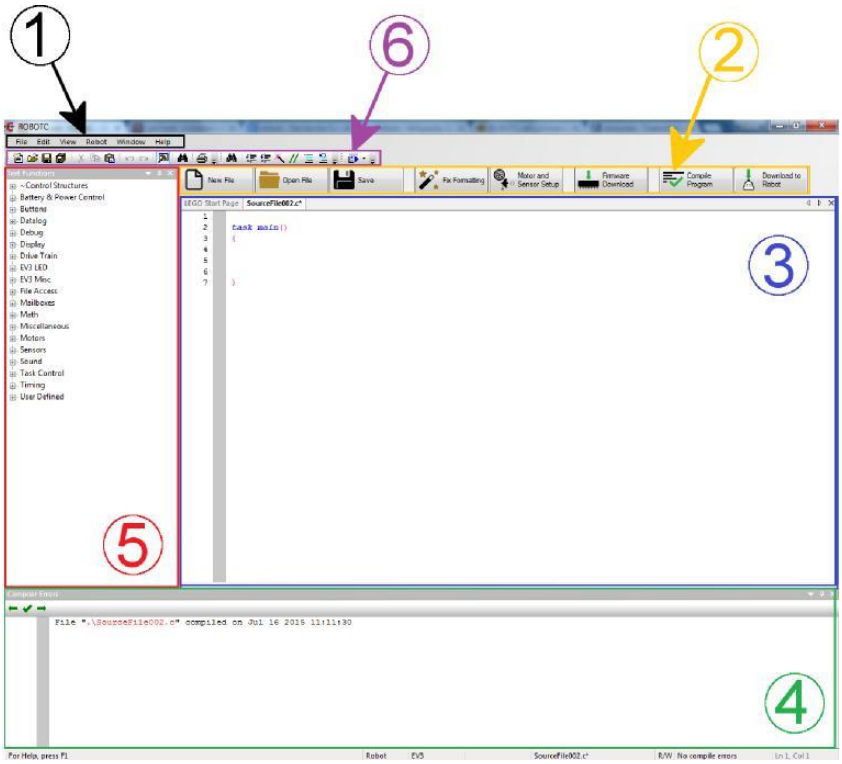

FIGURE 4 : ROBOT C INTERFACE

The RobotC interface (Figure 4) is divided in sections:

1. The menu bar
2. The big icon toolbar, with the most important commands such as save, compile, send the program to the robot. Note that clicking on "Download to robot" will automatically save and compile your code.
3. This is where you write your code.
4. This area displays compiler errors and warnings.
5. Here, you will find a list of RobotC functions.
6. An additional toolbar which may not be displayed when you start RobotC. Here there are icons for File functions (New, Open, Save all) and Edit functions (re-indent, comment, find).

2.2 Documentation and Help

Most commands are available on the left side of the interface. You can hover over them to get more information or press F1 to open the help (or *Help → Open Help*). Documentation for functions is given in "*Command Library - LEGO EV3 → Text-based ROBOTC commands"*.

2.3 Sensor and Motor Configuration

Before using RobotC, ensure that it is set up for EV3 robots in the following menu: *Robot → Platform Type.*

Different programs will not necessarily use the same sensors and motors. Before accessing them, you need to configure your own program. On the big icon toolbar, click on *Motor and Sensor setup*. The wizard as shown in (Figure 5) will appear.

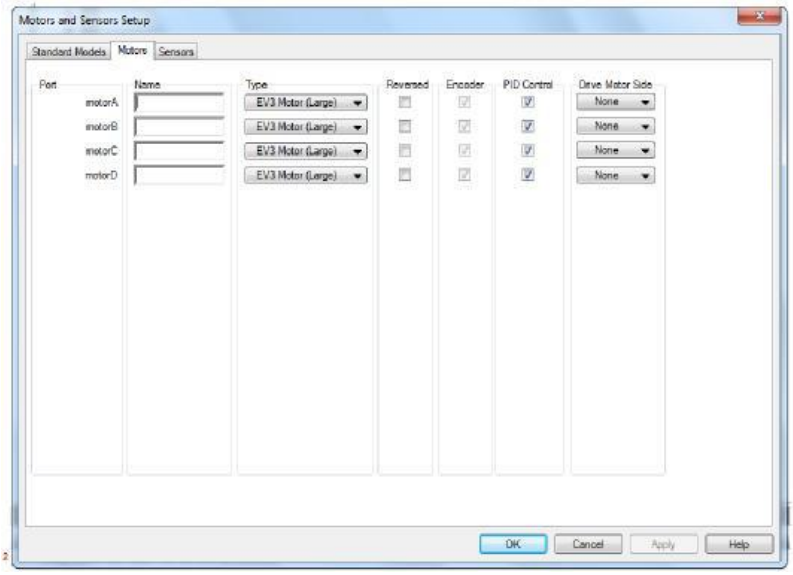

FIGURE 5 MOTOR AND SENSOR SETUP

There are 3 tabs. The first tab allows choosing a predefined configuration - we will mostly use custom configurations. Second and third tabs concern our setup. You can choose which ports will be used and what will be connected to them. Once you have completed the setup, validate. Pragma lines will be generated in your code (Code 1).

```
#pragma config(Sensor, S1,    gyro,      sensorEV3_Gyro)
#pragma config(Sensor, S2,    rightSonar, sensorEV3_Ultrasonic)
#pragma config(Sensor, S3,    HTCS2,     sensorI2CCustom)
#pragma config(Sensor, S4,    leftSonar,  sensorEV3_Ultrasonic)
#pragma config(Motor,  motorA, right, tmotorEV3_Large, PIDControl, encoder)
#pragma config(Motor,  motorC, left,  tmotorEV3_Large, PIDControl, encoder)
```

CODE 1: EXAMPLE OF PRAGMA LINES

2.4 How to use sensors and motors

On sensor function prototypes, you may notice multiple types such as Ultrasonic, and Gyro. You can refer to them as S1-4 or by the name you gave them in the configuration wizard. This is similar for motor commands with the tMotor type. The following example (Code 2) shows a basic use of motors and sensors. Note that your robot may have a different sensor

```
#pragma config (Sensor, S2 , SonarEv3 , sensorEV3_Ultrasonic)
#pragma config (Motor, motorA , leftMotor, tmotorEV3 Large, PIDControl)
#pragma config (Motor, motorC , rightMotor, tmotorEV3 Large , PIDControl )
//*!!Code automatically generated by 'ROBOTC' *//

task main()
{
  while (true) {
    if (getUSDistance(S2) > 10){  //read ultrasonic sensor
          //start Motors
          setMotorSpeed (leftMotor , 5);
          setMotorSpeed (rightMotor , 5);
    }
    else {
          // stop Motors
          setMotorSpeed (leftMotor , 0);
          setMotorSpeed (rightMotor , 0) ;
    }
  }
}
```

CODE 2: USING SENSORS AND MOTORS

When using the gyro, make sure you set up the sensor to use the "Rate and Angle" mode as in the screenshot in Figure 6. (If this was not set when running the first program, you may have to restart the brick or unplug and replug the gyro sensor.

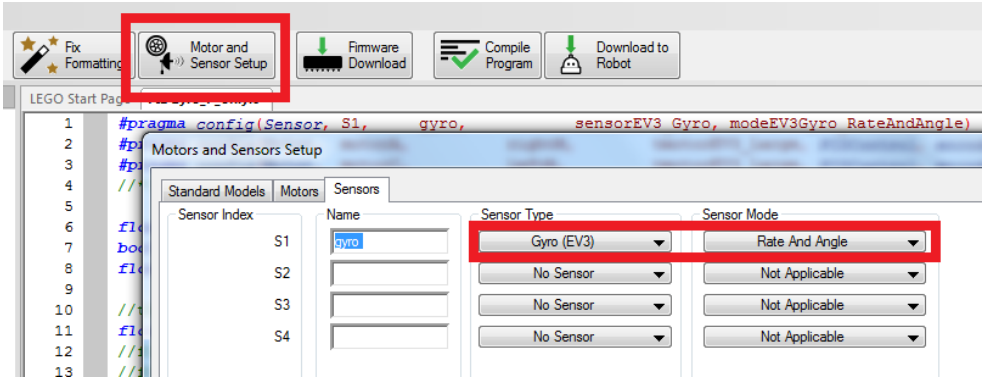

FIGURE 6 : GYRO SENSOR SETUP

2.5 Tasks

When you create a new file, a task called "main" is automatically created. This task is mandatory and the thread of execution of your program will start here. Tasks behave like “threads”, i.e. multiple tasks can run “concurrently”.

Each task has to be started from another running task, see for example (Code 3).

```
task firstTask()
{
  while(true){
    //....
  }
}

task secondTask()
{
  while(true){
    //....
  }
}

task main()
{
  startTask(firstask);
  startTask(secondTask);
  while(true){
    //....
  }
}
```

CODE 3: TASKS

2.6 Timer

RobotC has 4 timers, T1 through T4 which enable accurate timing.

- *clearTimer(the Timer) :* reinitialises the indicated timer to 0
- *time1(the Timer* : gives you the elapsed time in milliseconds

Others functions and variables are available in the Timing section of the functions list.

2.7 Third party sensors

In the labs, we will use a variety of sensors, many of which are not manufactured by LEGO but by specialised companies. See below for information on how to work with these sensors.

2.7.1 Settings

In order to use sensors which are not manufactured by LEGO (such as the HiTechnic Color Sensor or the Mindsensors Infrared sensor) with RobotC, you might need to add their drivers to the *include* directories of RobotC. To do this you have to select "Expert Level" in the menu *Window →Menu Level.* From here, you can access the Compiler Preferences in the menu *View → Preferences → Detailed Preferences*, choose the tab "Compiler" then "Include directories" and add the driver directory which should be located at:
"C:\Program Files (x86)\Robomatter Inc\ROBOTC Development Environment 4.X\Sample Programs\EV3\3rd Party Driver Library\include\" (Figure 7).

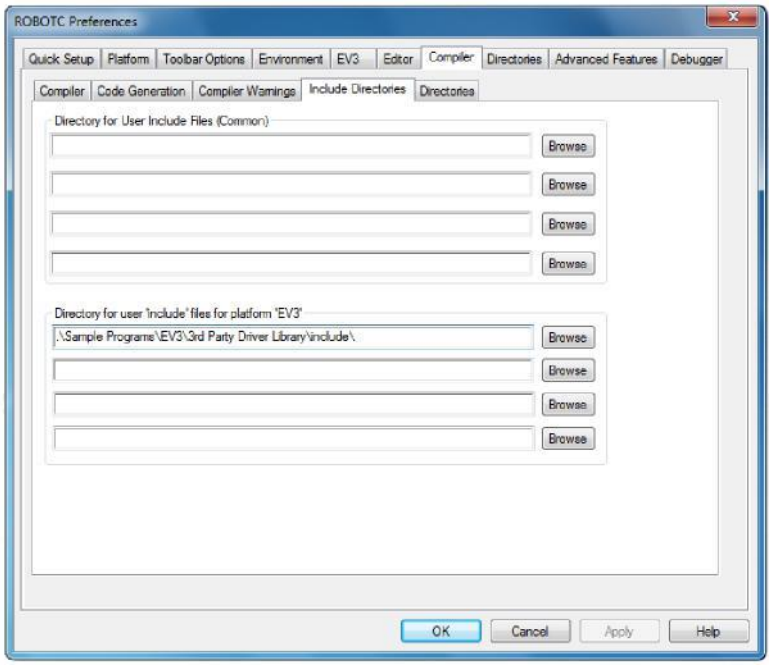

FIGURE 7 INCLUDE DIRECTORIES TAB

2.7.2 Mindsensors DIST-Nx-v3

To use the Mindsensors IR sensor (Figure 8) with RobotC, you need to declare it as a "sensorJ2CCustom" and include the file "mindsensors-irdist.h" (Code 4):

FIGURE 8 IR SENSOR

```
#pragma config (Sensor, S1, MSDIST, sensorI2CCustom)
#include "mindsensors-irdist.h"

task main()
  //return the distance measured in millimetres
  int dist = MSDISTreadDist(MSDIST);
}
```

CODE 4: MINDSENSORS DIST-NX-V3

2.7.3 HiTechnic Color Sensor V2

To get the ambient colour, you need to use the HiTechnic Color Sensor V2 (Figure 9) in passive mode (the LED is turned off).

FIGURE 9 HITECHNIC COLOR SENSOR

When it is used in passive Mode, you can only get the RAW values from the sensors (which are not scaled to 0-255), but if it is used in active mode you can get normalised RGB values with the function *HTCS2readNormRGB* or the ID of the detected colour (*HTCS2readColor*). Here is an example to get raw values from this sensor (Code 5).

```
#pragma config (Sensor, S3, HTCS2, sensorI2CCustom)
#include "HitechnicColorSensor.h"

task main(){
        long r , g , b;

        //use the sensor in passive mode (LED turned off)
        HTCS2readRawRGB(S3, true, r , g , b) ;
}
```

CODE 5: HITECHNIC COLOR SENSOR V2

The complete documentation about these sensors can be found at: http://botbench.com/driversuite/ (sensor-specific information is available under "modules".

2.8 Compiling, displaying and debugging

When you download your code to the EV3, a new window appears for debugging (Figure 10). You can run the program normally or by stepping through the code.

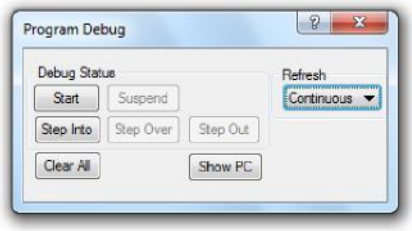

FIGURE 10 DEBUGGING WINDOW

There are many ways to read information from the robot to check if your program runs correctly:

- It is possible to write directly to the screen with the function *displayTextLine* (Code 6). Also, if the robot is connected to the computer (via USB or Bluetooth), it is possible to display a copy of the screen on the computer screen by using the *Ev3 Remote Screen* feature available in the menu *Robot → Debug.*
- If the robot is connected to the computer while a program is running, you can use the debug stream: you can write to the debug stream by using the function *writeDebugStreamLine* (Code 6). You can read the stream in RobotC by clicking *Robot ⟶ Debugger Windows*.
- If you want to record values in files, you can use the DataLog or Bluetooth as explained in Section Sensor Readings3.2.

```
#pragma config ( Sensor, S2 , SonarEv3 , sensorEV3_Ultrasonic)

task main()
  {
  while (true) {
     //read touch sensor
`    int d =  getUSDistance(S2);
    //display on Debug Stream
     writeDebugStreamLine ("%d cm", d);
    //display on the first line of the screen
    displayTextLine(1, "%d cm", d);
  }
}
```

CODE 6 WRITE IN DEBUG STREAM

2.9 Datalog

RobotC datalog functions let you create .txt files that can be exported to a PC and opened for example with Excel (values are separated by commas like .csv files).

To open or create a data log, use the following function:

bool **datalogOpen**(long **index,** long **columns,** bool **append = false**);

The parameter **index,** will determine the suffix in the name of your file, whereas **column** is the number of columns in the file. If the variable **append** is true and a file with the given index already exists, your new data will be appended to this file, otherwise this file will be overwritten.

Then you can add a number to a column with a function which depends on the type of the data, such as shown below. Note that column indices begin at position 0:

bool datalogAddLong(long column, long data), bool datalogAddFloat(long column, float data), bool datalogAddChar(long column, char data).

The parameter **data** will be written in the column indicated by the first parameter.

Once you have finished writing data, you must close your file by using the next function:

bool datalogClose();

An example is shown in (Code 7) and further details are available in the datalog documentation which can be found here:

```
#pragma config(Sensor, S1,    ,sensorEV3_Gyro)

task main()
{
	//create a datalog file called “datalog-0” with 4 columns
	bool datalogOK = datalogOpen(0, 4);
	if(datalogOK){
		//write the value of the gyro sensor in the 3rd column
		datalogAddLong(2, getGyroDegrees(S1));
		datalogClose();
	}

}
```

CODE 7: USING DATA LOGGING

When your programme has finished, use the file management utility to download the files created with the Datalog to the computer.

2.10 File Management

To gather files generated on the brick, you can use the file manager available on the menu: *Robot* ⟶ *LEGO Brick* ⟶ *File Management Utility* (Figure 11).

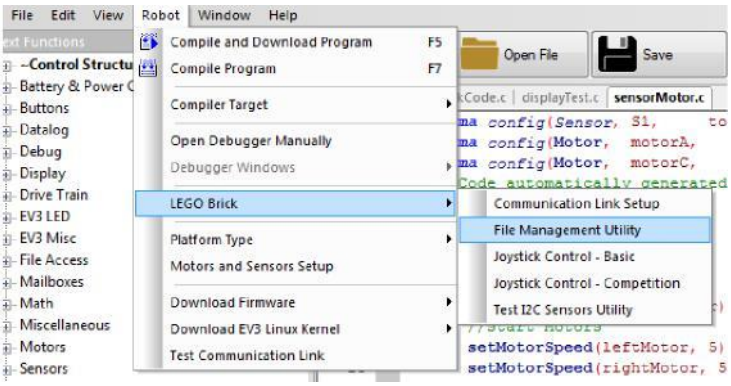

FIGURE 11 FILE MANAGER MENU

This utility lets you upload and download files, Datalog files are located in the folder rc-data.

3 Taking sensor readings via Bluetooth

We developed some purpose-built Ev3 Bluetooth software which allows you to record values measured by the sensors via Bluetooth and export them for example to Excel, by clicking on the icon (Figure 12) to launch it.

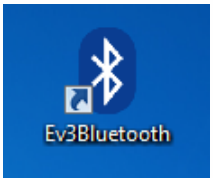

FIGURE 12
EV3BLUETOOTH ICON

3.1 Connection

First you need to activate the Bluetooth on the Ev3 brick (Figure 13), check that "Bluetooth" and "Visibility" are checked on the Ev3 menu.

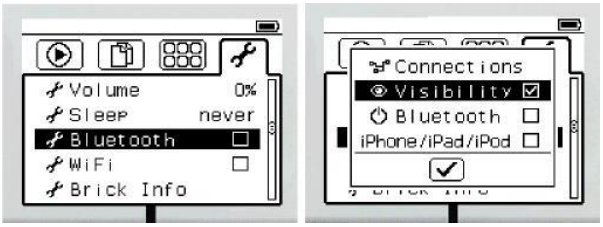

FIGURE 13 ENABLE BLUETOOTH ON THE BRICK

It is then possible to pair the Ev3 brick to the computer as any other Bluetooth device by right clicking on the Bluetooth icon which appears on the task bar when the Bluetooth dongle is inserted and select "Add device", then select the right device and follow the instructions to pair your brick (Figure 13).

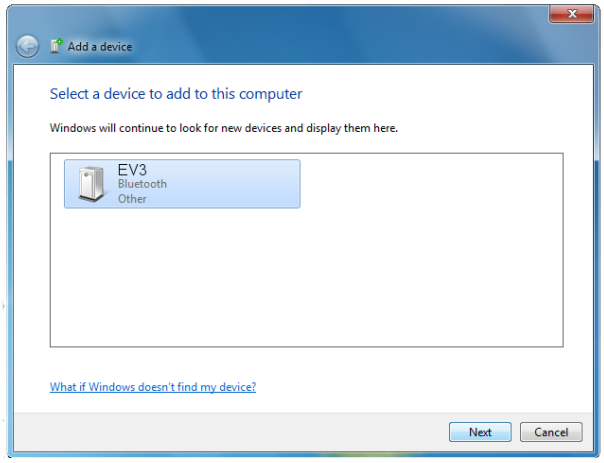

FIGURE 14 ADD A DEVICE

Once your brick has been paired, you will need to get the corresponding COM port on the computer so you can use EV3 Bluetooth software. Find your EV3 in the section "*Devices and printers*" *of* the control panel, right click and select *"Properties"* to find the COM port number in the tab "*Hardware*" (see Figure 15).

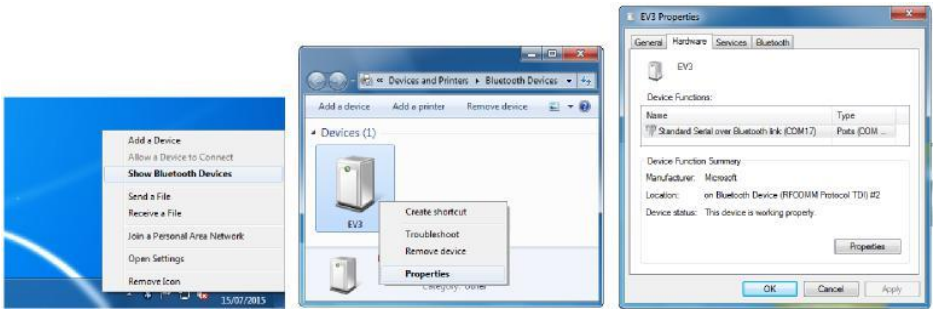

FIGURE 15

You can now run the software and enter this number in the top left corner *of* the window and then click on *"connect"* (Figure 16)**.**

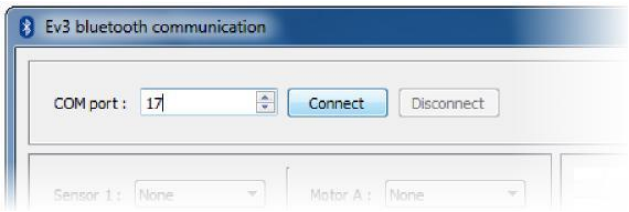

FIGURE 16

3.2 Sensor Readings

Once you are connected, on the right of the window you can see the sensors and motors connected to the Ev3 and the values generated by them. The program should detect them automatically but some sensors like the HiTechnic Color Sensor or the Mindsensors IR sensor are detected as *I2CCustom, so you will* need to adjust their types with the combo box next to each sensor (Figure 17).

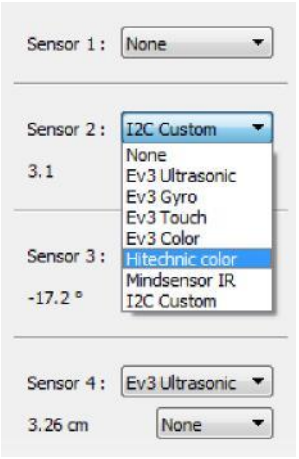

FIGURE 17 SELECT SENSOR

The value displayed for each sensor is the value of the current *mode* of the sensor. For example, if a program is currently using the ultrasonic sensor on the EV3 in the mode *centimeter,* the value displayed will also be in centimeters.

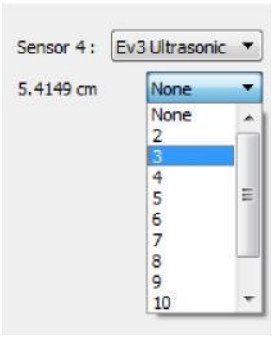

FIGURE 18 SELECT COLUMN

If you want to record the values measured by a given sensor, you have to choose the column in which they are going to be displayed in the table with the combo box next to the desired value (Figure 18). Once you have chosen a column for all the sensors that you want, you can record the values.

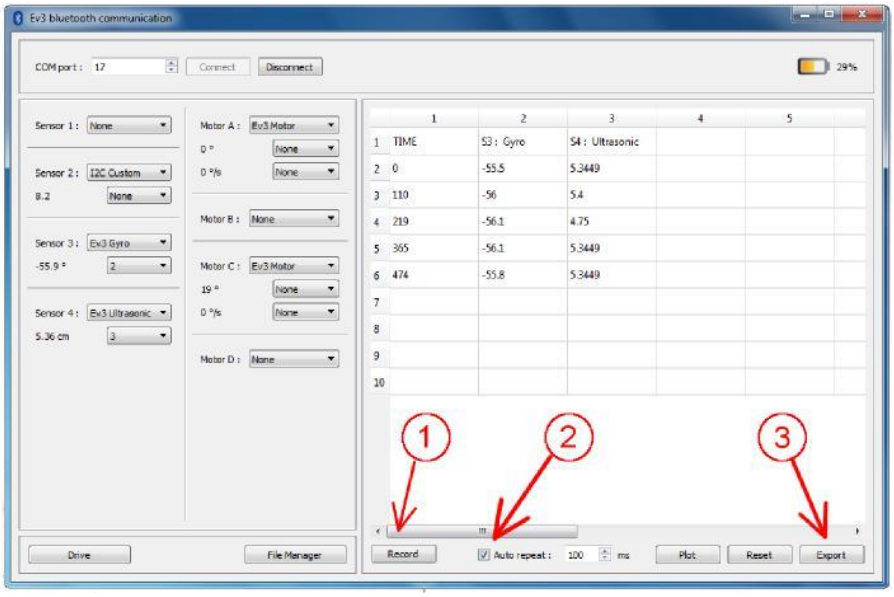

FIGURE 19 EV3BLUETOOTH INTERFACE

- When you click on the button *record* (1), you add the current values of all the sensors selected as a new row of the table.
- If the *Auto – repeat* box (2) is checked when you click on *record*, the program will automatically record new readings at regular interval (adjustable with the spin box) until you click on *stop*.
- You can export these data to .csv (with the export button (3)) and edit the file with excel.

It is also possible to draw plots by clicking on "plot" of the readings taken and to drive the Robot with the keyboard arrows by clicking on "drive" (Figure 20)**.**

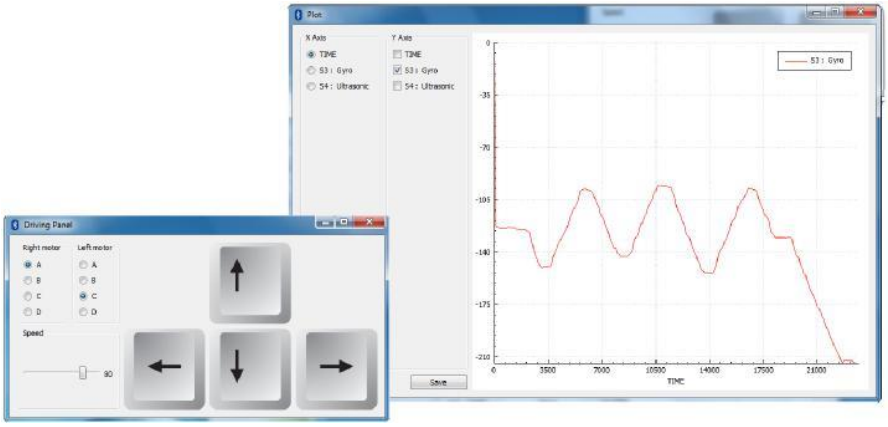

FIGURE 20

4 Using Bluetooth with RobotC

RobotC allows downloading programs and debugging by Bluetooth instead of USB. First, pair the EV3 to the computer (See 3.1). Then, in RobotC, click on *Robot → LEGO Brick → Communication Link Setup*. The following wizard will appear (Figure 21).

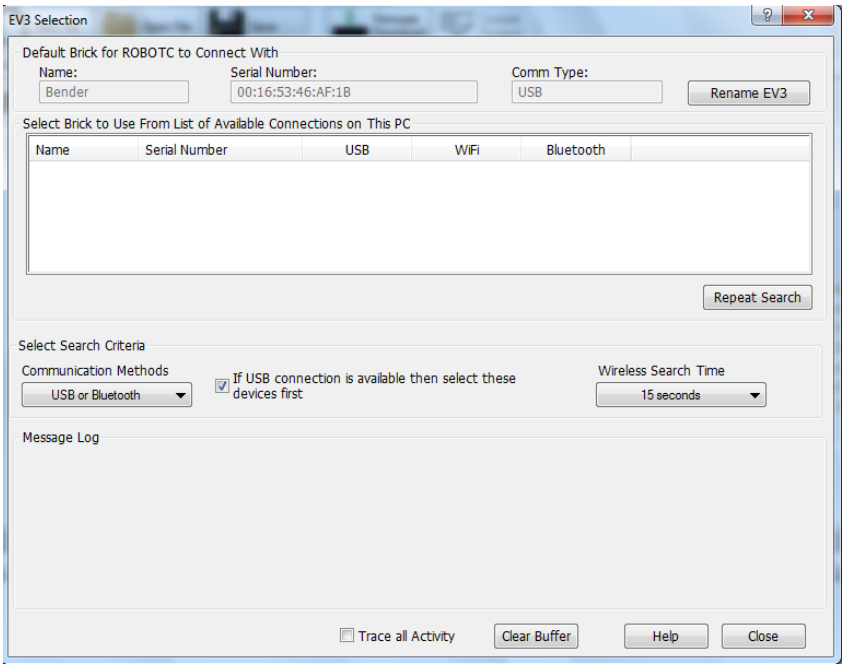

FIGURE 21: COMMUNICATION LINK SETUP

Check that the option chosen in Communication methods contains Bluetooth. To refresh, click on *Repeat Search*. The robot will appear on the list of detected robots (Figure 22).

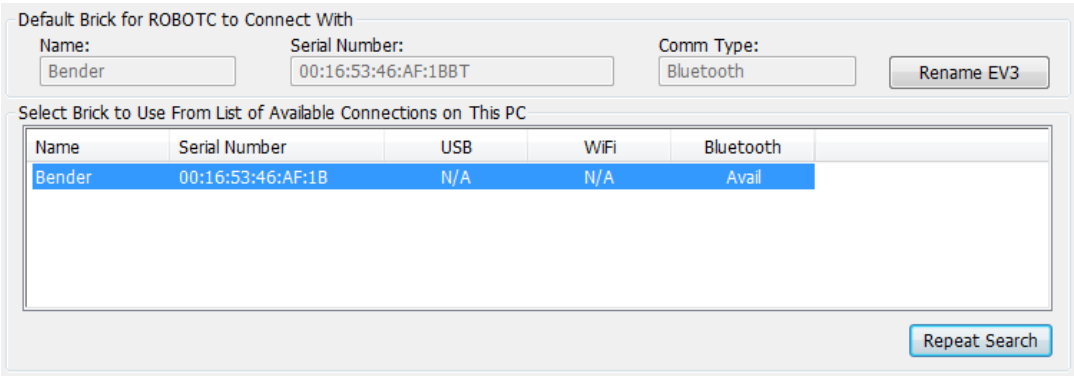

FIGURE 22: CHOOSING ROBOT

Click once on the brick name to set it as the default brick to connect with. Then you can close the wizard and download programs by Bluetooth. RobotC tools for debugging work by Bluetooth.

5 FAQ

1. **How can I get sensor readings?**
   In the example shown below you can see how to get the sensor readings of Ultrasonic, Touch and Gyros. However, for more information visit the RobotC API online (see question 4).

```
#include "HitechnicColorSensor.h"
//Get gyros, US, touch and light sensor readings

int gyro = getGyroDegrees(S1); // read Gyro with Sensor 1

int d =  getUSDistance(S2); // read US with Sensor 2

int t = getTouchValue(S3); // read Touch with Sensor 3

long r,g,b;
HTCS2readRawRGB(S4,true,r,g,b); // read RGB with Sensor 4
displayTextLine(0, "R: %5d", r); // print red value
displayTextLine(2, "G: %5d", g); // print green value
displayTextLine(4, "B: %5d", b); // print blue value
```

CODE 8 CODE SAMPLE FOR GETTING SENSOR READINGS

2. **How can I generate a random number?**
   Random integral numbers can be generated in the range to 0 to *n* by using the command **random** such as shown in the example below. Note: the range is **inclusive**.

```
//Generates a random number in the interval 0 to 10

int Random = random[10];
```

CODE 9 CODE SAMPLE FOR RANDOM NUMBER GENERATION

3. **How can I see/print a value in the robot screen?**
   You can display a text string on one of the 8 possible text lines by using the command **displayTextLine**. Below is a code sample:

```
//Displays ROBOTC on line 3 in normal text

displayTextLine(3, "ROBOTC");
```

CODE 10 CODE SAMPLE FOR DISPLAYING TEXT IN ROBOT SCREEN

4. **Where can I find more information about Robot C and general programming?**
   A complete RobotC API Guide is available online for programming tips and tricks:
   http://help.robotc.net/WebHelpMindstorms/index.htm
   Also see: http://www.robotc.net/wikiarchive/ (Consider both "General Programming" and "NXT" sections)

   Remember, information on third party sensors is available here:
   http://botbench.com/driversuite/ (sensor-specific information is available under “modules”.