How to command Otto with the voice ?

J. Lemaire

(Pierrefeu Mai 2021)

I recently discoverd OttoDIY+ and was totally amazed by this little Robot, really brilliant by its simplicity and its possibilities, in particular by its approach using only 4 micro servos.

Having been a teacher in a computer environment in another lifetime (I'm now retired), I got caught up in the game and tried to understand how it moves and tried to simplify its Arduino code. I published in https://github.com/jlemaire06/OttoMoves a little Arduino library and a report (Fr and En) about this work.

But, I admit I was a little frustated by the low possibilities of its microphone and so I asked me : « How can I command the robot with my voice » ?

First, I imagine to replace this microphone by a more complex system with speech recognition functionnalities such as https://wiki.seeedstudio.com/Grove-Speech_Recognizer/ or https://www.geeetech.com/wiki/index.php/Arduino_Voice_Recognition_Module. But I observed immediatly their limits. In fact, is it really reasonable to solve with such modules the large problem of speech recognition which generally involves complex deep learning methods ?

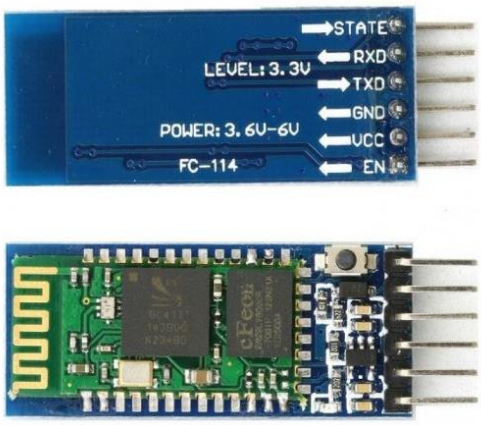
On the other hand, now most of the smarphones offer this possibility. For instance, with Android system (version 5 minimum), the Google Speech-to-Text service enables to enter text just saying it, and this recognition process is possible in a large number of languages : just need an internet access, which is now possible almost everywhere.

Furthermore, as we can install in Otto an Arduino sketch capable of parsing and handling a lot of simple serial commands, received by its Bluetooth module, the only problem is finally to find an application on the smarphone which can access to the previous Google service and send to a Bluetooth device, command texts corresponding to the recognized sentences, with the hability to customize these associations.

Here I found a very simple Android application on Google play which exactly solves exactly the previous problem : **Bluetooth Voice** by yashx :

But it has a restriction : it only works with **Bluetooth 2.0**, and not **Bluetooth LE** used by the module delivered with OttoDIY+. May be there is any other equivalent applications that accept Bluetooth LE, but I don't found such one… So I opted to replace the Otto's Bluetooth module by an « old » **HC-05** I

had (https://www.estudioelectronica.com/wp-content/uploads/2018/09/istd016A.pdf for instance), working with Bluetooth 2.0 (but you can use any other module such as HC-06, BlueSMiRF, etc.) :



There are different versions of the HC-05 module, with or without a push button, shorter or longer, but their main characteristics are the same :

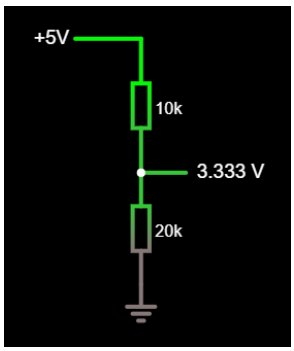
- Bluetooth 2.0 ;
- can be 5V powered ;
- 3.3V logic level on TX/RX.

Let us go now to precise the steps, using this module and my simplified code on github.

1. Install the **HC-05** module in Otto, in place of its Bluetooth LE module, using only its VCC, GND, TXD and RXD pins which are in the same order on these two modules :

HC-05	Nano shield
VCC	V (5V)
GND	G (GND)
TXD	S 11
RXD	S 12

   Here I recommand to reduce the voltage on the S 12 pin, because the Nano has a 5V logic level, using a resistance bridge for instance :



   Inversely, no problem on S 11, because the Nano works perfectly with 3.3V coming from TXD.

2. Using AT commands, set the following parameters of the module :
   - NAME=Otto
   - ROLE=0 (Slave))
   - PSWD=1234
   - UART=9600, 1, 0 (9600 bauds, 1 stopbit, no parity)

To do this, you have first to put the module in **command mode** by pushing its button (if present) or connecting the EN pin to 5V, while powering the module in 5V ; a diode then blinks with a period of 2s to indicate that the module enters in command mode.

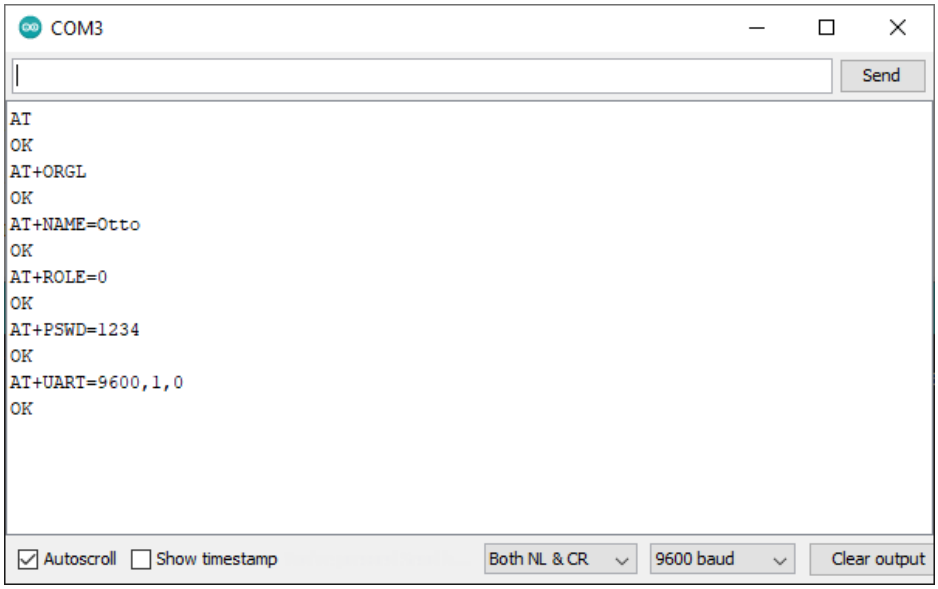
Then you have to send the following **AT commands**, using a serial connection at **38400 bauds** :

- AT (to verify that AT command are accepted)
- AT+ORGL (restore the default factory settings)
- AT+NAME=Otto
- AT+ROLE=0
- AT+PSWD=1234
- AT+UART=9600,1,0

The HC-05 module beeing connected to the Nano shield, the simplest way to send these commands consists in installing a serial communication sketch in Otto such as :

```
// SendAT.ino

#include <SoftwareSerial.h>

SoftwareSerial BTSerial(11,12);

void setup()
{
  Serial.begin(9600);
  BTSerial.begin(38400);
  delay(100);
}

void loop()
{
  while(BTSerial.available()>0)
  {
    Serial.print((char)BTSerial.read());
  }
  while(Serial.available()>0)
  {
    char c = (char)Serial.read();
    BTSerial.print(c);
    Serial.print(c); // For local echo
  }
}
```

and using the serial monitor :



3. Switch off the robot and after switch it on again, to put the HC-05 module in **communication mode** with these settings.

4. Load a Bluetooth command sktech in the Nano. Obviously you can use here **Blockly** an select :

In the Otto section. You can also load the **Otto_APP_V10.ino** sketch in the **OttoDIYLib** library on github (https://github.com/OttoDIY/OttoDIYLib).

But this code is a bit complex, and to well understand the Otto Bluetooth command mode, I suggest you to use first the Arduino IDE and my code in github (https://github.com/jlemaire06/OttoMoves), which is much more simple :

- Add my **Robot** library which defines and implements the **Movement**, **ServoE**, **TServoE**, **Robot**, **SerialCommand** and **RobotSC** classes.
- Load the **TestRobotBT.ino** sketch in the Nano :

```
// TestRobotBT.ino

#include <RobotSC.h>
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(11,12);
RobotSC r(2, 3, 4, 5, BTSerial);

void setup()
{
  BTSerial.begin (9600);
  delay(100);
}

void loop()
{
  r.processCommand();
  r.update();
}
```

Looking at the documentation, the **Robot** and the **RobotSC** classes, you can observe that I only have proposed methods and commands to move the robot :

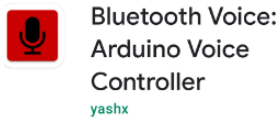
- C1# => on (attach the servos)
- C0# => off (detach the servos)
- M0# => home
- M1# => walk forward
- M2# => walk backward
- M3# => turn left
- M4# => turn right
- M13# => walk forward and turn left
- M14# => walk forward and turn right
- M23# => walk backward and turn left
- M24# => walk backward and turn right
- S1# => set speed low
- S2# => set speed normal

- S3# => set speed rapid
- I1# => stop (interrupt the movement)
- I0# => start (again)

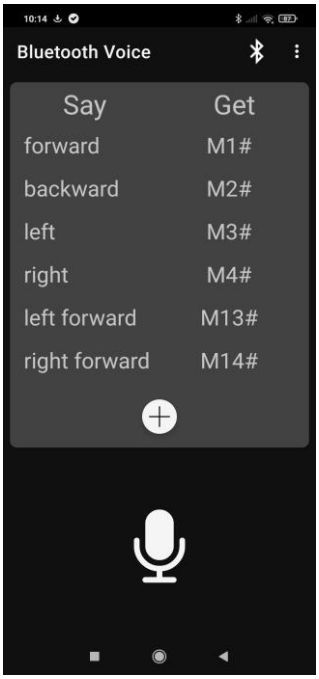
But, having understood how Otto moves and specially the oscillation concept, its very easy to add other movement methods or sensors.

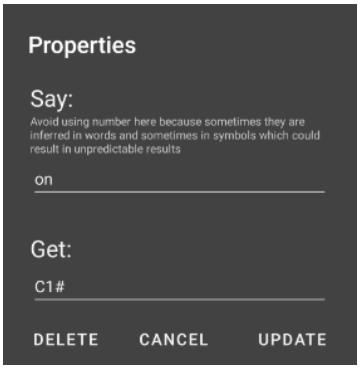
5. On the smartphone (here an Xiaomi MI9 SE), activate the Bluetooth communication functionnality and associate the **HC-05** module (named **Otto**), entering the pin code : **1234**.

6. Install and open the **Bluetooth Voice** application from Google play :

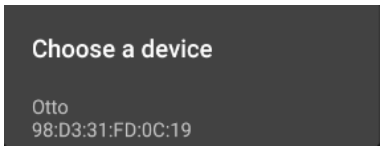


- Add the previous commands, using the Properties dialog box :





- Connect to Otto



7. Clic on :

and say a command, « **On** » for instance : when recognized, **C1#** is transmitted to the robot, which attaches the 4 servos.

8. Do the same with the other commands and enjoy !

Remarks

- If you prefer to use the **OttoDIY** library and the **Otto_APP_V10.ino** sketch, you can process in a similar way, after having understood what are the acceptable commands. You can discover simply the strings to be sent by the Smartphone application using the **Otto DIY** application on it and the **Otto9_BlueTest.ino** in the Nano (the received strings are displayed on the serial monitor).
- Obviously, the **Bluetooth voice** application could be improved and tailored ; for instance on accepting the Bluetooth LE protocol, avoiding the need to clic on the micro at each command, or enabling the possibility to send commands both with the voice and with clic on Otto icons : may be a subject for a future development…