Thymio and Python

Alessandro zonta

August 11, 2015

1 Thymio and Python

With this manual you will able to connect one Thymio to a computer and control it using Python script.

Figure 1: Thymio robot

To complete this manual you will need more or less 1 hour. With the python script provided with this manual you will control everything of the robot using only the keyboard.

2 Connect Thymio to a computer

To program your Thymio, you first need to download Aseba. Aseba is available on Windows [1] , Linux [2] , Mac OS [3] and Android [4].

Aseba provides *asebamedulla*, a command-line utility that allows you to access an Aseba robot through D-Bus. This permits the use of Aseba-enabled devices from third-party languages such as Python, Perl or any language that supports the D-Bus protocol. *asebamedulla* has only been tested under Linux. If you are using Windows or Mac OS you can only control Thymio II using Aseba Studio, using a Raspberry Pi or using Linux in a Virtual Machine.

To control your Thymio using Python you need medulla. Medulla is installed in your PC with Aseba package. Open a terminal and connect medulla with your Thymio:

```
$ sudo asebamedulla "ser:name=Thymio-II"
```

If Thymio is connected to your PC by USB cable, in the terminal will appear this message:

```
$ Found Thymio-II on port /dev/ttyACM0
```

Now Thymio is connected to your PC. This command recognize only **one** robot although there are more Thymio plugged in. To disconnect Thymio you only need to type *ctrl+c* in the Terminal. It is possible that the console returns some error. If the error is this:

```
$ pi@testthymio ~ $ asebamedulla "ser:name=Thymio-II"

$ terminate called after throwing an instance of
'Dashel::DashelException'
$ what():  Cannot bind socket to port, probably the
port is already in use.

$ Aborted
$ pi@testthymio ~ $
```

you will need to kill all the asebamedulla process before try again:

```
$ sudo killall -9 asebamedulla
```

3 Send Python command to Thymio

Now is possible to send command through Python, but you need another Terminal. You cannot use the same Terminal you used to connect with the Thymio because, if you write something in that Terminal, the connection will close.
If you want use only one terminal you can run the command in background:

```
$ (sudo asebamedulla "ser:name=Thymio-II" &)
```

To execute some Python code you need only a script and know how to execute it:

```
$ sudo python test.py
```

In A you find test.py. This is only an example script that needs also test.aesl to works. Later we will explain why you need also this file. You can also make your own script and execute it using this command.

Inside Python script, you need to connect to **D-Bus** in order to communicate with the robot. This code connects you with Thymio and create the object that you will use to send a command. In this case the object is *robot*

```
parser = OptionParser()
parser.add_option("-s", "--system", action="store_true", dest="system",
 default=False, help="use the system bus instead of the session bus")

(options, args) = parser.parse_args()

dbus.mainloop.glib.DBusGMainLoop(set_as_default=True)
```

```
if options.system:
bus = dbus.SystemBus()
else:
bus = dbus.SessionBus()

# Create Aseba network
robot = dbus.Interface(bus.get_object('ch.epfl.mobots.Aseba', '/'),
 dbus_interface='ch.epfl.mobots.AsebaNetwork')

# print in the terminal the name of each Aseba Node
print robot.GetNodesList()
```

You can use these three methods to control the robot:

- *SetVariable*
- *GetVariable*
- *SendEventName*

3.1 SetVariable

To move the robot, you have to use the *SetVariable* method. **motor.side.target** is used to specify the speed we want the motors to have. The values range from -500 to 500. A value of 500 approximately corresponds to a linear speed of 20 cm/s. We use *Time.sleep(1)* to stop the robot after 1 second.

```
robot.SetVariable("thymio-II", "motor.left.target", [300])
robot.SetVariable("thymio-II", "motor.right.target", [300])
time.sleep(1)
robot.SetVariable("thymio-II", "motor.left.target", [0])
robot.SetVariable("thymio-II", "motor.right.target", [0])
```

3.2 GetVariable

To obtain the value from sensors exist the method called *GetVariable*. With this method, you can retrieve information from distance sensors, accelerometer, temperature sensor and microphone intensity.
You can find the specification of all these sensors in Thymio wiki [5].
All the parameters inside GetVariable method are mandatory. The first parameter is the name of the robot that we are using (usually is thymio-II), second parameter is which information we want to know and the last two parameters are the handler for the result and if some errors are raised.

```
robot.GetVariable("thymio-II", "prox.horizontal", reply_handler=
      get_variables_reply, error_handler=get_variablacces_error)
def get_variables_reply(r):
#inside r you can find the value from the selected sensor

def get_variables_error(e):
print 'error:'
print str(e)
#put here code to manage the error
```

To retrieve information from other sensors you need only to change the name inside the method. This is all the possibility:

- prox.horizontal: distance sensors
- acc: 3-axes accelerometer
- prox.ground.delta: ambient light intensity at the ground,
- prox.ground.reflected: amount of light received when the sensor emits infrared
- prox.ground.ambiant: difference between reflected light and ambient light, linked to the distance and to the ground colour.
- temperature: current temperature in tenths of a degree Celsius
- mic.intensity: current microphone intensity

3.3 SendEventName

To control sounds and lights exist a method called *SendEventName* but it is different from the previous methods. You cannot control these functions with a direct interaction with them. You have to write an *aesl* file where you define the event and what action you want the robot will do. It is possible to write all the instruction that you want, also the one that moves the robot after one specific event. To control the light and the sound that Thymio can emit, you can define three event:

```
onevent SetColor
call leds.top(event.args[0], event.args[1], event.args[2])
call leds.bottom.left(event.args[0], event.args[1], event.args[2])
call leds.bottom.right(event.args[0], event.args[1], event.args[2])

onevent PlaySound
call sound.system(event.args[0])

onevent PlayFreq
call sound.freq(event.args[0], event.args[1])
```

You can find the specification of all these calls in Thymio wiki webpage. To use this event you have to load the *aesl* file inside Thymio. This is not difficult, only one line of code:

```
AESL_PATH = #insert here the path of the *.aesl file
robot.LoadScripts(AESL_PATH, reply_handler=dbusReply,
    error_handler=dbusError)

def dbusReply():
pass

def dbusError(e):
print 'error %s'
print str(e)
```

After load, you can call the event from python code like the previous method:

```
robot.SendEventName('SetColor',[randint(0, 32), randint(0, 32),
     randint(0, 32)], reply_handler=dbusReply, error_handler=dbusError)

robot.SendEventName('PlaySound', [4], reply_handler=dbusReply,
     error_handler=dbusError)

robot.SendEventName('PlayFreq', [700, 0], reply_handler=dbusReply,
     error_handler=dbusError)
```

If you want to call the same event more than one time, the use of *time.sleep()* is mandatory. This is necessary because you have to tell Thymio to wait until the end of the previous event before starts another one.

4 Python Example

In Appendix A, you can find a simple example of a python program. The command available in this script are:

- w: go forward
- a: turn left
- s: go backwards
- d: go right
- p: play sound (all the thymio sounds)
- o: play sound (two hardcoded frequences)
- i: show all the led

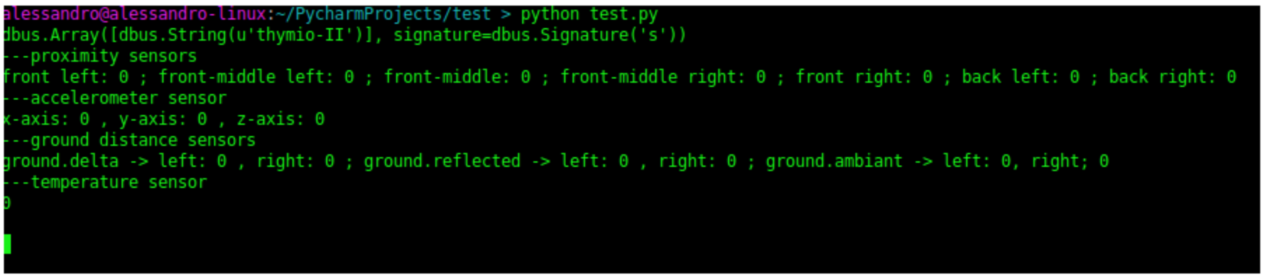
After every command, this example will print all the value from all the available sensors.

4.1 How it works?

How you can see in Appendix A, this script needs some library to work. The following libraries are the most important:

- dbus
- dbus.mainloop.glib
- gobject

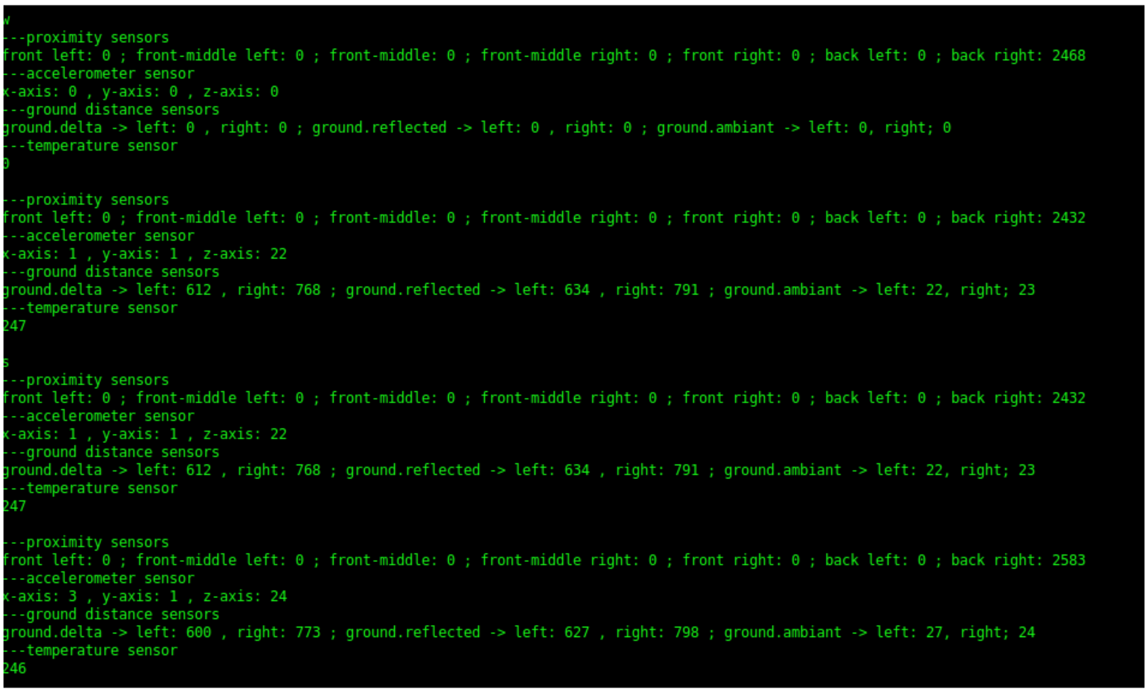
There are another libraries that the script needs but are very common and everyone should know how they work.
We need the first two libraries (D-Bus is a message bus system, a simple way for applications to talk to one another.) to let the computer talk with the Thymio. The algorithm is one big loop running every 0.1 sec waiting for some input. We need *gobject* to run the algorithm every 0.1 sec.



```
alessandro@alessandro-linux:~/PycharmProjects/test > python test.py
dbus.Array([dbus.String(u'thymio-II')], signature=dbus.Signature('s'))
---proximity sensors
front left: 0 ; front-middle left: 0 ; front-middle: 0 ; front-middle right: 0 ; front right: 0 ; back left: 0 ; back right: 0
---accelerometer sensor
x-axis: 0 , y-axis: 0 , z-axis: 0
---ground distance sensors
ground.delta -> left: 0 , right: 0 ; ground.reflected -> left: 0 , right: 0 ; ground.ambiant -> left: 0, right; 0
---temperature sensor
0
```

Figure 2: Script

After executed the code, in your console there should be something like Figure 2. Now it is waiting for some input. You can type every letter you want (only w,a,s,d,p,o,i are working) and after pressed "ENTER" the robot will execute your command. In Figure 3 you can see the results after command "w" and command "a"



```
w
---proximity sensors
front left: 0 ; front-middle left: 0 ; front-middle: 0 ; front-middle right: 0 ; front right: 0 ; back left: 0 ; back right: 2468
---accelerometer sensor
x-axis: 0 , y-axis: 0 , z-axis: 0
---ground distance sensors
ground.delta -> left: 0 , right: 0 ; ground.reflected -> left: 0 , right: 0 ; ground.ambiant -> left: 0, right; 0
---temperature sensor
0

---proximity sensors
front left: 0 ; front-middle left: 0 ; front-middle: 0 ; front-middle right: 0 ; front right: 0 ; back left: 0 ; back right: 2432
---accelerometer sensor
x-axis: 1 , y-axis: 1 , z-axis: 22
---ground distance sensors
ground.delta -> left: 612 , right: 768 ; ground.reflected -> left: 634 , right: 791 ; ground.ambiant -> left: 22, right; 23
---temperature sensor
247


s
---proximity sensors
front left: 0 ; front-middle left: 0 ; front-middle: 0 ; front-middle right: 0 ; front right: 0 ; back left: 0 ; back right: 2432
---accelerometer sensor
x-axis: 1 , y-axis: 1 , z-axis: 22
---ground distance sensors
ground.delta -> left: 612 , right: 768 ; ground.reflected -> left: 634 , right: 791 ; ground.ambiant -> left: 22, right; 23
---temperature sensor
247

---proximity sensors
front left: 0 ; front-middle left: 0 ; front-middle: 0 ; front-middle right: 0 ; front right: 0 ; back left: 0 ; back right: 2583
---accelerometer sensor
x-axis: 3 , y-axis: 1 , z-axis: 24
---ground distance sensors
ground.delta -> left: 600 , right: 773 ; ground.reflected -> left: 627 , right: 798 ; ground.ambiant -> left: 27, right; 24
---temperature sensor
246
```

Figure 3: Script

References

[1] Windows Aseba Download, `https://www.thymio.org/en:wininstall`

[2] Linux Aseba Download, `https://www.thymio.org/en:linuxinstall`

[3] Mac OSX Aseba Download, `https://www.thymio.org/en:macinstall`

[4] Android Aseba Download, `https://www.thymio.org/en:androidinstall`

[5] Thymio II wiki, `https://www.thymio.org/en:thymioapi`

A Python Script

```
import dbus
import dbus.mainloop.glib
import gobject
import os
from random import randint
import sys
import time
from optparse import OptionParser
import picamera

proxSensorsVal = [0, 0, 0, 0, 0, 0, 0]
accVal = [0, 0, 0]
ground = [0, 0]
ref = [0, 0]
amb = [0, 0]
temp = [0]
mic = [0]
CURRENT_FILE_PATH = os.path.abspath(os.path.dirname(__file__))
AESL_PATH = os.path.join(CURRENT_FILE_PATH, 'test.aesl')


def Test():
    # get the values of the sensors
    robot.GetVariable("thymio-II", "prox.horizontal", reply_handler=get_variables_
                      error_handler=get_variables_error)

    robot.GetVariable("thymio-II", "acc", reply_handler=accget_variables_reply,
                      error_handler=get_variables_error)

    robot.GetVariable("thymio-II", "prox.ground.delta", reply_handler=
        deltaget_variables_reply, error_handler=get_variables_error)
    robot.GetVariable("thymio-II", "prox.ground.reflected", reply_handler=
        reftaget_variables_reply, error_handler=get_variables_error)
    robot.GetVariable("thymio-II", "prox.ground.ambiant", reply_handler=
        ambtaget_variables_reply, error_handler=get_variables_error)

    robot.GetVariable("thymio-II", "temperature", reply_handler=
        tempget_variables_reply, error_handler=get_variables_error)
    robot.GetVariable("thymio-II", "mic.intensity", reply_handler=
        micget_variables_reply, error_handler=get_variables_error)

    # print the proximity sensors value in the terminal
    print "---proximity sensors"
    print "front left: %d ; front-middle left: %d ; front-middle: %d ; front-middl
        front right: %d ; back left: %d ; back right: %d " % ( proxSensorsVal[0],
        proxSensorsVal[1], proxSensorsVal[2], proxSensorsVal[3], proxSensorsVal[4
         proxSensorsVal[5], proxSensorsVal[6])
```

```
# print accelerometer sensor
print "---accelerometer sensor"
print "x-axis: %d , y-axis: %d , z-axis: %d " % (accVal[0], accVal[1], accVal

# print ground
print "---ground distance sensors"
print "ground.delta -> left: %d , right: %d ; ground.reflected -> left: %d ,
 right: %d ; ground.ambiant -> left: %d, right; %d  " % (
    ground[0], ground[1], ref[0], ref[1], amb[0], amb[1])

# print temp
print "---temperature sensor"
print temp[0]

# print sound intensitivi
print "---microphone intensity"
print mic[0]

print " "

char = sys.stdin.read(1)
if char == 'p':
    for x in range(0, 7):
        robot.SendEventName('PlaySound', [x], reply_handler=dbusReply,
                 error_handler=dbusError)
        time.sleep(1)
if char == 'l':
    for x in range(0, 32):
        robot.SendEventName('SetColor',
                             [randint(0, 32), randint(0, 32), randint(0, 32)],
                             reply_handler=dbusReply,
                             error_handler=dbusError)
        time.sleep(0.2)
    robot.SendEventName('SetColor', [0, 0, 0], reply_handler=dbusReply,
                        error_handler=dbusError)
if char == 'o':
    robot.SendEventName('PlayFreq', [700, 0], reply_handler=dbusReply,
         error_handler=dbusError)
    time.sleep(1.5)
    robot.SendEventName('PlayFreq', [700, -1], reply_handler=dbusReply,
         error_handler=dbusError)
    time.sleep(0.1)
    robot.SendEventName('PlayFreq', [1000, 0], reply_handler=dbusReply,
        error_handler=dbusError)
    time.sleep(1.5)
    robot.SendEventName('PlayFreq', [1000, -1], reply_handler=dbusReply,
        error_handler=dbusError)
if char == 'w':
    robot.SetVariable("thymio-II", "motor.left.target", [300])
    robot.SetVariable("thymio-II", "motor.right.target", [300])
    time.sleep(1)
```

```
            robot.SetVariable("thymio-II", "motor.left.target", [0])
            robot.SetVariable("thymio-II", "motor.right.target", [0])
        if char == 's':
            robot.SetVariable("thymio-II", "motor.left.target", [-300])
            robot.SetVariable("thymio-II", "motor.right.target", [-300])
            time.sleep(1)
            robot.SetVariable("thymio-II", "motor.left.target", [0])
            robot.SetVariable("thymio-II", "motor.right.target", [0])
        if char == 'a':
            robot.SetVariable("thymio-II", "motor.left.target", [-300])
            robot.SetVariable("thymio-II", "motor.right.target", [300])
            time.sleep(0.2)
            robot.SetVariable("thymio-II", "motor.left.target", [0])
            robot.SetVariable("thymio-II", "motor.right.target", [0])
        if char == 'd':
            robot.SetVariable("thymio-II", "motor.left.target", [300])
            robot.SetVariable("thymio-II", "motor.right.target", [-300])
            time.sleep(0.2)
            robot.SetVariable("thymio-II", "motor.left.target", [0])
            robot.SetVariable("thymio-II", "motor.right.target", [0])
        if char == "f":
            camera = picamera.PiCamera()
            camera.hflip = True
            camera.vflip = True
            camera.start_preview()
            time.sleep(2)
            camera.capture('image.jpg')

        return True


    def get_variables_reply(r):
        global proxSensorsVal
        proxSensorsVal = r


    def accget_variables_reply(r):
        global accVal
        accVal = r


    def deltaget_variables_reply(r):
        global ground
        ground = r


    def tempget_variables_reply(r):
        global temp
        temp = r
```

```
def reftaget_variables_reply(r):
    global ref
    ref = r


def ambtaget_variables_reply(r):
    global amb
    amb = r


def micget_variables_reply(r):
    global mic
    mic = r


def get_variables_error(e):
    print 'error:'
    print str(e)
    loop.quit()


def dbusReply():
    pass


def dbusError(e):
    print 'error %s'
    print str(e)


if __name__ == '__main__':
    parser = OptionParser()
    parser.add_option("-s", "--system", action="store_true", dest="system",
        default=False, help="use the system bus instead of the session bus")

    (options, args) = parser.parse_args()

    dbus.mainloop.glib.DBusGMainLoop(set_as_default=True)

    if options.system:
        bus = dbus.SystemBus()
    else:
        bus = dbus.SessionBus()

    # Create Aseba network
    robot = dbus.Interface(bus.get_object('ch.epfl.mobots.Aseba', '/'),
     dbus_interface='ch.epfl.mobots.AsebaNetwork')

    # print in the terminal the name of each Aseba NOde
    print robot.GetNodesList()
```

```
robot.LoadScripts(AESL_PATH, reply_handler=dbusReply,
 error_handler=dbusError)

# GObject loop
# print 'starting loop'
loop = gobject.MainLoop()
# call the callback of test algorithm
handle = gobject.timeout_add(100, Test)  # every 0.1 sec
loop.run()
```