Robot Workshop 2: Python

Introduction

We will connect to the Raspberry Pi computer, that is connected to the GoPiGo, and examine the example files to create an autonomous robot, using Python. Our connection will be via a command line interface (CLI), rather than a GUI interface.

Skills required

Before this workshop you will need to have:

- Understand some basic programming ideas e.g. loops, if-then-else, variables (If you've done a few Scratch projects, you'll probably know these)
- Understand the concept of Python functions (If you've done the Code Club 'Teaching turtles' project you'll be ok)

It may also be helpful to have:

- Linux command line experience e.g. changing directories, listing what files are there
- Command-line text editor experience e.g nano

Summary – the basic steps

The steps we will take are:

- Connect to the GoPiGo Robot
- Run an example Python script
- Identify useful code in the example script
- Create and run our own Python script

Connect to the GoPiGo robot

Connect to the network and connect to the robot via ssh from iterm/terminal in Mac OS X.

Once connected via ssh, you can type `ls` to see the files and directories.

```
pi@dex:~ $ ls
cc_test     error_log     nohup.out     python_games     Videos
Desktop     index.html    oldconffiles  Scratch          wifi
di_update   index.html.1  Pictures      selected_state   wpa_supplicant.conf
Documents   interfaces    Public        Templates
Downloads   Music         python        update_backup.sh
```

Use the `cd` command to change to the directory Desktop/GoPiGo/Software/Python. The commands are case-sensitive. You can use the tab key for autocomplete, so you only have to type a few characters. Again, use `ls` to list the files and directories.

```
pi@dex:~ $ cd Desktop/GoPiGo/Software/Python
pi@dex:~/Desktop/GoPiGo/Software/Python $ ls
basic_test_all.py  enc_val_read.py  hardware_test_2.py  other_scripts    tests
build              Examples         hardware_test.py    README.md
control_panel      GoPiGo.egg-info  ir_remote_control   sensor_examples
dist               gopigo.py        line_follower       setup.py
```

Notice the file named `gopigo.py`. This file contains some functions we'll use in our examples, with the `import` python command.

Looking into the gopigo functions

As you may already know, we have built-in functions like `print()`, but we also can create custom functions. These make our code easier to read, and save us re-writing code that is used lots of times.

We can use the less command to have a quick look at some of the functions in `gopigo.py`.

```
less gopigo.py
```

This will let us read the `gopigo.py` script without accidently editing it. Scroll the file up and down using the arrow keys, or page up and down by holding down the control key, and using `f` and `b` (forwards and back).

Scroll down to where you see the `import` statements. These import other build-in functions that are already written.

```
import serial, time
import smbus
import math
import RPi.GPIO as GPIO
import struct

import smbus
import time
import subprocess
```

Scroll down further to where you see a comment line that says `#Move the GoPiGo forward`. Under this line, you can see a function being defined, that tells the GoPiGo motors to drive the robot forward. The function name is `fwd()`. Don't worry if a lot of the code in this file doesn't make sense right now – we'll just be running the functions from our script.

Now hit the `q` key to quit out of the `less` reader.

Now `cd` to the Examples directory.

```
pi@dex:~/Desktop/GoPiGo/Software/Python $ cd Examples/
pi@dex:~/Desktop/GoPiGo/Software/Python/Examples $ ls
Basic_Robot_Control      Gamepad           PS3_Control
Basic_Robot_Control_GUI  GPS_Bot           README.md
Basic_Servo              GPS_Guided_Robot  Streaming_Video_Example
Browser_Streaming_Robot  LED_Blink         Ultrasonic_Basic_Obstacle_Avoider
Compass_Robot            Mouse_Control     Ultrasonic_Servo
Find_Hole                Office_Cannon
```

Run the example script

You can run the example script, which uses the functions we just looked at. Make sure the robot is on the floor, or a smooth area where it won't get damaged when it moves.

```
pi@dex:~/Desktop/GoPiGo/Software/Python/Examples/Basic_Robot_Control $ python basic_robot.py
This is a basic example for the GoPiGo Robot control
Press:
        w: Move GoPiGo Robot forward
        a: Turn GoPiGo Robot left
        d: Turn GoPiGo Robot right
        s: Move GoPiGo Robot backward
        t: Increase speed
        g: Decrease speed
        x: Stop GoPiGo Robot
        z: Exit

Enter the Command:
```

Try out some of the commands, and then press `z` to exit the script.

Identify useful commands in the example script

To start off, lets have a quick look at the basic_robot.py script using `less` again.

```
pi@dex:~/Desktop/GoPiGo/Software/Python/Examples/Basic_Robot_Control $ less basic_robot.py
```

Read through each line and try to work out how the script works. Try and explain this in plain language, for example, 'unless there's an error, keep asking for a command..'

A quick explanation of the basic_robot.py script

At the start of the basic_robot.py script, you'll see a lot of lines that start with `#`. These are just comments and don't do anything. Scroll down with the arrow keys, until you find the `import` command.

```
from gopigo import *    #Has the basic functions for controlling the GoPiGo Robot
```

You've seen these commands in the previous script we looked at. Here, the `basic_robot.py` script is importing the functions we saw in the previous script.

Further down, you'll see a `while` loop. Inside this loop, the script checks for keys entered, and then runs one of the functions. You may have seen this sort of structure before in Scratch or Python programs.

Press the `q` key once you've finished looking at the `basic_robot.py` script.

Using sensors

Now we can use the same process to look at how the `basic_obstacle_avoid.py` script in ` ~/Desktop/GoPiGo/Software/Python/Examples/Ultrasonic_Basic_Obstacle_Avoider` works.

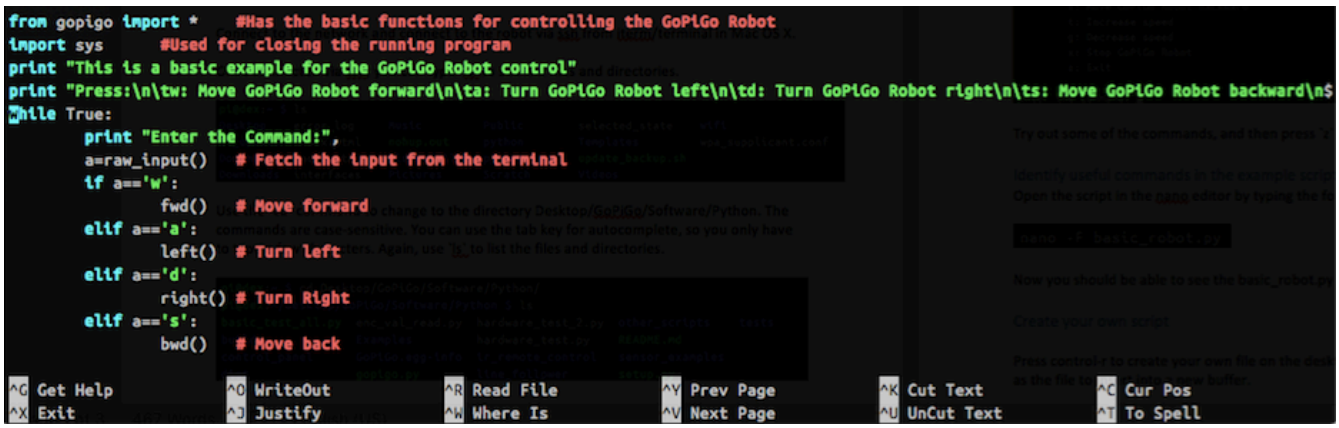
Create your own script

In this section we'll see how to open two scripts in the `nano` editor at the same time; the `basic_robot.py` script, and your own script.

Open the script in the nano editor by typing the following command.

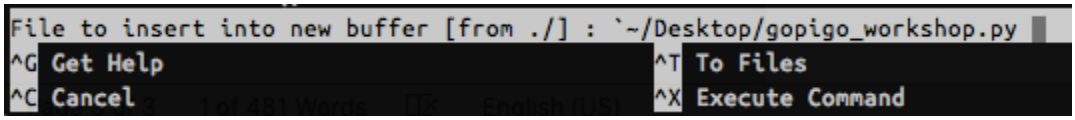
```
nano -F basic_robot.py
```

Now you should be able to see the basic_robot.py script in your `nano` editor.



Press hold down control and press ‘r’ (control-r) to create your own file on the desktop. The nano editor will ask you which file to load. Use *`~/Desktop/gopigo_workshop.py* as the file to insert into a new buffer.

Here’s how it will look at the bottom of the editor. Press enter to create the new file.



Now you can use `alt` with the `,` and `.` keys to switch between both of the files. Hold down `alt` and use the mouse to select text. And you can you use command c to copy and command v to paste in Mac OS X (remember, you’re still in a terminal on OS X).

Hold down `alt` and press `,` to go back to the basic_robot.py script. Be careful not to change anything in this file, since this is what we’re using to learn with.

Copy the import command across to your own file by selecting the text. Hold down the `alt` key and select it with your mouse, then press command-c to copy and command-v to paste. Remember to use `alt` and the `,` and `.` keys to swap between files.

Once you have your own file done, press control-x to save it and follow the prompts. Then press control-x to exit the editor. Now you can run it.