Programming Robots with Python

Shamyl Bin Mansoor
Co-founder & CTO LearnOBots
Is an EdTech company developing educational technology tools

Promotes Science Technology Engineering Arts & Mathematics (STEAM) in Pakistani Schools

Developed a Technology based school curriculum and educational kits for learning to Code, Make Robots, Electronics, Astronomy etc etc

Kids projects have been featured by Hackaday / MIT
Me?

• Teach CS at NUST
• Make “things” at LearnOBots
• Love making / playing with Technology / Hardware / Gadgets  
  • Was part of the recent CERN Hackathon
  • Work has been featured on Hackaday
• Hate publishing research papers, but still have a few
Outline

• Introduction
  • Python for Hardware
  • MicroPython
  • Motivation

• Programming Robots with Python
  • Python on the Raspberry PI
  • Accessing GPIOs on the PI
  • Sensors and Actuators interfacing
  • Programming Robot Movement

• Conclusion
Disclaimer

• Hardware prototype never works when you show it to someone!

“Anything that can go wrong will go wrong”

MURPHY’S LAW
Python for Hardware

• Programming hardware was hard!
  • 8051, C / Assembly
  • High Learning Curve

• Suitable for Hardware?
  • Easy to use
  • Packages available
  • Raspberry Pi

• What about Real time requirements?
  • Trade off
  • Use a dedicated Microcontroller for Real time requirements

• Who uses python with hardware?
MicroPython

- Lean and efficient implementation of Python3
- Small subset of Python Standard Library
- Optimized to run on microcontrollers
Why Robots with Python?

• Python is easy!
• Robots are Fun!
• Lots of Libraries!
Fun + Python = Fun
Robots: What do we need?

• A brain (Processor / Controller)
  • Raspberry PI or an Arduino

• Some Actuators
  • Motors (DC or Servo)

• Some Sensors
  • Sonar / Depth / Vision
What we will Use?

- Raspberry PI since we like Python
- Motors (DC, two of them)
- Sonar Sensor (For sensing obstacles)
Getting started with Raspberry PI

• Buy one! (instock.pk or ewall.com.pk)
• Download Raspbian Image (A Debian based OS for the Raspberry PI)
  • www.raspberrypi.org
• Copy Image to an SD Card
• Plug in Raspberry PI in a TV, connect keyboard and mouse and you’re good to go!
40 Pin GPIO Header

Broadcom BCM 2835 & 512MB RAM

Quad USB Ports

10/100 BaseT Ethernet Socket

DSI Display Connector

Micro SD Card Slot (on underside)

5V Micro USB

HDMI Port

CSI Camera Connector

4-pole 3.5mm jack (stereo audio & composite video)
Accessing GPIO Pins

```python
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.OUT)
```
Python on Raspberry PI

- IDLE and Python Come pre-installed on Raspbian
- We will use our favourite editor to write python scripts
Configuring VNC

• Install VNC viewer from https://www.realvnc.com/ on your MAC or Windows

• Raspbian already comes pre-installed with VNC

• You can connect to your PI over the same network or over the internet
  • Need to create an account with VNC

• Follow https://www.realvnc.com/en/connect/docs/raspberry-pi.html#raspberry-pi-connect-cloud to connect to Raspberry PI
Task 1: Hello World

• Blinking an LED
  • Connect an LED to IO Port of Raspberry PI
  • Make it Blink
  • We will use pin 11 (GPIO 17)
  • And pin 9 (Gnd)
LED Circuit
What’s in the Code?

#Blinking an LED
import RPi.GPIO as GPIO
import time

LedPin = 11  # pin11

def setup():
    GPIO.setmode(GPIO.BOARD)  # Numbers GPIOs by physical location
    GPIO.setup(LedPin, GPIO.OUT)  # Set LedPin's mode is output
    GPIO.output(LedPin, GPIO.HIGH)  # Set LedPin high(+3.3V) to turn on led
BLINK : HELLO WORLD!

```python
def blink():
    while True:
        GPIO.output(LedPin, GPIO.HIGH)  # led on
        time.sleep(0.2)
        GPIO.output(LedPin, GPIO.LOW)  # led off
        time.sleep(0.2)

def destroy():
    GPIO.output(LedPin, GPIO.LOW)    # led off
    GPIO.cleanup()                  # Release resource
```
Main Function

```python
if __name__ == '__main__':  # Program start from here
    setup()
    try:
        blink()
    except KeyboardInterrupt:  # When 'Ctrl+C' is pressed,
        destroy()
```
Output!
LED Blinking: Hellooo World!
Congratulations!

You are a Python Hardware Expert Now!
Task 2: Interfacing Motors

• Using CodiBot we will be using an L298 Hbridge to control the 2 motors for robot movement
• Moving Codi Forwards
• Moving Codi Backwards
• Turning Codi using Differential Drive
Have you ever tried this?
H-Bridge

• Changing Directions
Differential Drive
# Import required libraries
import sys
import time
import RPi.GPIO as GPIO

GPIO.cleanup()

# Define GPIO signals to use
# Physical pins 29,31,33,35,37,40
RightMotor = 29
RM1a = 31
RM1b = 33

LeftMotor = 40
LM1a = 35
LM1b = 37
```python
def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(RightMotor, GPIO.OUT)
    GPIO.setup(LeftMotor, GPIO.OUT)
    GPIO.setup(RM1a, GPIO.OUT)
    GPIO.setup(RM1b, GPIO.OUT)
    GPIO.setup(LM1a, GPIO.OUT)
    GPIO.setup(LM1b, GPIO.OUT)
```
def forward(x):
    # H-Bridge Pin Settings
    GPIO.output(RM1a, GPIO.HIGH)
    GPIO.output(RM1b, GPIO.LOW)
    GPIO.output(LM1a, GPIO.HIGH)
    GPIO.output(LM1b, GPIO.LOW)

    # Turning Motors ON
    GPIO.output(RightMotor, GPIO.HIGH)
    GPIO.output(LeftMotor, GPIO.HIGH)
    print("Moving Forward")
    time.sleep(x)
    GPIO.output(RightMotor, GPIO.LOW)
    GPIO.output(LeftMotor, GPIO.LOW)
def reverse(x):
    # H-Bridge Pin Settings
    GPIO.output(RM1a, GPIO.LOW)
    GPIO.output(RM1b, GPIO.HIGH)
    GPIO.output(LM1a, GPIO.LOW)
    GPIO.output(LM1b, GPIO.HIGH)

    # Turning Motors On
    GPIO.output(RightMotor, GPIO.HIGH)
    GPIO.output(LeftMotor, GPIO.HIGH)

    print("backwarding running motor")
    time.sleep(x)

    GPIO.output(RightMotor, GPIO.LOW)
    GPIO.output(LeftMotor, GPIO.LOW)
```python
def dance(x):
    # H-Bridge Pin Settings
    GPIO.output(RM1a, GPIO.LOW)
    GPIO.output(RM1b, GPIO.HIGH)
    GPIO.output(LM1a, GPIO.HIGH)
    GPIO.output(LM1b, GPIO.LOW)

    # Turning Motors On
    GPIO.output(RightMotor, GPIO.HIGH)
    GPIO.output(LeftMotor, GPIO.HIGH)

    print("Dancing!")
    time.sleep(x)

    GPIO.output(RightMotor, GPIO.LOW)
    GPIO.output(LeftMotor, GPIO.LOW)

def destroy():
    print("Stopping motor")
    GPIO.cleanup()
```
## Motor Connections H-Bridge

<table>
<thead>
<tr>
<th>HIGH</th>
<th>LOW</th>
<th>HIGH</th>
<th>LOW</th>
<th>HIGH</th>
<th>LOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW</td>
<td>HIGH</td>
<td>LOW</td>
<td>HIGH</td>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>HIGH</td>
<td>LOW</td>
<td>LOW</td>
<td>HIGH</td>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>LOW</td>
<td>HIGH</td>
<td>HIGH</td>
<td>LOW</td>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>CODI will move FORWARDS</td>
<td>CODI will move BACKWARDS</td>
<td>CODI will move LEFT</td>
<td>CODI will move RIGHT</td>
<td>CODI won’t move</td>
<td>CODI won’t move</td>
</tr>
</tbody>
</table>
Motors Movement
Task 3: Sensing the world

• Interfacing a Sonar Sensor with Raspberry PI
• Using sonar to detect obstacles
#Sonar interface with the Raspberry PI

#Import Python libraries
import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)

GPIO_TRIGGER = 29  #GPIO_24
GPIO_ECHO = 40  #GPIO_25

#Set Pins as output and input
GPIO.setup(GPIO_TRIGGER,GPIO.OUT)  # Trigger
GPIO.setup(GPIO_ECHO,GPIO.IN)  # Echo

#Set Trigger low
GPIO.output(GPIO_TRIGGER, False)

#Allow module to settle
time.sleep(0.5)
def sonar():
    #Send 10us pulse to trigger
    GPIO.output(GPIO_TRIG, True)
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIG, False)

    while GPIO.input(GPIO_ECHO)==0:
        start = time.time()

    while GPIO.input(GPIO_ECHO)==1:
        stop = time.time()

    #Calculate pulse length
    elapsed = stop-start

    #Distance pulse traveled in that time is time
    #multiplied by the speed of sound (cm/s)
    distance = elapsed * 34000

    #That was the total distance so half it for reaching the object
    distance = distance / 2

    return distance

while True:
    time.sleep(0.3)
    distance = sonar()
    print(distance)
Task 4: Obstacle Avoidance Robot

• Using our learning from the previous 3 tasks we can now program an obstacle avoidance robot.

• Robot moves randomly and turns as soon as it encounters an obstacle

• But let’s leave that task for you ☺️
Conclusion

• Python is Easy
• Lots of library support
• Raspberry PI is a natural tool to develop applications that use Python / Hardware and a GUI
Thank You

Find Today’s Code on
https://github.com/shamyl/Pycon

• t/@shamylmansoor
• shamyl@learnobots.com
• fb.com/learnobots