G54MRT Mixed Reality Technology

Coursework Technology Overview

Joe Marshall, Stuart Reeves

Technology for the coursework

- **Use Python (if you want help from us)**
- Raspberry PI
- GrovePI + Sensors, display
- GrovePI Emulator
- The internet

Raspberry PI

- Single board computer
- Runs Linux
- Network connected
- Headless (no keyboard, mouse, display)
  - Use another computer to talk to it (via SSH)
  - Output via network somehow (or grovePI display)
- Available in lab sessions only
  - Shared between people
    - Play nicely
    - Look at who is using a similar set of sensors to you, to save having to swap them over all the time.

Secure shell (ssh)

- Text only connection to a computer over a network.
- Secure, encrypted
- User name and password on PI **g54mrt**
- On lab computers, use ‘putty’ to do this
- You can also copy files (e.g. program files) across from your computer (use ‘winscp / CyberDuck / Filezilla’)
- (demo – copying a python script across to the PI, then ssh to the PI and running the script)

GrovePI + sensors

- Input/output board for Raspberry PI
- Has a bunch of sensors, which are easy to plug in, and a small LCD display.
- Python code for Raspberry PI to talk to the various sensors is preinstalled.
  - You can roll your own code if you hate python, but really not recommended.

Analog Sensors

- Connect to Analog ports (a0,a1,a2 on board)
- Loudness / sound sensor
- Light sensor
- Temperature
- Rotary angle
- Electricity Sensor
- Python:
- Light sensor example

```
import grovepi
value = grovepi.analogRead(5)   # read sensor
                                # on analog input 5
```

Digital Sensors

- Connect to digital ports (d1-d7)
- Button
- PIR Motion sensor
- Tilt switch

- Python:

```
import grovepi
value = grovepi.digitalRead(2)       # read sensor
                                     # on digital input 2
```

Ultrasonic distance measurer

- Measures distance of closest object in front of it, for example people walking past it.
- Connect to GrovePI digital input (D2,3,4,5,6,7,8)
- Python:

```
import grovepi
distance = grovepi.ultrasonicRead(2) # read ultrasonic sensor
                                     # on digital pin 2
```

Accelerometer & compass

  - Detect angle of sensor (tilt, compass direction)
  - Detect drops / taps

- Python:

```
import grove6axis
grove6axis.getAccel() # get raw accelerometer values
grove6axis.getMag() # get raw magnetometer values
grove6axis.getOrientation() # get orientation of accelerometer
                           as a tuple (yaw, pitch, roll)
```

Grove NFC Tag

- Programmable NFC tag
- Connects to I2C ports on Grove.
- Allows you to put some data (e.g. a URL) on the tag.
- Data is accessible by ‘bumping’ an NFC enabled device (e.g. android phone) onto it.
  - In theory, you can write a URL that auto-opens when you bump.
  - For this to be useful, you need to be able to deal with byte sequences and low level data structures (see examples at https://learn.adafruit.com/adafruit-pn532-rfid-nfc/ndef for what kind of thing you’re dealing with)
  - A bit of a pain to use (talk to Joe).
- Python:

```
import grovenfctag
grovenfctag.writeNFCData(5,[1,2,3,4,5]) # write bytes 1,2,3,4,5 to
                                           memory location 5
```

Grove NFC Reader

- NFC Tag reader
- Not to be confused with Grove NFC Tag
- This connects to the i2c ports of the GrovePi board
- Lets you read NFC tags
- Also does lots of other clever stuff (NFC data transfer etc.) but that is **HARD**

```
import grovenfcreader
# wait for up to 5 seconds and display the ID
# of an NFC tag that is
# put in front of the reader
print grovenfcreader.waitForTag(5)
```

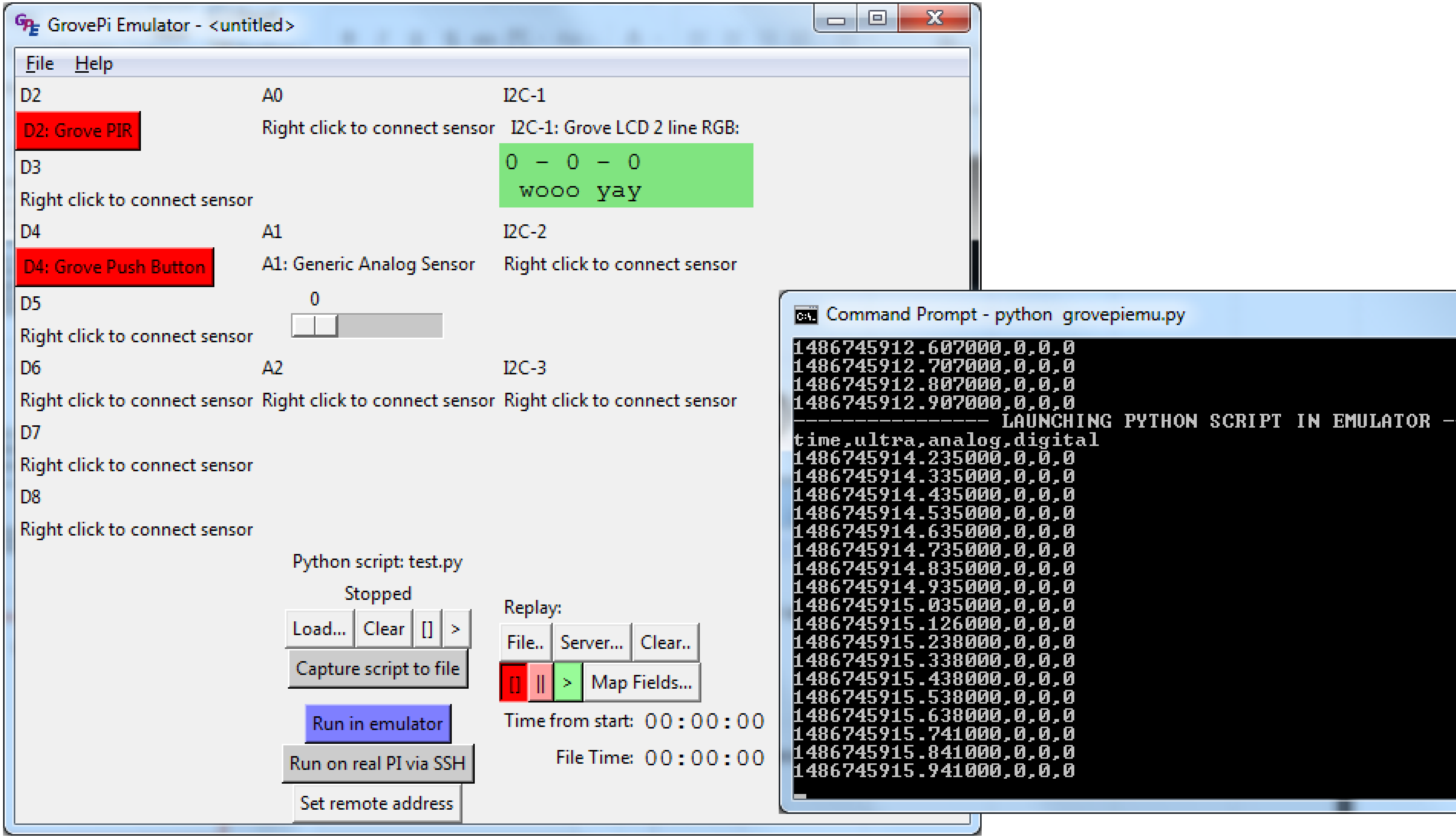
GrovePI display

- 16x2 LCD display, useful for status notifications etc.
- Coloured backlight
  - (you can set red, green, blue values)

- Python:

```
import grovelcd
grovelcd.setRGB(128,128,128) # set the backlight
grovelcd.setText("Hello World!") # set display text
```

GrovePI Emulator


GrovePi Emulator - <untitled>
File
Help
D2
D2: Grove PIR
D3
Right click to connect sensor
D4
D4: Grove Push Button
D5
Right click to connect sensor
D6
Right click to connect sensor
D7
Right click to connect sensor
D8
Right click to connect sensor
A0
Right click to connect sensor
A1
A1: Generic Analog Sensor
0
A2
Right click to connect sensor
I2C-1
I2C-1: Grove LCD 2 line RGB:
0 - 0 - 0
wooo yay
I2C-2
Right click to connect sensor
I2C-3
Right click to connect sensor
Python script: test.py
Stopped
Load...
Clear
[]
>
Capture script to file
Run in emulator
Run on real PI via SSH
Set remote address
Replay:
File..
Server...
Clear..
[]
||
>
Map Fields...
Time from start: 00:00:00
File Time: 00:00:00
Command Prompt - python grovepiemu.py

```
1486745912.607000,0,0,0
1486745912.707000,0,0,0
1486745912.807000,0,0,0
1486745912.907000,0,0,0
---------------- LAUNCHING PYTHON SCRIPT IN EMULATOR -
time,ultra,analog,digital
1486745914.235000,0,0,0
1486745914.335000,0,0,0
1486745914.435000,0,0,0
1486745914.535000,0,0,0
1486745914.635000,0,0,0
1486745914.735000,0,0,0
1486745914.835000,0,0,0
1486745914.935000,0,0,0
1486745915.035000,0,0,0
1486745915.126000,0,0,0
1486745915.238000,0,0,0
1486745915.338000,0,0,0
1486745915.438000,0,0,0
1486745915.538000,0,0,0
1486745915.638000,0,0,0
1486745915.741000,0,0,0
1486745915.841000,0,0,0
1486745915.941000,0,0,0
```

GrovePI Emulator

- A pretend GrovePI
- Runs on a PC (or Mac) with Python
- (or on flexible desktop)
- Can get it from
  - https://github.com/joemarshall/grovepi-emulator
- Run the same GrovePI python code
  - And interact with fake controls on the emulator
  - Or feed pre-recorded test data into it
  - Or put made up test data into it
- Also has other useful things that make development faster
  - Auto upload to Raspberry Pis
  - Generate skeleton data capture code.

Recording test data for emulator

- Test data must be in CSV format with a header
- You can output this from Python simply using a print statement

```
import time
import grovepi
print("timestamp,button4,analog1")
while True:
    timestamp=time.time()
    button4=grovepi.digitalRead(4)
    analog1=grovepi.analogRead(1)
    print("%f,%d,%d"%(timestamp,button4,analog1))
    time.sleep(0.001)
```

Run it with output sent to a file

```
python test.py > mytestdata.csv
```

Or, click on run on real pi via ssh in emulator and click 'capture script to file'.

CSV File

```
timestamp,button
1443535316.429,0
1443535316.929,0
1443535317.428,0
1443535317.929,1
1443535318.428,0
```

The internet

- The raspberry PIs have internet access.
- So they can:
  - Upload data to websites for visualisation
  - Tweet / facebook / email or whatever when an event is detected.
  - Download information (e.g. other contextual data such as local weather information)
  - Do other clever things that you think of…

(Very optional) machine learning

- I've put tensorflow on the raspberry Pis and tested it with my own models.
- They can *run* machine learning classifiers.
- You'd need to *train* any models on a PC in your own time.
- Need to record lots of train/test data in labs.
- More effort in some ways than the normal coursework, and limited help (ask me!).
- I just put this in because people asked me to.
- If you loooove machine learning, this is for you.

Worksheet

- There's a worksheet on moodle
- The hardware is on the table
- Go forth and play with the stuff, get used to it.
- Think about your proposal ideas this week (and next)