

DE2 Electronics 2

Tutorial 5

Lab 4 Explained

Prof Peter YK Cheung

Dyson School of Design Engineering

URL: www.ee.ic.ac.uk/pcheung/teaching/DE2_EE/

E-mail: p.cheung@imperial.ac.uk

Lab 4 – Task 1: Measuring Angel of tilt – the IMU

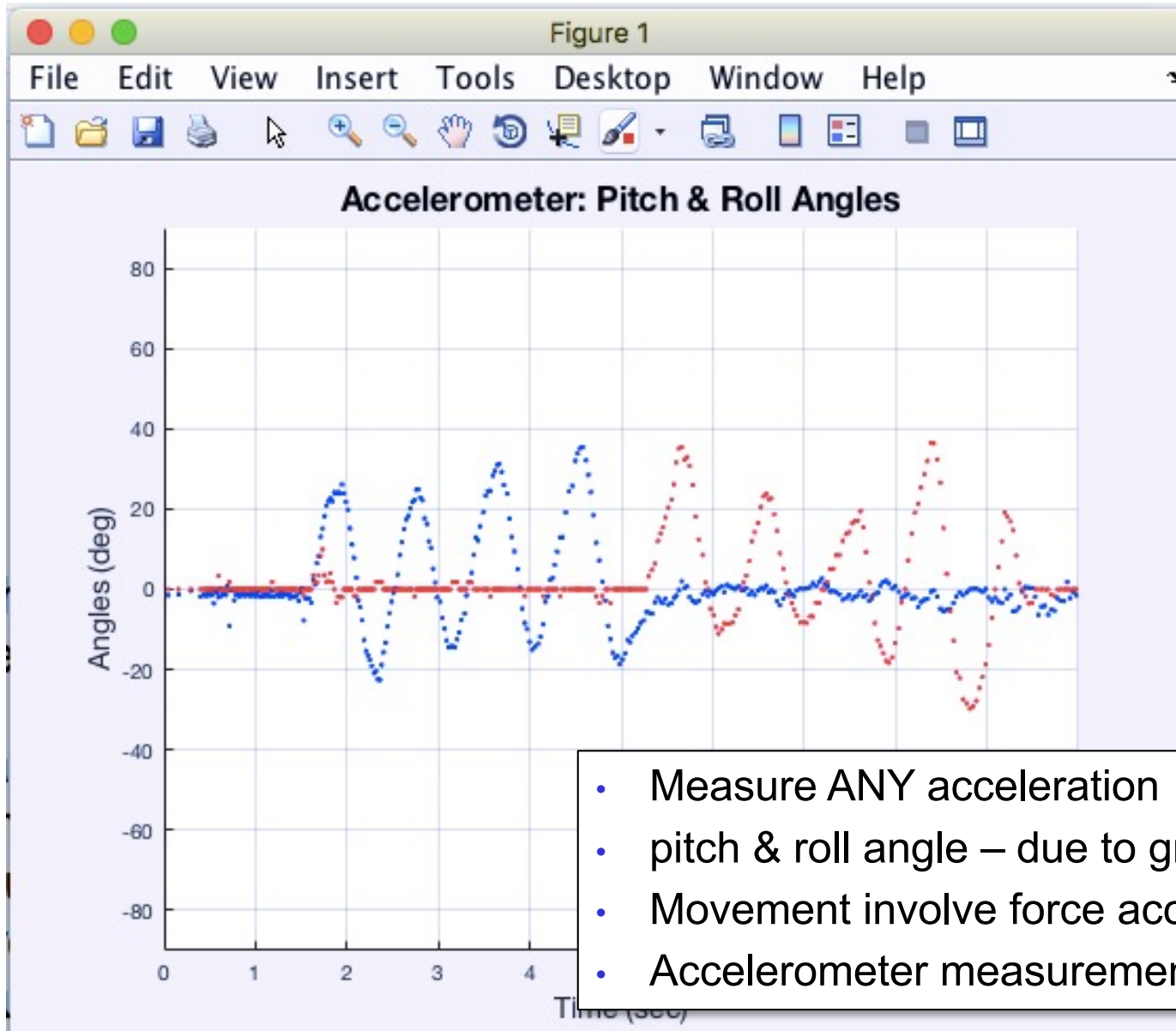
- ◆ The IMU – inertia measurement unit – has built in 3-axis accelerometer and 3-axis gyroscope
- ◆ Easy to access from Matlab using PyBench:.

```
[p, r] = pb.get_accel();           % p, r = pitch & roll angle in radians  
[x, y, z] = pb.get_gyro();        % x, y, z = rate of rotation in 3-axes in rad/sec
```

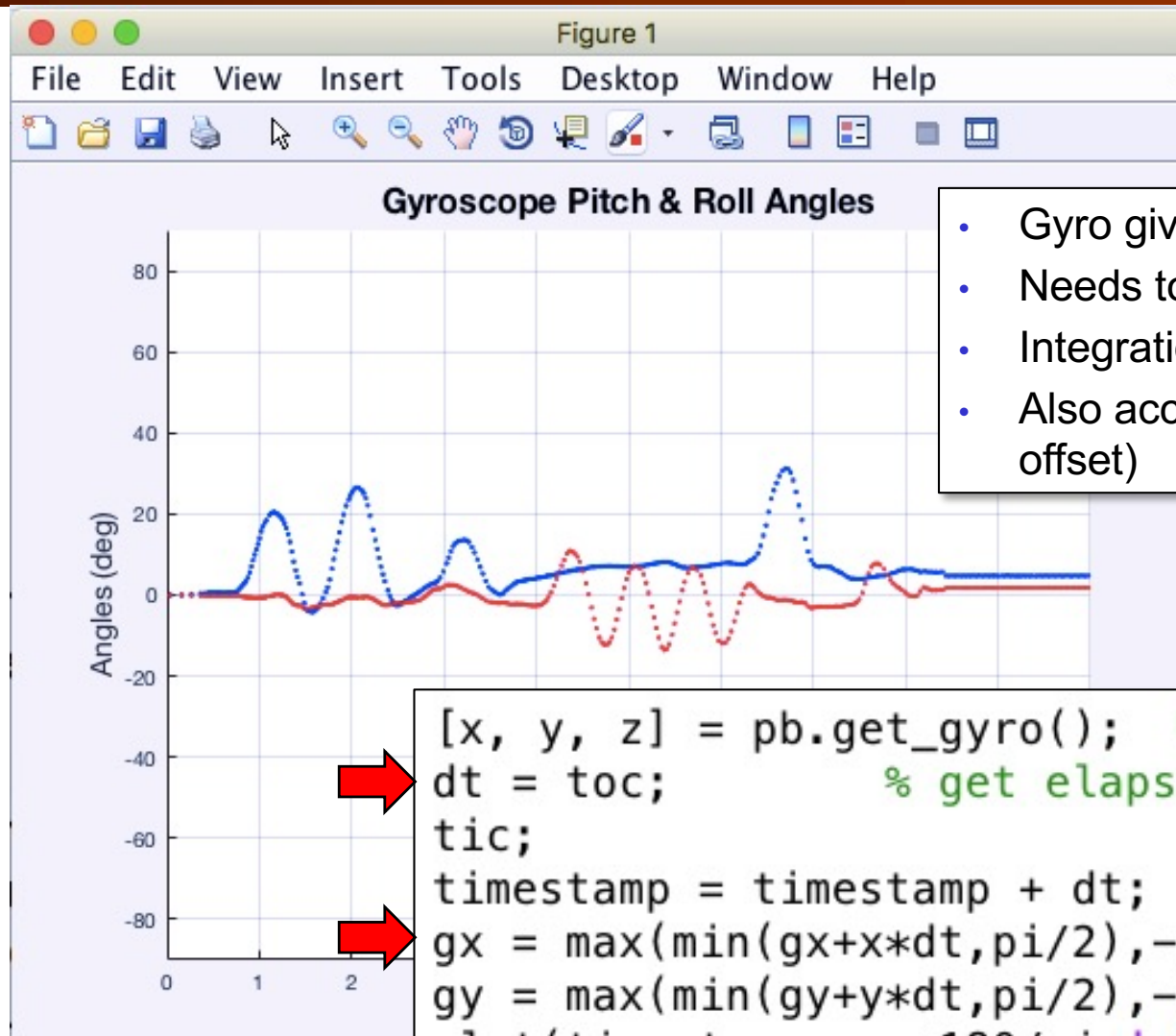
- ◆ Pitch angle – plane pointing up or down
- ◆ Roll angle – plane pointing left or right
- ◆ Angle can be in unit radian or degree: $\text{degrees} = \text{radians} * 180 / \pi$
- ◆ Generally use radian for calculations; use degree for display

- ◆ Learn usefulness and limitations of accelerometer and gyroscope

Lab 4 – Task 1a: Accelerometer



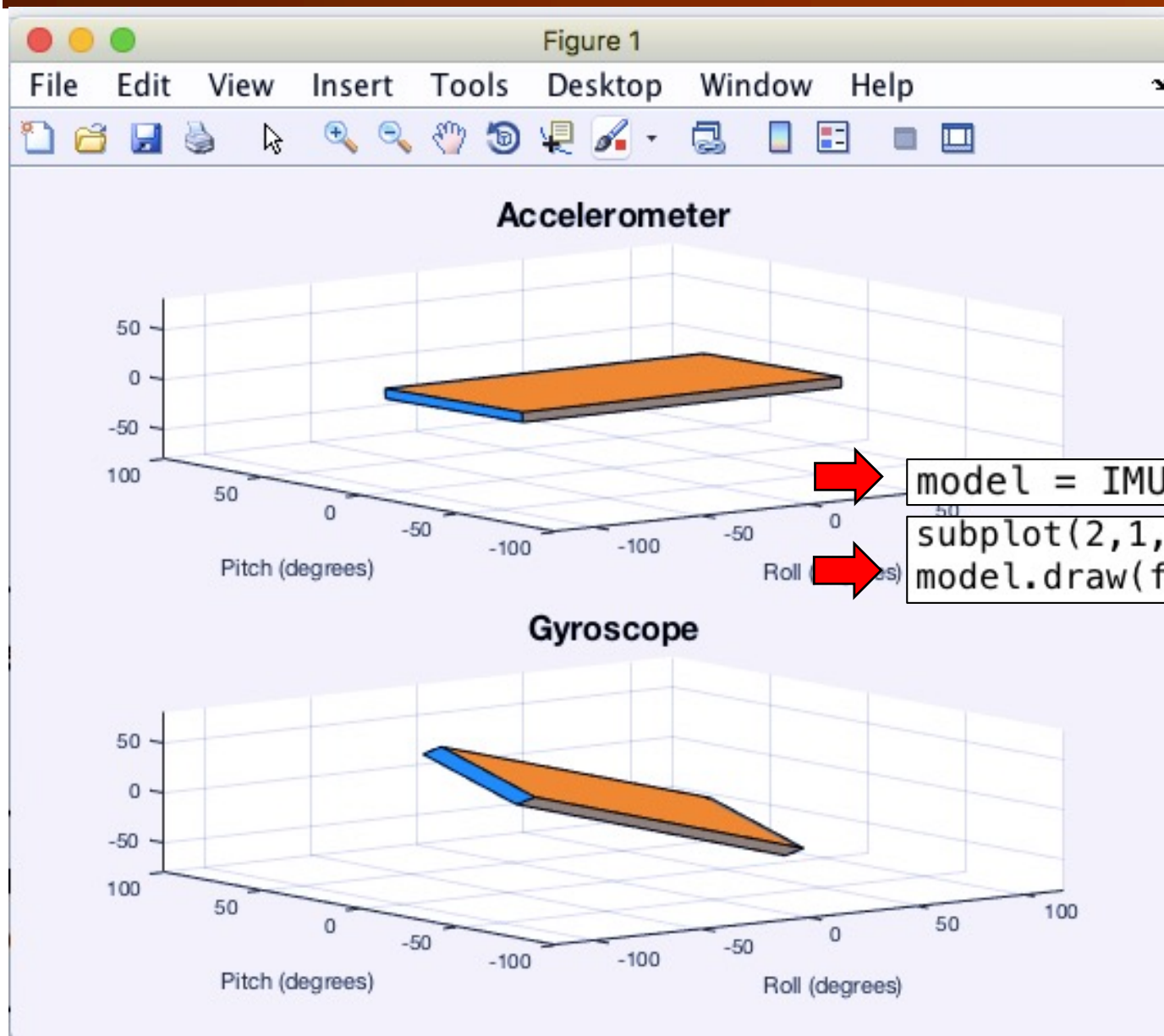
Lab 4 – Task 1b: Gyroscope



- Gyro gives angular velocity, not angle
- Needs to integrate to get angle
- Integration = accumulation
- Also accumulate errors – causing drift (or dc offset)

```
[x, y, z] = pb.get_gyro(); % angular rate in rad/sec
dt = toc; % get elapsed time
tic;
timestamp = timestamp + dt;
gx = max(min(gx+x*dt,pi/2),-pi/2); % limit to +/- pi/2
gy = max(min(gy+y*dt,pi/2),-pi/2);
plot(timestamp, gy*180/pi, '.b'); % plot pitch in blue
plot(timestamp, gx*180/pi, '.r'); % plot roll in red
pause(0.001); % delay for 1 ms, needed for plot
```

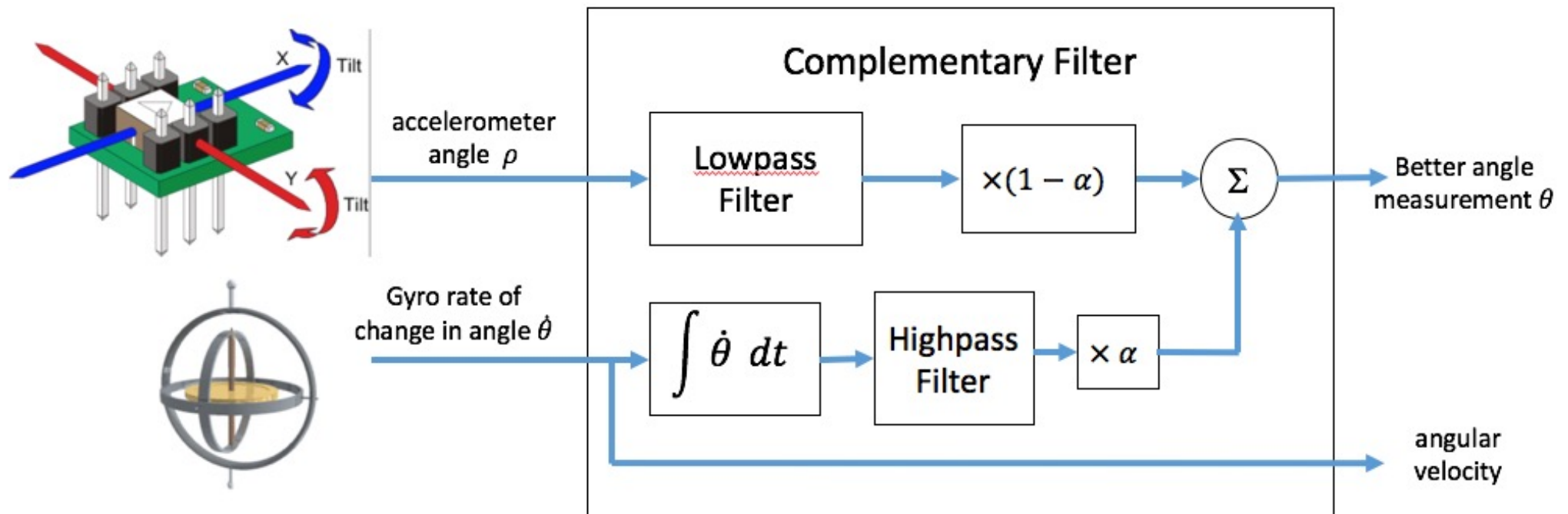
Lab 4 – Task 2: 3D visualization



```
model = IMU_3D();
```

```
subplot(2,1,1);  
model.draw(fig1, p, r, 'Accelerometer');
```

Lab 4 – Task 3: Complementary Filter - Concept



$$\text{angle } \theta = \alpha \times (\theta + \dot{\theta} dt) + (1 - \alpha) \times \rho$$

where

α = scaling factor chosen by users and is typically between 0.7 and 0.98

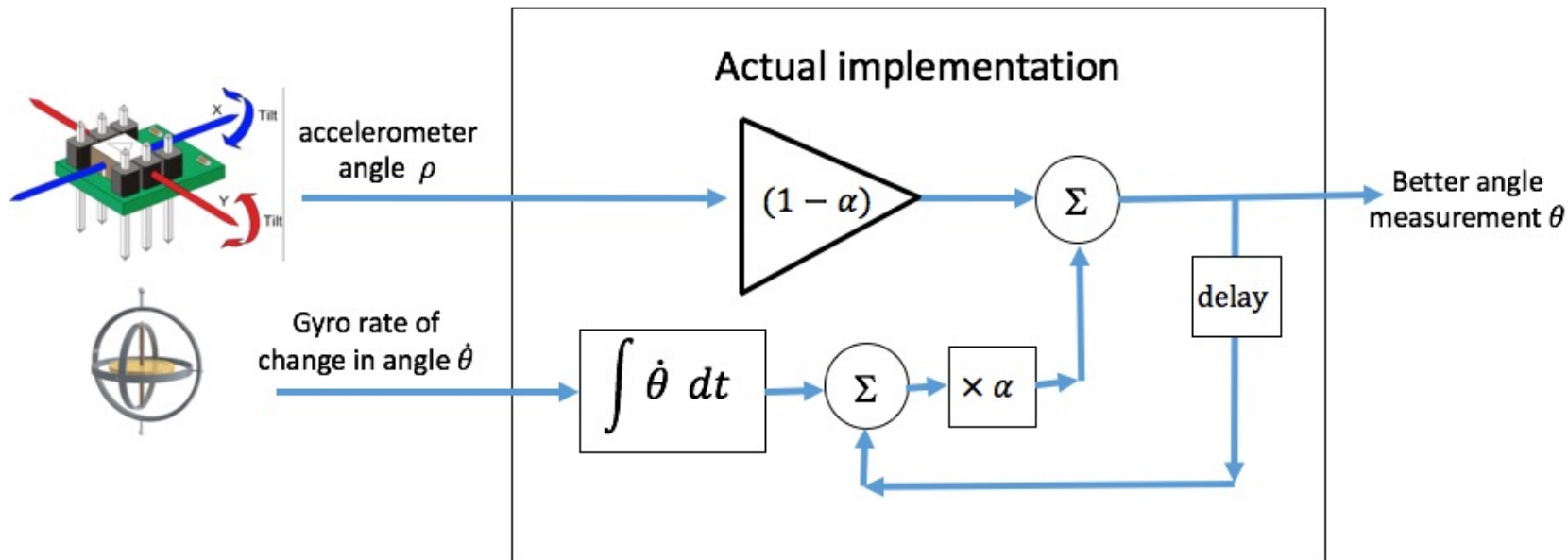
ρ = accelerometer angle

θ = output angle computed

$\dot{\theta}$ = gyroscope reading of the rate of change in angle

dt = time interval between gyro readings

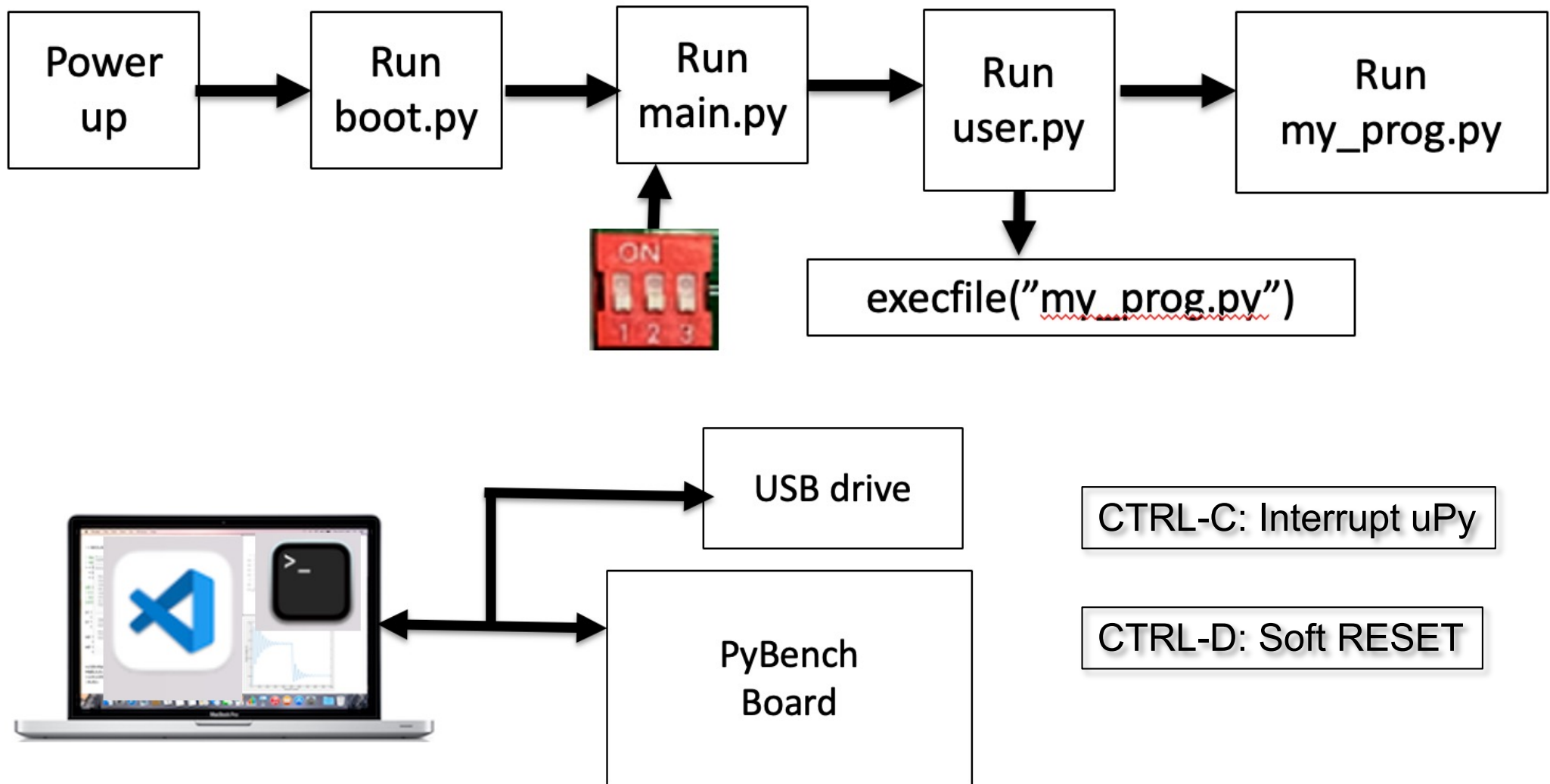
Lab 4 – Task 3: Complementary Filter - Implementation



$$\text{angle } \theta = \alpha \times (\theta + \dot{\theta} dt) + (1 - \alpha) \times \rho$$

- ◆ What happens if $\dot{\theta}$ is zero? Effectively average out the value of ρ
- ◆ What happens if $\dot{\theta}$ has a small error? Effectively reduce this error over time

Lab 4 – Pybench usage



Lab 4 – Task 4: Untethered – OLED Display

```
12 import pyb # Pyboard basic library
13 from pyb import LED, ADC, Pin # Use various class libraries in pyb
14 from oled_938 import OLED_938 # Use OLED display driver
15
16 # Create peripheral objects
17 b_LED = LED(4) # blue LED
18 pot = ADC(Pin('X11')) # 5k ohm potentiometer to ADC input on pin
19
20 # I2C connected to Y9, Y10 (I2C bus 2) and Y11 is reset low active
21 i2c = pyb.I2C(2, pyb.I2C.MASTER)
22 devid = i2c.scan() # find the I2C device number
23 oled = OLED_938(
24     pinout={"sda": "Y10", "scl": "Y9", "res": "Y8"},
25     height=64,
26     external_vcc=False,
27     i2c_devid=i2c.scan()[0],
28 )
29 oled.poweron()
30 oled.init_display()
```

Lab 4 – Task 4: Untethered – OLED Display

```
32 # Simple Hello world message
33 oled.draw_text(0,0,'Hello World!') # each character is 6x8 pixels
34
35 tic = pyb.millis() # store start time
36 while True:
37     b_LED.toggle()
38     toc = pyb.millis() # read elapsed time
39     oled.draw_text(0,20,'Delay time:{:6.3f}sec'.format((toc-tic)*0.001))
40     oled.draw_text(0,40,'POT5K reading:{:5d}'.format(pot.read()))
41     tic = pyb.millis() # start time
42     oled.display()
43     delay = pyb.rng()%1000 # Generate random number btw 0 and 999
44     pyb.delay(delay) # delay in milliseconds
```