1091 ESLAB Final Project
GRD Cultivator
cultivate your good habit of reading distance

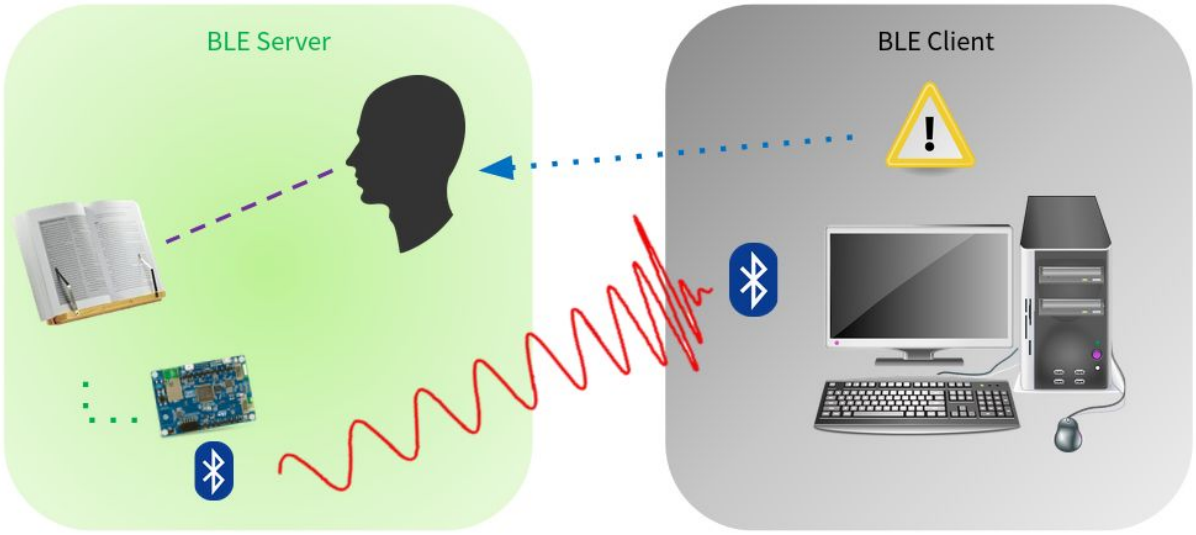
周光照 李達緯

Github URL: https://github.com/NTUEE-ESLab/2020-GRD-cultivator.git

一、動機

1. 我們利用這學期學到的技術，設計一個裝置，希望可以幫助人們建立良好的閱讀習慣
2. 現今有些民眾會使用閱讀架擺放書籍進行閱讀，若測距器放在書架上即可測量頭部和書架距離，用藍芽傳輸給BLE Client即可得知是否距離過近或者讀書時間過長
3. 學習利用網頁呈現結果和輸出警訊

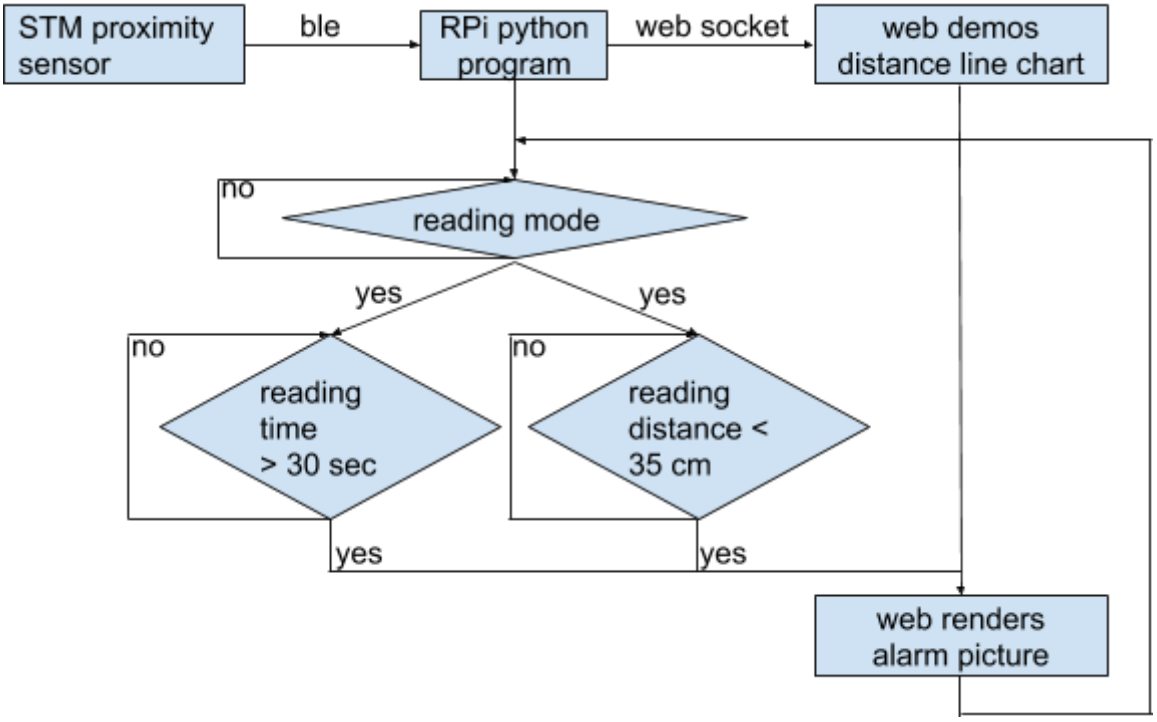
二、情境



□情境圖步驟：

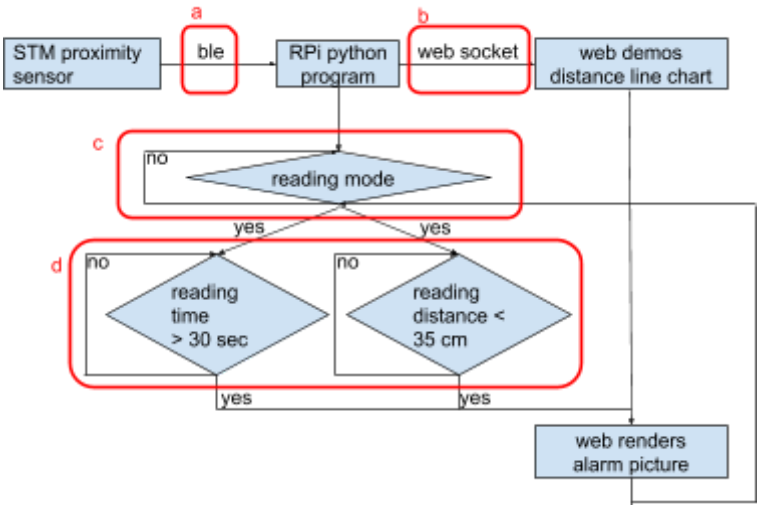
1. 測距器測量距離
2. 將距離資料以藍芽傳送
3. 電腦接收藍芽資料
4. 進行資料處理及判斷
5. 選擇是否發出警示
6. 將距離和警訊傳給Websocket
7. 網頁圖像化呈現資料和警訊
8. 使用者得知自身閱讀狀態

三、流程圖



四、實作內容

1. RPi



(1) 簡易濾波器

我們每十筆資料做moving average，簡易快速的濾掉雜訊，存到ave_dis的list內。

```
if(((len(data) % 10) == 0) and (data_len != len(data))):
    data_len = len(data)
    average_ = 0
    for i in range(10):
        average_ = average_ + data[-i - 1][0]
    average_ = average_ / 10.0
    ave_dis.append([average_, data[-i][1]])
```

(2) multi-thread

我們用4個thread來獨立運作：

a. BLE接受資料

```
def ble_receive():
    global data
    global run
    global but_ch
    global time_start
    global time_from_start
    data_len = 0

    while(run):
        if (but_ch.supportsRead()):
            time_from_start = time.perf_counter() - time_start
            data.append(
                [int.from_bytes(but_ch.read(), byteorder='little'), time_from_start])
            #simple filter
            if(((len(data) % 10) == 0) and (data_len != len(data))):
                data_len = len(data)
                average_ = 0
                for i in range(10):
                    average_ = average_ + data[-i - 1][0]
                average_ = average_ / 10.0
                ave_dis.append([average_, data[-i][1]])

        time.sleep(0.05)
```

b. socket傳輸

傳送以下資料給.html，.html可以再將reading_near或reading_long的alarm渲染出來

data_val：距離量測

data_time：時間

reading_near：紀錄是否閱讀距離太近的變數

reading_long：紀錄是否閱讀時間太久的變數

```
async def web_socket(websocket, path):
    global data
    global run
    global reading_near
    global reading_long
    data_time = 0
    data_len = 0
    while(run):

        if(len(data) > data_len):
            data_len = len(data)
            data_web = data[-1]
            if(data_web[1] > data_time + 1):

                data_time = int(data_web[1])
                data_val = data_web[0]
                # send data to socket
                data_web = "{\"val\":%d,\"time\":%d,\"near\":%d,\"long\":%d}" % (
                    data_val, data_time, reading_near, reading_long)

                line = await websocket.recv()
                # wait for notification
                if line is None:
                    print("line is None")
                    return
                await websocket.send(data_web)
```

c. reading_mode判斷

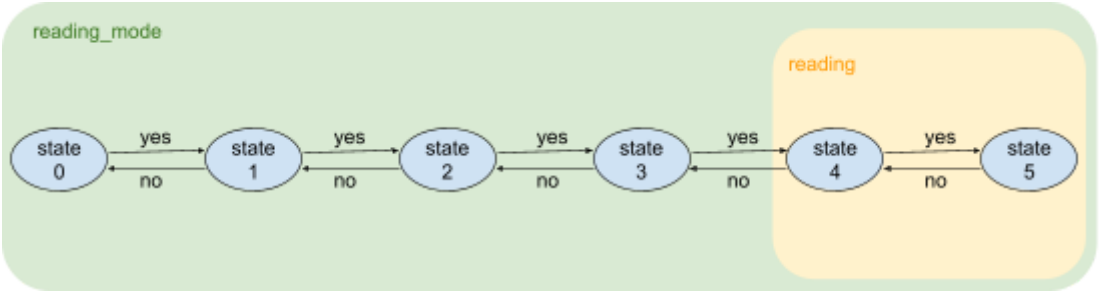
```
def data_process():
    global data
    global run
    global ave_dis
    global reading
    global reading_state
    threshold = 100
    ave_dis_len = 0
    # print("data_process")
    while(run):
        if((len(ave_dis) > 2) and (len(ave_dis) > ave_dis_len)):
            ave_dis_len = len(ave_dis)

            if ((abs(ave_dis[-1][0] - ave_dis[-2][0]) < threshold)
                    and (abs(ave_dis[-1][0] < 10000)) and (reading_state < 5)):
                reading_state = reading_state + 1
            elif(((abs(ave_dis[-1][0] - ave_dis[-2][0]) > threshold) or
                    ((ave_dis[-1][0] == 10000) and (ave_dis[-2][0] == 10000)
                     and (ave_dis[-3][0] == 10000))) and (reading_state > 0)):
                reading_state = reading_state - 1

        if(ave_dis_len > 0):
            if((reading_state == 4) or (reading_state == 5)):
                reading = True
            else:
                reading = False
        time.sleep(1)
```

每秒鐘判斷一次當前閱讀狀態，我們以finite states實做reading_mode判斷，可以避免一時的振動（極值）造成的偏誤。若當前狀態正在閱讀，state+=1；反之則state-=1，只有當state為4或5時才判斷真的在閱讀，設reading_mode為真。

threshold可以調整，讓我們的系統靈敏度適中，單位也是mm。



d. 判斷閱讀狀態太近或太久

當判斷閱讀距離太近，設全域變數reading_near為真；當判斷閱讀時間太久，設全域變數reading_long為真。反之則設為假。

```
def reading_judge():
    global reading
    global ave_dis
    global run
    global reading_near
    global reading_long
    global time_from_start
    mode = False
    time_rec = 0
    while(run):
        # record time
        if(mode == False):
            if((reading == True)):
                mode = True
                time_rec = time_from_start
            else:
                mode = False
        else:
            if((reading == False)):
                mode = False
        # reading_long
        if((mode == True) and (reading == True)
           and (abs(time_rec - time_from_start) > 0.25 * 60)):  # 0.25min
            reading_long = True
            print("reading_long! time: ", abs(
                time_rec - time_from_start), " sec")
        else:
            reading_long = False
```

```
        else:
            reading_near = False

        time.sleep(1)
```

我們的時間threshold設為15秒是為了方便開發，正確閱讀習慣是每30分鐘請休息5分鐘！

2. STM32

```
uint32_t distance;
int status = range.get_distance(&distance);


if (status == VL53L0X_ERROR_NONE) {
    printf("VL53L0X [mm]:           %6d\r\n", distance);
    signal_state = 0;
    if(pre_distance != distance)
    {
        event_queue.call(Callback<void(int)>(demo.get_service_pointer(), &ButtonService::updateButtonState), distance);
        pre_distance = distance;

    }
}
else
{
    printf("VL53L0X [mm]:               --\r\n");
    //printf("signal state= %d\n", signal_state);
    if(signal_state != 2)
    {
        signal_state = 1;
    }
   // if (ii ++ == 0)
    //event_queue.call(Callback<void(int)>(demo.get_service_pointer(), &ButtonService::updateButtonState), distance);
    // pre_distance = 10000;


}
if(signal_state == 1)
{
    event_queue.dispatch(500);
    event_queue.call(Callback<void(int)>(demo.get_service_pointer(), &ButtonService::updateButtonState), 10000);
    pre_distance = 10000;
    signal_state = 2;
}
```

(1) Ranging sensor

a. 測到目標物距離
紀錄前一次的距離，和當前測到的距離比較，如果相同就不再傳送data

b. 未測到距離
由於ranging sensor偵測最大角度只有25度，很容易測不到距離，此時傳送10000mm給RPi，使其知道當前並未得到任何數值

(2) BLE

```
class ButtonService {
public:
    const static uint16_t BUTTON_SERVICE_UUID              = 0xA000;
    const static uint16_t BUTTON_STATE_CHARACTERISTIC_UUID = 0xA001;
```

a. Device name = Distance

b. 設定兩組UUID，一個代表提供的service，另一個是service characteristic，即為目前最新得到的距離資料（須注意由於是根據網路上button service的原始碼修改而得，故參數名稱並未改變，buttonstate即是ranging sensor測到的距離）

3.HTML網頁

(1) 視覺呈現
為了讓警示更為醒目，讀書過久旱距離過近分別用兩張圖片來呈現，此外也將頭部和書架距離的資料動態上傳至網頁中。由於讀書有可能會遇到同時太近和過久的情況，當兩種情況同時為真，則以提醒休息為主，所以只顯示讀書過久的圖片。

五、成果demo

1. 正常姿勢閱讀
https://youtu.be/_5SVkgfkps0

2. 閱讀距離太近
https://youtu.be/xJ9MZhZMEdM

3. 閱讀時間太久
https://youtu.be/fguUoNTII1U

六、實作瓶頸

1. Ranging sensor利用I2C傳輸寫死的問題
由於是基於原始碼修改程式，結果原本的sensor函式庫已經寫好呼叫constructor時必定要指定I2C的連接埠，以便其他相關函式使用，但是我們並沒有需要利用I2C傳輸，結果似乎會出現Invalid new address的錯誤。雖然不知實際上這個錯誤的意義與影響，但是程式中斷時再重跑常會出現Hardware fault的狀況，有時必須將USB線拔掉重接數次才能回復原狀。上網查詢發覺可能原因為系統在程式中斷會出現不停reboot的動作，似乎因為接腳會出現懸空狀態，導致電路板進入異常狀態。這甚至有可能造成電路板重接電腦時會出現AP error情形。也正因為如此，我們的成品缺陷在於不能把電路板接到行動電源運作，因為程式一離開電腦即當場中斷，連帶燒進去的程式碼也被破壞掉，因此接到行動電源時無法正常運作。

七、組員分工

1. 李達緯
    a. Python資料處理
    b. Web socket實作
2. 周光照
    a. Mbed program整合各樣devices
    b. 網頁設計

八、參考資料

http://os.mbed.com/teams/ST/code/HelloWorld_ST_Sensors/
http://os.mbed.com/teams/mbed-os-examples/code/mbed-os-example-ble-Button/