物理實驗專題 - 藍牙空間定位

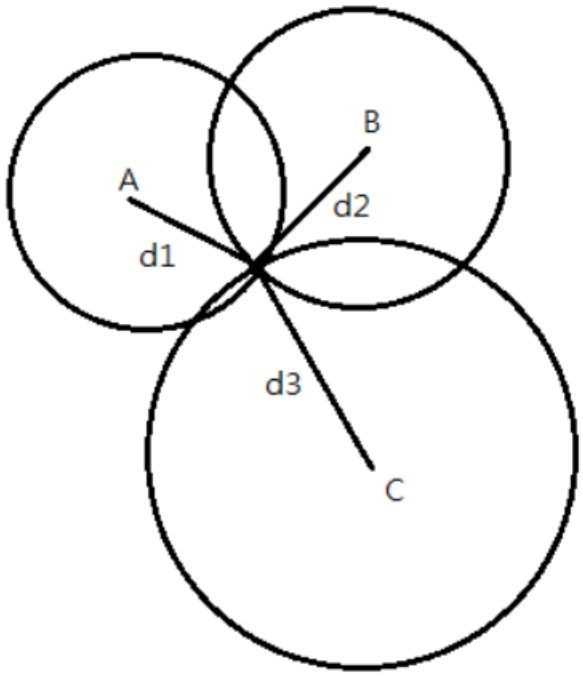
第 13 組，李國成 **C24085016**、何律煒 **C24081614**

摘要

透過收集 設備的 訊號強度(RSSI) 來估算距離，並用三角定位來判斷目標的位置。

三角定位算法

已知距離為半徑，在參考點劃圈以得到交點便是該物體位置。



圖源： 洪銘鴻,「iBeacon 定位校正方法設計與實作」, 東海大學資訊工程研究所

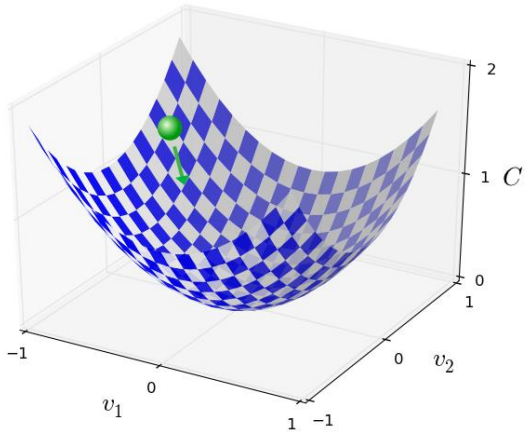
用數值分析的方法求得近似的交點位置。

一開始是用類似“勢能場”的概念，$U_i = 1/(\frac{r}{d_i})^2$，當位置和距離一樣遠時，勢能為 1，並且越遠勢能越低。

當三個或以上的點之勢能相加時，會在交點有一勢能谷。也就是說我們只要對勢能取 gradient 就會得到“力”，這個力會把物體往低處推。

而我們只要這個模型里建構一個物理定律：動量無作用、不守恆，也不會有動能儲存在慣性運動里。或者換一個方式說：力會改變速度，但不會改變加速度；亦或是終端速度的體現：$F = kv$。

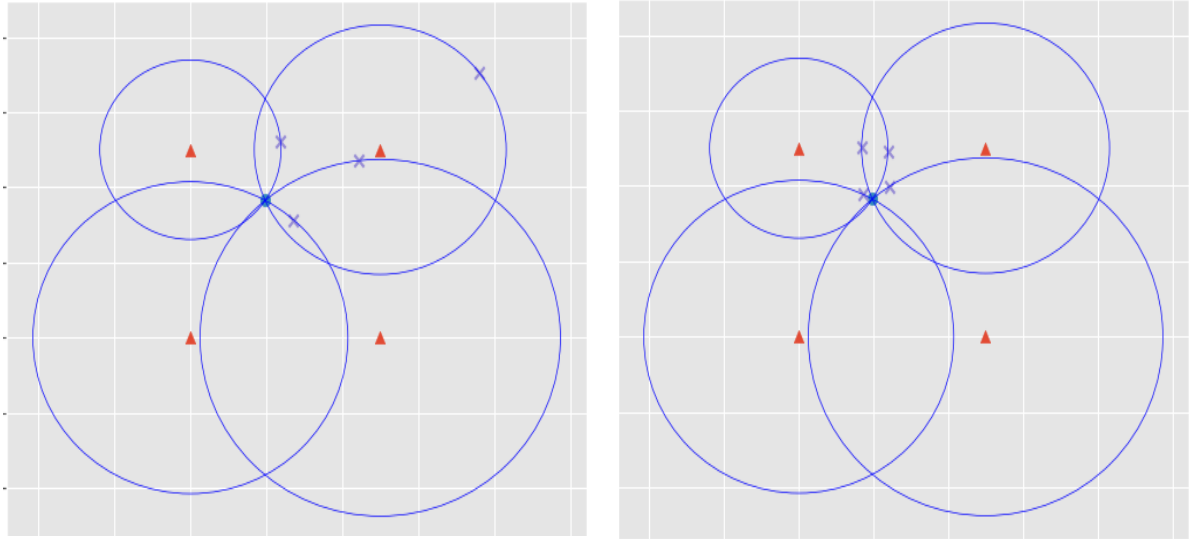
這樣的物理世界在理想情況下會讓物體滾向谷底也就是交點。



圖源： Michael A. Nielsen, “Neural Networks and Deep Learning”, Determination Press, 2015

不過這個模型收斂很慢，而且不收斂的情況居多。所以又設定另一個較有效的物理模型，並加了限制條件：

- 每個參考點畫出的圓上都有一個專屬的物體
- 三個物體之間有引力，為了加快收斂，引力和距離的平方成正比。
- 沒有慣性作用
- 限制條件：物體只能在所屬參考點的圓上移動，可用角度來表示該參考點上物體的位置。
- 為了方便，設定引力會直接作用在角度的變化上而不是位置的變化。



動畫模擬可以看到，四個隨機點最後會因為引力而逐漸靠近交點。
至此，我們的三角定位算法算是完成。

RSSI 距離換算

參考： *https://journals.sagepub.com/doi/full/10.1155/2014/371350*

RSSI 的公式和天文學的視星等觀念是一樣的：

$$RSSI(R) = -k\ log_{10}(R) + RSSI_0$$

在比較理想的情況下，RSSI 可以寫成

$$\bar{\text{RSSI}} = A - 10n\log(d).$$

其中 A 是在某個固定距離(1m)量測到的 RSSI，又或者是 RSSI 的“絕對星等”，而 n 是環境因子。

我們便可以推算距離：

$$d = 10^{(A-\bar{\text{RSSI}})/10n}.$$

其中，A 和 n 可以透過大量測量收集數據來得到，且

$$n = \frac{A-\bar{\text{RSSI}}}{10*\log(d)}$$

另外實際情況下，距離越遠，測得的 RSSI 值波動會越大，RSSI 便會成：

$$\text{RSSI} = A - 10n\log\left(\frac{d}{d_0}\right) - X_\sigma$$

其中 X_σ 是 RSSI 在該距離下的高斯標準差。

系統架構

- 三台 Raspberry Pi 做位置參考點來幫助定位
- 一台手機 / Raspberry Pi 做被定位的目標

收集 RSSI 的方向有倆個：

1）目標設備 --- RSSI >>>> 參考點設備
（**參考點設備** 接收 **目標設備** 的 RSSI）
好處：需要被定位的設備是被動的，不需要做什麼操作便可以被定位。
壞處：不同設備的發射功率不一樣，雖然可以通過索引數據庫從 mac address 來獲得發射功率，但相對困難(還需判斷對方的發射功率)，且若對方設備不在數據庫內便無法定位。

2）參考點設備 --- RSSI >>>> 目標設備
（**目標設備** 接收 **參考點設備** 的 RSSI ）
好處：參考點設備是固定的，也就是已知發射功率/絕對星等。定位起來相對方便
壞處：被定位的設備需要安裝且支援指定的軟體。還需要提前知道參考點設備的訊息。

為了方便，之後應該會選擇第二種模式。但設計通訊架構時也加入了第一個模式的考量。故在已知發射功率的情況下，只需要稍微調整便可以改成第一種模式。

通訊架構

獲得 RSSI

藍牙通訊協議下，即使還沒配對，設定後還是可以通過廣播 (broadcast) 一些訊息讓其它設備接收到。這樣可獲得廣播設備的 mac address，幾乎是設備專屬的編號，我們便可以以此得知特定設備的 RSSI 值是多少。

客戶端

以 shell script 執行，在一般的 Linux Console 需要先與被測 RSSI 值 的設備建立 baseband connection，再獲得 RSSI。

然後用 necat 以 TCP socket 把 該設備的訊息 和 RSSI 發給服務端。

一直循環以上步驟。

服務端

負責接受客戶端發來的包含 參考點設備 和 RSSI 值 的信息，計算出對應的距離後再用算法求出大致的位置。

另外服務器進程用 異步 I/O (Asynchronous I/O) 來處理多個客戶端的請求。

若要使用第一個模式（**參考點設備** 接收 **目標設備** 的 RSSI），只需要在三台作為基站的 Raspbery Pi 執行客戶端的 shell script 便可。

若使用第二個模式 （**目標設備** 接收 **參考點設備** 的 RSSI ），則在目標設備執行客戶端的 shell script 便可。若目標設備換成是 Android 而不是 Raspberry Pi，則暫時無法實現。

程序操作

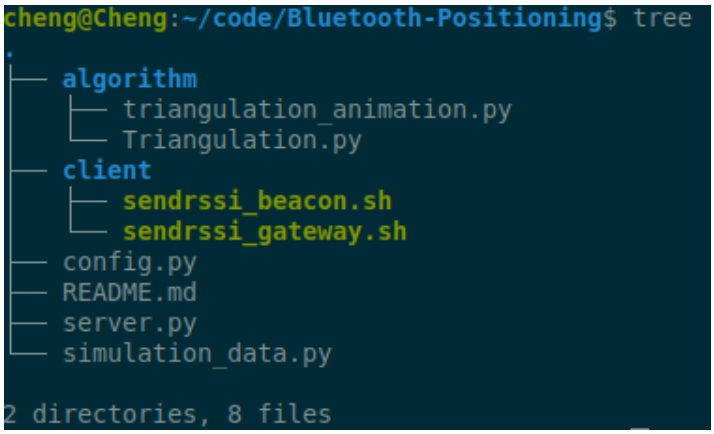
源代碼： https://github.com/cheng1999/Bluetooth-Positioning

設定 RSSI 發射設備至 discoverable

終端執行： `bluetoothctl discoverable on`

如圖：

程序目錄



- /client/

  說明： 有 sendrssi_beacon.sh 和 sendrssi_gateway.sh 分別對應倆種模式。

  備註： 第一次執行前先編輯該文件來設定 mac address。

  執行： sudo ./<file>

- /server.py

  說明： 是服務器，負責接收 RSSI 數據並計算位置。

  備註： 第一次執行前需要先編輯 /config.py 來根據環境和設備設定參數。

  執行： python3 <file>

- /simulation_data.py

  說明： 模擬 RSSI 數據，不測量而直接發送偽 RSSI 給服務器，方便測試。

  執行： python3 <file>

- /algoritim/triangulation_animation.py

  說明： 動畫模擬三角定位算法的過程。

  執行： python3 <file>

- **其它文件**

  說明： 相關的函數庫或文檔。

實做

因為期末很多報告、專題和作業，另有疫情影響，無法在時限內有更完整的完成度。

雖然理論和程式的部分已經完善。但收集數據求 RSSI 的環境因子和絕對星等還未實現程式自動化，也還沒開始實驗收集數據。

實際預測距離的部分應該還需要再拖幾天才有空完成。

其它

心得

Labview 好處

可以快速針對一些**儀器做編程**還有**使用者界面的重新整合**。對儀器和電腦的整合可以非常快速。再來就是它的編程裡**以資料流為主的設計**很不錯，並且做到**資料流可視化**。預設以 Async 異步/平行進程處理在實際實驗上也較為實際。

遇到的問題和看法

複雜的邏輯運算需要花**大量時間**來**拖拽線條**，而且物件很佔空間也**不能放大縮小編程界面**。以目前的熟練度來說以一般程式語言編寫複雜邏輯絕對比用 Labview 快不止一個數量級。不過上網找過別人的心得，發現這主要是自己不熟悉 Labview。

Labview 的學習曲線其實比一般程式語言還陡，不過架構的部分因為以資料流為主，很容易讓人掌握程式語言裡類似 MVC 編程架構的思維。而一般程式語言初學者要過渡到使用 MVC 反而就沒有這麼容易。

再來介面前後整合的部分和資料流的心得。個人小時候曾經 Full Stack 開發過網頁前後端小項目、Android 小程序、VisualBasic，因此對資料流、前後端處理和編程架構有一些要求。

對於 Labview 前後端還有架構的設計只會覺得它是 Android 開發架構設計上的半殘品(不過 Labview 多了資料流可視化)：後端的部分過度強調資料流。連一個 ifelse 邏輯都必須以資料流呈現，很佔空間(礙眼)，也影響代碼可閱讀性，除非一直寫 subvi (這點和 Java 很多 class 是一樣的道理)，並且一些效率很高的**編程架構和思維**在 labview 上是**根本無法實現**的。

另外現在科技發展的趨勢已經是以資料交換為核心，而不是以主機為核心。大量以資料流為主的架構甚至系統已經流行，陸續崛起。再來，Labview 作為可視化編程的 IDE 其實太過臃腫，而且執行起來有很多莫名其妙的 Bug。而且還取消了早期對 linux 系統有的 DAQ 設備支持，頗有 Microsoft 壟斷 OS 市場的那股俗氣。**以上**，個人認為在未來 Labview 會被替代（但絕對不會在這幾年內）或轉型（後來查了一下發現有 Labview NXG，是近 4 年來的轉型嘗試）。

叫我改進 Labview 的話，我還是會希望 labview 有 textbased programming 的空間。基礎單元/邏輯可以用一般代碼實現，必要時再讓程式給代碼做資料流可視化。這樣在宏觀處理架構層面的資料流時可以保留 Labview 的優勢。在處理複雜算法時可以保留 textbased programming 的優勢。

建議

（國成）

Labview 工程師來上課的幫助感覺不大。倆個助教的協助適時地幫助反而讓我們學得比較快少走很多彎路，可以的話由助教根據自己學習的歷程來設計 Labview 作業並給予一點關鍵提示，這樣在缺乏諮詢的資源時同學也不至於一直繞路。

（律煒）

關於這學期的 labview 課程，個人的改進建議是希望之後學期初那種請專業人士來教學的課可以變成專題化或者少量多餐的形式。把所有東西都塞在一起要我們快速吸收明顯是不切實際的，特別是它是我們以前沒有接觸過的軟體，而且使用的型態又與我們以前所學的 python 等等程式軟體相去甚遠，卻要我們跟著工程師連續上三四個小時，這會有兩個問題：

第一，連工程師自己都說只給三四個小時是明顯不夠的，他只能講重點。
而這樣子就會沿生出一些問題：比如說助教給的寫好的 labview 要上課使用的時候，若是出現 BUG，我們甚至連 debug 的能力都沒有，因為這個的迴圈形式我們極度不熟悉，若只有幾個還好，當一堆東西在跑的時候，可能光介面上不懂(沒看過)的區塊就已經超過一半了，光看懂整個程式運作的過程就可能要花上幾個小時，何談需要去 debug，所以我們只能在一知半解的情況拼拼湊湊，猜對改對就成功，猜錯改錯就繼續猜，超級浪費時間，

第二點是我們當天上課之後，回去還能記住多少?

時間給不夠的情況，工程師只能趕，那就會更容易跟不上，而只要一個跟不上後面也不用聽了(我甚至還在下課的時候被主任問到工程師上課特別講要注意的地方，但主任不會所以做不下去，連主任應該有需要用 labview 都可以對這東西如此不瞭解?)

我個人建議之後的 labview 課程可以改成先看過影片，然後:
- 再找個時間請工程師來回答問題(不推薦),
- 或者多讓工程師上幾次課(推薦，大家吸收消化一定比較好，這是這一年上網課的經驗),
- 不然就是專題化，讓學長或者幫我們寫上課 labview 的人告訴我們各個部分怎麼寫的,
- 或把邏輯跟工具交給我們讓我們做，在引導下的實作比較不會那麼吃力學起來也比較有成就感。

這大概就是我這學期的意見，謝謝。