SDC Localization Competition Report

A. Pipeline

撰寫 icp_locolization 類別，在主程式中宣告，並使用 ros::spin()執行緒。

(1). 初始化

1.讀進 Map
2.Map 降低取樣
3.設定初值 x,y,z 和 yaw
4.以初值計算出 initial_guass

(2). lidar_points callback 函式

1.將收到的 msg 轉存為 point cloud
2.座標轉換（velodyne 到 base_link）
3.使用濾波器，降低取樣以利於運算，以及從 x,y,z 方向過濾掉一些點。
4.進行 ICP，配對 map 和 ladar 的 point cloud。計算出 base_link 和 world 之間的座標轉換矩陣。
5.TF 廣播座標轉換
6.依據座標轉換矩陣的結果，計算 odometry
7.publish 結果，包含處理後的 lidar_points,odometry 到各 topic 上

B. Contribution

1. Point Cloud Library

PCL 是用於 point cloud 處理任務和 3D 幾何處理（例如發生在三維計算機視覺中）的算法的開源函式庫。
在這次作業中同樣使用 PCL 來做 point cloud 的處理和 registration。PCL 缺點是引入了許多其他第三方函式庫，造成配置環境的肥大。還有 PCL 已經久未更新。

2. ICP 演算法

最近點迭代演算法，是最為經典的資料配准演算法。其特徵在於，通過求取源點雲和目標點雲之間的對應點對，基於對應點對構造旋轉平移矩陣，並利用所求矩陣，將源點雲變換到目標點雲的座標系下，估計變換後源點雲與目標點雲的誤差函式，若誤差函式值大於門檻值，則迭代進行上述運算直到滿足給定的誤差要求。ICP 演算法採用最小二乘估計計算變換矩陣，原理簡單且具有較好的精度，但是由於採用了迭代計算，導致演算法計算速度較慢，而且採用 ICP 進行配準計算時，其對待配準點雲的初始位置有一定要求，若所選初始位置不合理，則會導致演算法陷入區域性最優。

在這次作業中，主要的策略就是一直修改以下參數，嘗試達成更好的 locolization 效果。

(a). PCL VoxelGrid 下采样
VoxelGrid 方法是將 3D 空間劃分成多個很小的區塊，然後讓處於在同一區塊內之所有 point cloud 的中心點之該 point cloud，作為該區域內的唯一一個點，如此來達到降低取樣的目的。特別是，此方法的結果並不是原本的 point cloud 集的子集合，因為很多點的位置已經變化了。其中 leafsize 越大，表示降得採樣的幅度越大，輸出結果的點越少。

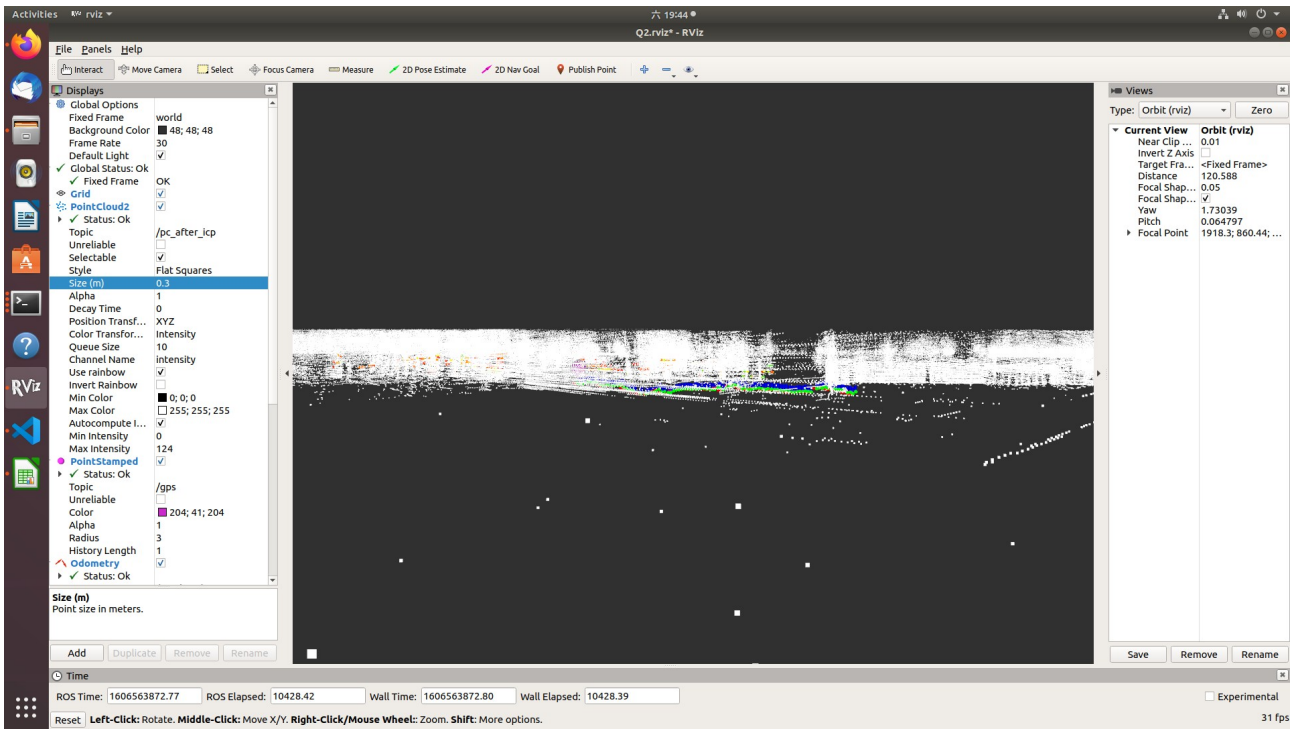
(b). setMaximumIterations，最大迭代次數
icp 是一個迭代的方法，設定上限最多就迭代這麼多次。此值我設的很高，以免在初值不精準的時候跌不出精確的結果。

(c). setEuclideanFitnessEpsilon 收斂條件
當均方根誤差小於此門檻值時，停止迭代。如果前進效果不佳，調謹此參數似乎能改善。

(d). setTransformationEpsilon
前一個座標轉換矩陣和當前矩陣的差小於此值時，就認為已收斂，是另一個收斂條件；如果轉彎效果不佳，調謹此參數似乎能改善。

(e). setMaxCorrespondenaceDistance
設定對應點對之間的最大距離，此值對配準結果影響較大，過鬆會配對不精確，如果過緊會讓它容易被某些點拉走。

(f). filter 的 x,y,z 上下限
ICP 的缺點之一就是要自己去掉不合適的點。透過設定不同的上下限，濾掉 map,lidar point 的一些點後再進行配對，同樣的 ICP 參數會有很不一樣的結果。尤其是二，三題的 map，z 方向太亂了，必須經過過濾才能使用。



圖一、過濾 z 方向後的 map 側視畫面

C. Problem and Solution

1. 初值設定
以 gps 作為 x,y,z 的初值
在 rviz 上觀察 yaw 的方向，微調出適合的初值

2. 車子不會前進
在第二題時，一開始遇到這個問題，經過反覆調整參數，尤其是 icp.setEuclideanFitnessEpsilon，使 match 結果更緊密，讓 locolization 效果變好。

3. 轉彎幅度
在第一題時和第三題，剛開始都曾經遇到這個問題。轉彎時，odomtry 方向的修改幅度不夠，經過反覆調整參數，尤其是 setTransformationEpsilon，使 match 結果更緊密，讓 locolization 效果變好。

4. 整個轉換被某些點拉走，造成極大誤差
因為不想讓 ICP 為了配對某些點而整個偏掉，會在過程中不斷觀察 rviz 畫面，找出是哪些點造成問題，使用 filter 將它們過濾掉。在這個作業中，setFilterLimits 也是很關鍵的參數。

5. 運算量過大，容易錯失幾個 message frame
ICP 演算法每次迭代都要搜尋最近點，計算代價高昂。因為把參數調很緊，需要多次迭代，轉換矩陣的誤差才能收斂到設定範圍內。需要將播放 bag 檔的速度放慢再放慢，確保每次收到 message 後，都確實運算完成後，才再收到下一個 frame 的 massege。更需要一台 CPU 夠好的電腦來做運算。

D. Others

1. 程式執行
在建立 workspace，執行完 `catkin_make` 和 `source devel/setup.bash` 後
使用 roslaunch 來執行相關 node

第一題：`roslaunch localization 309605008 Q1.launch`
第二題：`roslaunch localization 309605008 Q2.launch`
第三題：`roslaunch localization_309605008 Q3.launch`

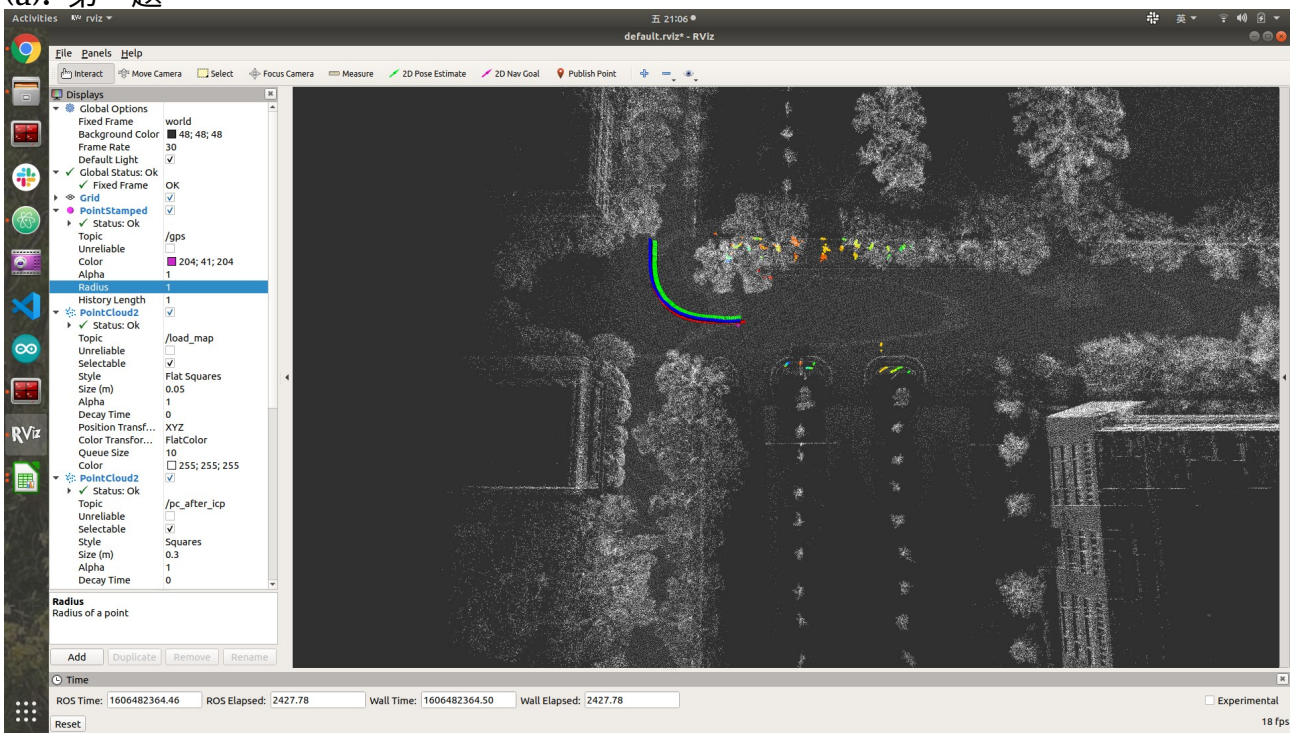
2. 讀取 map 路徑修改
第一題 catkin_ws/src/localization 309605008/src/icp_locolization1.cpp57 行
`if (pcl::io::loadPCDFile<pcl::PointXYZI> ("/home/bory/sdc/localization/map/itri_map.pcd", *load_map) == -1)`

第二題 catkin_ws/src/localization 309605008/src/icp_locolization2.cpp59 行
`if (pcl::io::loadPCDFile<pcl::PointXYZI> ("/home/bory/sdc/localization/map/nuscenes_map.pcd", *load_map) == -1)`

第三題 catkin_ws/src/localization 309605008/src/icp_locolization3.cpp59 行
`if (pcl::io::loadPCDFile<pcl::PointXYZI> ("/home/bory/sdc/localization/map/nuscenes_map.pcd", *load_map) == -1)`

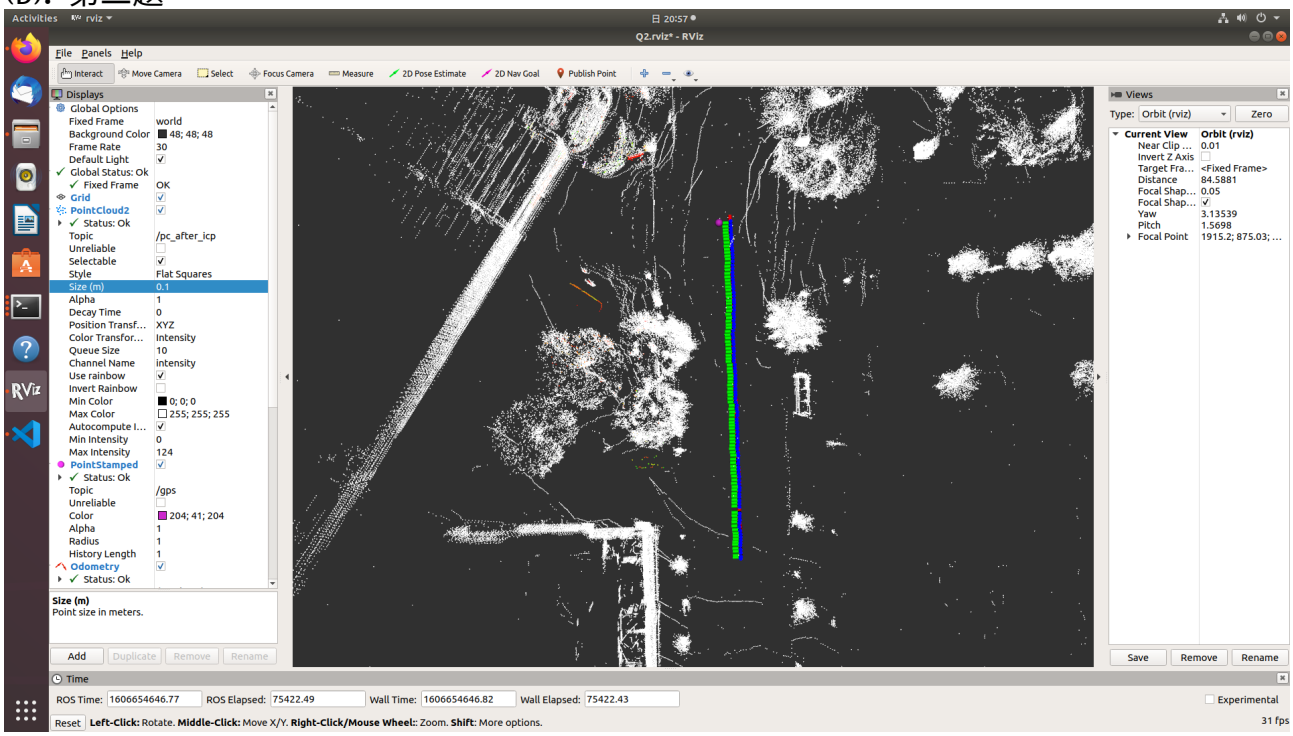
3. Rviz 畫面截圖

(a). 第一題



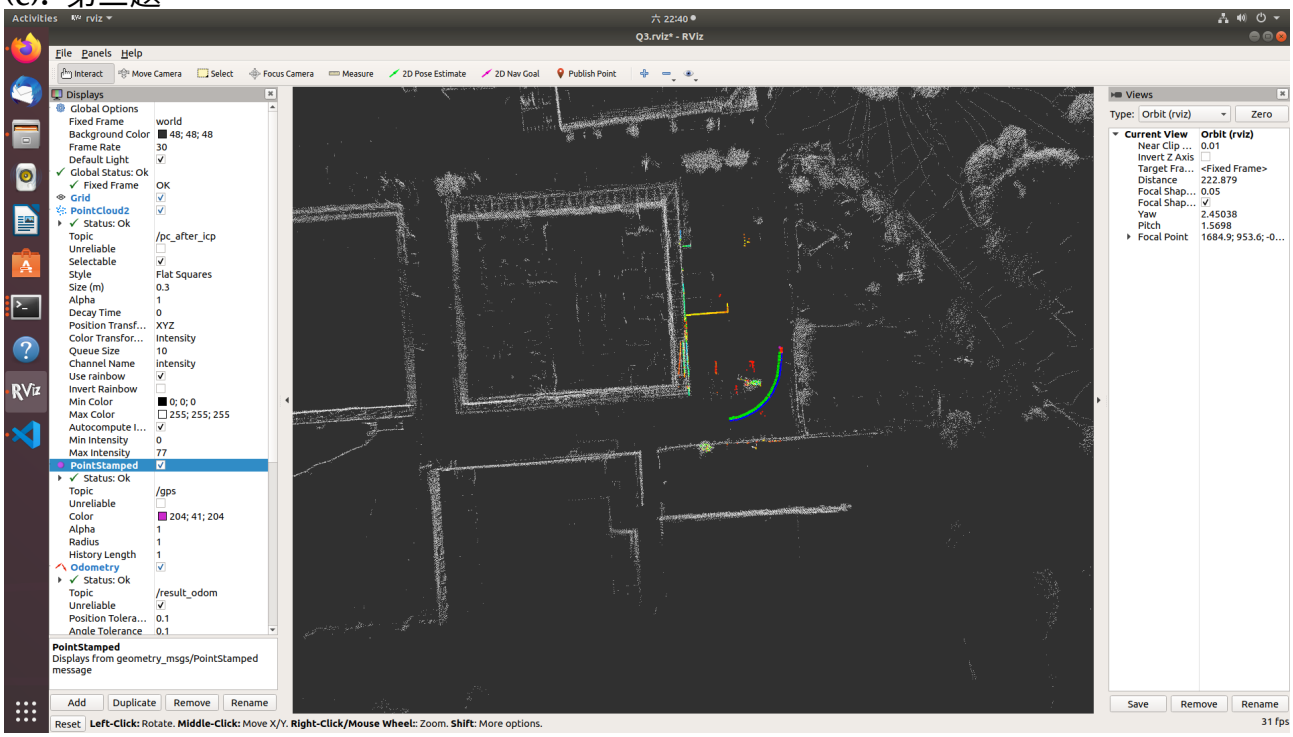
圖二、第一題執行結果

(b). 第二題



圖三、第二題執行結果

(c). 第三題



圖四、第三題執行結果