Homework #4 Report

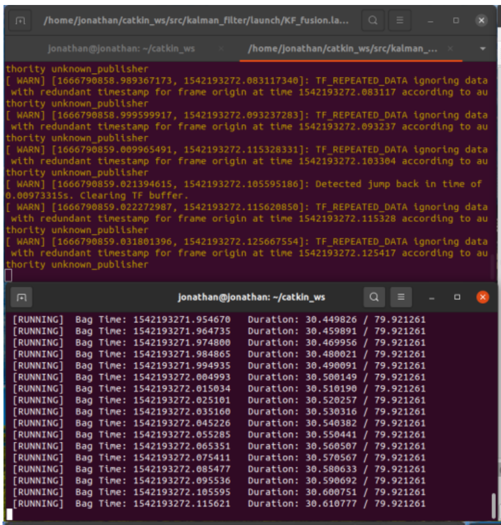
[EECN30168] Self-Driving Cars 2022

Student ID: 311605004　　**Name:** 劉子齊　　**Date:** 2022.10.26

1. Introduction

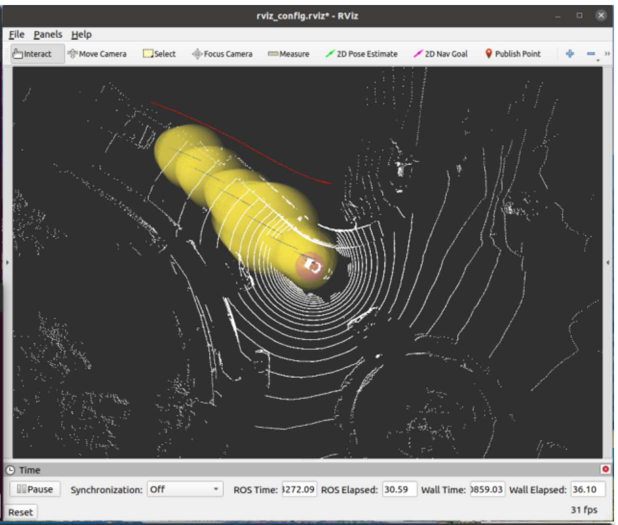
In this homework, our goal is to apply the Kalman filter we implemented in the previous homework on a real set of self-driving car data. Furthermore, we need to visualize the result through “rviz”, and make further adjustments to our program through the observations to the output result, as shown in the following session.

2. Result

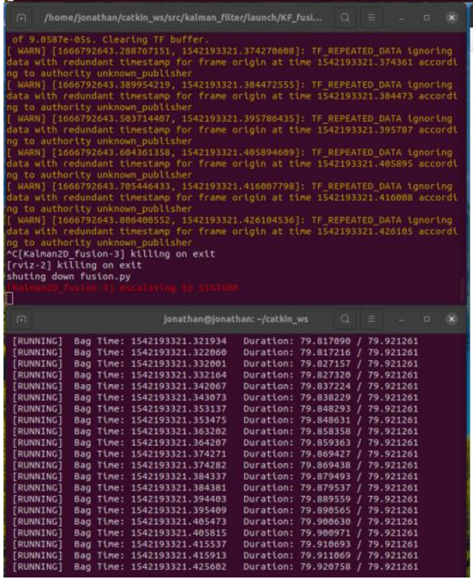

/home/jonathan/catkin_ws/src/kalman_filter/launch/KF_fusion.la...
jonathan@jonathan: ~/catkin_ws
/home/jonathan/catkin_ws/src/kalman_...

```
thority unknown_publisher
[ WARN] [1666790858.989367173, 1542193272.083117340]: TF_REPEATED_DATA ignoring data
 with redundant timestamp for frame origin at time 1542193272.083117 according to au
thority unknown_publisher
[ WARN] [1666790858.999599917, 1542193272.093237283]: TF_REPEATED_DATA ignoring data
 with redundant timestamp for frame origin at time 1542193272.093237 according to au
thority unknown_publisher
[ WARN] [1666790859.009965491, 1542193272.115328331]: TF_REPEATED_DATA ignoring data
 with redundant timestamp for frame origin at time 1542193272.103304 according to au
thority unknown_publisher
[ WARN] [1666790859.021394615, 1542193272.105595186]: Detected jump back in time of
0.00973315s. Clearing TF buffer.
[ WARN] [1666790859.022272987, 1542193272.115620850]: TF_REPEATED_DATA ignoring data
 with redundant timestamp for frame origin at time 1542193272.115328 according to au
thority unknown_publisher
[ WARN] [1666790859.031801396, 1542193272.125667554]: TF_REPEATED_DATA ignoring data
 with redundant timestamp for frame origin at time 1542193272.125417 according to au
thority unknown_publisher
```

```
 [RUNNING]  Bag Time: 1542193271.954670   Duration: 30.449826 / 79.921261
 [RUNNING]  Bag Time: 1542193271.964735   Duration: 30.459891 / 79.921261
 [RUNNING]  Bag Time: 1542193271.974800   Duration: 30.469956 / 79.921261
 [RUNNING]  Bag Time: 1542193271.984865   Duration: 30.480021 / 79.921261
 [RUNNING]  Bag Time: 1542193271.994935   Duration: 30.490091 / 79.921261
 [RUNNING]  Bag Time: 1542193272.004993   Duration: 30.500149 / 79.921261
 [RUNNING]  Bag Time: 1542193272.015034   Duration: 30.510190 / 79.921261
 [RUNNING]  Bag Time: 1542193272.025101   Duration: 30.520257 / 79.921261
 [RUNNING]  Bag Time: 1542193272.035160   Duration: 30.530316 / 79.921261
 [RUNNING]  Bag Time: 1542193272.045226   Duration: 30.540382 / 79.921261
 [RUNNING]  Bag Time: 1542193272.055285   Duration: 30.550441 / 79.921261
 [RUNNING]  Bag Time: 1542193272.065351   Duration: 30.560507 / 79.921261
 [RUNNING]  Bag Time: 1542193272.075411   Duration: 30.570567 / 79.921261
 [RUNNING]  Bag Time: 1542193272.085477   Duration: 30.580633 / 79.921261
 [RUNNING]  Bag Time: 1542193272.095536   Duration: 30.590692 / 79.921261
 [RUNNING]  Bag Time: 1542193272.105595   Duration: 30.600751 / 79.921261
 [RUNNING]  Bag Time: 1542193272.115621   Duration: 30.610777 / 79.921261
```

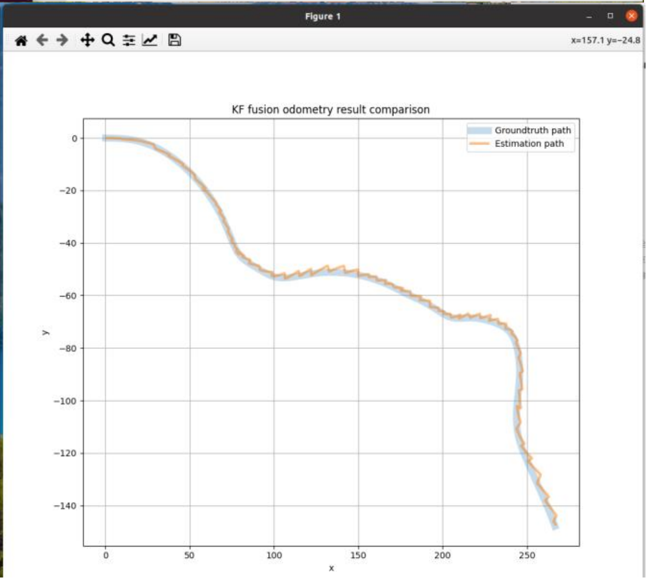


Synchronization: Off　ROS Time:)272.09　ROS Elapsed: 30.59　Wall Time:)859.03　Wall Elapsed: 36.10


/home/jonathan/catkin_ws/src/kalman_filter/launch/KF_fusi...

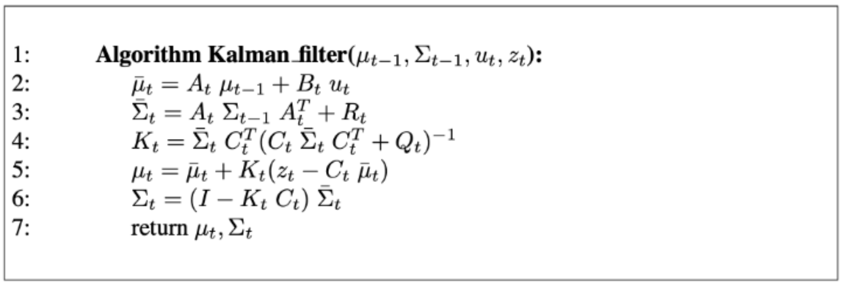
```
 of 9.0587e-05s. Clearing TF buffer.
[ WARN] [1666792643.288767151, 1542193321.374270608]: TF_REPEATED_DATA ignoring
data with redundant timestamp for frame origin at time 1542193321.374361 accordi
ng to authority unknown_publisher
[ WARN] [1666792643.389954219, 1542193321.384472555]: TF_REPEATED_DATA ignoring
data with redundant timestamp for frame origin at time 1542193321.384473 accordi
ng to authority unknown_publisher
[ WARN] [1666792643.503714407, 1542193321.395786435]: TF_REPEATED_DATA ignoring
data with redundant timestamp for frame origin at time 1542193321.395787 accordi
ng to authority unknown_publisher
[ WARN] [1666792643.604361358, 1542193321.405894609]: TF_REPEATED_DATA ignoring
data with redundant timestamp for frame origin at time 1542193321.405895 accordi
ng to authority unknown_publisher
[ WARN] [1666792643.705446433, 1542193321.416007798]: TF_REPEATED_DATA ignoring
data with redundant timestamp for frame origin at time 1542193321.416008 accordi
ng to authority unknown_publisher
[ WARN] [1666792643.806400552, 1542193321.426104536]: TF_REPEATED_DATA ignoring
data with redundant timestamp for frame origin at time 1542193321.426105 accordi
ng to authority unknown_publisher
^C[Kalman2D_fusion-3] killing on exit
[rviz-2] killing on exit
shuting down fusion.py
[Kalman2D_fusion-3] escalating to SIGTERM
```

```
 [RUNNING]  Bag Time: 1542193321.321934   Duration: 79.817090 / 79.921261
 [RUNNING]  Bag Time: 1542193321.322060   Duration: 79.817216 / 79.921261
 [RUNNING]  Bag Time: 1542193321.332001   Duration: 79.827157 / 79.921261
 [RUNNING]  Bag Time: 1542193321.332164   Duration: 79.827320 / 79.921261
 [RUNNING]  Bag Time: 1542193321.342067   Duration: 79.837224 / 79.921261
 [RUNNING]  Bag Time: 1542193321.343073   Duration: 79.838229 / 79.921261
 [RUNNING]  Bag Time: 1542193321.353137   Duration: 79.848293 / 79.921261
 [RUNNING]  Bag Time: 1542193321.353475   Duration: 79.848631 / 79.921261
 [RUNNING]  Bag Time: 1542193321.363202   Duration: 79.858358 / 79.921261
 [RUNNING]  Bag Time: 1542193321.364207   Duration: 79.859363 / 79.921261
 [RUNNING]  Bag Time: 1542193321.374271   Duration: 79.869427 / 79.921261
 [RUNNING]  Bag Time: 1542193321.374282   Duration: 79.869438 / 79.921261
 [RUNNING]  Bag Time: 1542193321.384337   Duration: 79.879493 / 79.921261
 [RUNNING]  Bag Time: 1542193321.384381   Duration: 79.879537 / 79.921261
 [RUNNING]  Bag Time: 1542193321.394403   Duration: 79.889559 / 79.921261
 [RUNNING]  Bag Time: 1542193321.395409   Duration: 79.890565 / 79.921261
 [RUNNING]  Bag Time: 1542193321.405473   Duration: 79.900630 / 79.921261
 [RUNNING]  Bag Time: 1542193321.405815   Duration: 79.900971 / 79.921261
 [RUNNING]  Bag Time: 1542193321.415537   Duration: 79.910693 / 79.921261
 [RUNNING]  Bag Time: 1542193321.415913   Duration: 79.911069 / 79.921261
 [RUNNING]  Bag Time: 1542193321.425602   Duration: 79.920758 / 79.921261
```



3. Discussions

For the implementation of the Kalman filter, I followed the algorithm in the following figure.



```
1:    Algorithm Kalman_filter($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):
2:        $\bar{\mu}_t = A_t\, \mu_{t-1} + B_t\, u_t$
3:        $\bar{\Sigma}_t = A_t\, \Sigma_{t-1}\, A_t^T + R_t$
4:        $K_t = \bar{\Sigma}_t\, C_t^T (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$
5:        $\mu_t = \bar{\mu}_t + K_t (z_t - C_t\, \bar{\mu}_t)$
6:        $\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$
7:        return $\mu_t, \Sigma_t$
```

For the prediction function, I first calculated the predicted state estimate x. I obtain the expected state estimate x by adding the dot product of the transition matrix A and the state estimate x, and the dot product of the transition matrix B and the control vector u. Then I calculate the error matrix P by adding up the covariance of the state transition error Q with the dot product of the transition matrix A, the error matrix P, and the transform of the transition matrix A.

For the update function, I first calculate the innovation covariance S, which can be found in the parentheses in the equation of the optimal Kalman gain K in the algorithm above. For the innovation covariance S, I added up the measurement error R with the dot product of the observation matrix H, the error matrix P, and the transform of the observation matrix H. Then I obtained the optimal Kalman gain K by calculating the dot product of the error matrix P, the transform of the observation matrix H, and the inverse of the innovation covariance S matrix.

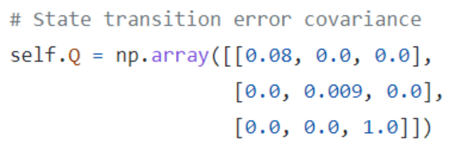
Then I'll calculate the measurement post-fit residual y, which is the subtraction of the dot product of the observation matrix H and the state estimate x to the matrix z. With the measurement post-fit residual y, we can get the final state estimate x by adding the state estimate x itself with the dot product of the optimal Kalman gain K and the measurement post-fit residual y.

Lastly, for the error matrix P, we first initialize a 3x3 matrix I with np.eye(), which with ones on the diagonal and zeros elsewhere. Then we calculate the error matrix P by calculating the dot product of the subtraction of the dot product of the optimal Kalman gain K and the observation matrix H from the 3x3 matrix we just created and the error matrix P itself. Finally, we get the final error matrix P, and the following figure is the plotting result of the Kalman filter I implemented.

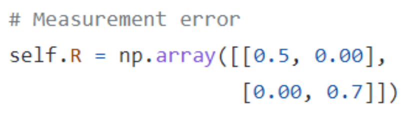
Besides, the matrix “A” is the identity matrix, and then matrix “B” will calculate the middle angle according to the position of the last msg received from the GPS, the current position of the radar odometry and the position of the msg of the GPS this time. This corrects the direction error accumulated by the radar odometry.

The matrix “u” in the Kalman filter can be obtained by subtracting the previous point from the current odometry point. However, since the radar odometry is very inaccurate, we will need the rotation matrix mentioned above to correct it.

For the covariance matrices of GPS & radar odometry are R and Q matrix, which are set as below. When the value of Q is larger, it means that the prediction based on the current state is more inaccurate. When R is larger, it means that the observation is more inaccurate. That is to say, the ratio of Q to R will determine the offset of the result to the observation point.



```
# State transition error covariance
self.Q = np.array([[0.08, 0.0, 0.0],
                   [0.0, 0.009, 0.0],
                   [0.0, 0.0, 1.0]])
```



```
# Measurement error
self.R = np.array([[0.5, 0.00],
                   [0.00, 0.7]])
```