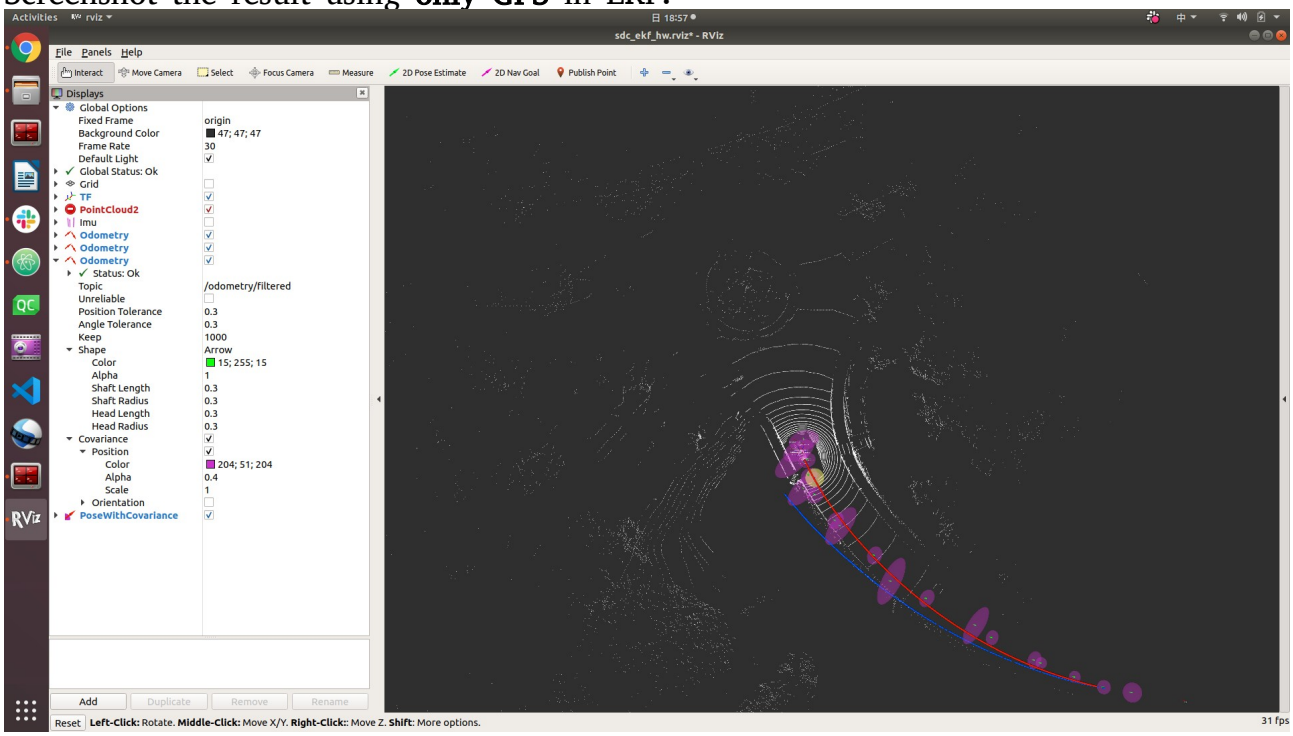
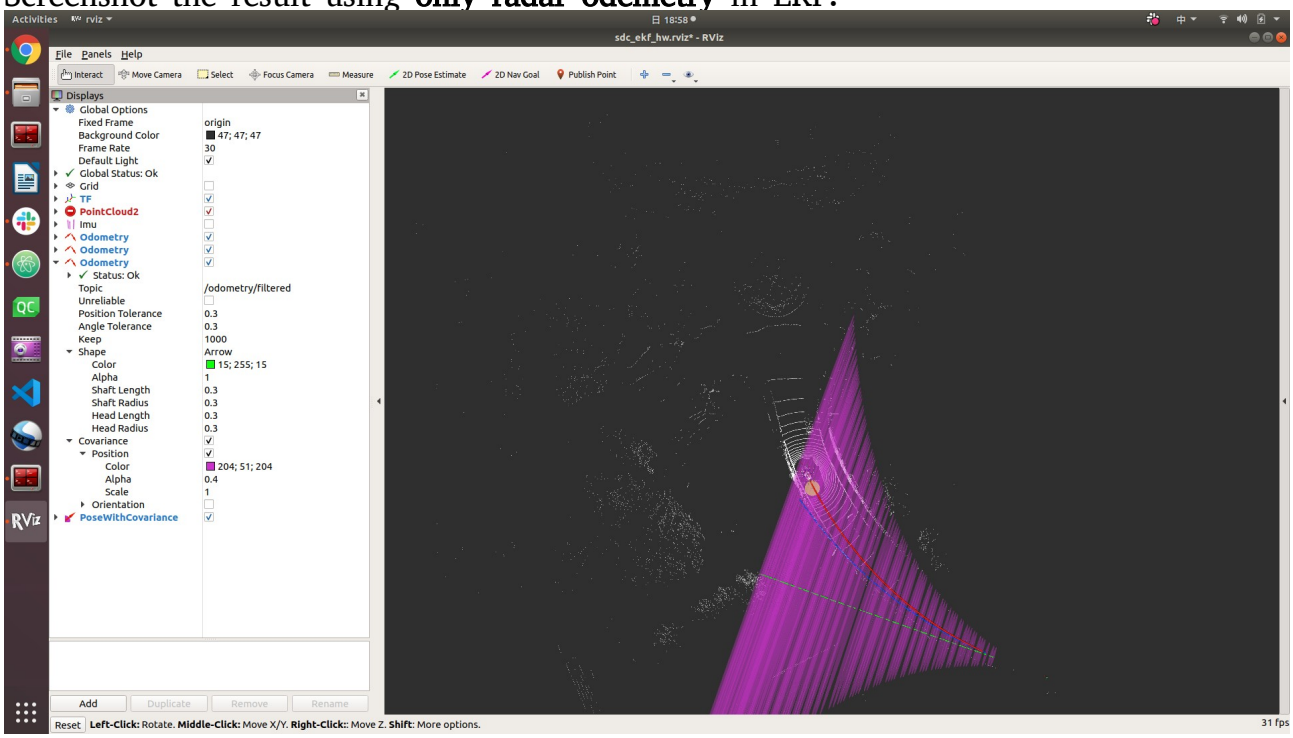
A. Screenshot Result

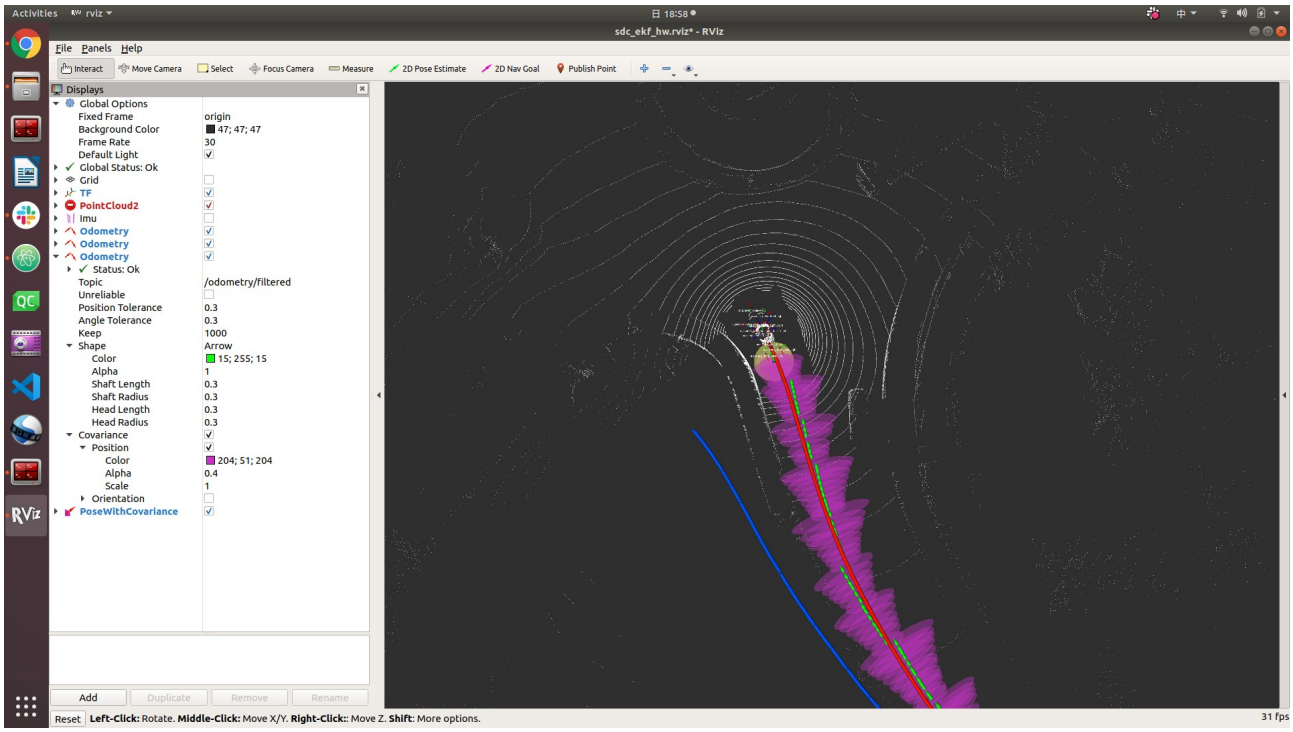
Screenshot the result using **only GPS** in EKF.



Screenshot the result using **only radar odemetry** in EKF.

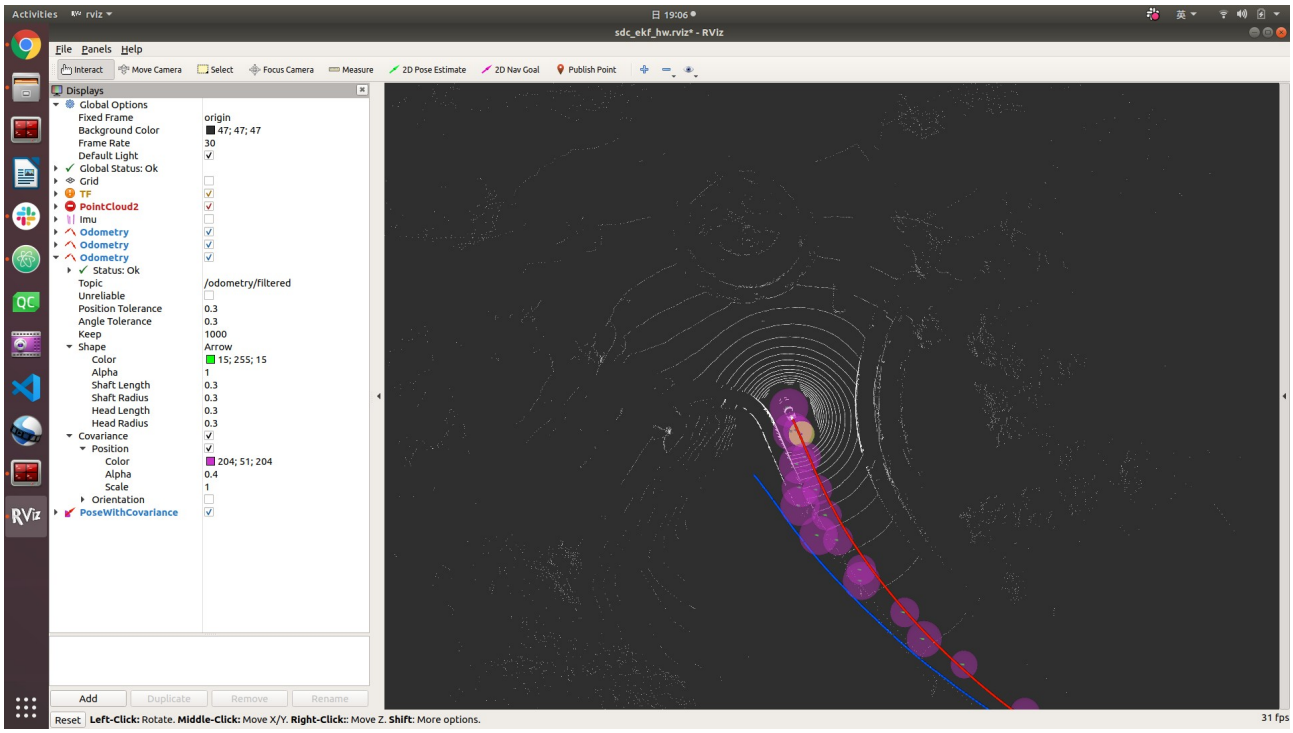


Screenshot the result using **both GPS and radar odemetry** in EKF.

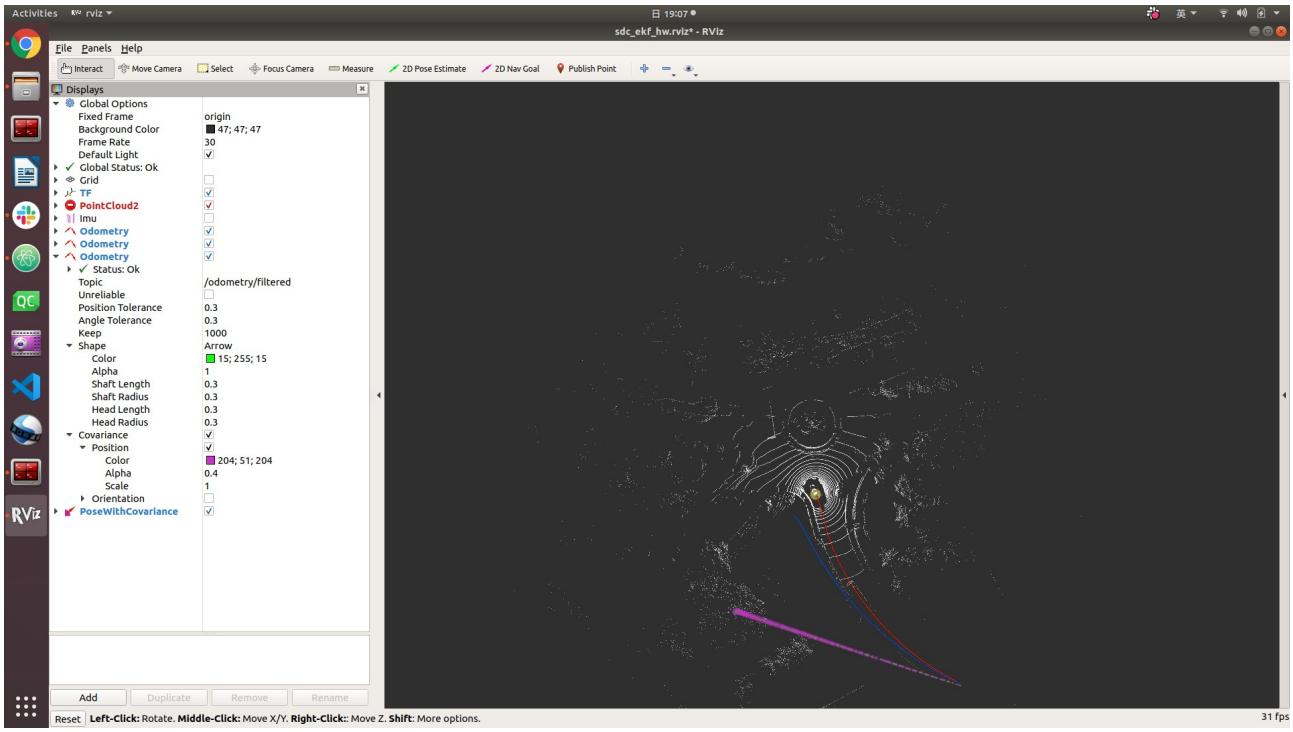


B. Bonus

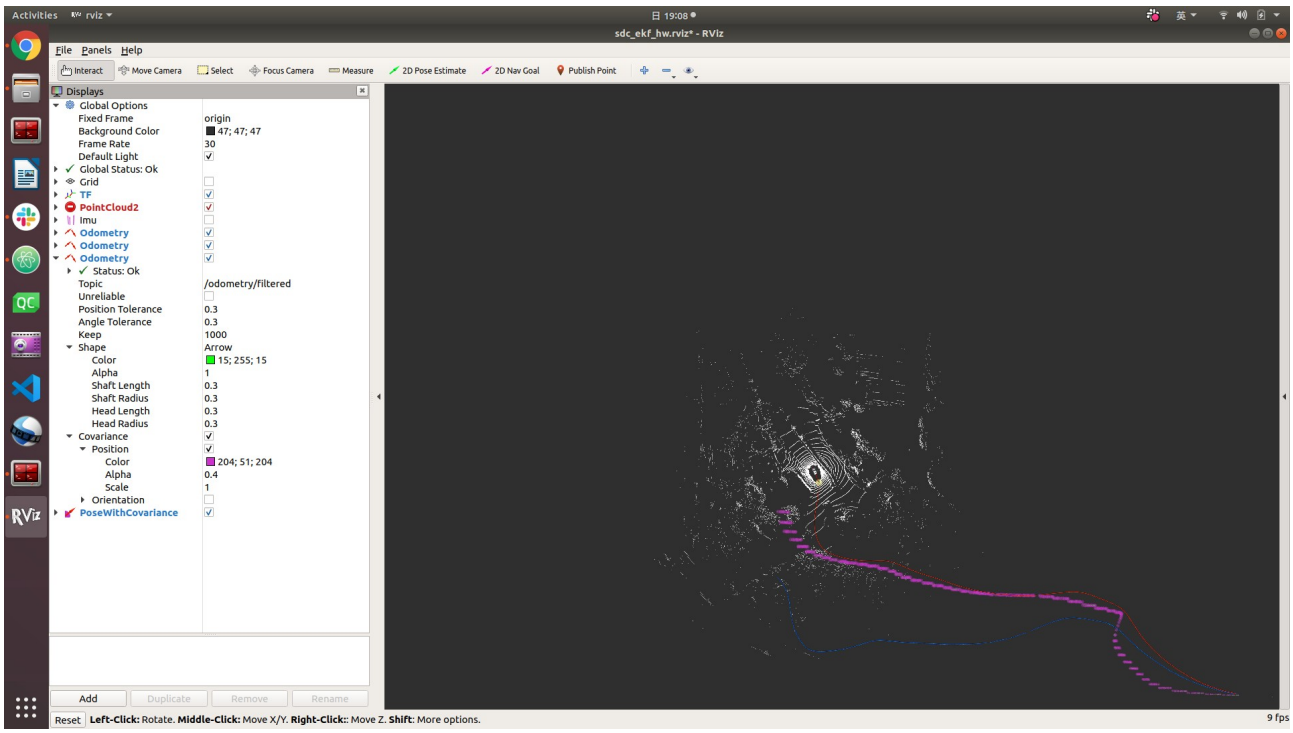
Screenshot the result using **only GPS** in UKF.



Screenshot the result using **only radar odemetry** in UKF.



Screenshot the result using **both GPS and radar odemetry** in UKF.



Compare the EKF result with the UKF result, describe your findings, and explain why.

EKF is better. The result (green line) of EKF is very close to the ground truth. There is theoretical proof though, that the UKF will perform better than the EKF in estimating the mean of the state estimates by one order in the Taylor Series expansion of the nonlinear transformation. In practice, however, the answer to which estimator will perform better than others will be system dependent. Maybe this system is linear nearly, so EKF works well, or the numerical condition is bad, like the state is small floating value: $10^{(-3)}$, UKF will lose precision in square root computation seriously.

C. Question Discussion

How do robot_localization package know the covariance matrix of GPS and radar odometry?

Which has been included in the ROS message of the topic.

What is the covariance matrix of GPS and what does it mean?

```
  covariance: [3.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

The covariance matrix is a powerful tool for expressing what remains uncertain about a group of variables that have already been measured many times. This tool can be applied to help make smart decisions about systems which are messy and complex because they have many different kinds of uncertainty piled on top of each other.

In yaml file, you set differential parameter of odometry and GPS to true or false? Why?

For each of the sensor messages defined above that contain pose information, users can specify whether the pose variables should be integrated differentially. If a given value is set to true, then for a measurement at time 't' from the sensor in question, we first subtract the measurement at time 't − 1', and convert the resulting value to a velocity.

In this case, this setting is especially useful. We has two sources of absolute pose information : pose from radar odometry and GPS. If the variances on the input sources are not configured correctly, these measurements may get out of sync with one another and cause oscillations in the filter, but by integrating one or both of them differentially, we avoid this scenario.

From my experience, I have to set odom0_differential to true, otherwise, the "ekf_localization_node" will not publish any message to the topic '/odometry/filtered'. I have to set pose0_differential to false, because it is not continuous, otherwise, the covariance of the pose will will be divergence.

D. Report summarizing the paper

This paper provides a practical introduction to the discrete Kalman filter, especially EKF. The Kalman filter addresses the general problem of trying to estimate the state of a discrete-time controlled process that is governed by a linear stochastic difference equation. However, if the process is non-linear, we can linearize the estimation around the current estimate with Taylor series. A Kalman filter that linearizes about the current mean and covariance is referred to as an extended Kalman filter or EKF.

To help in developing a better feel for the operation and capability of the filter, this paper also present a very simple example and the simulation. In the result of its simulation, the Kalman "filtering" is evident as the estimate appears considerably smoother than the noisy measurements.