第7章作业

一. 基础部分：补全代码

FILE: lidar_localization/src/model/kalman_filter/error_state_kalman_filter.cpp

滤波算法主要包括预测(Update函数)和观测(Correct函数)两个部分：

预测部分接收imu数据，基于惯性解算更新名义值，基于状态方程更新误差值。
观测部分同时接收imu数据和定位数据，首先利用imu数据进行预测，保证状态与定位数据时间同步，然后基于观测方程计算误差值，最后利用误差值对名义值进行修正，并将误差值清零。

修改1

FUNCTION: ErrorStateKalmanFilter::UpdateOdomEstimation

```
void ErrorStateKalmanFilter::UpdateOdomEstimation(
    Eigen::Vector3d &linear_acc_mid, Eigen::Vector3d &angular_vel_mid) {
  //
  // TODO: this is one possible solution to previous chapter, IMU Navigation,
  // assignment
  //
  // get deltas:
    size_t   index_curr_  = 1;
    size_t   index_prev_ = 0;
    Eigen::Vector3d  angular_delta;
    GetAngularDelta(index_curr_,  index_prev_,   angular_delta,
 angular_vel_mid);           //   获取等效旋转矢量，   保存角速度中值
  // update orientation:
    Eigen::Matrix3d  R_curr_;
    Eigen::Matrix3d  R_prev_;
    UpdateOrientation(angular_delta, R_curr_, R_prev_);
//      更新四元数
  // get velocity delta:
    double   delta_t_ = 0.0;
    Eigen::Vector3d  velocity_delta_;
    GetVelocityDelta(index_curr_, index_prev_,  R_curr_,  R_prev_, delta_t_,
 velocity_delta_,  linear_acc_mid);
  // save mid-value unbiased linear acc for error-state update:

  // update position:
    UpdatePosition(delta_t_,  velocity_delta_);

}
```

修改2

FUNCTION:ErrorStateKalmanFilter::SetProcessEquation

```
void ErrorStateKalmanFilter::SetProcessEquation(const Eigen::Matrix3d &C_nb,
                                                const Eigen::Vector3d &f_n,
                                                const Eigen::Vector3d &w_b) {
 // TODO: set process / system equation:
  // a. set process equation for delta vel:
  F_.block<3,  3>(kIndexErrorVel,   kIndexErrorOri)  =  - C_nb *
 Sophus::SO3d::hat(f_n).matrix();
  F_.block<3,  3>(kIndexErrorVel,   kIndexErrorAccel) =  -C_nb;
  // b. set process equation for delta ori:
  F_.block<3,  3>(kIndexErrorOri,   kIndexErrorOri) =   -
Sophus::SO3d::hat(w_b).matrix();
  B_.block<3,  3>(kIndexErrorVel,  kIndexErrorAccel)  =    C_nb;
}
```

修改3

FUNCTION:ErrorStateKalmanFilter::UpdateErrorEstimation

```
void ErrorStateKalmanFilter::UpdateErrorEstimation(
    const double &T, const Eigen::Vector3d &linear_acc_mid,
    const Eigen::Vector3d &angular_vel_mid) {
  static MatrixF F_1st;
  static MatrixF F_2nd;
// TODO: update process equation:
  UpdateProcessEquation(linear_acc_mid ,  angular_vel_mid);   //  更新状态方程

  // TODO: get discretized process equations:  //   非线性化
    F_1st  =  F_ *  T;        //  T kalman 周期
    MatrixF   F = MatrixF::Identity()  +   F_1st;
    MatrixB  B =  B_*T;
  // TODO: perform Kalman prediction
      P_ =  F * P_ * F.transpose()   +  B * Q_ * B.transpose();           //
只有方差进行了计算
}
```

修改4

FUNCTION:ErrorStateKalmanFilter::CorrectErrorEstimationPose

```
void ErrorStateKalmanFilter::CorrectErrorEstimationPose(
    const Eigen::Matrix4d &T_nb, Eigen::VectorXd &Y, Eigen::MatrixXd &G,
    Eigen::MatrixXd &K) {
  //
  // TODO: set measurement: 计算观测 delta pos  、 delta  ori
  //
  Eigen::Vector3d  dp  =  pose_.block<3,  1>(0,  3)  -   T_nb.block<3,  1>(0,
 3);
  Eigen::Matrix3d  dR  =  T_nb.block<3,  3>(0, 0).transpose() *  pose_.block<3,
3>(0, 0) ;
  // TODO: set measurement equation:
  Eigen::Vector3d  dtheta  =  Sophus::SO3d::vee(dR  -
 Eigen::Matrix3d::Identity() );
  YPose_.block<3, 1>(0, 0)  =  dp;          //   delta  position
  YPose_.block<3, 1>(3, 0)  =  dtheta;          //   失准角
  Y = YPose_;
```

```
  G  =   GPose_;
   //   set measurement  C

  // TODO: set Kalman gain:
  K =  P_  *  G.transpose() * ( G  *  P_  *  G.transpose( )  +   RPose_
).inverse() ;
}
```

修改5

FUNCTION:ErrorStateKalmanFilter::EliminateError

```
void ErrorStateKalmanFilter::EliminateError(void) {
  //  误差状态量  X_  :   15*1
  // TODO: correct state estimation using the state of ESKF
  //
  // a. position:
  // do it!
  pose_.block<3, 1>(0, 3)  =  pose_.block<3, 1>(0, 3) - X_.block<3, 1>
(kIndexErrorPos, 0 );   //  减去error

  // b. velocity:
  // do it!
  vel_ = vel_ - X_.block<3,1>(kIndexErrorVel, 0 );

  // c. orientation:
  // do it!
  Eigen::Matrix3d   delta_R =  Eigen::Matrix3d::Identity() -
Sophus::SO3d::hat(X_.block<3,1>(kIndexErrorOri, 0)).matrix();         //   失准角
的反对称矩阵
  Eigen::Quaterniond dq = Eigen::Quaterniond(delta_R);
  dq = dq.normalized();        //  为了保证旋转矩阵是正定的
  pose_.block<3, 3>(0, 0) = pose_.block<3,3>(0,0) * dq.toRotationMatrix();
  // d. gyro bias:
  gyro_bias_ -= X_.block<3, 1>(kIndexErrorGyro, 0);
  // e. accel bias:
  accl_bias_ -= X_.block<3, 1>(kIndexErrorAccel, 0);

}
```

运行

代码运行命令：

```
roslaunch lidar_localization kitti_localization.launch
```

播放数据集命令：

```
rosbag  play kitti_lidar_only_2011_10_03_drive_0027_synced.bag
```

保存里程计：

```
rosservice call /save_odometry
```

数据位于:

```
src/lidar_localization/slam_data/trajectory
```

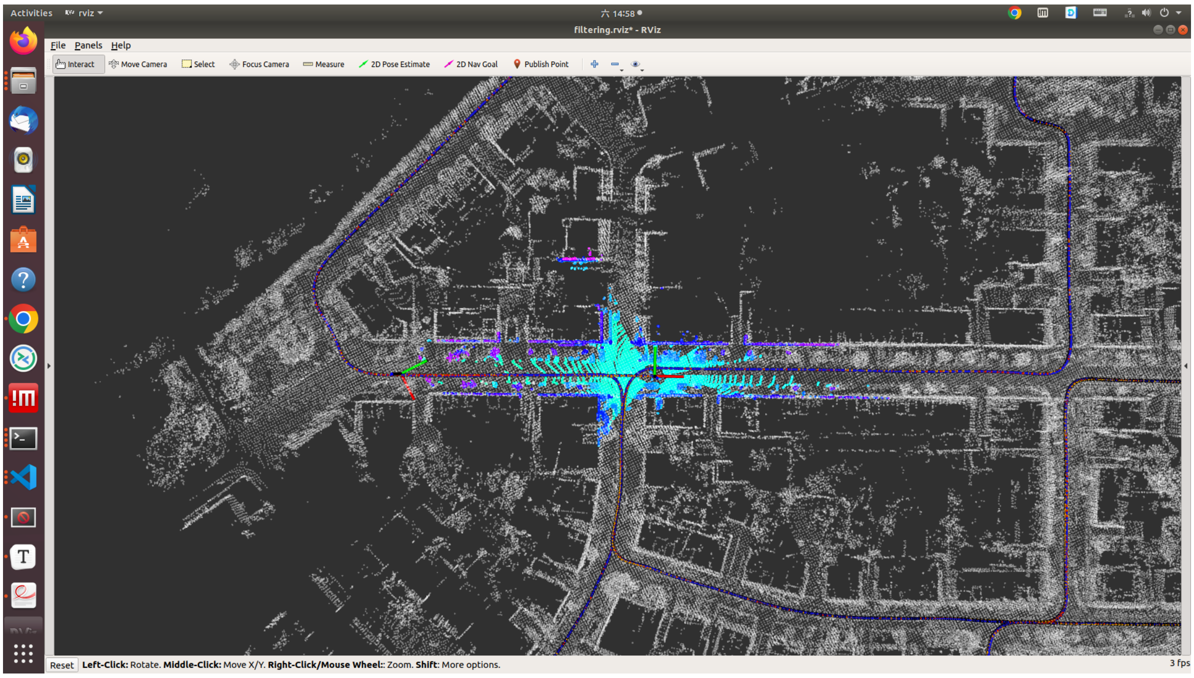
evo工具运行命令：

```
# a. laser  输出评估结果，并以zip的格式存储:
evo_ape kitti ground_truth.txt laser.txt -r full --plot --plot_mode xy  --
save_results ./laser.zip
# b. fused 输出评估结果，并以zip的格式存储:
evo_ape kitti ground_truth.txt fused.txt -r full --plot --plot_mode xy  --
save_results ./fused.zip
#c. 比较 laser  fused  一并比较评估
evo_res  *.zip -p
```

默认参数（参数1）

```
    covariance:
        prior:
            pos: 1.0e-6
            vel: 1.0e-6
            ori: 1.0e-6
            epsilon: 1.0e-6
            delta: 1.0e-6
        process:
            gyro: 1.0e-4
            accel: 2.5e-3
            bias_accel: 2.5e-3
            bias_gyro: 1.0e-4
        measurement:
            pose:
            pos: 1.0e-2
            ori: 1.0e-2
            pos: 1.0e-2
            vel: 2.5e-1
```

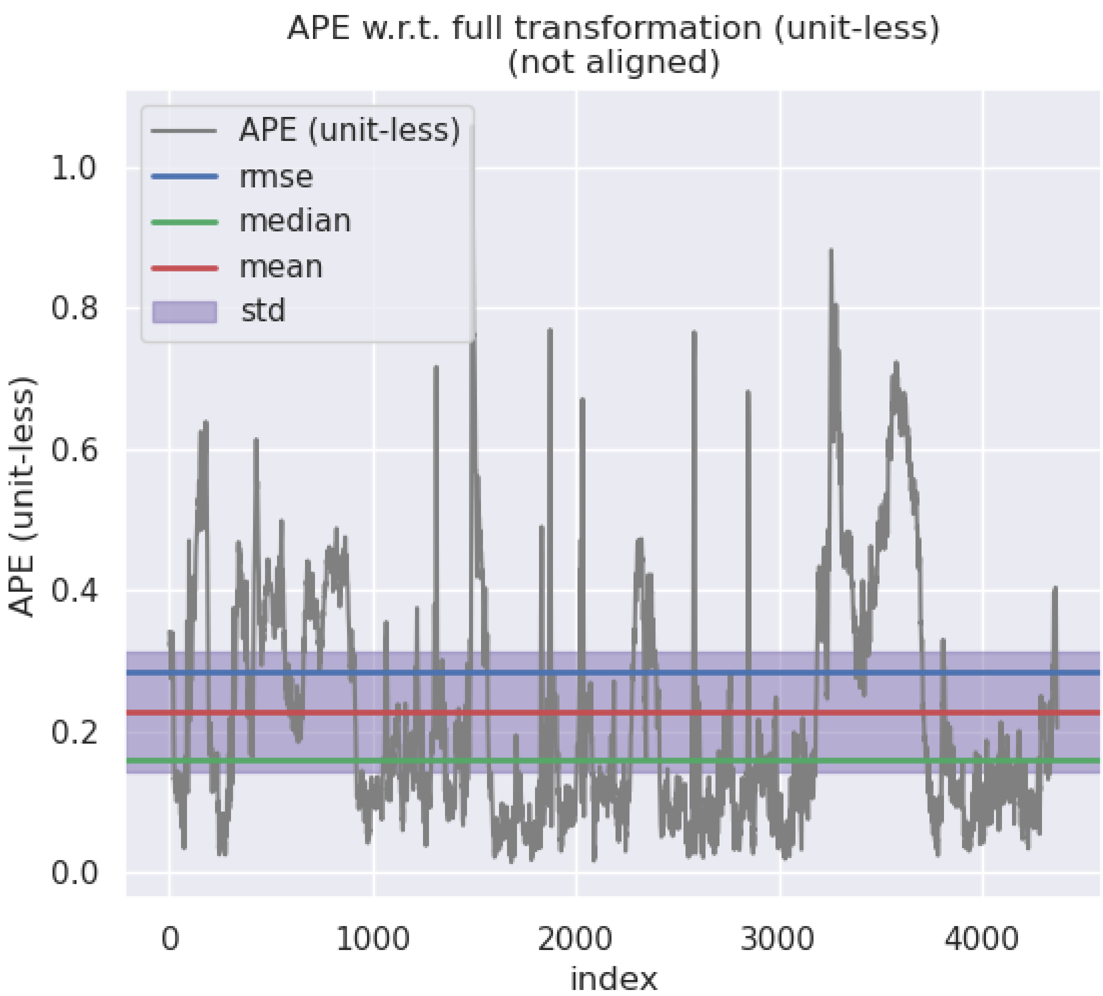
RVIZ效果



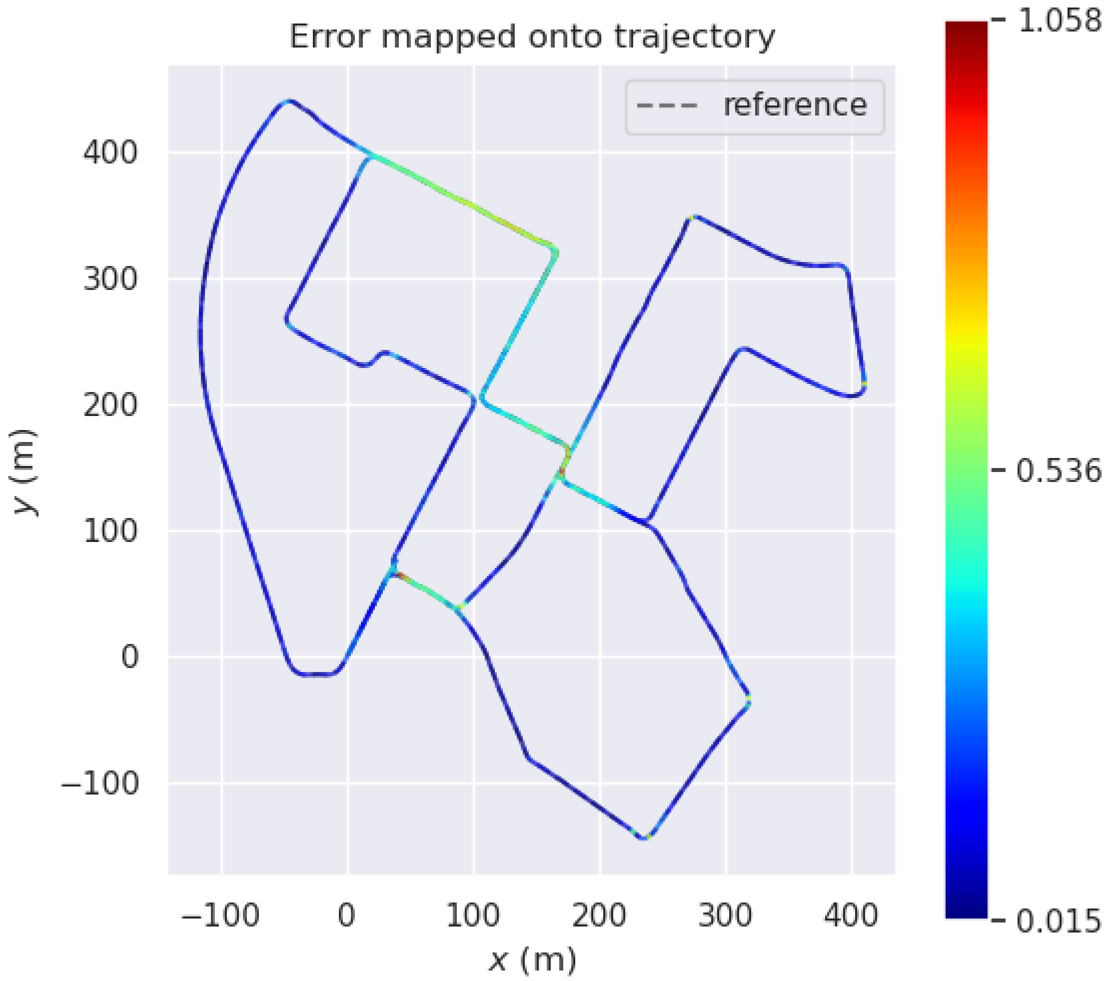
Activities
rviz
filtering.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
3 fps

EVO评估

雷达：

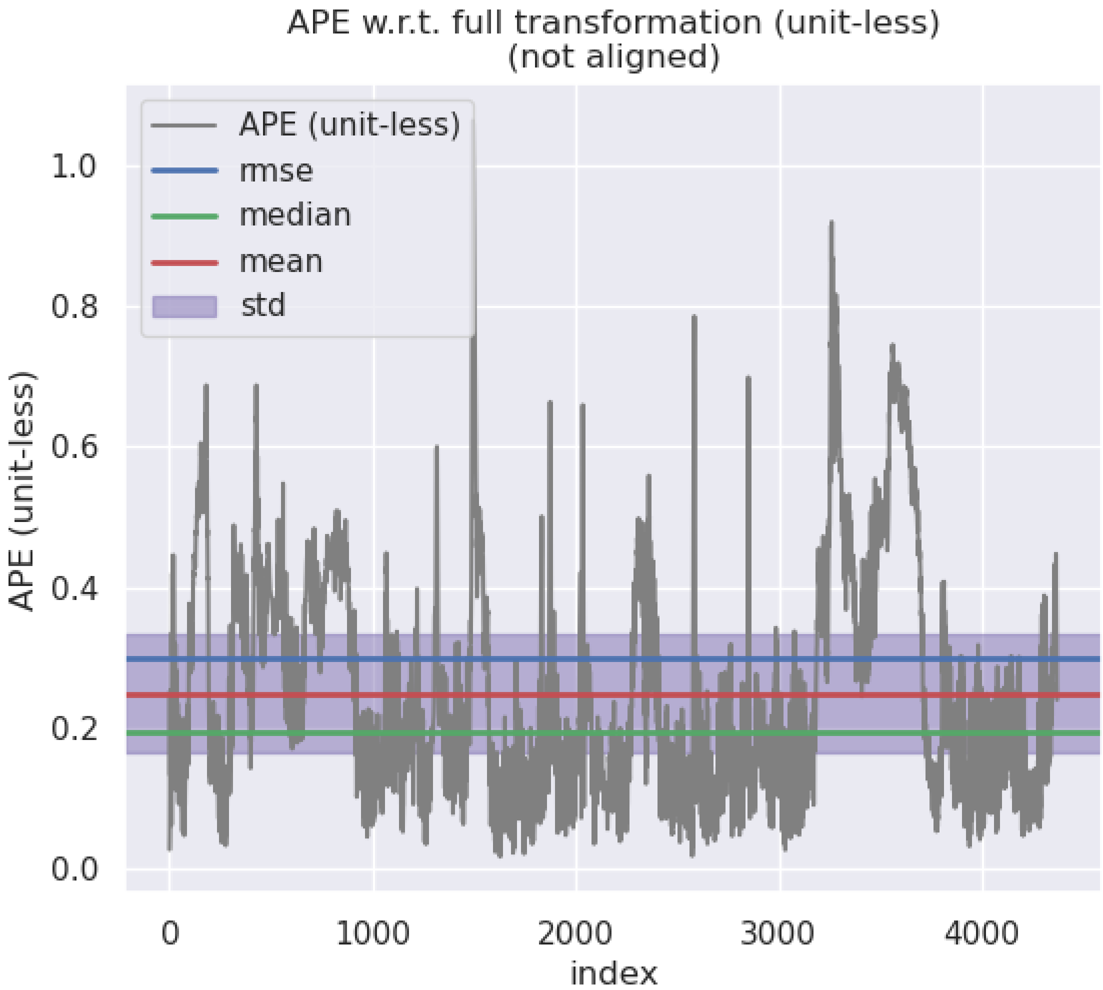


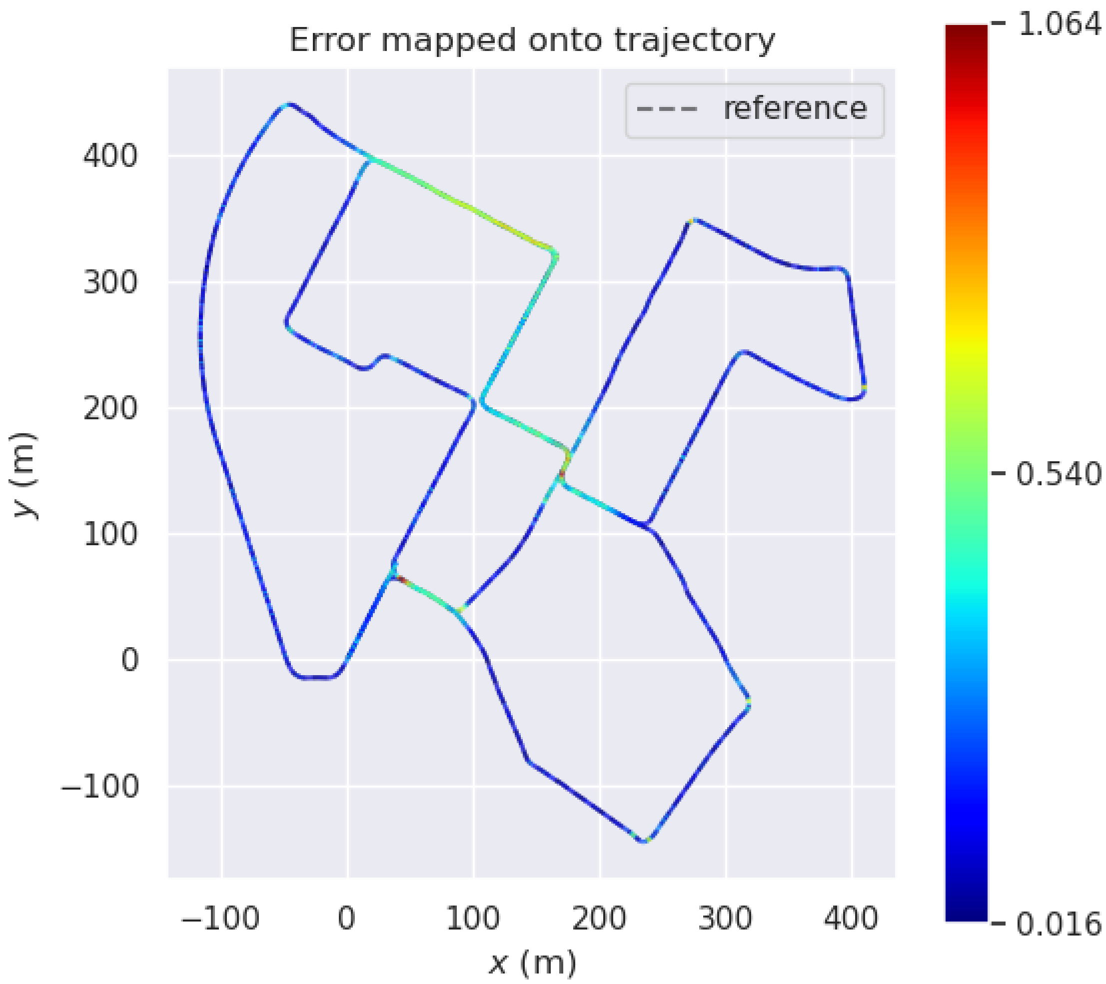
APE w.r.t. full transformation (unit-less)
(not aligned)
APE (unit-less)
rmse
median
mean
std
1.0
0.8
0.6
0.4
0.2
0.0
APE (unit-less)
0
1000
2000
3000
4000
index




```
   max   1.058145
  mean   0.228117
median   0.160389
   min   0.014763
  rmse   0.284866
   sse   354.780742
   std   0.170620
```

融合后:


APE w.r.t. full transformation (unit-less)
(not aligned)
APE (unit-less)
rmse
median
mean
std
1.0
0.8
0.6
0.4
0.2
0.0
0
1000
2000
3000
4000
index

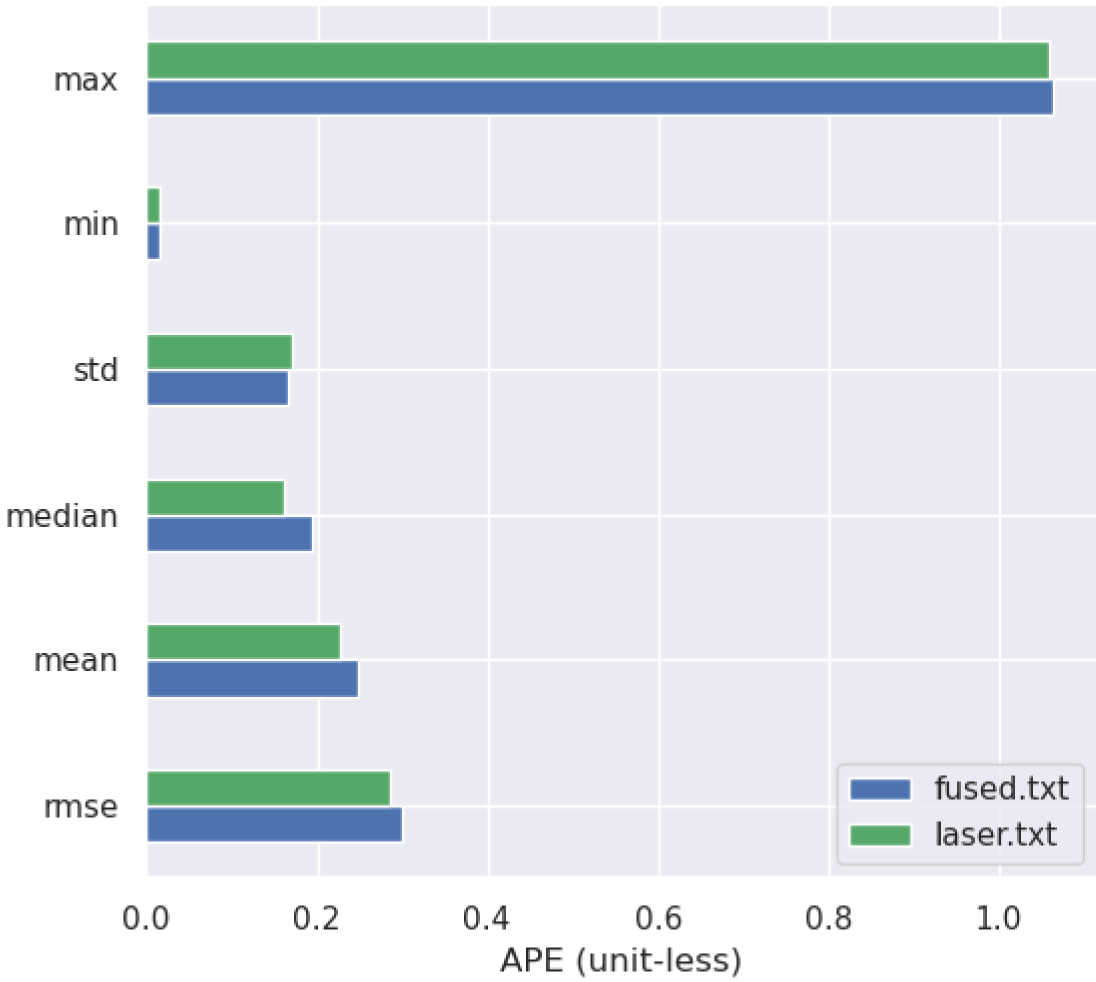

Error mapped onto trajectory
reference
400
300
200
100
0
−100
−100
0
100
200
300
400
x (m)
y (m)
1.064
0.540
0.016

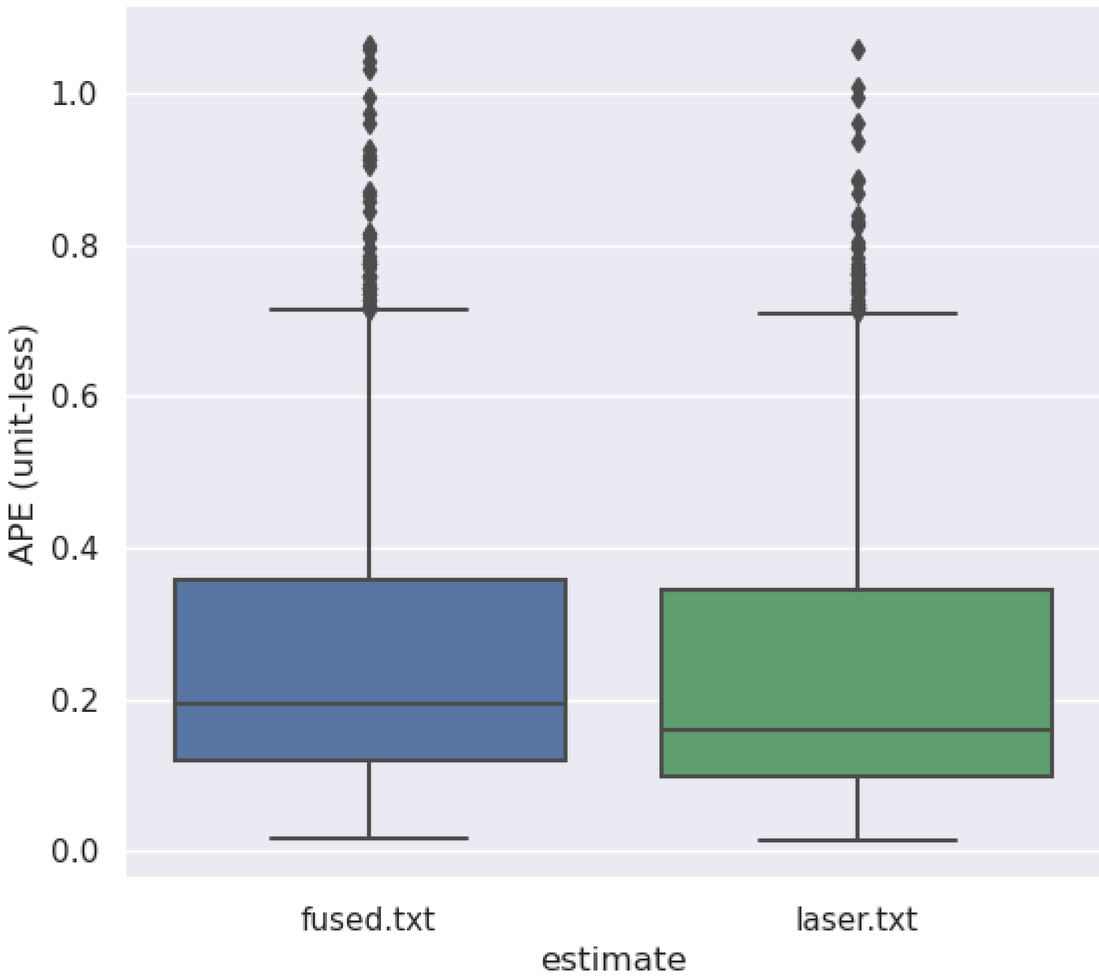
```
       max   1.063830
      mean   0.248405
    median   0.193894
       min   0.016452
      rmse   0.299318
       sse   391.693699
       std   0.166992
```

对比:

```
                rmse      mean    median       std       min       max  \
fused.txt  0.299318  0.248405  0.193894  0.166992  0.016452   1.06383
laser.txt  0.284866  0.228117  0.160389   0.17062  0.014763  1.058145

                  sse
fused.txt  391.693699
laser.txt  354.780742
```





结论

默认参数下，融合的效果是差于单雷达的效果的，基本差不多。

二.参数优化

参数修改（参数6）

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-4
        accel: 2.5e-3
        bias_accel: 2.5e-3
        bias_gyro: 1.0e-4
    measurement:
        pose:
        pos: 1.0e-4
        ori: 1.0e-4
        pos: 1.0e-4
        vel: 2.5e-3
```

EVO评估

```
                 rmse      mean    median       std       min       max  \
fused.txt  0.284697  0.227977  0.160314  0.170526  0.014329  1.058306
laser.txt  0.284696  0.227975  0.160564  0.170527  0.014763  1.058145

                  sse
fused.txt  354.848121
laser.txt  354.846029
```

结论

经过参数调整，融合的效果基本和单雷达的效果一致。

三.不考虑随机游走的模型推导

公式推导

状态方程：

$$\delta \dot{x} = F_t \delta x + B_t w$$

状态量：

$$\begin{aligned}
\delta \dot{p} &= \delta \boldsymbol{v} \\
\delta \dot{\boldsymbol{v}} &= -\boldsymbol{R}_t [\boldsymbol{a_t} - \boldsymbol{b}_{\boldsymbol{a}_t}]_\times \delta \boldsymbol{\theta} + \boldsymbol{R_t} \left(\boldsymbol{n_a} - \delta \boldsymbol{b_a} \right) \\
\delta \dot{\boldsymbol{\theta}} &= -[\boldsymbol{\omega}_t - \boldsymbol{b}_{\omega_t}]_\times \delta \boldsymbol{\theta} + \boldsymbol{n}_\omega - \delta \boldsymbol{b}_\omega \\
\delta \dot{\boldsymbol{b}}_a &= 0 \\
\delta \dot{\boldsymbol{b}}_w &= 0
\end{aligned}$$

过程噪声部分：

线性kalman：

$$\boldsymbol{Q} = [\boldsymbol{Q}_a \quad Q_\omega \quad 0 \quad 0]$$

$$\boldsymbol{B}_t = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \boldsymbol{R}_t & 0 & 0 & 0 \\ 0 & \boldsymbol{I}_3 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

非线性kalman：

$$\boldsymbol{Q} = [\boldsymbol{Q}_a \quad Q_\omega \quad 0 \quad 0]$$

$$\boldsymbol{B}_{k-1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \boldsymbol{R}_{k-1} T & 0 & 0 & 0 \\ 0 & \boldsymbol{I}_3 T & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

工程实现

在 kitti_filtering.yaml 中添加bias_flag选项，以选择是否考虑使用随机游走。

FILE: src/lidar_localization/config/filtering/kitti_filtering.yaml

```
covariance:
       prior:
           pos: 1.0e-6
           vel: 1.0e-6
           ori: 1.0e-6
           epsilon: 1.0e-6
           delta: 1.0e-6
       process:
           gyro: 1.0e-4
           accel: 2.5e-3
           bias_accel: 2.5e-3
           bias_gyro: 1.0e-4
           bias_flag: true
       measurement:
           pose:
               pos: 1.0e-4
               ori: 1.0e-4
           pos: 1.0e-4
           vel: 2.5e-3
```

代码修改

FUNCTION:ErrorStateKalmanFilter::ErrorStateKalmanFilter

```
  COV.PROCESS.BIAS_FLAG =
      node["covariance"]["process"]["bias_flag"].as<bool>();
```

FUNCTION:ErrorStateKalmanFilter::ErrorStateKalmanFilter

```
 // c. process noise:
  Q_.block<3, 3>(kIndexNoiseAccel, kIndexNoiseAccel) = COV.PROCESS.ACCEL *
Eigen::Matrix3d::Identity();
  Q_.block<3, 3>(kIndexNoiseGyro, kIndexNoiseGyro) = COV.PROCESS.GYRO *
Eigen::Matrix3d::Identity();
  if (COV.PROCESS.BIAS_FLAG ){
  Q_.block<3, 3>(kIndexNoiseBiasAccel, kIndexNoiseBiasAccel) =
COV.PROCESS.BIAS_ACCEL * Eigen::Matrix3d::Identity();
  Q_.block<3, 3>(kIndexNoiseBiasGyro, kIndexNoiseBiasGyro) =
COV.PROCESS.BIAS_GYRO * Eigen::Matrix3d::Identity();
  }
```

FUNCTION:ErrorStateKalmanFilter::SetProcessEquation

```
  // b. set process equation for delta ori:
  F_.block<3,  3>(kIndexErrorOri,   kIndexErrorOri) =   -
Sophus::SO3d::hat(w_b).matrix();
  B_.block<3,  3>(kIndexErrorVel,  kIndexErrorAccel)  =    C_nb;
  if(COV.PROCESS.BIAS_FLAG){      //  判断是否考虑随机游走
    B_.block<3,  3>(kIndexErrorAccel,  kIndexNoiseBiasAccel)  =
Eigen::Matrix3d::Identity();
    B_.block<3,  3>(kIndexErrorGyro,  kIndexNoiseBiasGyro)    =
Eigen::Matrix3d::Identity();
  }
```

FUNCTION:ErrorStateKalmanFilter::UpdateErrorEstimation

```
    if(COV.PROCESS.BIAS_FLAG)     B =  B_*sqrt(T);
```

四.对比分析

当 Q/R 越大，则表示Q越大，预测的噪声越大，系统更相信观测；当 Q/R 越小，表示 R 越大，观测的噪声越大，系统更相信预测；
即当 Q 减少，R 不变时，更相信预测值；当 Q 增大，R 减小时，更相信观测值；当 R 减少，Q 不变时，更相信观测值；当 R 增大，Q 不变时，更相信预测值。

参数1：

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-4
        accel: 2.5e-3
        bias_accel: 2.5e-3
        bias_gyro: 1.0e-4
    measurement:
        pose:
        pos: 1.0e-2
        ori: 1.0e-2
        pos: 1.0e-2
        vel: 2.5e-1
```

不考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.952761  0.721738  0.526481  0.621971  0.028239   3.90681
laser.txt  0.284661  0.227867  0.160084  0.170612  0.014763  1.058145

                   sse
fused.txt  3954.177354
laser.txt   352.974659
```

考虑随机游走：

```
              rmse      mean    median       std       min       max  \
fused.txt  0.299318  0.248405  0.193894  0.166992  0.016452   1.06383
laser.txt  0.284866  0.228117  0.160389   0.17062  0.014763  1.058145

                  sse
fused.txt  391.693699
laser.txt  354.780742
```

参数2：

```
covariance:
        prior:
            pos: 1.0e-6
            vel: 1.0e-6
            ori: 1.0e-6
            epsilon: 1.0e-6
            delta: 1.0e-6
        process:
            gyro: 1.0e-4
            accel: 2.5e-3
            bias_accel: 2.5e-3
            bias_gyro: 1.0e-4
        measurement:
            pose:
            pos: 1.0e-4
            ori: 1.0e-4
            pos: 1.0e-4
            vel: 2.5e-3
```

不考虑随机游走：

```
                rmse      mean    median       std       min       max  \
fused.txt  0.438888  0.357005  0.285806  0.255285  0.026705  2.178491
laser.txt  0.284607  0.227924  0.160545  0.170445  0.014763  1.058145

                  sse
fused.txt   839.64202
laser.txt  353.083373
```

考虑随机游走：

```
                rmse     mean    median       std       min       max  \
fused.txt  0.299679  0.24884  0.194298  0.166991  0.018342  1.065207
laser.txt   0.28465  0.22786  0.160084  0.170604  0.014763  1.058145

                  sse
fused.txt  391.201546
laser.txt  352.948792
```

参数3：

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-4
        accel: 2.5e-3
        bias_accel: 2.5e-3
        bias_gyro: 1.0e-4
    measurement:
        pose:
        pos: 1.0e-6
        ori: 1.0e-6
        pos: 1.0e-6
        vel: 2.5e-5
```

不考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.354642  0.289052  0.226475  0.205474  0.021593  1.695696
laser.txt  0.284835  0.228046  0.160133  0.170663  0.014763  1.058145

                  sse
fused.txt  548.486621
laser.txt  353.811309
```

考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.289648  0.235745  0.175245  0.168287  0.020645  1.116669
laser.txt  0.284479  0.227683  0.159882  0.170555  0.014763  1.058145

                  sse
fused.txt  365.955408
laser.txt  353.010395
```

参数4：

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-2
        accel: 2.5e-1
        bias_accel: 2.5e-1
```

```
        bias_gyro: 1.0e-2
    measurement:
        pose:
        pos: 1.0e-6
        ori: 1.0e-6
        pos: 1.0e-6
        vel: 2.5e-5
```

不考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt   0.32493  0.263788  0.204127  0.189725  0.017245  1.290073
laser.txt  0.284226  0.227177    0.1588  0.170808  0.014763  1.058145

                  sse
fused.txt  460.115907
laser.txt  352.059608
```

考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt   0.28542  0.229484   0.16391  0.169711   0.01556  1.062555
laser.txt  0.284606  0.227953  0.160545  0.170405  0.014763  1.058145

                  sse
fused.txt  356.895876
laser.txt  354.863676
```

参数5：

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-1
        accel: 2.5e-0
        bias_accel: 2.5e-0
        bias_gyro: 1.0e-1
    measurement:
        pose:
        pos: 1.0e-6
        ori: 1.0e-6
        pos: 1.0e-6
        vel: 2.5e-5
```

不考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.305093  0.248036  0.187818  0.177651  0.018315  1.099339
laser.txt  0.284907  0.228132  0.160604  0.170669  0.014763  1.058145

                  sse
fused.txt  404.905921
laser.txt  353.098237
```

考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.284888  0.228233  0.159983  0.170501  0.015389  1.059236
laser.txt  0.284809   0.22809  0.160186  0.170561  0.014763  1.058145

                  sse
fused.txt   354.99839
laser.txt  354.803164
```

参数6：

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-1
        accel: 2.5e-0
        bias_accel: 2.5e-0
        bias_gyro: 1.0e-1
    measurement:
        pose:
        pos: 1.0e-7
        ori: 1.0e-7
        pos: 1.0e-7
        vel: 2.5e-6
```

不考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt   0.29411  0.237169   0.17132   0.17393  0.010965  1.015445
laser.txt  0.284321  0.227608  0.159898  0.170391  0.014763  1.058145

                  sse
fused.txt  378.613827
laser.txt  353.830077
```

考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.284697  0.227977  0.160314  0.170526  0.014329  1.058306
laser.txt  0.284696  0.227975  0.160564  0.170527  0.014763  1.058145

                  sse
fused.txt  354.848121
laser.txt  354.846029
```

参数7：

```
covariance:
    prior:
        pos: 1.0e-6
        vel: 1.0e-6
        ori: 1.0e-6
        epsilon: 1.0e-6
        delta: 1.0e-6
    process:
        gyro: 1.0e-6
        accel: 2.5e-5
        bias_accel: 2.5e-5
        bias_gyro: 1.0e-6
    measurement:
        pose:
        pos: 1.0e-6
        ori: 1.0e-6
        pos: 1.0e-6
        vel: 2.5e-5
```

不考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.452069  0.378197  0.327996  0.247655  0.022795  1.967658
laser.txt  0.284223    0.2275  0.159953  0.170371  0.014763  1.058145

                  sse
fused.txt   891.03615
laser.txt  352.212622
```

考虑随机游走：

```
               rmse      mean    median       std       min       max  \
fused.txt  0.294391  0.242138  0.185901  0.167438  0.013392   1.10459
laser.txt  0.284545  0.227908  0.160545  0.170365  0.014763  1.058145

                  sse
fused.txt  378.471748
laser.txt  353.578696
```

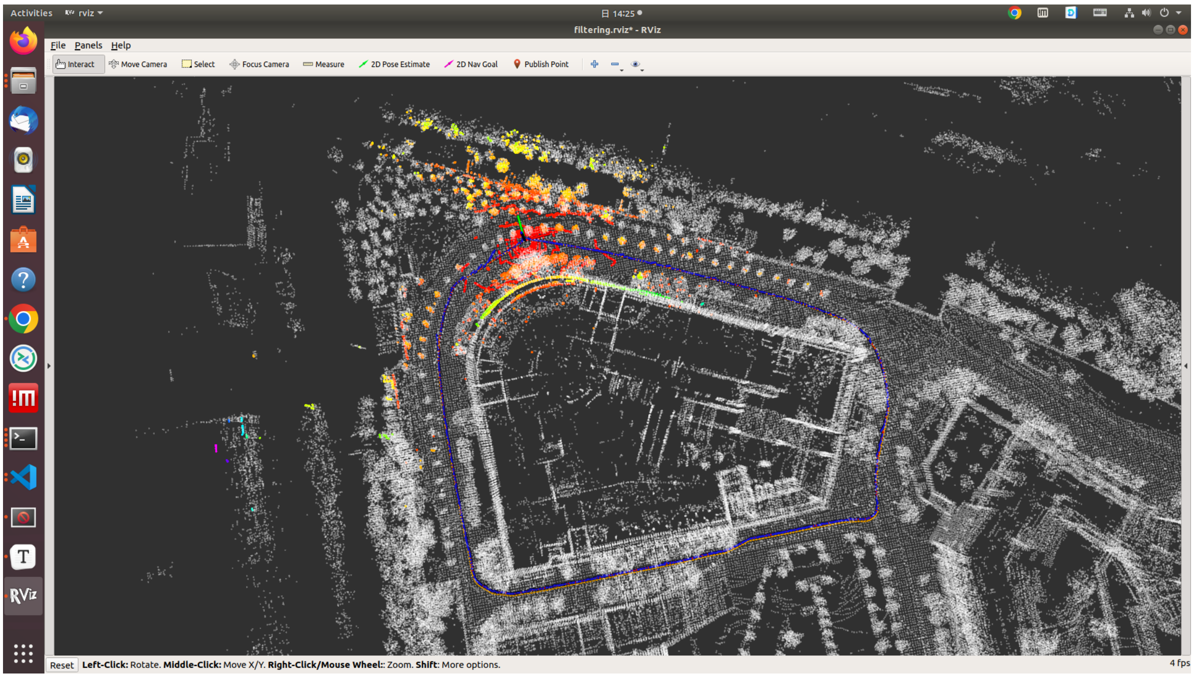
五.实车部署

实车硬件如下：

1. 松灵Scout2，车速为1.5m/s

2. 速腾16线雷达
3. SBG-ellipse-N 九轴惯导+单天线RTK

rviz效果



evo评估

```
               rmse       mean     median        std        min        max  \
fused.txt  0.674251    0.62103  0.606449  0.262556  0.118306  1.116956
laser.txt  0.673647   0.620549  0.605487  0.262144  0.118514  1.114238

                  sse
fused.txt  276.405189
laser.txt  275.910958
```

总结和思考

总结

参数表：

	参数1	参数2	参数3	参数4	参数5	参数6	参数7
Q	gyro: 1.0e-4 accel: 2.5e-3 bias_accel: 2.5e-3 bias_gyro: 1.0e-4	gyro: 1.0e-4 accel: 2.5e-3 bias_accel: 2.5e-3 bias_gyro: 1.0e-4	gyro: 1.0e-4 accel: 2.5e-3 bias_accel: 2.5e-3 bias_gyro: 1.0e-4	gyro: 1.0e-2 accel: 2.5e-1 bias_accel: 2.5e-1 bias_gyro: 1.0e-2	gyro: 1.0e-1 accel: 2.5e-0 bias_accel: 2.5e-0 bias_gyro: 1.0e-1	gyro: 1.0e-1 accel: 2.5e-0 bias_accel: 2.5e-0 bias_gyro: 1.0e-1	gyro: 1.0e-6 accel: 2.5e-5 bias_accel: 2.5e-5 bias_gyro: 1.0e-6
R	pos: 1.0e-2 ori: 1.0e-2 pos: 1.0e-2 vel: 2.5e-1	pos: 1.0e-4 ori: 1.0e-4 pos: 1.0e-4 vel: 2.5e-3	pos: 1.0e-6 ori: 1.0e-6 pos: 1.0e-6 vel: 2.5e-5	pos: 1.0e-6 ori: 1.0e-6 pos: 1.0e-6 vel: 2.5e-5	pos: 1.0e-6 ori: 1.0e-6 pos: 1.0e-6 vel: 2.5e-5	pos: 1.0e-7 ori: 1.0e-7 pos: 1.0e-7 vel: 2.5e-6	pos: 1.0e-6 ori: 1.0e-6 pos: 1.0e-6 vel: 2.5e-5

kitti数据集不同参数融合指标对比，选取sse作为评判指标参数，考虑随机游走：

parameter/trajectory	参数1	参数2	参数3	参数4	参数5	参数6	参数7
laser	354.780742	352.948792	353.010395	354.863676	354.803164	354.846029	353.578696
fused	391.693699	391.201546	365.955408	356.895876	354.99839	354.848121	378.471748

kitti数据集不同参数融合指标对比，选取sse作为评判指标参数，不考虑随机游走：

parameter/trajectory	参数1	参数2	参数3	参数4	参数5	参数6	参数7
laser	52.974659	353.083373	353.811309	352.059608	353.098237	353.830077	352.212622
fused	3954.177354	839.64202	548.486621	460.115907	404.905921	378.613827	891.03615

从指标得出这次实验的结论：

1. 考虑随机游走能够提高精度；
2. 单雷达下的精度比融合后的精度高，则说明观测更加可靠；增大Q，减小R；或者增加Q，R不变；或者Q不变，减小R；这些都代表更相信观测，能提高精度。相反，减小Q，R不变；Q不变，增大R；减小Q，增大R；这些代表更相信预测，则减小精度。

思考和疑问

1. 为什么调不到一组融合后比单雷达下精度高的参数？无论是在kitti数据集还是自采数据集，融合后的精度都是比单雷达要低一些？原因是？