补充代码部分

SetProcessEquation

```
// 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).matr
F_.block<3, 3>(kIndexErrorVel, kIndexErrorAccel) = -C_nb;
F_.block<3, 3>(kIndexErrorOri, kIndexErrorOri) = -Sophus::SO3d::hat(w_b).matrix();
// b. set process equation for delta ori:
B_.block<3, 3>(kIndexErrorVel, kIndexNoiseAccel) = C_nb;
```

UpdateOdomEstimation 基本和第六章的作业一样

```
size_t index_curr = 1;
size_t index_prev = 0;
// get deltas:
Eigen::Vector3d angular_delta = Eigen::Vector3d::Zero();
GetAngularDelta(index_curr, index_prev, angular_delta, angular_vel_mid);
// update orientation:
Eigen::Matrix3d R_curr = Eigen::Matrix3d::Zero();
Eigen::Matrix3d R_prev = Eigen::Matrix3d::Zero();
UpdateOrientation(angular_delta, R_curr, R_prev);

// get velocity delta:
double T;
Eigen::Vector3d velocity_delta = Eigen::Vector3d::Zero();
GetVelocityDelta(index_curr, index_prev, R_curr, R_prev, T, velocity_delta, linear_a

// save mid-value unbiased linear acc for error-state update:

// update position:
UpdatePosition(T, velocity_delta);
```

UpdateErrorEstimation 对应ppt中状态方程离散化,注意其中w_k为零均值，故w_k为0

```
// TODO: update process equation:
UpdateProcessEquation(linear_acc_mid, angular_vel_mid);
// TODO: get discretized process equations:
F_1st = F_ * T;
F_2nd = MatrixF::Identity() + F_1st;

MatrixB B = MatrixB::Zero();
B.block<3, 3>(kIndexErrorVel, kIndexNoiseAccel) = B_.block<3, 3>(kIndexErrorVel, kIr
B.block<3, 3>(kIndexErrorOri, kIndexNoiseGyro) = B_.block<3, 3>(kIndexErrorOri, kInc
B.block<3, 3>(kIndexErrorAccel, kIndexNoiseBiasAccel) = B_.block<3, 3>(kIndexErrorAc
B.block<3, 3>(kIndexErrorGyro, kIndexNoiseBiasGyro) = B_.block<3, 3>(kIndexErrorGyrc
// TODO: perform Kalman prediction
X_ = F_2nd * X_;
P_ = F_2nd * P_ * F_2nd.transpose() + B * Q_ * B.transpose();
```

CorrectErrorEstimationPose 对应观察方程计算。这里直接根据ppt中计算后，初始化Y,计算K

```
Eigen::Vector3d delta_p = pose_.block<3, 1>(0, 3) - T_nb.block<3, 1>(0, 3);
Eigen::Matrix3d delta_R = T_nb.block<3, 3>(0, 0).transpose() * pose_.block<3, 3>(0,
Eigen::Vector3d delta_ori = Sophus::SO3d::vee(delta_R - Eigen::Matrix3d::Identity())

YPose_.block<3, 1>(0, 0) = delta_p;
YPose_.block<3, 1>(3, 0) = delta_ori;

Y = YPose_;
// TODO: set measurement equation:
G = GPose_;

// TODO: set Kalman gain:
K = P_ * G.transpose() * (G * P_ * G.transpose() + CPose_ * RPose_ * CPose_.transpos
```

然后计算P与X

```
P_ = (MatrixP::Identity() - K * G) * P_;
X_ = X_ + K * (Y - G * X_);
```

EliminateError 对应ppt中的更新后验位姿

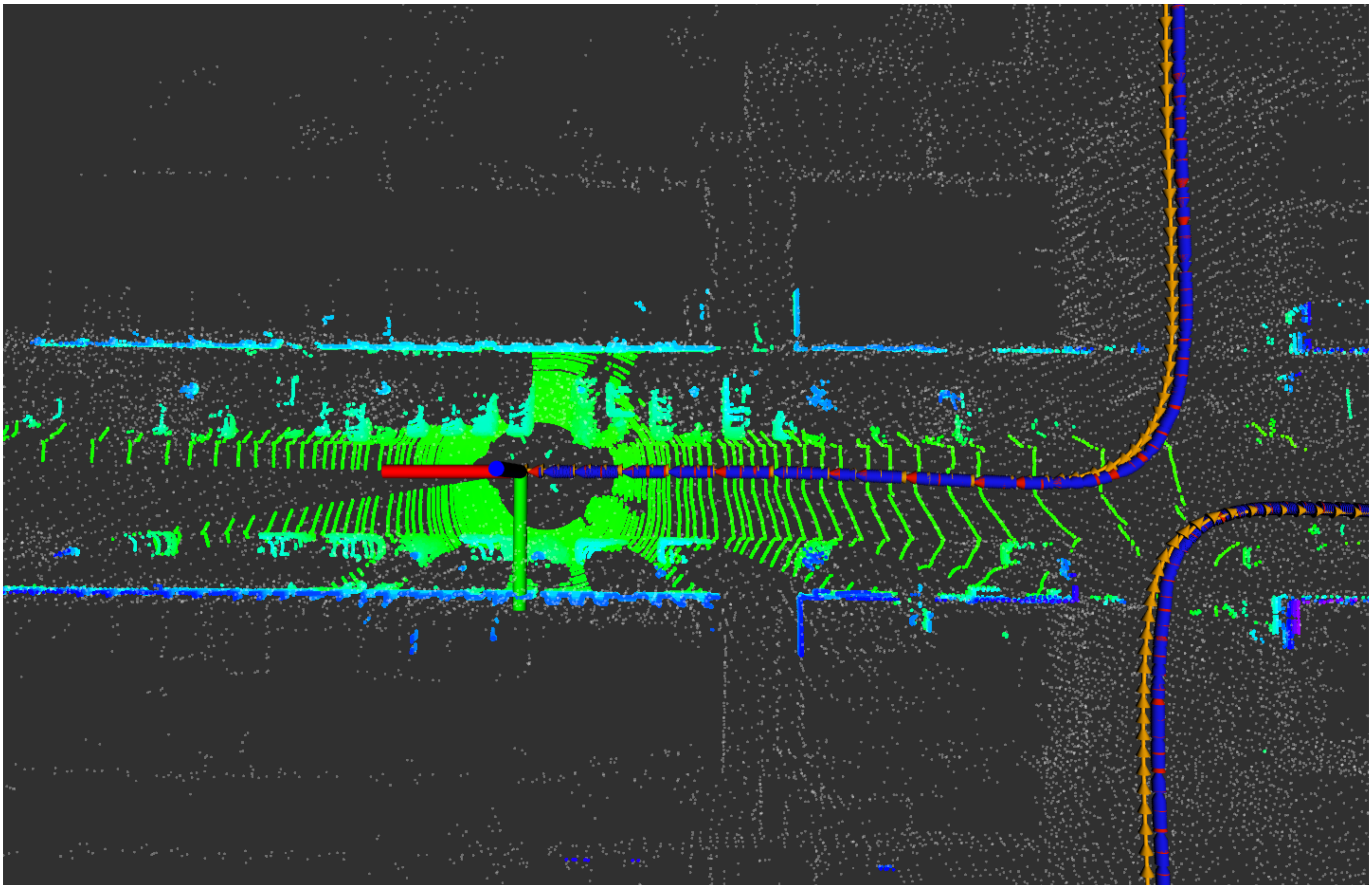
```
    // a. position:
    // do it!
    pose_.block<3, 1>(0, 3) -= X_.block<3, 1>(kIndexErrorPos, 0);
    // b. velocity:
    // do it!
    vel_ -= X_.block<3, 1>(kIndexErrorVel, 0);
    // c. orientation:
    // do it!
    pose_.block<3, 3>(0, 0) = pose_.block<3, 3>(0, 0) * (Eigen::Matrix3d::Identity() - S

    Eigen::Quaterniond q(pose_.block<3, 3>(0, 0));
    q.normalize();
    pose_.block<3, 3>(0, 0) = q.toRotationMatrix();

    // d. gyro bias:
    if (IsCovStable(kIndexErrorGyro)) {
        gyro_bias_ -= X_.block<3, 1>(kIndexErrorGyro, 0);
    }

    // e. accel bias:
    if (IsCovStable(kIndexErrorAccel)) {
        accl_bias_ -= X_.block<3, 1>(kIndexErrorAccel, 0);
    }
}
```

效果如下



不考虑随机游走

定义bias_flag,然后在构造函数做如下修改:

```
// c. process noise:
Q_.block<3, 3>(kIndexNoiseAccel, kIndexNoiseAccel) = COV.PROCESS.ACCEL * Eigen::Matr
Q_.block<3, 3>(kIndexNoiseGyro, kIndexNoiseGyro) = COV.PROCESS.GYRO * Eigen::Matrix3
if (bias_flag) {
    Q_.block<3, 3>(kIndexNoiseBiasAccel, kIndexNoiseBiasAccel) = COV.PROCESS.BIAS_AC
    Q_.block<3, 3>(kIndexNoiseBiasGyro, kIndexNoiseBiasGyro) = COV.PROCESS.BIAS_GYRC
}
// d. measurement noise:
RPose_.block<3, 3>(0, 0) = COV.MEASUREMENT.POSE.POSI * Eigen::Matrix3d::Identity();
RPose_.block<3, 3>(3, 3) = COV.MEASUREMENT.POSE.ORI * Eigen::Matrix3d::Identity();

// e. process equation:
F_.block<3, 3>(kIndexErrorPos, kIndexErrorVel) = Eigen::Matrix3d::Identity();
F_.block<3, 3>(kIndexErrorOri, kIndexErrorGyro) = -Eigen::Matrix3d::Identity();

B_.block<3, 3>(kIndexErrorOri, kIndexNoiseGyro) = Eigen::Matrix3d::Identity();
if (bias_flag) {
    B_.block<3, 3>(kIndexErrorAccel, kIndexNoiseBiasAccel) = Eigen::Matrix3d::Identi
    B_.block<3, 3>(kIndexErrorGyro, kIndexNoiseBiasGyro) = Eigen::Matrix3d::Identity
}
```

对比odometry数据

```
evo_rpe kitti ground_truth.txt fused.txt -r trans_part --delta 100 --plot --plot_mode xy
```

对比rmse.考虑随机游走：2.687764，不考虑随机游走：2.396791。从结果来看不考虑随机游走误差小些，当然也跟kitti数据有关，任老师说过实际中可以都尝试下，那个优用那个

参数调整

```
#保存雷达数据
evo_ape kitti ground_truth.txt laser.txt -r full --plot --plot_mode xy  --save_results .
#保存fuse数据
evo_ape kitti ground_truth.txt fused.txt -r full --plot --plot_mode xy  --save_results .
#数据比对
evo_res *.zip -p
```

1)原始参数结果

```
➜  i_true evo_res *.zip -p

APE w.r.t. full transformation (unit-less)
(not aligned)


               rmse      mean    median       std        min       max       sse
fused.txt  0.294197  0.238266  0.167785  0.172572  0.0188841    1.1044   390.867
laser.txt  0.291739  0.236865  0.170568  0.170313  0.0204833   1.10824   384.365
```

2)可以看到lidar精度与fuse精度差不多，但总体精度较低。考虑增加imu置信度，即减小imu的误差
修改参数如下

```yaml
error_state_kalman_filter:
    earth:
        # gravity can be calculated from https://www.ser
        gravity_magnitude: 9.80943
        # rotation speed, rad/s:
        rotation_speed: 7.292115e-5
        # latitude:
        latitude:   48.9827703173
    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-5 #1.0e-4
            accel: 2.5e-4 #2.5e-3
            bias_accel: 2.5e-4 #2.5e-3
            bias_gyro: 1.0e-5 #1.0e-4
        measurement:
            pose:
                pos: 1.0e-4
                ori: 1.0e-4
            pos: 1.0e-4
            vel: 2.5e-3
    motion_constraint:
        activated: true
        w_b_thresh: 0.13
    bias_flag: true
```

结果如下

	rmse	mean	median	std	min	max	sse
fused.txt	0.345421	0.259673	0.172774	0.227784	0.0247766	2.14991	538.948
laser.txt	0.337891	0.252635	0.172358	0.224378	0.0214211	1.97228	515.706

3)误差反而增大，并且laser精度变差，理论上应该不变，可能是数据问题。尝试减小laser置信度，即增大lidar误差

修改参数如下

```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-5 #1.0e-4
            accel: 2.5e-4 #2.5e-3
            bias_accel: 2.5e-4 #2.5e-3
            bias_gyro: 1.0e-5 #1.0e-4
        measurement:
            pose:
                pos: 1.0e-3 #1.0e-4
                ori: 1.0e-3 #1.0e-4
            pos: 1.0e-4
            vel: 2.5e-3
    motion_constraint:
        activated: true
```

结果如下

APE w.r.t. full transformation (unit-less)
(not aligned)

	rmse	mean	median	std	min	max	sse
fused.txt	0.301861	0.245489	0.173519	0.175656	0.019027	1.05933	411.406
laser.txt	0.291676	0.236798	0.170619	0.170298	0.0204833	1.10824	384.113

4)lidar精度与初始精度相同，fuse精度优于上次修改，但低于初始参数精度。继续增大imu置信度，进行测试

修改参数如下

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

结果如下

```
➜ i_true4 evo_res *.zip -p

APE w.r.t. full transformation (unit-less)
(not aligned)

              rmse      mean    median       std        min       max      sse
fused.txt  0.382928  0.297588  0.224845  0.240988   0.041234  2.24481  662.345
laser.txt   0.33803  0.252781  0.172358  0.224423  0.0204833  1.97228  516.131
```

5)效果不是很理想。还原默认参数，减少P置信度，即增大误差，测试下

修改参数如下

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

结果如下

```
➜ i_true5 evo_res *.zip -p

APE w.r.t. full transformation (unit-less)
(not aligned)

              rmse      mean    median       std        min       max      sse
fused.txt   0.29334  0.237863  0.168349  0.171667  0.0203239  1.10506  389.111
laser.txt  0.291716  0.236849  0.170627  0.170296  0.0204833  1.10824  384.813
```

6)fuse精度比默认环境稍微好一丢丢，尝试再次降低状态量置信度

修改参数如下

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

结果如下

```
→ i_true6 evo_res *.zip -p

APE w.r.t. full transformation (unit-less)
(not aligned)


              rmse      mean    median       std         min       max      sse
fused.txt  0.339823  0.253752  0.170784   0.22603  0.0190589  2.13008   521.16
laser.txt  0.338035  0.252712  0.172255  0.224509  0.0214211  1.97228   515.69
```

精度变差

多次调整，只有第5次稍微好一点。个人觉得调参需要多次修改参数，进行数据验证，最终确认合适的参数