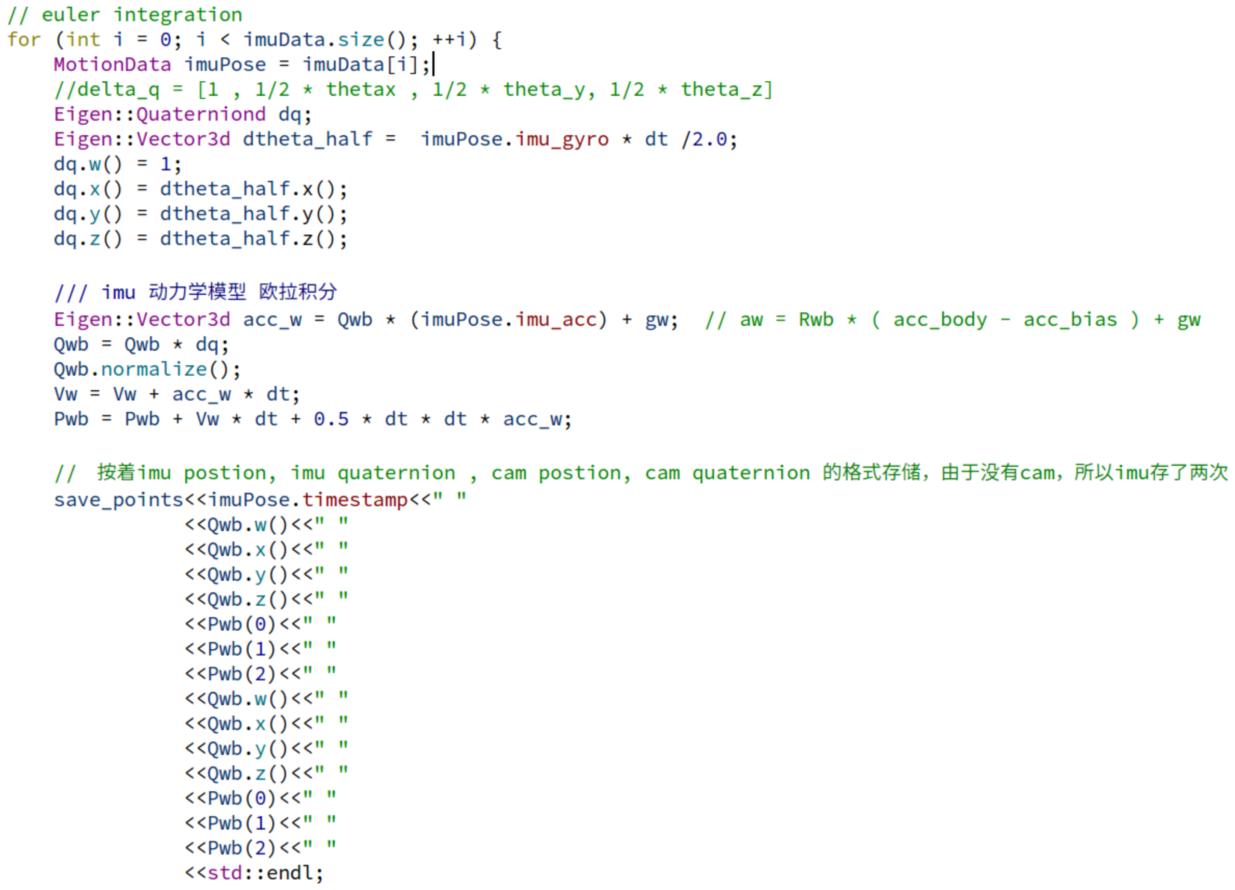
ReadMe

对IMU数据利用动力学模型，采用中值积分得到IMU的轨迹。原来的欧拉积分更新四元数后没有归一化，所以也做了修改，另外积分for循环变量都改成了从0开始。下面分别附上欧拉积分和中值积分的代码：

1.欧拉积分：



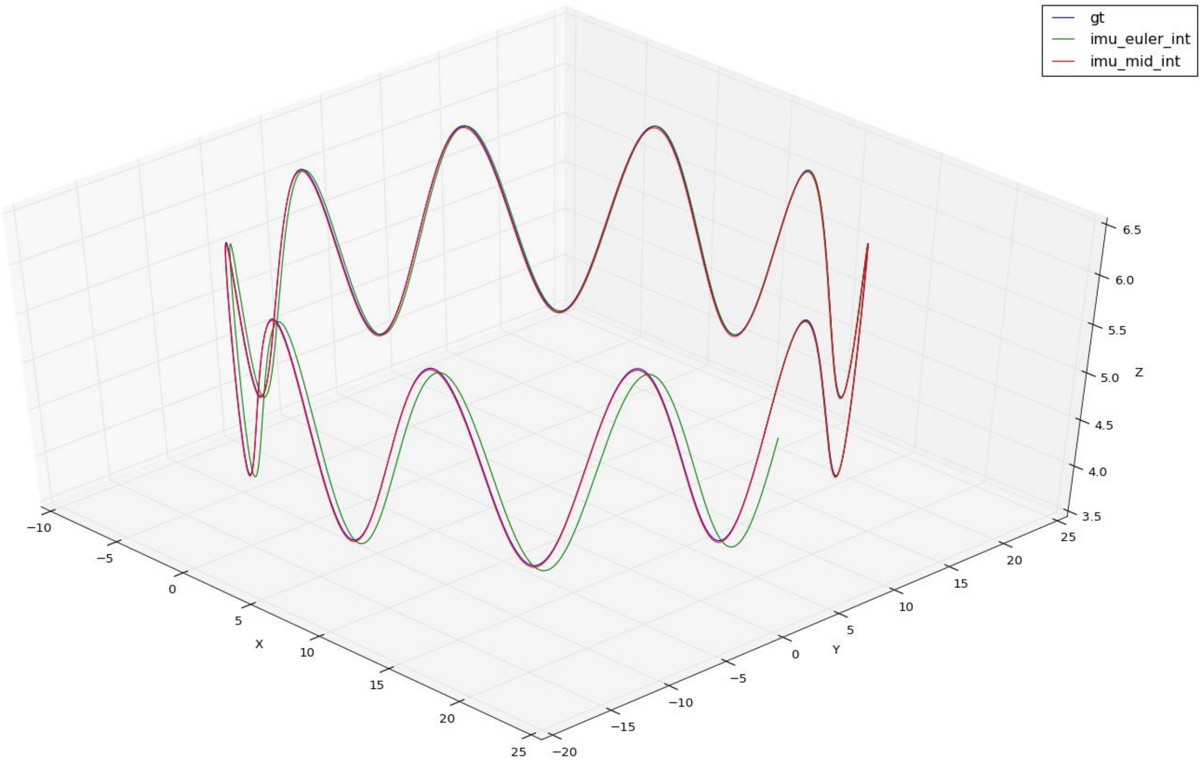
```
// euler integration
for (int i = 0; i < imuData.size(); ++i) {
    MotionData imuPose = imuData[i];
    //delta_q = [1 , 1/2 * thetax , 1/2 * theta_y, 1/2 * theta_z]
    Eigen::Quaterniond dq;
    Eigen::Vector3d dtheta_half =  imuPose.imu_gyro * dt /2.0;
    dq.w() = 1;
    dq.x() = dtheta_half.x();
    dq.y() = dtheta_half.y();
    dq.z() = dtheta_half.z();

    /// imu 动力学模型 欧拉积分
    Eigen::Vector3d acc_w = Qwb * (imuPose.imu_acc) + gw;  // aw = Rwb * ( acc_body - acc_bias ) + gw
    Qwb = Qwb * dq;
    Qwb.normalize();
    Vw = Vw + acc_w * dt;
    Pwb = Pwb + Vw * dt + 0.5 * dt * dt * acc_w;

    //  按着imu postion， imu quaternion , cam postion， cam quaternion 的格式存储，由于没有cam，所以imu存了两次
    save_points<<imuPose.timestamp<<" "
               <<Qwb.w()<<" "
               <<Qwb.x()<<" "
               <<Qwb.y()<<" "
               <<Qwb.z()<<" "
               <<Pwb(0)<<" "
               <<Pwb(1)<<" "
               <<Pwb(2)<<" "
               <<Qwb.w()<<" "
               <<Qwb.x()<<" "
               <<Qwb.y()<<" "
               <<Qwb.z()<<" "
               <<Pwb(0)<<" "
               <<Pwb(1)<<" "
               <<Pwb(2)<<" "
               <<std::endl;
```

2.中值积分：

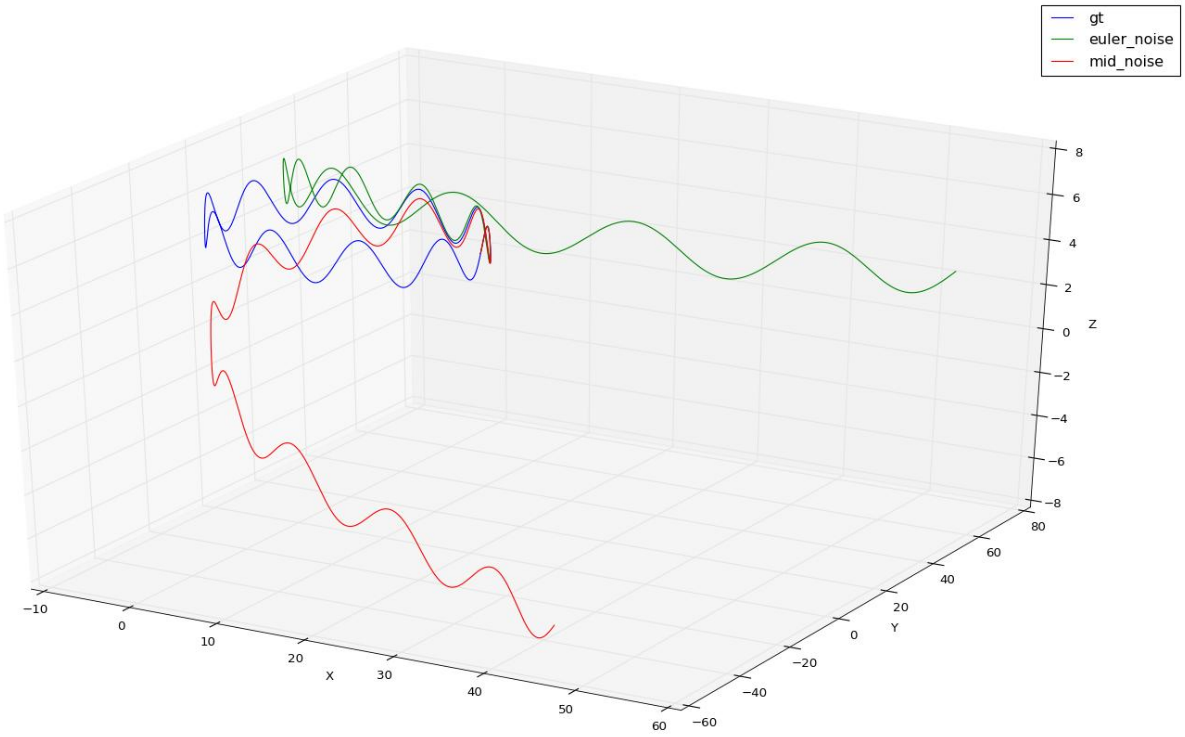
```
// midpoint integration
for (int i = 0; i < imuData.size(); ++i) {

    MotionData imuPose_i = imuData[i];
    MotionData imuPose_j = imuData[i+1];

    Eigen::Quaterniond Qwb_i = Qwb;

    // update orientation first
    Eigen::Vector3d w = 0.5*(imuPose_i.imu_gyro + imuPose_j.imu_gyro);
    Eigen::Quaterniond deltaQ;
    Eigen::Vector3d delta_phi = 0.5 * w * dt;
    deltaQ.w() = 1;
    deltaQ.x() = delta_phi(0);
    deltaQ.y() = delta_phi(1);
    deltaQ.z() = delta_phi(2);

    Eigen::Quaterniond Qwb_j = Qwb_i * deltaQ;
    Qwb_j.normalize();

    // then update velocity and position
    Eigen::Vector3d acc_w = 0.5*(Qwb_i * (imuPose_i.imu_acc) + gw + Qwb_j * (imuPose_j.imu_acc) + gw);

    Vw = Vw + acc_w * dt;
    Pwb = Pwb + Vw * dt + 0.5 * dt * dt * acc_w;

    Qwb = Qwb_j;
```

下面给出欧拉和中值积分的对比曲线：

3.no_noise_euler_midpoint.jpg是欧拉积分和中值积分对于不含噪声的数据积分的轨迹：



4.with_noise_euler_midpoint.jpg是欧拉积分和中值积分对于含噪声的数据积分的轨迹。



综上，从no_noise_euler_midpoint.jpg可以发现，中值积分对于不含噪声的数据积分的轨迹跟ground truth非常接近，误差要明显小于欧拉积分方法。

但是，欧拉积分和中值积分对含噪声的IMU数据积分，误差比较大，有明显的累计漂移，因此需要融合视觉等位置传感器。