

# VIO 第二课作业

靳波  
2019/6/18

## 基础作业

### 1 设置 IMU 仿真代码中的不同的参数，生成 Allen 方差标定曲线

- IMU 仿真代码为 VIO\_data\_simulation-ros\_version

仿真程序运行结果：

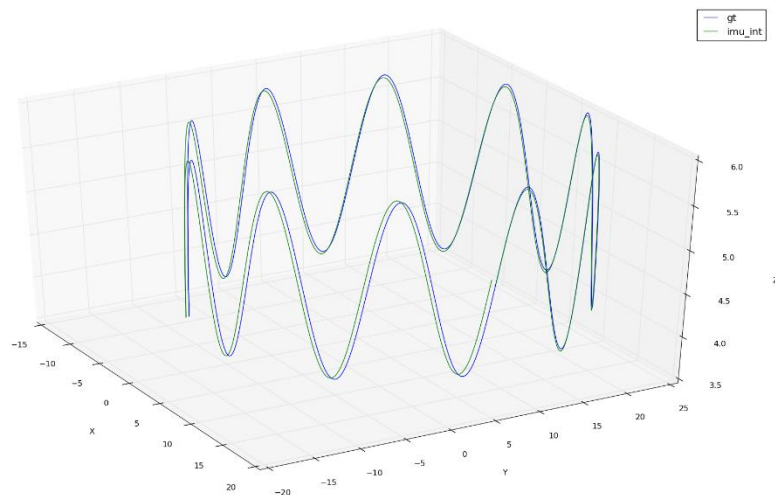


图 1 IMU 仿真曲线

为了得到不同的 Allen 方差曲线，设置两种不同的仿真参数，以得到两个不同的 IMU 数据，参数设置如下：

- 设置仿真参数 1:

加速度高斯白噪声:  $\text{acc\_noise\_sigma}=0.019(\text{m/s}^2)$

陀螺仪的高斯白噪声:  $\text{gyro\_noise\_sigma}=0.015(\text{rad/s})$

加速度 bias 的随机游走噪声:  $\text{acc\_bias\_sigma}=0.0005$

陀螺仪的 bias 随机游走噪声:  $\text{gyro\_bias\_sigma}=0.00005$

- 设置仿真参数 2:

加速度高斯白噪声:  $\text{acc\_noise\_sigma}=0.02(\text{m/s}^2)$

陀螺仪的高斯白噪声:  $\text{gyro\_noise\_sigma}=0.02(\text{rad/s})$

加速度 bias 的随机游走噪声:  $\text{acc\_bias\_sigma}=0.0001$

陀螺仪的 bias 随机游走噪声:  $\text{gyro\_bias\_sigma}=0.00001$

这里生成 allan 方差曲线的工具为 imu\_utils

程序运行结果以及方差曲线图如图 2 至图 7 所示

● 仿真参数 1 下的运行结果：

```

bobo:~/catkin_ws
acc x num of Cluster 100
acc y num of Cluster 100
acc z num of Cluster 100
wait for imu data.
gyr x numData 1064454
gyr x start_t 1.56113e+09
gyr x end_t 1.56114e+09
gyr x dt
-----7204.35 s
-----120.073 min
-----2.00121 h
gyr x freq 147.751
gyr x period 0.00676812
gyr y numData 1064454
gyr y start_t 1.56113e+09
gyr y end_t 1.56114e+09
gyr y dt
-----7204.35 s
-----120.073 min
-----2.00121 h
gyr y freq 147.751
gyr y period 0.00676812
gyr z numData 1064454
gyr z start_t 1.56113e+09
gyr z end_t 1.56114e+09
gyr z dt
-----7204.35 s
-----120.073 min
-----2.00121 h
gyr z freq 147.751
gyr z period 0.00676812
Gyro X
C -1.88235 3623.88 -44.8578 7.62286 -0.10599
Bias Instability 0.000318969 rad/s
Bias Instability 0.000921414 rad/s, at 719.052 s
White Noise 765.701 rad/s
White Noise 0.212711 rad/s
bias -0.180639 degree/s
-----
Gyro y
C -0.691828 3605.98 -18.4343 2.88296 -0.015829
Bias Instability 0.00049272 rad/s
Bias Instability 0.000771115 rad/s, at 1224.2 s
White Noise 765.766 rad/s
White Noise 0.21275 rad/s
bias 0.00790486 degree/s
-----
Gyro z
C 0.865254 3592.54 -10.9217 1.11356 0.00439529
Bias Instability 0.000675746 rad/s
Bias Instability 0.000604063 rad/s, at 3548.45 s
White Noise 748.247 rad/s
White Noise 0.207985 rad/s
bias -0.0556269 degree/s
-----
=====
acc x numData 1064454
bobo:~/catkin_ws
C 0.865254 3592.54 -10.9217 1.11356 0.00439529
Bias Instability 0.000675746 rad/s
Bias Instability 0.000604063 rad/s, at 3548.45 s
White Noise 748.247 rad/s
White Noise 0.207985 rad/s
bias -0.0556269 degree/s
-----
=====
acc x numData 1064454
acc x start_t 1.56113e+09
acc x end_t 1.56114e+09
acc x dt
-----7204.35 s
-----120.073 min
-----2.00121 h
acc x freq 147.751
acc x period 0.00676812
acc y numData 1064454
acc y start_t 1.56113e+09
acc y end_t 1.56114e+09
acc y dt
-----7204.35 s
-----120.073 min
-----2.00121 h
acc y freq 147.751
acc y period 0.00676812
acc z numData 1064454
acc z start_t 1.56113e+09
acc z end_t 1.56114e+09
acc z dt
-----7204.35 s
-----120.073 min
-----2.00121 h
acc z freq 147.751
acc z period 0.00676812
acc X
C -1.77727e-05 0.0222957 -0.000386404 0.00011085 2.86111e-06
Bias Instability 0.00324924 m/s^2
White Noise 0.264904 m/s^2
-----
acc y
C -4.59718e-05 0.0225408 -0.000919927 0.000264675 1.69962e-06
Bias Instability 0.00368449 m/s^2
White Noise 0.268724 m/s^2
-----
acc z
C -5.35544e-05 0.022545 -0.000979309 0.00020729 6.10528e-06
Bias Instability 0.00346861 m/s^2
White Noise 0.267345 m/s^2
-----
[imu_an-1] process has finished cleanly
log file: /home/bobo/.ros/log/97514ebe-9421-11e9-9c81-0071cc46bdef/t
all processes on machine have died, roslaunch will exit
shutting down processing monitor...
... shutting down processing monitor complete
done
bobo@bobo:~/catkin_ws$

```

图 2 仿真参数 1 情况下的噪声估计结果

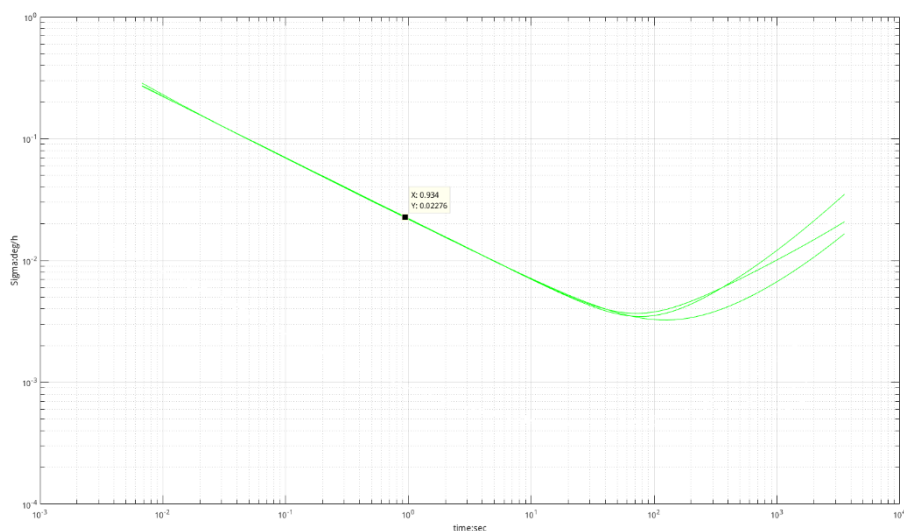


图 3 仿真参数 1 情况下加速度计 Allen 方差曲线

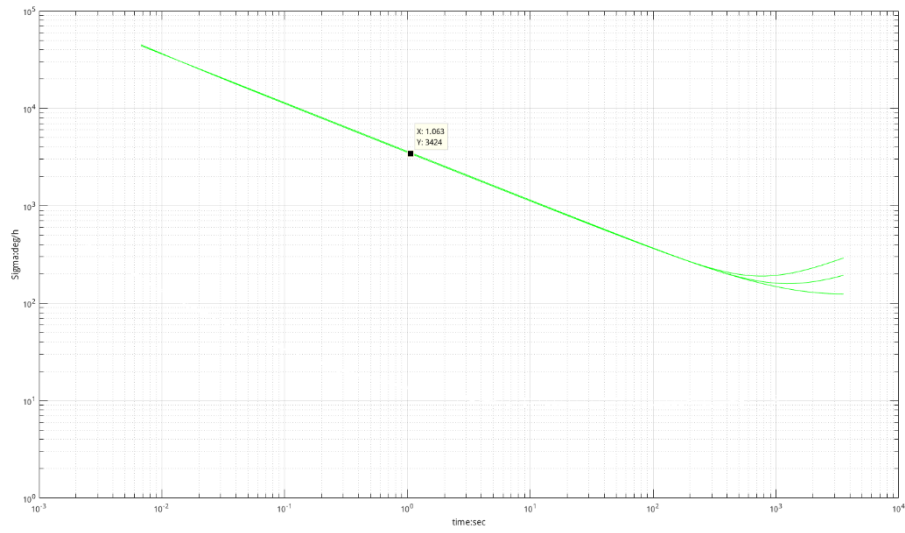


图 4 仿真参数 1 情况下陀螺仪 Allen 方差曲线

- 仿真参数 2 下的运行结果:

```
bobo: ~/catkin_ws/src/imu_utils/launch
acc x num of Cluster 100
acc y num of Cluster 100
acc z num of Cluster 100
wait for imu data.
gyr x numData 955299
gyr x start_t 1.56111e+09
gyr x end_t 1.56111e+09
gyr x dt
-----7205.6 s
-----120.093 min
-----2.00155 h
gyr x freq 132.577
gyr x period 0.00754277
gyr y numData 955299
gyr y start_t 1.56111e+09
gyr y end_t 1.56111e+09
gyr y dt
-----7205.6 s
-----120.093 min
-----2.00155 h
gyr y freq 132.577
gyr y period 0.00754277
gyr z numData 955299
gyr z start_t 1.56111e+09
gyr z end_t 1.56111e+09
gyr z dt
-----7205.6 s
-----120.093 min
-----2.00155 h
gyr z freq 132.577
gyr z period 0.00754277
Gyro X
C 2.02801 3777.39 23.5052 -3.52569 0.244821
Bias Instability 0.000597849 rad/s
Bias Instability 0.00106877 rad/s, at 494.331 s
White Noise 767.452 rad/s
White Noise 0.21323 rad/s
bias -0.14705 degree/s
-----
Gyro y
C 0.47325 3794.28 -3.15999 3.49286 -0.0156823
Bias Instability 0.000839765 rad/s
Bias Instability 0.0010269 rad/s, at 928.274 s
White Noise 765.785 rad/s
White Noise 0.213035 rad/s
bias -0.068478 degree/s
-----
Gyro z
C -0.350795 3799.4 39.3295 -7.5744 0.426947
Bias Instability 0.000613489 rad/s
Bias Instability 0.001171 rad/s, at 435.791 s
White Noise 771.386 rad/s
White Noise 0.214336 rad/s
bias -0.194018 degree/s
-----
=====
acc x numData 955299
C -0.350795 3799.4 39.3295 -7.5744 0.426947
Bias Instability 0.000613489 rad/s
Bias Instability 0.001171 rad/s, at 435.791 s
White Noise 771.386 rad/s
White Noise 0.214336 rad/s
bias -0.194018 degree/s
-----
=====
acc x numData 955299
acc x start_t 1.56111e+09
acc x end_t 1.56111e+09
acc x dt
-----7205.6 s
-----120.093 min
-----2.00155 h
acc x freq 132.577
acc x period 0.00754277
acc y numData 955299
acc y start_t 1.56111e+09
acc y end_t 1.56111e+09
acc y dt
-----7205.6 s
-----120.093 min
-----2.00155 h
acc y freq 132.577
acc y period 0.00754277
acc z numData 955299
acc z start_t 1.56111e+09
acc z end_t 1.56111e+09
acc z dt
-----7205.6 s
-----120.093 min
-----2.00155 h
acc z freq 132.577
acc z period 0.00754277
acc X
C -9.94519e-06 0.0234354 -0.000232431 2.78417e-05 1.26881e-05
Bias Instability 0.00338259 m/s^2
White Noise 0.268285 m/s^2
-----
acc y
C -4.26899e-05 0.023681 -0.000433095 0.000153828 2.27939e-06
Bias Instability 0.00369277 m/s^2
White Noise 0.265252 m/s^2
-----
acc z
C -4.50898e-05 0.0237352 -0.000644177 0.000116038 8.31735e-06
Bias Instability 0.00343527 m/s^2
White Noise 0.268385 m/s^2
-----
[imu_an-1] process has finished cleanly
log file: /home/bobo/.ros/log/97514ebe-9421-11e9-9c81-0071cc46bdef/im
all processes on machine have died, roslaunch will exit
shutting down processing monitor...
... shutting down processing monitor complete
done
bobo@bobo:~/catkin_ws/src/imu_utils/launch$
```

图 5 仿真参数 2 情况下的噪声估计结果

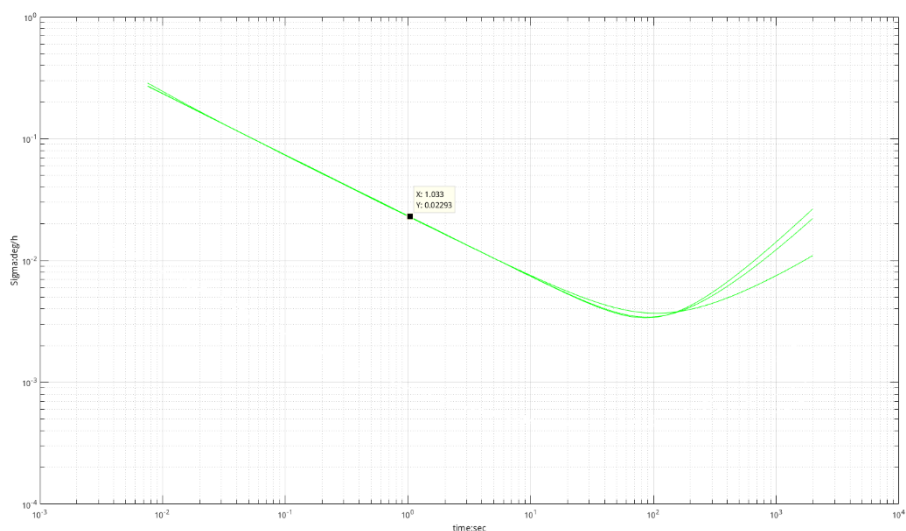


图 6 仿真参数 2 情况下加速度计 Allen 方差曲线

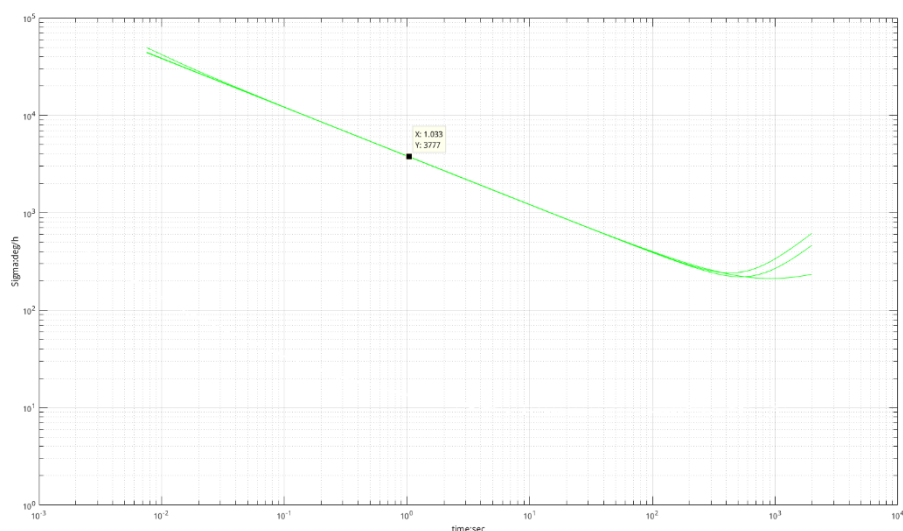


图 7 仿真参数 2 情况下陀螺仪 Allen 方差曲线

分析，在参数 1 下，标定的加速度为 0.02276 这与设定值 0.019 相差不大。标定的陀螺仪为 3424 deg/h 换算单位后为 0.01659 rad/s，这与设定值 0.015 接近。

在参数 2 下，标定的加速度为 0.02293 这与设定值 0.02 相差不大。标定的陀螺仪为 3777 deg/h 换算单位后为 0.0183 rad/s，这与设定值 0.02 接近。

## 2 将 IMU 仿真代码中的欧拉积分替换成中值积分

关键代码如下所示：

```

130 Eigen::Quaterniond Qwb(init_Rwb); // quaterniond: from imu measurements
131 Eigen::Vector3d Vw = init_velocity; // velocity : from imu measurements
132 Eigen::Vector3d gw(0,0,-9.81); // ENU frame
133 Eigen::Vector3d temp_a;
134 Eigen::Vector3d theta;
135 for (int i = 1; i < imudata.size()-1; ++i) {
136     MotionData imupose = imudata[i];
137     MotionData imupose1=imudata[i+1];
138     //delta_q = [1, 1/2 * thetax, 1/2 * theta_y, 1/2 * theta_z]
139     // Eigen::Quaterniond dq;
140     //Eigen::Vector3d dtheta_half = imupose.imu_gyro * dt / 2.0;
141     // dq.w() = 1;
142     //dq.x() = dtheta_half.x();
143     //dq.y() = dtheta_half.y();
144     //dq.z() = dtheta_half.z();
145     // imu 动力学模型 欧拉积分
146     //Eigen::Vector3d acc_w = Qwb * (imupose.imu_acc) + gw; // aw = Rwb * ( acc_body - acc_bias ) + gw
147     //Qwb = Qwb * dq;
148     //Vw = Vw + acc_w * dt;
149     //Pwb = Pwb + Vw * dt + 0.5 * dt * dt * acc_w;
150
151     /// 中值积分
152     Eigen::Quaterniond dq;
153     Eigen::Vector3d dtheta_half = (imupose.imu_gyro+imupose1.imu_gyro) * dt / 4.0;
154     dq.w() = 1;
155     dq.x() = dtheta_half.x();
156     dq.y() = dtheta_half.y();
157     dq.z() = dtheta_half.z();
158
159     Eigen::Quaterniond Qwb1;
160     Qwb1=Qwb*dq;
161     Eigen::Vector3d acc_w = (Qwb * (imupose.imu_acc)+Qwb1*(imupose1.imu_acc))/2.0 + gw;
162
163     Qwb=Qwb*dq;
164     Vw = Vw + acc_w * dt;
165     Pwb = Pwb + Vw * dt + 0.5 * dt * dt * acc_w;
166

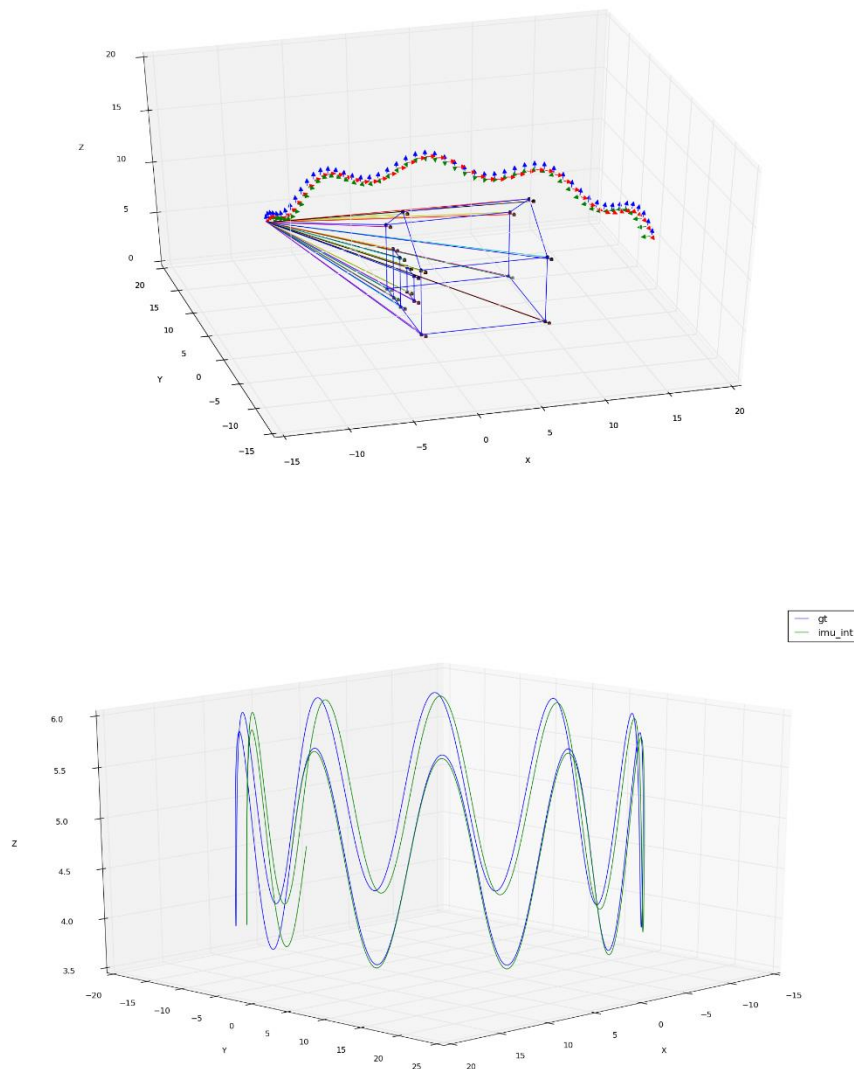
```

这里计算 dq 是使用 imu 陀螺仪在第 i 和第 i+1 时刻的中值  
 计算 acc\_wy 使用的是加速度计在第 i 和第 i+1 时刻的中值  
 这里严格按照公式。

这里 for 循环重第 1 帧开始，我试过从第零帧开始，二者差别不大。

代码见附件，由于只改动了 imu.cpp 文件，因此附件中只放了 imu.cpp 文件，而没有放整个工程。

运行结果：



这里由于没有对仿真得到的轨迹和真实轨迹做出误差曲线，所以不能明显看出中值法与欧拉法之间的差异。但是从理论上来说中值法更为精确，但中值法的计算相比欧拉法要更复杂。

## 提升作业

阅读从已有轨迹生成 imu 数据的论文，撰写总结推导

- 2013 年 BMVC, Steven Lovegrove ,Spline Fusion: A continuous-time representation for



visual-inertial fusion with application to rolling shutter cameras.

这篇文章介绍了一种通用的用于视觉惯导 SLAM 标定的时间一致性框架。我们将演示如何使用与传感器的扭矩最小运动紧密匹配的样条参数化。与传统的离散时间解相比，连续时间公式对于解决高帧速率传感器和多个非同步器件的问题特别有用，我们通过精确确定相对姿态和多个非同步设备的内参，论证了该方法在多传感器视觉 SLAM 和校准中的适用性。我们还通过评估和统一处理视觉和视觉惯性 SLAM 系统中的全局和滚动快门相机，展示了该方法的优势。

公式推导

$$p(t) = \sum_{i=0}^n p_i B_{i,k}(t) \quad \begin{matrix} p_i \text{ 是 } t_i \text{ 时刻的控制点 } i \in [0, n] \\ B_{i,k}(t) \text{ 是基函数} \end{matrix} \quad (1)$$

$$p(t) = p_0 \tilde{B}_{0,k}(t) + \sum_{i=1}^n (p_i - p_{i-1}) \tilde{B}_{i,k}(t) \quad (2)$$

这里将 (1) 式分开:  $p(t) = \sum_{i=0}^n p_i B_{i,k}(t) = p_0 B_{0,k}(t) + \sum_{i=1}^n p_i B_{i,k}(t)$

$$= p_0 \sum_{i=0}^n B_{i,k}(t) + \sum_{i=1}^n (p_i - p_{i-1}) \left( \sum_{j=1}^n B_{j,k}(t) \right)$$

$$= p_0 \tilde{B}_{0,k}(t) + \sum_{i=1}^n (p_i - p_{i-1}) \tilde{B}_{i,k}(t)$$

将 (2) 映射到 SE3 可得:

$$T_{wis}(t) = \exp(\tilde{B}_{0,k}(t) \log(T_{w,0})) \prod_{i=1}^n \exp(\tilde{B}_{i,k}(t) \log(T_{w,i-1}^{-1} T_{w,i}))$$

$$= \exp(\tilde{B}_{0,k}(t) \log(T_{w,0})) \prod_{i=1}^n \exp(\tilde{B}_{i,k}(t) \Omega_i) \quad \begin{matrix} \text{记 } \Omega_i = \log(T_{w,i-1}^{-1} T_{w,i}) \end{matrix} \quad (3)$$

当  $k=4$  时,  $T_{wis}(t) = T_{w,i-1} \prod_{j=1}^3 \exp(\tilde{B}_{i,j}(t) \Omega_{i+j}) \quad (4)$

这里求导公式不再赘述

将(2)代入到(3)可得:

$$T_{w,s}(u) = \exp(\tilde{B}_{w,s}(u) \log(T_{w,s,0})) \prod_{i=1}^n \exp(\tilde{B}_{w,s}(u) \log(T_{w,s,i-1}^{-1} T_{w,s,i}))$$

$$= \exp(\tilde{B}_{w,s}(u) \log(T_{w,s,0})) \prod_{i=1}^n \exp(\tilde{B}_{w,s}(u) \Omega_i) \quad \text{其中 } \Omega_i = \log(T_{w,s,i-1}^{-1} T_{w,s,i}) \quad (3)$$

当  $k=4$  时,  $T_{w,s}(u) = T_{w,s,0} \prod_{j=1}^3 \exp(\tilde{B}_{w,s}(u) \Omega_j) \quad (4)$

这里求导就不再难了.

$\dot{T}_{w,s}(u), \ddot{T}_{w,s}(u)$  均如论文求法.

☆: 仿真的  $Gyro(u)$  和  $Accel(u)$  得到方法:

$$p_b = W(p_a; T_{b,a}, p) = \pi([k_b Z] T_{b,a} [k_a^T [p]; p]) \quad (5)$$

$p$  为逆深度. 相机  $a$  在图像坐标系中为:  $p_a, \quad \pi(p) = \frac{1}{p} Z p_0, p_1^T$  单位矩阵求导数  $k_a, k_b \in \mathbb{R}^{3 \times 3}$

$$Gyro(u) = R_{w,s}^T(u) \cdot \dot{R}_{w,s}(u) + bias \quad (6)$$

$$Accel(u) = p_w^T(u) \cdot (\ddot{s}_w(u) + g_w) + bias \quad (7)$$

$\dot{R}_{w,s}$  来自  $\dot{T}_{w,s}$   
 $\ddot{s}_w$  来自  $\ddot{T}_{w,s}$

这个很好理解.  $R_{w,s}(u)$  表旋转, 而  $Gyro$  的逆为角速度. 逆旋转求角速度  
 $\dot{s}_w$  表线速度,  $T_{w,s}$  为速度,  $\ddot{T}_{w,s}$  为加速度.

面的公式其实就相当在线性优化. 通过构建目标函数, 使其最小, 来求最优解.

$$E(u) = \sum_{p_m} (\hat{p}_m - u) (p_r; T_{c,s} T_{w,s}(u_m)^T T_{w,s}(u_r) T_{s,c} p)^2_{\epsilon_p} \rightarrow \text{相机误差}$$

$$+ \sum_{\hat{w}_m} (\hat{w}_m - Gyro(u_m))^2_{\epsilon_w} + \sum_{\hat{a}_m} (\hat{a}_m - Accel(u_m))^2_{\epsilon_a}$$

陀螺仪误差  
 加速度计误差

总结: 本文基于摄像机轨迹的连续时间模型, 对卷帘门序列的滑动窗视觉里程计进行了仿真验证. 在这种模式下, 我们能够仅从一台相机估算出准确的相机轨迹和场景结构, 并证明忽略相机的卷帘会导致糟糕的结果. 本文还展示了摄像机内部和 IMU 的视觉惯性校准, 从真实数据到摄像机外部, 给出了精确的对应关系. 虽然这种校准是用卷帘照相机显示的, 但是方法是灵活的, 并且自然支持比以前的方法更广泛的传感器范围. 通过在一个大窗口内对传感器数据进行联合融合, 我们减少了卡尔曼滤波器内线性化可能引起的偏差.



助教辛苦了  
祝好