注意：先认真用理论分析，然后设计实验及保存的数据。反复调整实验方法。

1. TUM数据集handled\_slam-rgbd\_dataset\_freiburg2\_large\_with\_loop轨迹每次跑的结果不一样的问题
2. 数据集信息：5182张图片，图片名即拍摄时间，单位为s；1311875745.126861-1311875572.406161=172.7207s，两帧间隔大约0.033s；
3. 猜想可能原因
4. 自己改动过程序

解决：重新下载源程序，跑的结果和自己注释过代码差不多。

1. Ransac方法不同造成
2. 特征点每次提取不一样
3. 和笔记本散热是否有关，笔记本放瓷砖上似乎表现好
4. 设计验证：
5. 查找程序中会出现跟踪丢失的地方，注释//yilu lost。编写小程序打印重要信息，重点关注658时丢失情况。
6. 编写python小程序读取每条轨迹保存的帧序号，并统计不同帧出现的次数，观察次数多的帧和次数少的帧在实验中的表现，画出曲线，观察哪里断了。
7. 统计并分析每次开始丢失的帧
8. 实验结果分析
9. 原程序和注释程序分别得到4/11，2/10次好结果；关键帧最多结果保留210张图片：579-658=79s，658-734=76（丢失），734-744=10s（loop）

跟踪时间内约5182\*89/172=2681张图，得到210/2681=7.83%的关键帧，关键帧间隔89/210=0.424s。

因为中间丢失一段，故最后是重定位，并非闭环

* 其它好结果情况：

nomodified：跟踪 丢失 重定位

3（203）：579-658，658-737，737-745

6（202）：580-658，658-738，738-744

10（203）584-663，663-739，739-745

modified：

4（209）：573-658，658-733，733-745

6（205）：580-658，658-737，737-744

* 其它结果（只给出关键帧段，两帧间隔超过5s算丢失）：

nomodified：

1（50）：579-592，734-744

2（9）：572-573

4（33）：574-581，735-744

5（59）：573-593，736-744

7（74）：581-611，738-744

8（166justnoloop）：594-659（594初始化太晚了，没能重定位）

9（80）：581-614，738-745

11（141在中间丢失一次重定位）：595-630，640-659（重定位，初始化太晚第二个重定位没看到）

modified：

1（93）：573-603，733-744

2（194）：583-658，739-745 接近好情况

3（40）：581-592，734-744

5（83）：579-612，734-745

7（94）：573-609，734-744

8（127）：573-625，733-745

9（36）：582-592，737-744

重数据来看：好的结果是在658处发生丢失，其它丢失见数据，2次初始化太晚，到了594，因而不能在最后重定位。

10（196）：582-658，738-744 接近好情况

1. 1
2. 关键方法
3. 通过4-1）数据分析，找出了主要原因就是丢失，然后分析观察丢失发生的图片，通过代码分析可能原因
4. TUM中robot\_slam初始化问题
5. 问题：4条轨迹初始化总是失败
6. 实验方法：在初始化失败的地方将数据打印出来
7. 解决：通过打印数据发现，即使场景是3D的，但总是使用单应矩阵H来重建，故打印阈值RH观察，发现范围在0.41-0.48，主要在0.44-0.46，最后将阈值从0.4设置为0.46，能够初始化。
8. 实验总结（实验4条轨迹）

* 常在H、F重建环节失败
* F重建成功常在0.43-0.45，阈值可再提高到0.47左右，重建失败原因常是not a clear winner，且相似数常是2。
* H失败原因常是第一好的没有明显优于第二好的和视差不足（三角化2射线夹角小于1°）。
* 小车在转弯时特别容易跟踪丢失

1. 疑问

* 转弯时跟踪丢失的原因可能时转得比较快没有足够时间新建新的3D点，试设计实验验证猜想：设定程序给定相同运动曲线控制指令，速度不一样。r=v/w。
* 初始化时的2帧间隔多帧会不会更好，基线更长

1. 测试树莓派和笔记本连在同一个WiFi上图片实时性和是否存在掉帧的情况

使用rosbag分别在笔记本和树莓派记录数据，用于对比是否掉帧，在笔记本使用image\_view观察实时性。

1. 实验方法：将摄像头装在树莓派上，笔记本和树莓派连接同一个路由器通过ros通信，分别在树莓派和笔记本上实验rosbag记录/image\_raw话题
2. 实验结果：结果存在~/experiment/LAN\_speed,传输图片画面流畅，肉眼看不出明显延迟，但树莓派rosbag记录的文件2G，而笔记本记录数据372.5MB，重放时在转弯时会明显模糊。最后将图片用rosbag记录下来，对比两个数据，掉帧3-4帧（如1-5中间掉3帧），真实帧速0.05-0.06s，实际0.15-0.26s。
3. 测试更改阈值RH为0.46后，小车能否正常初始化，以及跑一个闭环试试，
4. 实验方法：将摄像头装在树莓派上，笔记本和树莓派连接同一个路由器通过ros通信，在笔记本上跑ORBSLAM2源码，帧率较低，8HZ左右，一直在房间跑。
5. 实验结果：最后轨迹存在

~/experiment/home/trajectory/bedroom\_about\_2\_hours

* 可以初始化
* 转弯容易丢失，尤其是在较窄的直角的位置，如床角
* 随着时间增加，在房间重建的关键帧很多（820帧），一直在增加，跑了2小时左右，内存从2G左右达到4.2G左右

1. 实验总结：转弯时，若路标点比较远没那么容易丢失，较远的路标点更容易跟踪；若速度太快时，等不及localmapping线程三角化更多的路标点
2. 在房间和测试速度多大是ORB\_SLAM会失败：将bag记录的图片加快播放速度
3. 相机对着家里窗不动时初始化问题：在源码加代码记录初始化两帧及其对应未校正的特征点位置，然后自己读取和显示这两帧图片观察，记录图片数据，方便重复实验。
4. 静止是打印IMU数据观察偏差
5. 实验时要注意像VINS图10那样打印数据出来观察，然后选择合适的解决方法
6. 单独用相机和IMU进行测量，以相机作为标准，观察imu漂移与时间（更新频率）的关系，同理可测量相机的漂移，使用laser作为参考
7. 画一个典型的角点，通过改变摄像头的角度、远近看检测结果如何，先仔细用理论分析会带来什么结果，
8. 对共视图进行可视化，认真分析错匹配等情况。
9. vins实时性和主要部分时间测试，MH04-20h-1min44s，实验数据见experiment/vins/runningdata

* rviz、vins、loop全打开：初始化时间244-282ms左右
* vins

1. 图像特征提取与跟踪线程：期望处理频率20HZ（50ms），相机帧率。初始化完成时间点后，稳定在30-40ms，主要时间在跟踪，其他时间1ms左右，跟踪过程中，如39.76ms-19.05光流跟踪，0.94ms设置mask，15.65探测新的角点，32.73ms-13.78光流跟踪，0.64ms设置mask，15.14探测新的角点，
2. 处理imu和image线程： 期望处理频率10HZ（100ms），相机帧率一半
3. 初始化前1-5ms
4. 初始化后平均时间80ms左右，50-140ms波动比较大，处理imu数据1ms左右，1ms左右处理其他，剩下时间主要在处理图像，处理图像主要时间在优化，其他就1ms左右，优化中预边缘-7ms左右，边缘化-30-35ms。
5. 处理的基本都是关键帧，非关键帧处理时间比关键帧少20-30ms，最后快停下时，连续的非关键帧，处理总时间降到30ms左右

* Loop

1. 添加KF线程：30-270ms，波动较大，中间有很多退出条件，随运行时间变长而变长，关键帧库不断变大
2. 4DOF优化：8-24-127-158-255-196ms，随运行时间增长时间变长，

* 只打开vins时

1. 处理imu和image时间：平均时间55ms左右，40-70ms波动比较小，主要集中在55-66ms

* leica-20hz速度计算：525-535：0.468998138/时间=0.782968539m/s，505-515：0.584261813/时间=1m/s
* 总结

1. 都是关键帧的原因是无人机运动速度比较快，相隔两帧10hz运动距离较大
2. 话题的回调函数给输入图像提供了缓冲。将输入图像和处理图像（后端优化）作为两个线程，输入图像特征提取和跟踪处理线程稳定，可防止丢帧，也给后端优化处理时间的不稳定提供了一定的缓冲的时间。
3. 关闭loop、rviz线程处理速度提高了20-30ms，且波动较小，因而在时间充足的情况下，没必要开太多线程
4. 1