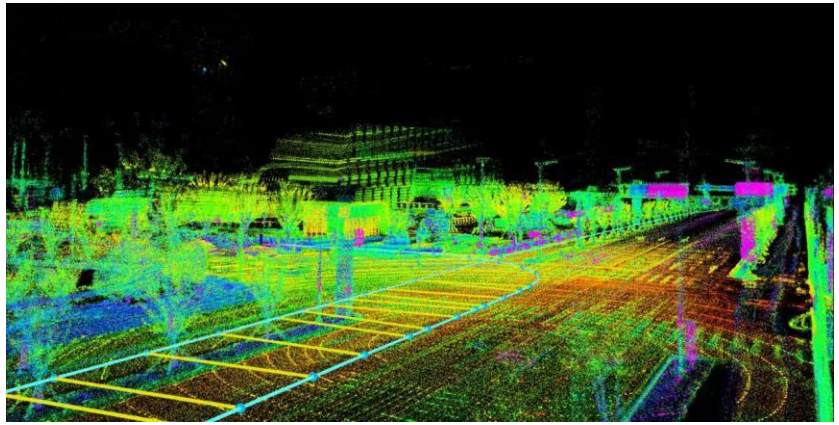
微信扫码加入星球

自动驾驶中实战课之Lidar与IMU的同步实战

Camera + LiDAR + Radar + IMU



主　讲　人：帅的丑小鸭

公　众　号：3D 视 觉 工 坊

内容

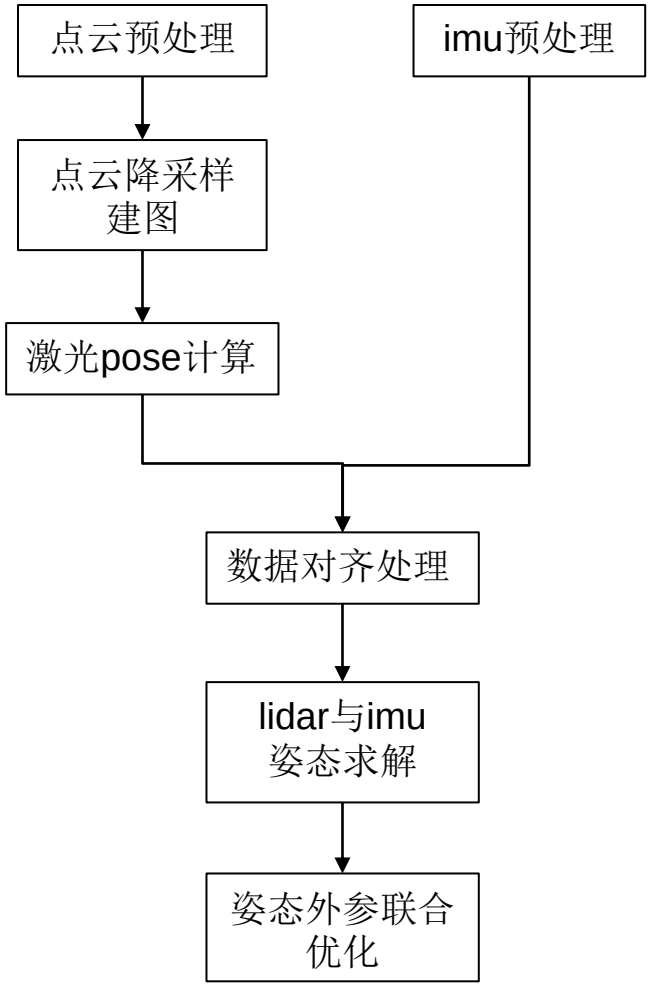
一、 Lidar与IMU的同步方法

二、 代码讲解

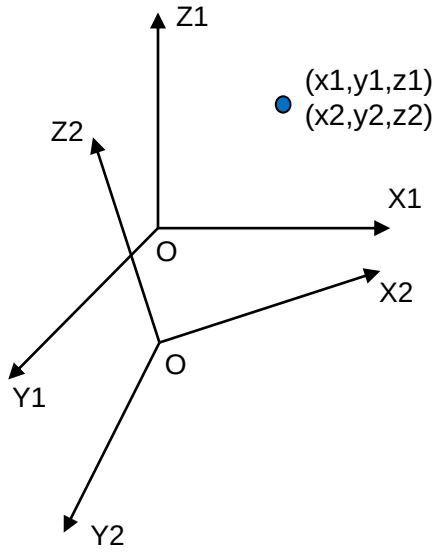
激光与IMU之间的时空同步：

- IMU一般与GPS相结合
- 通过GPS授时完成激光与IMU的时间同步
- 基于状态估计的时空同步方案
- 激光SLAM方法

工程改进流程图



坐标系示意图



给出上图两个坐标系之间的关系：

假设$(OX1Y1Z1)$为IMU坐标系，$(OX2Y2Z2)$ 为激光雷达坐标系，

假设某点在两个坐标系之间的坐标分别为(x1,y1,z1), (x2,y2,z2)，

两者之间的坐标转换关系为4x4的包含旋转矩阵R和平移矩阵T组成的矩阵，记为T_{3D}。即：

$$(x1,y1,z1)^T = T_{3D}(x2,y2,z2)^T$$

这里三维坐标转换矩阵T_{3D}可以用下式表示：

$$T_{3D}=\left[\begin{array}{ccc|c} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ \hline t_{41} & t_{42} & t_{43} & t_{44} \end{array}\right]=\left[\begin{array}{c|c} \boldsymbol{R} & 0 \\ \hline \boldsymbol{T} & 1 \end{array}\right]$$

其中R对应的是比例、旋转、错切等几何变换，

T为平移矩阵，对应$[t_{41}, t_{42}, t_{43}]$，

$[t_{14}, t_{24}, t_{34}]$为对应投影变换，

$[t_{44}]$反映的是整体比例的变换。

由于lidar和IMU之间连接是刚性的，所以有：

$$[t_{14}, t_{24}, t_{34}]^{\mathrm{T}} = [0\ 0\ 0]^T$$

$$[t_{44}] = 1$$

激光坐标系相对IMU坐标系欧拉角为俯仰角θ，翻滚角γ，方位角φ；

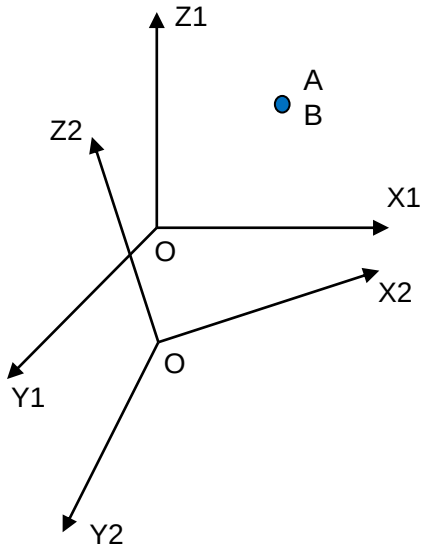
相对于轴向的平移量为t_x、t_y、t_z，所以有：

$$R = \begin{bmatrix} c\gamma c\varphi + s\gamma s\theta s\varphi & -c\gamma s\varphi + s\gamma s\theta c\varphi & -s\gamma c\theta \\ c\theta s\varphi & -c\theta c\varphi & s\theta \\ s\gamma c\varphi - c\gamma s\theta s\varphi & -s\gamma s\varphi - c\gamma s\theta c\varphi & c\gamma c\theta \end{bmatrix}^T$$

其中，c表示cos，s表示sin。

根据旋转矩阵R，求解欧拉角为

$$\begin{cases} \theta = \arcsin(R^T(23)) \\ \gamma = \arctan(-\frac{R^T(13)}{R^T(33)}) \\ \varphi = arctan(-\frac{R^T(21)}{R^T(22)}) \end{cases}$$



假设P点在IMU坐标系下的坐标为A，在激光坐标系下的坐标为B，则两者之间的关系可以表示为：

$$A = T_{3D}B$$

使用最小二乘法进行系统坐标标定，求解的坐标转换公式如下：

$$T_{3D} = \mathrm{AB}^{+}$$

其中B^{+}为B的广义逆矩阵，定义为：

$$\mathrm{B}^{+} = \mathrm{B}^{\mathrm{T}}(\mathrm{BB}^{T})^{T}$$

由此方法能够得到使三维坐标误差最小的坐标系转换矩阵。

连续时间状态法

采用B样条曲线进行拟合，比如使用d自由度的曲线拟合，对于 $p(t)(t \in [t_i, t_j)$ ，给定 $t_i, t_{i+1}, \cdots, t_{i+d}$ 时刻对应的状态 $p_i, p_{i+1}, \cdots, p_{i+d}$ ，

则有：

$$p(t) = \sum_{j=0}^{d} u^T M_j^{d+1} p_{i+j}$$

其中，$u^T = [1 u \cdots u^d]$，$u = (t - t_i)/(t_{i+1} - t_i)$ 是多项式系数。

如何基于连续状态法来标定外参?

标定初始旋转外参

首先基于纯激光SLAM方法得到每一帧的姿态，然后使用三次B样条曲线拟合得到R(t)，这样就可以利用IMU测量值进一步修正R(t),如下：

$$q_0, \cdots, q_N = \arg\min \sum_{k=0}^{M} ||\omega_m^{I_k} - {}_L^{L_0}R^T(t_k) {}_L^{L_0}\dot{R}(t_k)||$$

上式表示拟合出的激光转向姿态求解出的对应IMU时刻下角速度与IMU测量值的残差，最小化残差优化R(t)。

基于R(t)可以得到每两帧IMU时刻之间 $[t_k, t_{k+1}]$ 对应的激光雷达相对旋转姿态 ${}^{L_k}_{L_{k+1}}q$

然后同手眼标定法一样求解出对应IMU的相对旋转姿态 ${}^{I_k}_{I_{k+1}}q$ ，那么满足如下手眼标定关系：

$${}^{I_k}_{I_{k+1}}q \, {}^{I}_{L}q = {}^{I}_{L}q \, {}^{L_k}_{L_{k+1}}q$$

将所有这样时刻的关系联立即可得到如下超定方程：

$$\begin{bmatrix} \vdots \\ \alpha_k([{}^{I_k}_{I_{k+1}}q]_L - [{}^{L_k}_{L_{k+1}}q]_R) \\ \vdots \end{bmatrix} {}^{I}_{L}q = Q_N \, {}^{I}_{L}q = 0$$

可以通过SVD等方法即可得到初始外参 。

环境和代码配置说明

1.配置内容均是在docker 下进行，在docker下/home/下：

```
root@6dc69d3505b5:/home# ls
3rdparty  workspace
root@6dc69d3505b5:/home#
```

```
root@6dc69d3505b5:/home/workspace# ls
ad_sensor_fusion  data
root@6dc69d3505b5:/home/workspace#
```

2. 重新pull代码，更新到最新版本，查看log，确认更新了lidar_imu_sync和ndt_omp

lidar_imu_sync	update lidar_imu_sync and ndt_omp
ndt_omp	update lidar_imu_sync and ndt_omp

这里的ndt_omp是一个pcl ndt的一个多线程优化库

3. 数据集

```
path:        test.bag
version:     2.0
duration:    24.5s
start:       Aug 03 2021 16:45:05.47 (1627980305.47)
end:         Aug 03 2021 16:45:29.00 (1627980330.00)
size:        3.4 GB
messages:    10553
compression: none [737/737 chunks]
types:       geometry_msgs/TwistStamped [98d34b0043a2093cf9d9345ab6eef12e]
             nav_msgs/Odometry          [cd5e73d190d741a2f92e81eda573aca7]
             sensor_msgs/Image          [060021388200f6f0f447d0fcd9c64743]
             sensor_msgs/Imu            [6a62c6daae103f4ff57a132d6f95cec2]
             sensor_msgs/NavSatFix      [2d3a8cd499b9b4a0249fb98fd05cfa48]
             sensor_msgs/PointCloud2    [1158d486dd51d683ce2f1be655c3c181]
topics:      /cgi610/imu          2454 msgs    : sensor_msgs/Imu
             /cgi610/nav_fix      2454 msgs    : sensor_msgs/NavSatFix
             /cgi610/twist        2454 msgs    : geometry_msgs/TwistStamped
             /localization/odom   2454 msgs    : nav_msgs/Odometry
             /raw_image            491 msgs    : sensor_msgs/Image
             /velodyne_points      246 msgs    : sensor_msgs/PointCloud2
```

```
lidar_imu_sync/
├── CMakeLists.txt
├── include
│   └── lidar_imu_sync
│       ├── lidar_imu_calib.hpp
│       └── lidar_imu_sync.hpp
├── package.xml
└── src
    ├── lidar_imu_calib.cpp
    ├── lidar_imu_sync.cpp
    └── main.cpp
```

main.cpp：
程序主函数
lidar_imu_sync.cpp：
同步模块，主要是将rosbag中的数据进行解析和处理，并送入到待标定模块队列中
lidar_imu_calib.cpp：
标定模块，主要将标定数据队列中的数据进行标定处理，主要采用状态估计的方式进行同步

```cpp
struct LidarData
{
    double stamp;
    CloudT::Ptr cloud;
};

struct LidarFrame
{
    double stamp;
    Eigen::Matrix4d T;
    Eigen::Matrix4d gT;
    CloudT::Ptr cloud{nullptr};

    EIGEN_MAKE_ALIGNED_OPERATOR_NEW
};

struct ImuData
{
    double stamp;
    Eigen::Vector3d acc;
    Eigen::Vector3d gyr;
    Eigen::Quaterniond rot;

    EIGEN_MAKE_ALIGNED_OPERATOR_NEW
};
```

具体代码讲解和结果展示

购买该课程请扫描二维码

微信扫码加入星球

感谢聆听

Thanks for Listening