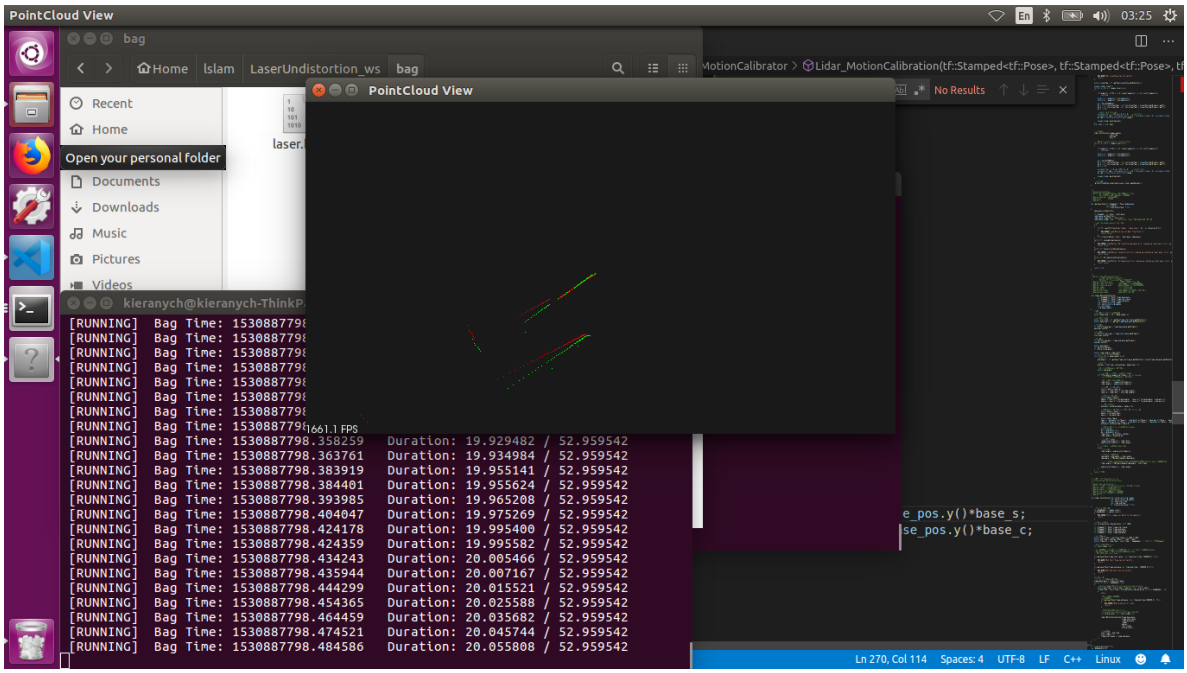
第三章作业

一、运动畸变模块

```
void Lidar_MotionCalibration(
            tf::Stamped<tf::Pose> frame_base_pose,
            tf::Stamped<tf::Pose> frame_start_pose,
            tf::Stamped<tf::Pose> frame_end_pose,
            std::vector<double>& ranges,
            std::vector<double>& angles,
            int startIndex,
            int& beam_number)
    {
        //TODO
         //每个位姿进行线性插值时的步长
         double beam_step = 1.0 / (beam_number-1);
         //起始和基准角度
         double start_Yaw = tf::getYaw(frame_start_pose.getRotation());
         double base_Yaw = tf::getYaw(frame_base_pose.getRotation());
         //基础位姿
         tf::Vector3 base_pos = frame_base_pose.getOrigin();
         base_pos.setZ(0);
         //起始位姿
         tf::Vector3 start_pos = frame_start_pose.getOrigin();
         start_pos.setZ(0);
         //最终位姿
         tf::Vector3 end_pos = frame_end_pose.getOrigin();
         end_pos.setZ(0);

         tf::Vector3 mid_pos;
         tf::Vector3 mid_point;
         double mid_angle, lidar_angle, lidar_dist;

         //插值计算出来每个点对应的位姿
         for(int i = 0; i< beam_number;i++){
             //角度插值
             mid_angle =
 tf::getYaw(frame_start_pose.getRotation().slerp(frame_end_pose.getRotation
(), beam_step * i));
             //线性插值
```

```
            mid_pos = start_pos.lerp(end_pos, beam_step * i);

            //得到激光点在odom坐标系中的坐标
            double tmp_angle;

            //如果激光雷达不等于无穷,则需要进行矫正.
            if( tfFuzzyZero(ranges[startIndex + i]) == false){
                //计算对应的激光点在odom坐标系中的坐标

                //得到这帧激光束距离和夹角
                lidar_dist  =  ranges[startIndex+i];
                lidar_angle =  angles[startIndex+i];
                //激光雷达坐标系下的坐标
                double laser_x,laser_y;
                laser_x = lidar_dist * cos(lidar_angle);
                laser_y = lidar_dist * sin(lidar_angle);
                //里程计坐标系下的坐标
                double odom_x,odom_y;
                odom_x = laser_x * cos(mid_angle) - laser_y *
sin(mid_angle) + mid_pos.x();
                odom_y = laser_x * sin(mid_angle) + laser_y *
cos(mid_angle) + mid_pos.y();

                //转换到类型中去
                mid_point.setValue(odom_x, odom_y, 0);
                //把在odom坐标系中的激光数据点转换到基础坐标系
                double base_s, base_c;
                base_s = sin(base_Yaw);
                base_c = cos(base_Yaw);

                double tmp_x,tmp_y;
                tmp_x =  mid_point.x()*base_c  + mid_point.y()*base_s -
base_pos.x()*base_c - base_pos.y()*base_s;
                tmp_y = -mid_point.x()*base_s  + mid_point.y()*base_c  +
base_pos.x()*base_s - base_pos.y()*base_c;
                mid_point.setValue(tmp_x,tmp_y,0);

                //然后计算以起始坐标为起点的 dist_angle
                double dx,dy;
                dx = (mid_point.x());
                dy = (mid_point.y());
                lidar_dist = sqrt(dx*dx + dy*dy);
                lidar_angle = atan2(dy,dx);

                //激光雷达被矫正
                ranges[startIndex+i] = lidar_dist;
                angles[startIndex+i] = lidar_angle;
            }
            //如果等于无穷,则随便计算一下角度
            else{
                //激光角度
                lidar_angle = angles[startIndex+i];
                //里程计坐标系的角度
                tmp_angle = mid_angle + lidar_angle;
                tmp_angle = tfNormalizeAngle(tmp_angle);
                //如果数据非法 则只需要设置角度就可以了。把角度换算成start_pos坐标系
内的角度
                lidar_angle = tfNormalizeAngle(tmp_angle - start_Yaw);
```

```
88                  angles[startIndex+i] = lidar_angle;
89              }
90          }
91          //end of TODO
92      }
```



二、ICP求解

ICP.

假设两个点集为 $P=\{p_i\}$，和 $P'=\{p_i'\}$；$i=1,2,\cdots,N$.

又：y_i，$p_i=Rp_i'+t+N_i$，其中 R 为旋转矩阵，t 为平移向量，N_i 为噪声向量.

误差项 $e_i=p_i-(Rp_i'+t)$

$$\min_{R,t} J=\frac{1}{2}\sum_{i=1}^{n}\|(p_i-(Rp_i'+t)\|_2^2$$

定义质心，$p=\frac{1}{n}\sum_{i=1}^{n}(p_i)$，$p'=\frac{1}{n}\sum_{i=1}^{n}(p_i')$

则 ICP 求解步骤如下：

① 去质心坐标.

$$q_i=p_i-p \qquad q_i'=p_i'-p'$$

② 根据优化问题计算旋转矩阵：

$$R^*=\arg\min_R \frac{1}{2}\sum_{i=1}^{n}\|q_i-Rq_i'\|^2$$

③ 根据②的 R 计算 t：

$$t^*=p-Rp'$$

其中 $\frac{1}{2}\sum_{i=1}^{n}\|q_i-Rq_i'\|^2=\frac{1}{2}\sum_{i=1}^{n}q_i^Tq_i+q_i'^TR^TRq_i'-2q_i^TRq_i'$

忽略与 R 无关项，则 R^* 的优化目标函数为

$$\sum_{i=1}^{n}-q_i^TRq_i'=\sum_{i=1}^{n}-tr(Rq_i'q_i^T)=-tr\left(R\sum_{i=1}^{n}q_i'q_i^T\right)$$

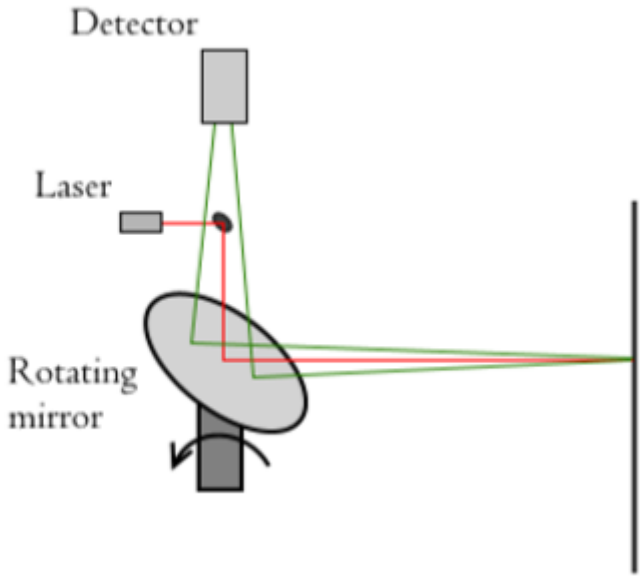
定义 $W=\sum_{i=1}^{n}q_iq_i'^T$，$W$ 进行 SVD 分解，得

$$W=U\Sigma V^T$$

则当 W 满秩时，$R=UV^T$，再根据③求得 t.

三、激光雷达相关

3.1 简述激光雷达测距原理

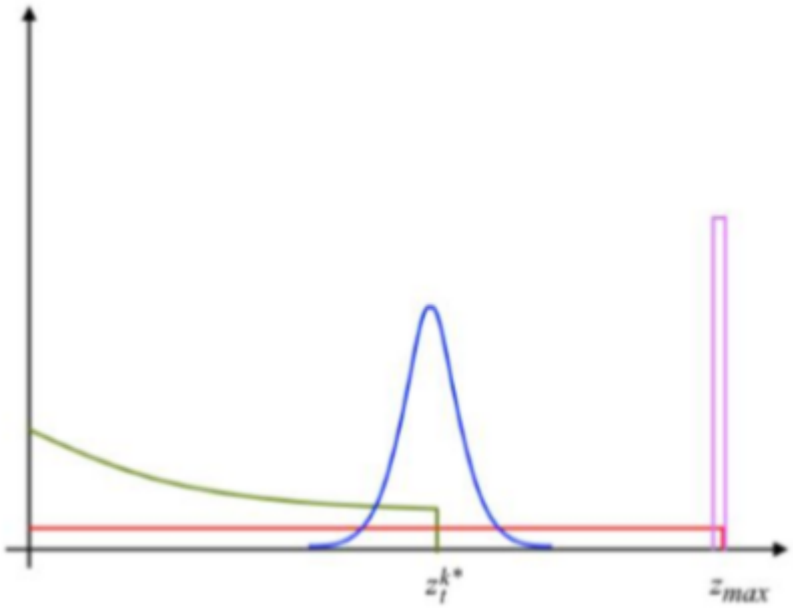


激光雷达的测距，一种基于飞行时间(tof)的测距，另一种基于测量波形相位差的测距。

1）TOF测量是根据短激光脉冲发射时，发送和接收之间的时间 Δt，然后，然后根据光速恒定为c计算测量距离$r = \frac{\Delta t \times c}{2 \times n}$， 这里的n是空气的折射率 。

2）波形相位差的测距中 Δt 的测量是根据，将连续调幅光束发射到目标上，测量发射和接收波形的相位差$\Delta\varphi$ 。$\Delta t = \frac{\Delta\varphi \times f_m}{2 \times \pi}$ ，这里的f_m是调制频率。之后就根据1）的方式计算距离。

3.2 图片解读



通常测量可以通过条件概率密度$p(z_t|x_t, m)$来建模，其中z_t是单激光扫描的测量值，x_t是激光装置的姿态，m是地图。激光装置扫描的各个测量值z_t可以被认为是相互独立的，因此综合概率可以计算如公式$p(z_t|x_t, m) = \prod_{k=1}^{K} p({z_t}^k|x_t, m)$所示，其中$k$表示单个测量值的指标。

该图片是光束模型的混合模型示意图，${z_t}^{k*}$是光线投射计算出的距离，z_{max}是传感器的最大距离。蓝色表示实际测量中的测量不确定度(Gaussian分布)，绿色表示目标前方可能存在的动态障碍物(指数分布)，红色表示来自意外来源的随机噪声，粉色表示测量失败的可能性。

四、去畸变设计方案

4.1 仅用IMU去除运动畸变可能会有哪些不足之处?

激光帧去畸变原理，一遍是把一帧激光中的每个激光点坐标变换到不同时刻的机器人里程计上，如果仅仅依靠IMU充当里程计，则需要考虑在实际使用中，IMU线加速度精度差的问题，以及IMU本身标定的困难程度。

4.2 在仅有IMU和激光雷达传感器的情况下，你会如何设计运动畸变去除方案(平移+旋转)，达到较好的畸变去除效果?