# ROS数据展示系统

## 首先启动ros：

文本

描述已自动生成

## 使用命令行窗口显示小车的IMU数据与里程计数据

首先对于bag包，我们需要先了解其内部信息，使用rosbag info命令查看内部的topic以及消息类型：（下面任务同理）

文本

描述已自动生成

查找对应的消息类型所对应的数据结构：

IMU

文本

描述已自动生成

Odometry：

文本

描述已自动生成

再根据对应的数据结构编写相关cpp接收节点：

文本

描述已自动生成

文本

描述已自动生成

初始化节点，根据rosbag info获取：（下面同理）

文本

描述已自动生成

再cmakelist中添加相关内容：



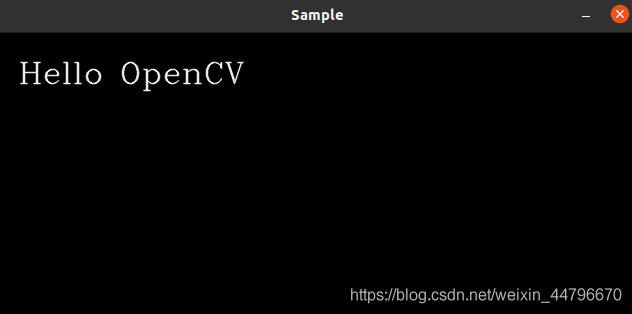
运行接收节点接收bag包播放的imu数据与odometry数据：

文本

描述已自动生成

## ⽤图形界⾯显示颜⾊相机和深度相机的数据（利⽤OpenCV库）

首先安装opencv库，在这里我使用了编译安装的方式，然后验证opencv是否安装成功：



然后我们需要知道如何使用opencv库播放颜色相机与深度相机的数据：ros的图像与opencv的图像是不一样的，因此我们需要：

cv\_bridge用于ROS图像和OpenCV图像的转换

ROS以自己的sensor\msgs/Image消息格式传递图像，但用户希望将图像与OpenCV结合使用。在OpenCV中，图像以Mat矩阵的形式存储，与ROS定义的图像消息的格式有一定的区别，CvBridge是一个ROS库，提供ROS和OpenCV之间的接口。

图示

描述已自动生成

回调函数，接收数据，转换数据：

文本

描述已自动生成深度相机与彩色相机只有内部参数以及接收的节点不一样的少量区别，其他均一样：

电脑屏幕的截图

描述已自动生成

图像展示：

电脑萤幕画面

中度可信度描述已自动生成

## ⽤图形界⾯显示激光雷达的点云数据（利⽤PCL库）;

同样是编译安装PCL库，然后再camkelist中链接到该库。

同上，通过rosbag info寻找雷达点云图的节点，编写回调函数：

文本

描述已自动生成

通过PCL内置的库显示点云：

图片包含 游戏机

描述已自动生成

拓展任务

使用gmapping（slam）建立激光雷达图像

根据不同的点在图上绘制出不一样的颜色的图像并利用gmapping来建立图像

文本

描述已自动生成

运行slam\_gampping

图片包含 文本

描述已自动生成

初始化节点并检测是否打开bag包：

文本

描述已自动生成

运行编写好的节点：

电脑萤幕画面

描述已自动生成

效果图：

地图

描述已自动生成