RFlySim 激光 SLAM 的应用

❖ 基本步骤：

1. 先配置 RFlySim 传感器；
2. 配置 tf 树所用到的 frame_id;
3. 根据自身需是构建 2D 地图还是 3D 地图，如是用 2D 地图则要把激光雷达点云数据转换成 laserScan 数据；
4. 配置激光 SLAM 开源算法；
5. 启动程序；

❖ 配置 RFlySim 激光雷达传感器

先参考 PPT 第六讲，激光雷达篇。这里有些两点需要说明，如果是构建 3 维地图，那么按照多线束激光雷达配置，如果是构建 2 维地图,则需要做些不同的处理(尽管 pointcloud_to_laserscan 功能包能够在三维点云上做些过滤，但是不建议这样做),垂直方向我们设置 0 度，线束设置 1(单线激光雷达)

```
"otherParams":[200,0.05,-90,90,0,0,0,0]
"DataHeight":1,
"DataWidth":720,
 "DataCheckFreq":50,
```

DataWidth 参数为单线点的个数。假设为例水平角分辨率为 0.25,那么根据上配置的角度范围[-90,90],那么 DataWidth 的值为 720，推荐配置角分辨率小于 0.5，角度配置范围要大于 180 度，数据发布频率为 50 为宜。

除了以上参数外，其他参数按照通用参数来配置。

❖ 配置 tf 树所用到的 frame_id

在配置消息数据 frame_id 之前，最好对 ROS tf 树有一定的了解，当然不了解也没关系，按照说明来配置一样可以跑。该功能在 RflySimAPIs/PythonVisionAPI/1-APIUsageDemos/18-ConfigROSTFAPIDemos 中有说明，以该例程中配置文件说明，这个例程配置 3 个视觉传感器，两个灰度相机，一个激光雷达，其中激光在 SeqID 为 2 的位置，那么 tf_cfg.yaml 文件可以配置成这样：

```
sensors_num: 3
sensors_frame_id:  ["left_rgb","right_rgb","horizental_laser"]
imu_frame_id: "imu"
```

sensors_num 为 config.json 中配置传感器的个数，
sensors_frame_id 分别对应各自的 SeqID 的 frame_id 可以自由设置,
imu_frame_id 为 imu 话题对应的 frame_id。

❖ PointCloud_to_LaserScan(仅适在构建二维地图中使用)

我们输出的点云数据是 Sensor/PointCloud2 类型，而二维激光 SLAM 通常使用的是 sensor/LaserScan 数据类型，因此我们需要转换数据类型，ROS 官方提供了对应的功能包 http://wiki.ros.org/pointcloud_to_laserscan ，下面对里面官网的配置参数说明做一个补充
前面小节中提到,改功能包会对三维点云做过滤,如果过滤效果不好,会影响到建图效果,在我们数据做配置好传感器的前提下,这个高度范围值设置成合适的值即可,只要能包含所有点的数据.
~min_height (double, default: 0.0):
~max_height (double, default: 1.0)

这两个值就是与我们设置水平角保持一致,需要注意单位是弧度制,这与我们的 config.json 文件配置是不同的
~angle_min (double, default: -π/2)
~angle_max (double, default: π/2)

这里就是上面提到水平角度分辨率,但是要注意的是单位
~angle_increment (double, default: π/360)

这里数据数据发布频率
~scan_time (double, default: 1.0/30.0)

点云有效距离的最大值与最小值,对应着我们配置文件 otherParams[1]与 otherParams[0]
~range_min (double, default: 0.45)
~range_max (double, default: 4.0)

这里 farme_id 配置,与 config.json 文件内保持一致即可
~target_frame (str, default: none)
...
其他的配置参数不再另做说明

launch 文件配置：
以 sample_node.launch 文件为准修改对应的话题名称，比如

```
<remap from="cloud_in" to="/rflysim/sensor2/vehicle_lidar"/>
<remap from="scan" to="laser_scan"/>
```

❖ 配置激光 SLAM 开源算法

开源的激光 SLAM 有很多，这里以应用比较广泛的 Cartographer 为例，算法功能性以及鲁棒性等参数配置参考官网,这里只讲 RflySim 在 激光 SLAM 中的应用，在配置 Cartographer 的过程讲几点需要注意的，避免踩坑；

cartographer 官方文档地址：https://google-cartographer.readthedocs.io/en/latest/
cartographer-ros 官方文档地址：https://google-cartographer-ros.readthedocs.io/en/latest/

- 强烈建议先源码编译安装 cartographer 库,然后再从 github 上下载 cartographer-ros 源码编译，而不是按照 cartographer-ros 官方文档地址说明的那个操作，一个移动机器人项目不仅仅只有 SLAM 功能，所以如果按照 cartograpehr-ros 官方说明的那样操作会有很多麻烦;

- 在编译安装 Cartographer 源码时，不建议使用网页文档说的使用 git clone 去下载依赖库的源码，而是要到对应 github 地址上下载对应的依赖库版本分支源码。
- 在编译 cartographer-ros 源码时，需要采用 Release 模式编译。

配置 launch 文件(以 backpack_2d.launch,和 backpack_3d.launch 为例)

backpack_2d.launch 对于构建二维地图,也就是使用单线激光雷达的配置,

```
<remap from="echoes" to="laser_scan
" />
```

红色字体为功能包 pointcloud_to_laserscan 发布的 lanserScan 消息类型的话题，在该例程中为 laser_scan

backpack_3d.launch 构建三维地图,

```
<remap from="points2" to="/rflysim/sensor2/vehicle_lidar" />
```

修改 tf 需要的 urdf 模型文件：
这里讲怎么把 urdf 与我们的 tf_config 文件中 frame_id 以及后续讲的 lua 配置文件结合起来构建 tf 树，关于更深层次的不再这里阐述。

修改 backpack_2d.lua: 更多算法上的参数可以根据官网上说明配置，这里只讲应用性参数，因此这讲以下几个参数：

```
  tracking_frame = "imu",
  published_frame = "horizental_laser",
  odom_frame = "odom",
  provide_odom_frame = false,
```

tf 构建方案有很多，根据项目中需要进行更改，这里提供可行的方案.tracking_frame 这里设置成话题 imu 中的 frame_id.当然可以是其他的，但是必须要有一个 imu 到其他某个 frame_id 的连接。published_frame 这里是我们在 tf_config.yaml 文件中的激光雷达 frame_id 或者是 pointcloud_to_laserscan 中的 target_frame 值。
provide_odom_frame 这里设置成 false。

构建 tf 连接关系
修改 backpack_2d.urdf 文件，

```
<robot name="cartographer_backpack_2d">
 <material name="orange">
  <color rgba="1.0 0.5 0.2 1" />
 </material>
 <material name="gray">
  <color rgba="0.2 0.2 0.2 1" />
 </material>

 <link name="imu">
  <visual>
   <origin xyz="0 0 0" /> <!--imu 相对于载体的坐标位置 -->
   <geometry>
    <box size="0.06 0.04 0.02" />
   </geometry>
   <material name="orange" />
  </visual>
 </link>
 <link name="horizental_laser"> <!-- 为 laserscan 话题的 frame_id 或者 tf_config 中三维点云 frame_id -->
  <visual>
   <origin xyz="0 0 0" /> <!-- 激光雷达在载体中坐标位置-->
   <geometry>
    <cylinder length="0.05" radius="0.03" />
```

```
    </geometry>
    <material name="gray" />
  </visual>
 </link>


 <joint name="imu2lidar" type="fixed">
  <parent link="horizental_laser" />
  <child link="imu" />
  <origin xyz="0 0 0" />
 </joint>

</robot>
```

配置 tf 一定要注意 frame_id 的的 link 连接，这里话题 frame_id，urdf 模型文件，lua 文件配置，三个结合起来配置。
另外以上修改的配置文件仅仅是作为一个模板讲解，用户可以定义自己的配置文件。三维建图 TF 配置和上面讲解的一样。