1.概述

实际使用中会出现轮子打滑和累计误差的情况，这里单单使用编码器得到里程计会出现一定的偏差，虽然激光雷达会纠正，但一个准确的里程对这个系统还是较为重要

2.IMU数据获取

IMU即为 惯性测量单元，一般包含了三个单轴的加速度计和三个单轴的陀螺仪，简单理解通过加速度二次积分就可以得到位移信息、通过角速度积分就可以得到三个角度，实时要比这个复杂许多

2.1 PIBOT IMU

PIBOT在嵌入式程序提供出原始的数据接口，通过配置可以输出原始raw_imu topic，

```
apollo@apollo-desktop:~$ rostopic info /raw_imu
Type: pibot_msgs/RawImu

Publishers:
 * /pibot_driver (http://172.16.8.45:38089/)

Subscribers:
 * /pibot_imu (http://172.16.8.45:36209/)
```

该topic类型为自定义具体如下，即为三轴加速度三轴陀螺仪和三轴磁力计的原始数据

```
apollo@apollo-desktop:~$ rosmsg info pibot_msgs/RawImu
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
bool accelerometer
bool gyroscope
bool magnetometer
geometry_msgs/Vector3 raw_linear_acceleration
  float64 x
  float64 y
  float64 z
geometry_msgs/Vector3 raw_angular_velocity
  float64 x
  float64 y
  float64 z
geometry_msgs/Vector3 raw_magnetic_field
  float64 x
  float64 y
  float64 z
```

```
apollo@apollo-desktop:~$ rostopic echo /raw_imu
header:
  seq: 51511
  stamp:
    secs: 1529562342
    nsecs: 178633017
  frame_id: "imu_link"
accelerometer: True
gyroscope: True
magnetometer: True
raw_linear_acceleration:
  x: 0.1953125
  y: -0.5859375
  z: 8.671875
raw_angular_velocity:
  x: -0.05706467852
  y: 0.0109272785485
  z: -0.00971313659102
raw_magnetic_field:
  x: 53.3600006104
  y: -379.959991455
  z: -140.759994507
```

通过对原始数据处理得到一个/imu/data_raw数据类型为sensor_msgs/Imu，

```
apollo@apollo-desktop:~$ rostopic info /imu/data_raw
Type: sensor_msgs/Imu

Publishers:
 * /pibot_imu (http://172.16.8.45:36209/)

Subscribers:
 * /complementary_filter_gain_node (http://172.16.8.45:42697/)
```

通过ROS提供的相关包imu_tools进行滤波 可以看到complementary_filter_gain_node会订阅该topic，即该topic作为输入滤波得到最终数据(发布/imu/data topic 类型同样为sensor_msgs/Imu)

```
apollo@apollo-desktop:~$ rostopic info /imu/data
Type: sensor_msgs/Imu

Publishers:
 * /complementary_filter_gain_node (http://172.16.8.45:42697/)

Subscribers:
 * /imu_with_covariance (http://172.16.8.45:37981/)
```

输出该topic可以看到得到的值波动已经较小了，且静止的时候接近于0

```
apollo@apollo-desktop:~$ rostopic echo /imu/data
header:
  seq: 141230
  stamp:
    secs: 1529563285
    nsecs: 340643435
  frame_id: "imu_link"
orientation:
  x: 0.000382586626582
  y: -0.000185662087256
  z: 0.0315045424193
  w: 0.999503518235
orientation_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: 0.000147970471465
  y: -0.000647745602036
  z: -0.000344794580046
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
  x: -0.004301312
  y: -0.029196343
  z: 9.794302301
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
header:
  seq: 141231
  stamp:
    secs: 1529563285
    nsecs: 350941115
  frame_id: "imu_link"
orientation:
  x: 0.00036464393009
  y: -0.000185323242321
  z: 0.0315027870373
  w: 0.999503580333
orientation_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: 0.000146490766751
```

3.两种融合的方法

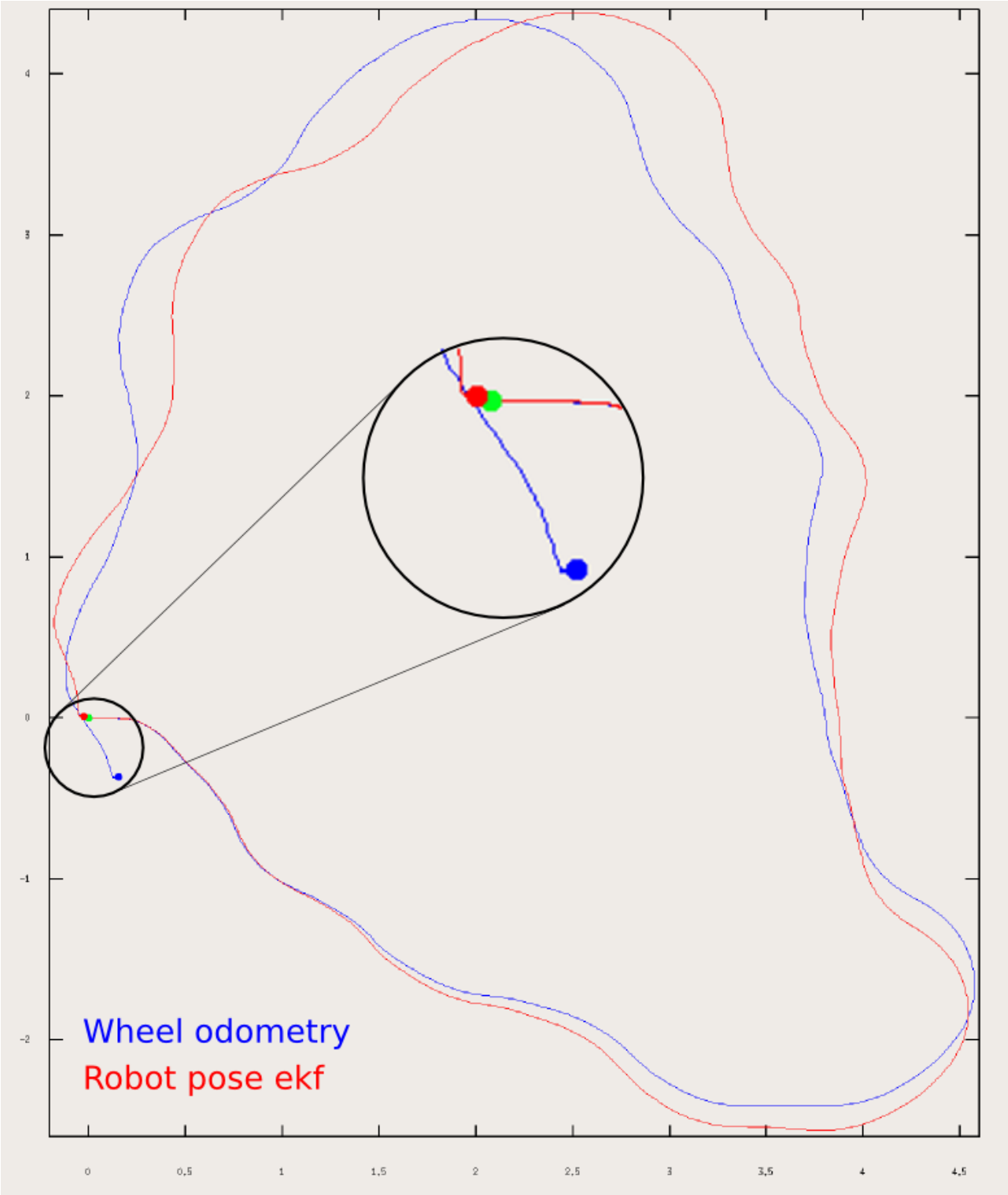
3.1 一种简单的方法

从imu得到的数据为一个相对角度(主要使用yaw，roll和pitch 后面不会使用到)，使用该角度来替代由编码器计算得到的角度。 这个方法较为简单，出现打滑时候因yaw不会受到影响，即使你抬起机器人转动一定的角度，得到的里程也能正确反映出来

3.2 扩展的卡尔曼滤波

官方提供了个扩展的卡尔曼滤波的包robot_pose_ekf，robot_pose_ekf开启扩展卡尔曼滤波器生成机器人姿态，支持

- odom（编码器）
- imu_data（IMU）
- vo（视觉里程计） 还可以支持GPS 引用官方图片



PR2从实际初始点(绿色)溜达一圈回到初始点(绿色)，编码器的里程(蓝色)发生了漂移，而使用robot_pose_ekf融合出来的里程(红色)则跟实际位置基本重合(后面我们会针对这个测试下效果)

> 中间的圆是小圆放大的展示效果

再回去看下该包的输出

2.1.2 Published Topics

robot_pose_ekf/odom_combined (geometry_msgs/PoseWithCovarianceStamped)
The output of the filter (the estimated 3D robot pose).

2.1.3 Provided tf Transforms

odom_combined → base_footprint

- 发布一个topic， 类型需要注意下是PoseWithCovarianceStamped并非Odometry 后面会用到这个作为显示，所以还需要一个转换

```
apollo@apollo-desktop:~$ rostopic info /robot_pose_ekf/odom_combined
Type: geometry_msgs/PoseWithCovarianceStamped

Publishers:
 * /robot_pose_ekf (http://172.16.8.45:42889/)

Subscribers:
 * /odom_ekf (http://172.16.8.45:41001/)
```

查看该topic信息可以看到odom_ekf订阅了该topic 再次查看该节点信息可以看到

```
apollo@apollo-desktop:~$ rosnode info /odom_ekf
--------------------------------------------------------------------------------
Node [/odom_ekf]
Publications:
 * /odom [nav_msgs/Odometry]
 * /rosout [rosgraph_msgs/Log]

Subscriptions:
 * /robot_pose_ekf/odom_combined [geometry_msgs/PoseWithCovarianceStamped]

Services:
 * /odom_ekf/get_loggers
 * /odom_ekf/set_logger_level


contacting node http://172.16.8.45:41001/ ...
Pid: 29572
Connections:
 * topic: /rosout
    * to: /rosout
    * direction: outbound
    * transport: TCPROS
 * topic: /robot_pose_ekf/odom_combined
    * to: /robot_pose_ekf (http://172.16.8.45:42889/)
    * direction: inbound
    * transport: TCPROS

apollo@apollo-desktop:~$
```

，他会发出一个Odometry的topic

- 发出一个tf

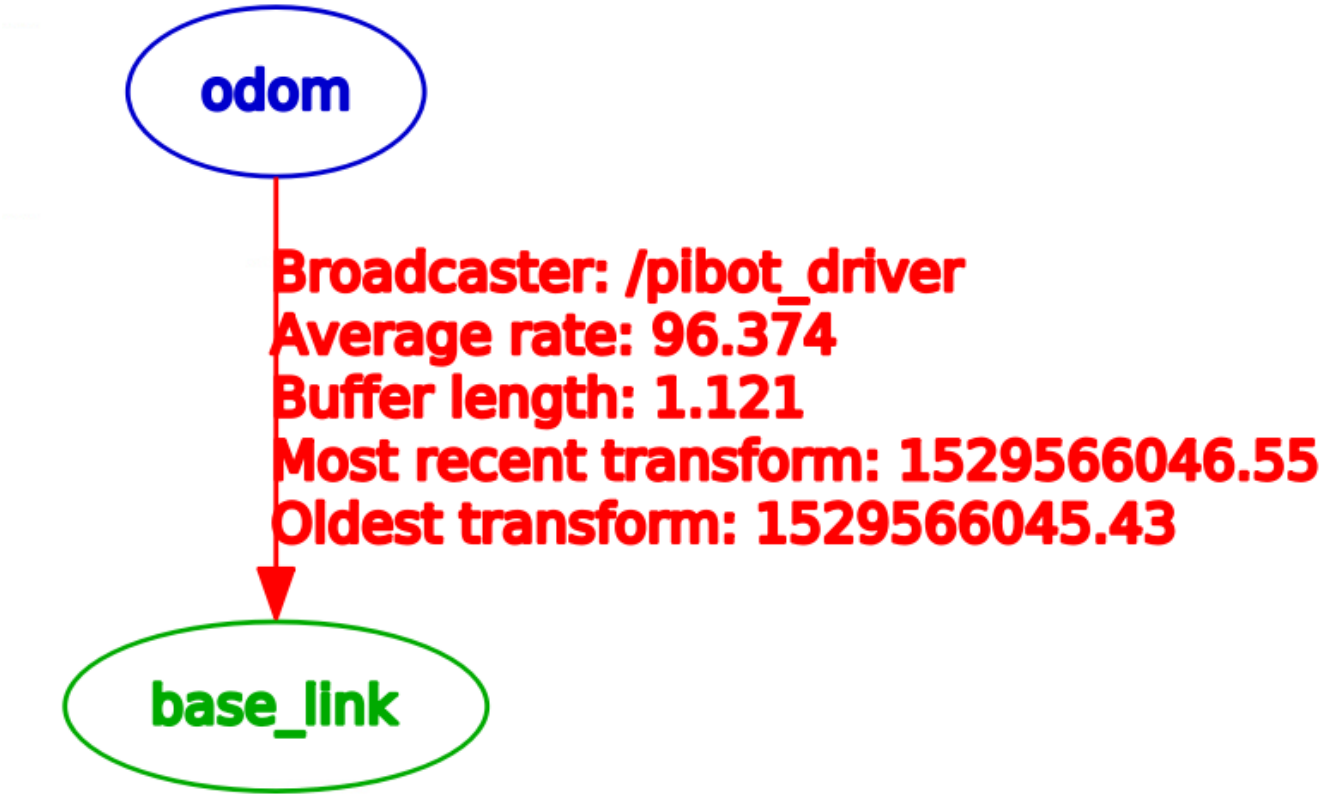
> 在robot_pose_ekf配置时，做了些映射处理，这样可以保证导航层在使用和不用imu的时候无需修改就可以工作

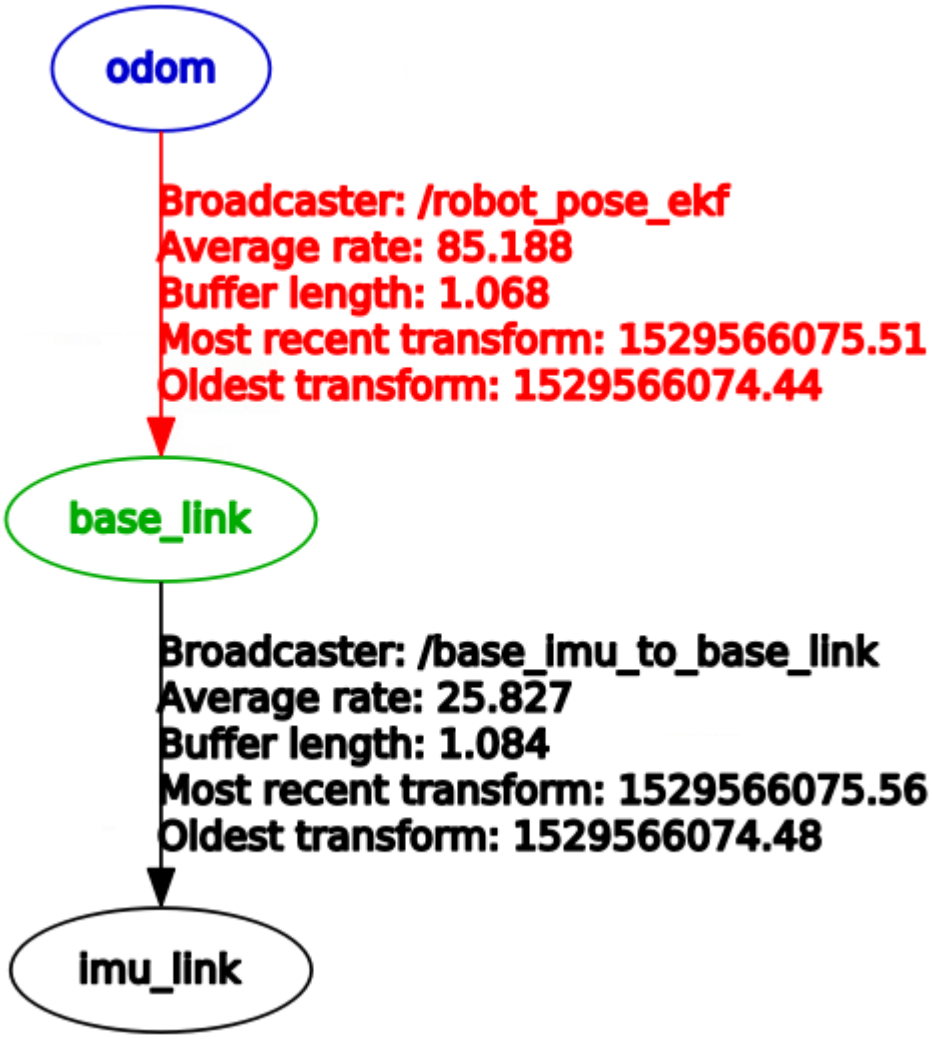
```
<node pkg="robot_pose_ekf" type="robot_pose_ekf" name="robot_pose_ekf" output="screen">
    <param name="output_frame" value="odom" />
    <param name="base_footprint_frame" value="base_link"/>
    <param name="freq" value="100.0"/>
    <param name="sensor_timeout" value="1.0"/>
    <param name="odom_used" value="true"/>
    <param name="imu_used" value="true"/>
    <param name="vo_used" value="false"/>
    <param name="debug" value="true"/>
    <remap from="odom" to="wheel_odom" />
    <remap from="imu_data" to="imu_with_covariance/data" />
</node>
```

bringup.lauch或者bringup_with_imu.launch 输出的tf都为odom → base_footprint ;发出的里程也都是odom bringup_with_imu.launch轮子的里程topic 映射为wheel_odom

> 这里很重要，后面的对该包的验证会使用到

下2张图展示了未使用IMU和使用IMU时候的tf tree情况, 可以看到用了一致的frame




odom
Broadcaster: /robot_pose_ekf
Average rate: 85.188
Buffer length: 1.068
Most recent transform: 1529566075.51
Oldest transform: 1529566074.44
base_link
Broadcaster: /base_imu_to_base_link
Average rate: 25.827
Buffer length: 1.084
Most recent transform: 1529566075.56
Oldest transform: 1529566074.48
imu_link