apriltag 二维码追踪

rmtt_ros 中所包含的功能包如下图所示：

```
tianbot@ros2go:~/robomaster_ws/src/rmtt_ros$ tree -L 1
.
├── README.md
├── rmtt_apriltag
├── rmtt_description
├── rmtt_driver
├── rmtt_ros
├── rmtt_teleop
└── rmtt_tracker
```

rmtt_apriltag 是二维码检测功能包，依赖于 apriltag_ros 功能包。该功能包已经更新在了 rmtt_ros 中，也可以自行安装：

sudo apt install ros-noetic-apriltag-ros

编译：catkin_make_isolated

配置环境变量：source devel_isolated/setup.bash

rmtt_description 是无人机的描述文件功能包，提供相机和机体间坐标变换。

rmtt_driver 是无人机驱动。

rmtt_teleop 是无人机遥控功能包。

rmtt_tracker 包含无人机 apriltag 二维码跟踪程序。

无人机跟踪程序功能包依赖于 PyPI 的 simple-pid，输入以下指令安装：

pip3 install simple-pid

原理说明：

我们需要控制 TT 无人机到达相对 apriltag 二维码的某一位置姿态。TT 的摄像头可以检测到二维码并且发布相机坐标系到二维码坐标系的平移旋转坐标变换（camera_link -> tag_xxx）。我们可以将理想无人机的位置在二维码的坐标系中发布出来，这样只需要得到无人机实际机体坐标系（base_link）到理想无人机位置坐标系的平移旋转坐标变化关系，用 pid 将这一坐标变换转换成无人机升降、前后（俯仰）、左右（滚转）、转动（偏航）四个通道的速度控制量，就可以实现无人机对二维码的跟踪。

无人系统技术研究院

实验步骤：

1、连接无人机到计算机，在新的终端启动无人机驱动：

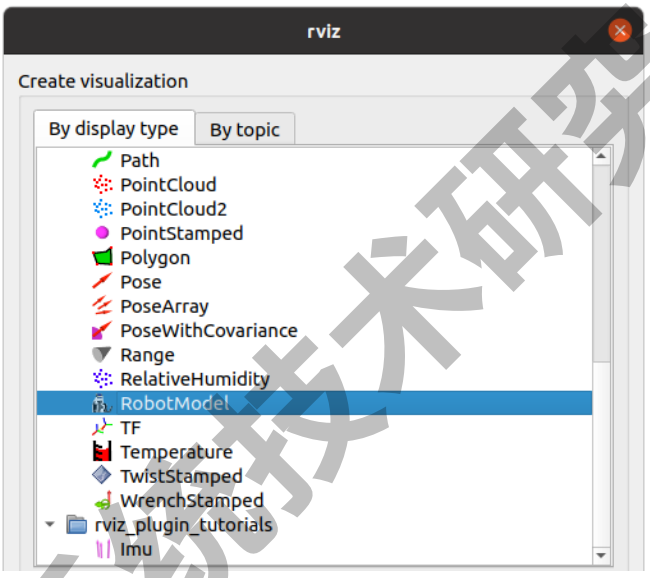
roslaunch rmtt_driver rmtt_bringup.launch

2、在新的终端启动无人机的描述文件：

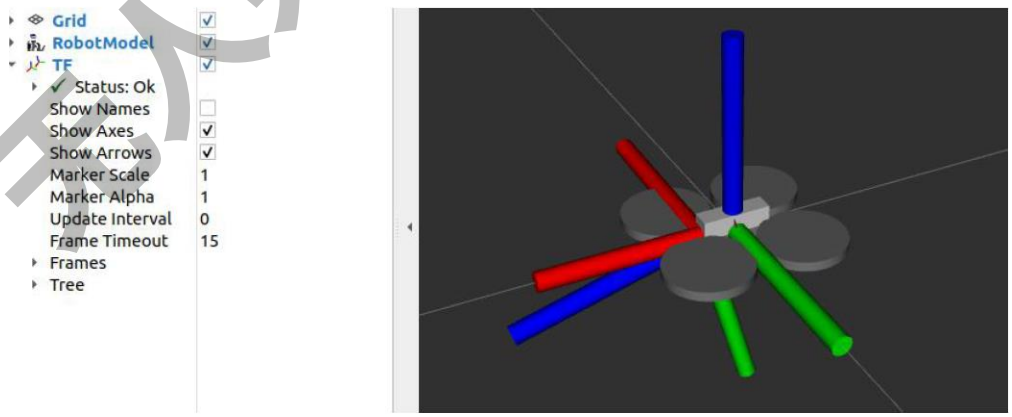
roslaunch rmtt_description rmtt_description.launch

~rmtt_ws\src\rmtt_ros\rmtt_description\urdf\tello_talent.xacro 描述文件包括无人机绘制，以及相机坐标系和机体坐标系的坐标变换，请自行阅读。

打开 rviz 添加 RobotModel 和 TF：



将 Global Options->fixed frame 更改为 base_link，可以看到 rviz 的显示区域如下图所示：



3、启动二维码检测程序

根据需要修改 rmtt_apriltag/config/tags.yaml 文件中可以检测的二维码（id：5）：

无人系统技术研究院

```
standalone_tags:
  [
    {id: 10, size: 0.12},
    {id: 20, size: 0.12},
    {id: 30, size: 0.12},
    {id: 5, size: 0.12}
  ]
```

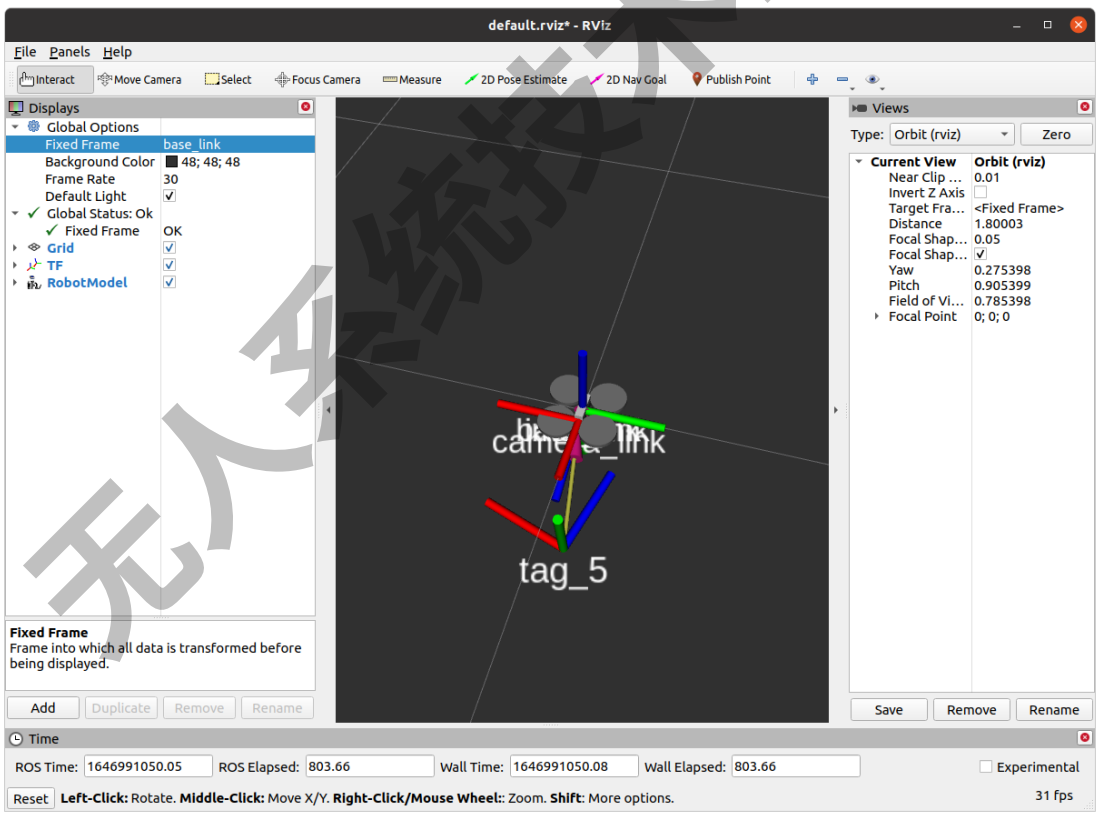
在新的终端输入：

roslaunch rmtt_apriltag detection.launch

这时候飞机就开始检测二维码了。

```
NODES
  /rmtt/
    apriltag_ros (apriltag_ros/apriltag_ros_continuous_node)

ROS_MASTER_URI=http://localhost:11311

process[rmtt/apriltag_ros-1]: started with pid [137326]
[ INFO] [1620805973.352163585]: Initializing nodelet with 8 worker threads.
[ INFO] [1620805973.430806447]: Loaded tag config: 10, size: 0.12, frame_name: tag_10
[ INFO] [1620805973.430851508]: Loaded tag config: 20, size: 0.12, frame_name: tag_20
[ INFO] [1620805973.430864964]: Loaded tag config: 30, size: 0.12, frame_name: tag_30
[ INFO] [1620805973.430876467]: Loaded tag config: 586, size: 0.12, frame_name: tag_586
[ WARN] [1620805973.431182101]: No tag bundles specified
[ WARN] [1620805973.431470684]: remove_duplicates parameter not provided. Defaulting to true
```

如图所示，现在可以检测 id = 5 的二维码。



如上图所示，二维码标签的坐标可以在 base_link 参考系中显示。在 Fixed Frame 中切换参考坐标系，看看有什么不同。

4、设置无人机理想位置目标坐标系并且跟踪。

将 rmtt_tag_tracker.launch 中的 tag_id 设置为 5：

无人系统技术研究院

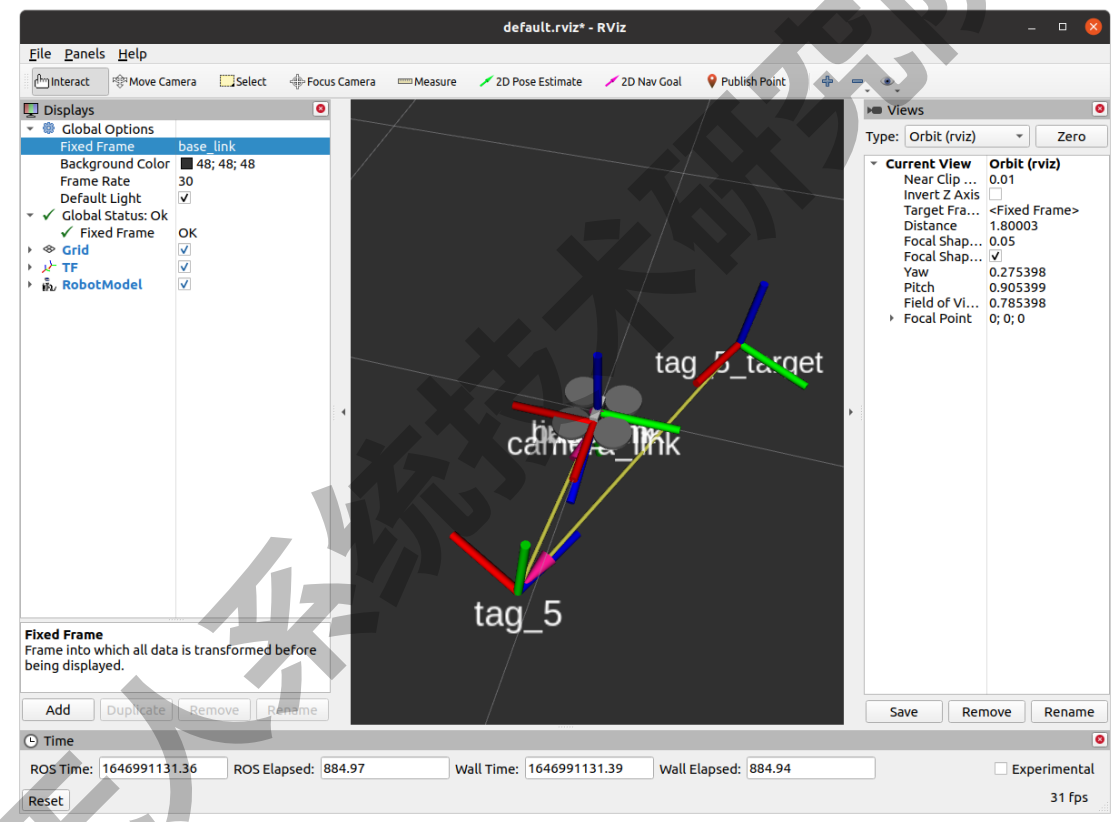
打开(O) rmtt_tag_tracker.launch ~/rmtt_ws/src/rmtt_ros/rmtt_tracker/launch 保存(S)

```
<?xml version="1.0"?>

<launch>
  <arg name = "drone_name" default="$(optenv RMTT_NAMESPACE /)" />
  <arg name = "tag_id" default="$(optenv RMTT_TAG_ID 5)" />
  <group ns="$(arg drone_name)">
    <node pkg="rmtt_tracker" type="rmtt_tracker.py" name="tag_tracker" output="screen"
 respawn="true">
      <param name="tag_id" value="$(arg tag_id)"/>
    </node>
  </group>
</launch>
```

在新的终端输入：

roslaunch rmtt_tracker rmtt_tag_tracker.launch

可以看到我们设置了一个 tag_5_target 的坐标系，目标就是让飞机追踪这个坐标系。



这时候可以用 rostopic echo /cmd_vel 命令查看飞机速度控制话题，应该有相应的输出。不过飞机还没有起飞，并不会动。

```
linear:
  x: -0.2693690677789881
  y: 0.014825360960796018
  z: 0.023664418727708594
angular:
  x: 0.0
  y: 0.0
  z: 0.04764233377645793
```

5、在新的终端启动遥控程序

roslaunch rmtt_teleop rmtt_teleop_key.launch

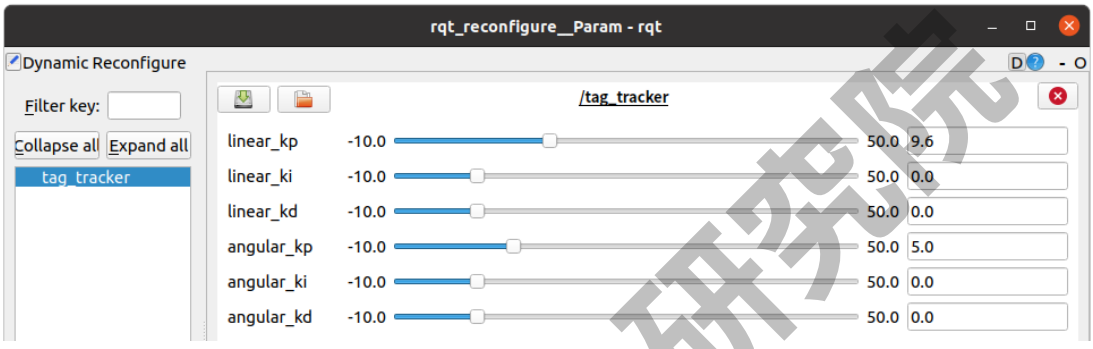
用键盘遥控飞机，飞到能够观察到二维码的地方，飞机就开始追踪二维码了。

6、动态 PID 参数调节

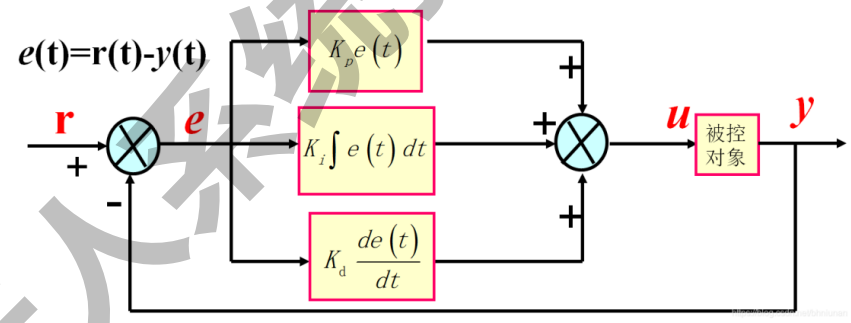
在新的终端输入：

rosrun rqt_reconfigure rqt_reconfigure

可以在弹出的窗口中动态调节几个通道的 PID 控制参数，注意升降、俯仰、滚转三个通道均使用 linear 对应的参数，偏航使用 angular 对应的参数。



PID 控制器是一种线性控制器，它主要根据给定值和实际输出值构成控制偏差，然后利用偏差给出合理的控制量。PID 里的 P 是 Proportion 的首字线，是比例的意思，I 是 Integral 的首字线，是积分的意思，D 是 Differential 的首字母，是微分的意思。



以无人机为例在没有控制系统的情况下，直接用信号驱动电机带动螺旋桨转动产生控制力，会出现动态响应太快，或者太慢，或者控制过冲或者不足的现象，无人机无法很好的完成跟踪、转弯等动作。为了解决这些问题，就需要在控制系统回路中加入 PID 控制器算法。在位置信息、速度信息和电机转速之间建立比例、积分和微分的关系，通过调节各个环节的参数大小，使无人机系统控制达到动态响应迅速、既不过冲、也不欠缺，可以完美的跟踪与转弯。

以位置控制为例，比例控制信号为系数 P 乘以指定位置和受控对象位置的差值。P 值越大，被控对象反应越快，P 值越小，被控对象反应越慢。虽然 P 值大能够较快的到达指

无人系统技术研究院

定位置，但是反应比较剧烈，总是因为过快冲过了头。相反 P 值小的反应比较平缓，但是它反应太慢，我们有时候接受不了。

为了不让被控对象冲过头，我们再给它加一个力，这个力就是 D，让这个力来起一个作用，就是让被控对象越接近指定位置的时候，接近的速度越慢，越远离指定位置的时候，接近的速度相对较快。这个 D 大家可以理解为受控对象靠近指定位置的一个阻力。

设置合适的 P 值和 D 值后，一些干扰如有风等就会让受控对象发生偏移，因为此时 P 和 D 产生的控制信号值均较小。所以我们再给它加一个力，这个力就是 I，积分控制。设置了 I 以后，I 会根据误差和误差经历的时间进行积分，然后决定施加给目标方向的力的大小，就能够让受控对象回到指定位置上。

代码讲解：

追踪程序的源代码在 rmtt_tracker/scripts/rmtt_tag_tracker.py。

我们从主函数看起，首先我们定义了要检测的二维码默认的 id，此处为 tag_5。track_distance 处需要设定二维码到无人机理想位置的距离，此处为 0.8m。我们还定义了发布器和订阅器，以及一个服务。订阅器中的回调函数是 tag_callback，一旦检测到二维码将调用这个函数，并计算无人机速度控制量并发布。服务用来接收动态调节的 PID 参数（可自行查阅 ROS 中的 dynamic reconfigure 使用方法）。

```
72 if __name__ == '__main__':
73
74     # init
75     rospy.init_node('tag_tracker')
76
77     # params
78     tag_id = rospy.get_param('~tag_id', '5')
79     tag_name = "tag_" + str(tag_id)
80     track_distance = rospy.get_param("~track_distance", 0.8)
81
82     # sub and pub
83     tag_sub = rospy.Subscriber('tag_detections', AprilTagDetectionArray, tag_callback, queue_size=1)
84     vel_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)
85
86     # dynamic reconfigure
87     srv = Server(tracker_pidConfig, pid_cb)
```

接着我们发布了一个静态坐标变换。注意我们发布这个静态变换的时候，做了欧拉角的一个转换，将欧拉角变换为了四元数。这是因为在 tf 变换中，坐标系的旋转是已四元数表示的和储存的，四元数和欧拉角等价。这个坐标变化是从二维码坐标系（父坐标系）到无人机理想位置坐标系（子坐标系）的变换关系。

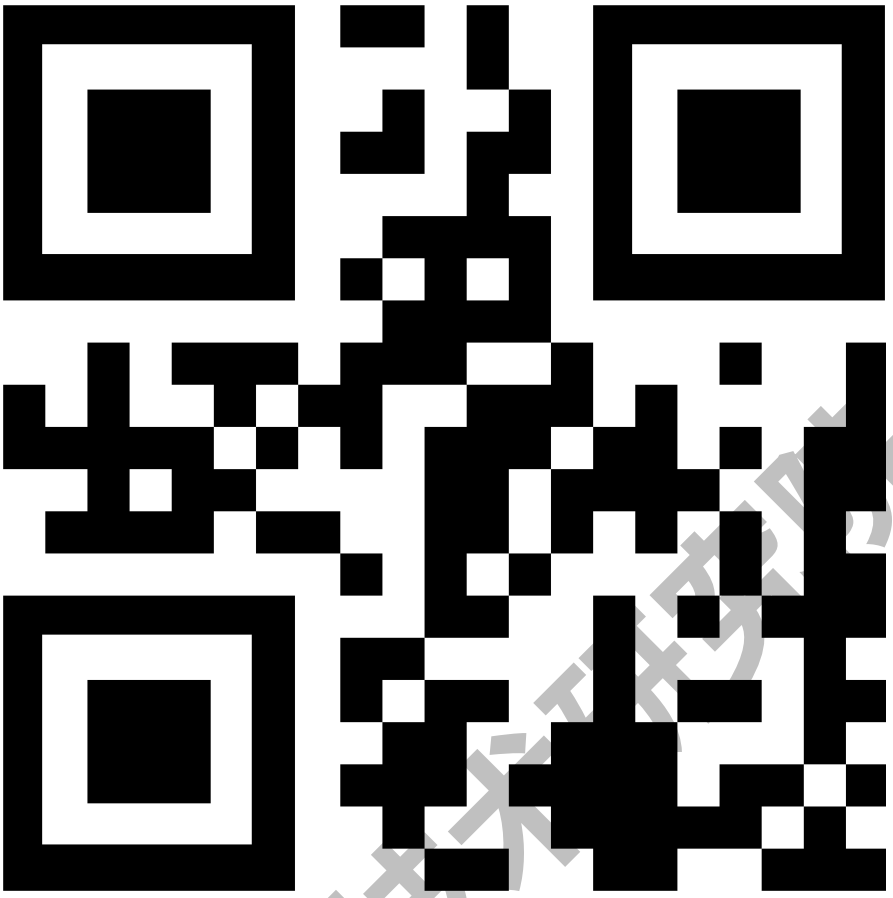
无人系统技术研究院

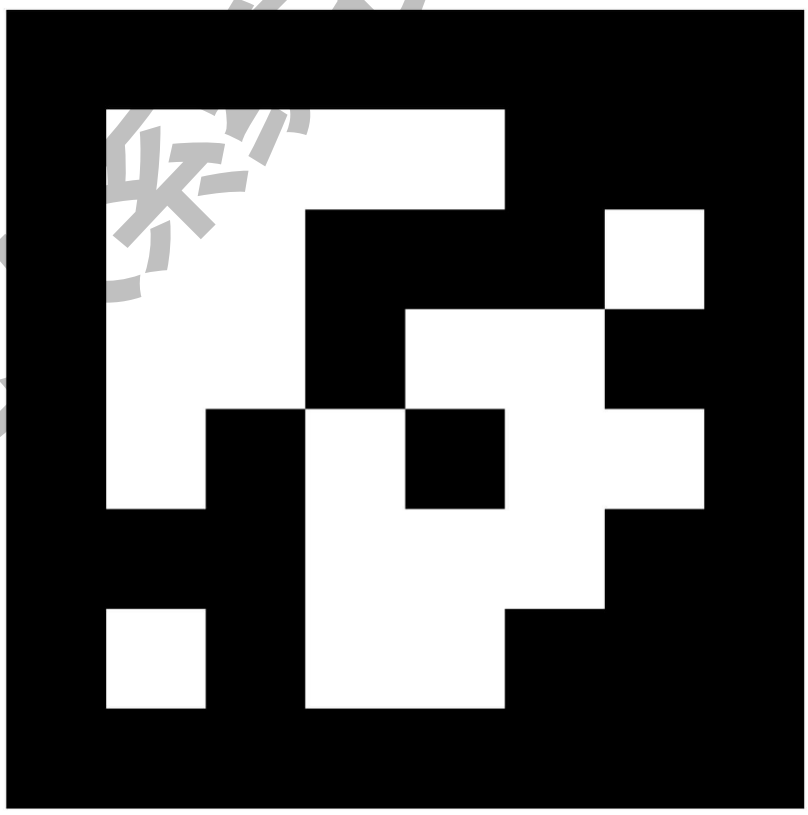
```
    # a static broadcaster to pub the target frame
    broadcaster = tf2_ros.StaticTransformBroadcaster()

    static_transformStamped = TransformStamped()
    static_transformStamped.header.stamp = rospy.Time.now()
    static_transformStamped.header.frame_id = tag_name
    target_frame = tag_name + "_target"
    static_transformStamped.child_frame_id = target_frame

    static_transformStamped.transform.translation.y += track_distance*np.tan(np.deg2rad(15))
    static_transformStamped.transform.translation.z += track_distance

    quat = quaternion_from_euler(np.deg2rad(-90), np.deg2rad(90), 0)
    static_transformStamped.transform.rotation.x = quat[0]
    static_transformStamped.transform.rotation.y = quat[1]
    static_transformStamped.transform.rotation.z = quat[2]
    static_transformStamped.transform.rotation.w = quat[3]
    broadcaster.sendTransform(static_transformStamped)
```

最后，订阅一个监听器，监听无人机机体 base_link 坐标系到 tag_5_target 坐标系的平移旋转关系，利用 PID 计算几个通道的速度控制量。这一监听和速度计算在 while 循环中以 10Hz 频率（rate）执行。

```
    # tf listener
    tfBuffer = tf2_ros.Buffer()
    listener = tf2_ros.TransformListener(tfBuffer)

    # rate
    rate = rospy.Rate(10.0)

    while not rospy.is_shutdown():
        if tag_detected:
            try:
                trans = tfBuffer.lookup_transform(target_frame, (rospy.get_namespace()+'base_link').strip("/"), rospy.Time(), rospy.Duration(0.2))
                vel = Twist()
                vel.linear.x = pid_x(trans.transform.translation.x)
                vel.linear.y = pid_y(trans.transform.translation.y)
                # vel.linear.z = pid_z(trans.transform.translation.z)
                quaternion = [trans.transform.rotation.x, trans.transform.rotation.y, trans.transform.rotation.z, trans.transform.rotation.w]
                # vel.angular.z should be calculate according to the tag position in cam. same as the tag link in base link. only angle matters.
                # this can be applied to linear.z.

                # vel.angular.z = pid_a(euler_from_quaternion(quaternion)[2])
                # vel.angular.z = - vel.linear.y / (track_distance + trans.transform.translation.x)
                rate.sleep()
            except (tf2_ros.LookupException, tf2_ros.ConnectivityException, tf2_ros.ExtrapolationException) as e:
                rospy.logwarn(e)
                continue

    rospy.spin()

```

无人系统大研究院





ID = 5

无锡科技有限公司