实物测试说明

MAVROS

mavros是连接飞控与上位机的重要桥梁，读取飞控中的IMU，电池，遥控信号等各种数据，同时将我们计算得到的控制指令发送给飞控。

树莓派安装mavros

```
sudo apt-get install ros-noetic-mavros ros-noetic-mavros-extras ros-noetic-control-toolbox
cd /opt/ros/noetic/lib/mavros
sudo ./install_geographiclib_datasets.sh
sudo chmod 777 /dev/ttyACM0
```

用文件夹中的文件来替换mavros的默认启动文件，从而屏蔽飞控发出的一部分不需要的数据：

```
cd real_ws
sudo cp src/mavros_launch_files/px4_pluginlists.yaml /opt/ros/noetic/share/mavros/launch
sudo cp src/mavros_launch_files/px4.launch /opt/ros/noetic/share/mavros/launch
```

启动mavros，检查是否正常通信：

```
roslaunch mavros px4.launch
```

在另一个终端

```
rostopic hz /mavros/imu/data
```

应该能得到如图所示的50hz imu数据

```
subscribed to [/mavros/imu/data]
average rate: 49.963
        min: 0.014s max: 0.026s std dev: 0.00188s window: 49
average rate: 50.013
        min: 0.014s max: 0.026s std dev: 0.00167s window: 99
average rate: 50.007
        min: 0.014s max: 0.026s std dev: 0.00170s window: 149
average rate: 50.008
        min: 0.014s max: 0.026s std dev: 0.00162s window: 199
average rate: 50.004
        min: 0.014s max: 0.026s std dev: 0.00157s window: 250
average rate: 49.994
        min: 0.014s max: 0.026s std dev: 0.00155s window: 300
average rate: 50.001
        min: 0.014s max: 0.026s std dev: 0.00156s window: 350
average rate: 50.010
        min: 0.014s max: 0.026s std dev: 0.00156s window: 400
average rate: 49.992
        min: 0.014s max: 0.026s std dev: 0.00157s window: 450
average rate: 49.991
        min: 0.014s max: 0.026s std dev: 0.00156s window: 500
average rate: 50.007
```

启动

已经将所有要启动的程序写在一个脚本run.sh中，只需要一键即可启动，但需要先检查各个ros节点的topic是否正确：

首先根据在动捕中创建的刚体名称，修改src/ekf_pose/launch/PX4_vicon.launch中接收的位姿topic

```
<launch>

    <node pkg="ekf" type="ekf" name="ekf" output="screen">
        <remap from="~imu" to="/mavros/imu/data"/>
        <remap from="~bodyodometry" to="/vrpn_client_node/TA/pose"/>
        <remap from="~ekf_odom" to="/vicon_imu_ekf_odom"/>

        <!-- parms -->
        <rosparam file="$(find ekf)/launch/PX4_vio_drone.yaml" command="load" />

        <!--body in IMU frame-->>
        <param name="imu_trans_x" type="double" value="0.0"/>
        <param name="imu_trans_y" type="double" value="0.0"/>
        <param name="imu_trans_z" type="double" value="-0.03"/>

        <!-- Qt -->
        <param name="gyro_cov" type="double" value="0.02"/>
        <param name="acc_cov" type="double" value="0.5"/>
        <!-- Rt -->
        <param name="position_cov" type="double" value="0.01"/>
        <param name="q_rp_cov" type="double" value="0.01"/>
        <param name="q_yaw_cov" type="double" value="0.01"/>

    </node>

</launch>
```

改节点利用EKF扩展卡尔曼滤波，对imu和动捕的数据进行融合估计，我们以此结果作为无人机的里程计信息给到控制器。

随后修改run.sh中启动vrpn的ip为动捕主机的ip

```
roscore & sleep 5;
roslaunch vrpn_client_ros sample.launch server:=192.168.1.102 & sleep 3;
roslaunch mavros px4.launch & sleep 3;
roslaunch ekf PX4_vicon.launch & sleep 3;
roslaunch px4ctrl run_ctrl.launch
#rosbag record /vicon_imu_ekf_odom /debugPx4ctrl
```

`catkin_make` 编译

`source devel/setup.bash`

然后启动即可

`./run.sh`
可通过rqt_graph 来检查各模块间的通讯是否正常。

代码说明

要编写的控制器部分代码位于real_ws/src/px4ctrl/src/linear_control.cpp

```
/*
  compute u.thrust and u.q, controller gains and other parameters are in param_
*/
quadrotor_msgs::Px4ctrlDebug
LinearControl::calculateControl(const Desired_State_t &des,
    const Odom_Data_t &odom,
    const Imu_Data_t &imu,
    Controller_Output_t &u)
{
  /* WRITE YOUR CODE HERE */
      //compute disired acceleration
      Eigen::Vector3d des_acc(0.0, 0.0, 0.0);

      //supposed to be readonly, compute thrust by acc
      u.thrust = computeDesiredCollectiveThrustSignal(des_acc);

      //compute control attitude in the BODY frame
      u.q = Eigen::Quaterniond(1.0,0.0,0.0,0.0);
  /* WRITE YOUR CODE HERE */
```

其中computeDesiredCollectiveThrustSignal函数用来根据加速度计算油门百分比，在实际过程中通过在线估计参数，这部分不需要同学们实现，只需要给定加速度即可。

另外需要计算的是无人机的姿态u.q。

程序提供了debug的接口，通过rostopic向外发送，可通过run.sh中最后一行注释记录rosbag，通过plotjuggler进行后续分析（详见ros仿真中的说明）

```
30  //used for debug
31  debug_msg_.des_p_x = des.p(0);
32  debug_msg_.des_p_y = des.p(1);
33  debug_msg_.des_p_z = des.p(2);
34
35  debug_msg_.des_v_x = des.v(0);
36  debug_msg_.des_v_y = des.v(1);
37  debug_msg_.des_v_z = des.v(2);
38
39  debug_msg_.des_a_x = des_acc(0);
40  debug_msg_.des_a_y = des_acc(1);
41  debug_msg_.des_a_z = des_acc(2);
42
43  debug_msg_.des_q_x = u.q.x();
44  debug_msg_.des_q_y = u.q.y();
45  debug_msg_.des_q_z = u.q.z();
46  debug_msg_.des_q_w = u.q.w();
47
48  debug_msg_.des_thr = u.thrust;
```

参数文件位于real_ws/src/px4ctrl/config/ctrl_param_fpv.yaml，可调节增益参数。

注意事项

1.注意实验安全！！！

2. 四元数、旋转矩阵、欧拉角之间的转换
3. 世界系、机体系的坐标变换
4. 建议在实物控制器测试前确保其它模块功能正常