1.PIBOT的ros中driver

1.1目标

- 串口数据发送与接收
- 订阅`cmd_vel topic`下发至下位机
- 根据下位机的反馈发布`odom topic`和`odom tf`

1.2串口数据发送与接收

```
boost::shared_ptr<Transport> trans;
boost::shared_ptr<Dataframe> frame;
```

```
trans = boost::make_shared<Serial_transport>(bdg.port, bdg.buadrate);
frame = boost::make_shared<Simple_dataframe>(trans.get());
```

> 这里实现一个上位机的Serial_transport和一个Simple_dataframe即可完成

1.3subscribe cmd_vel

```
cmd_vel_sub = nh.subscribe(bdg.cmd_vel_topic, 1000, &BaseDriver::cmd_vel_callback,
this);
```

```
void BaseDriver::cmd_vel_callback(const geometry_msgs::Twist& vel_cmd)
{
    ROS_INFO_STREAM("cmd_vel:[" << vel_cmd.linear.x << " " << vel_cmd.linear.y <<
" " << vel_cmd.angular.z << "]");

    Data_holder::get()->velocity.v_liner_x = vel_cmd.linear.x*100;
    Data_holder::get()->velocity.v_liner_y = vel_cmd.linear.y*100;
    Data_holder::get()->velocity.v_angular_z = vel_cmd.angular.z*100;
```

```
    need_update_speed = true;
}
```

```
void BaseDriver::update_speed()
{
    if (need_update_speed)
    {
        ROS_INFO_STREAM("update_speed");
        need_update_speed = !(frame->interact(ID_SET_VELOCITY));
    }
}
```

> 代码容易理解，订阅消息，回调函数中设置标识，循环中根据标识下发设置指令

1.4publish odom

9. base_link、odom、map关系中有个odom publisher的例子，基本拿过来就可以

```
void BaseDriver::update_odom()
{
    frame->interact(ID_GET_ODOM);

    ros::Time current_time = ros::Time::now();

    float x = Data_holder::get()->odom.x*0.01;
    float y = Data_holder::get()->odom.y*0.01;
    float th = Data_holder::get()->odom.yaw*0.01;

    float vxy = Data_holder::get()->odom.v_liner_x*0.01;
    float vth = Data_holder::get()->odom.v_angular_z*0.01;

    //ROS_INFO("odom: x=%.2f y=%.2f th=%.2f vxy=%.2f vth=%.2f", x, y ,th,
vxy,vth);

    geometry_msgs::Quaternion odom_quat = tf::createQuaternionMsgFromYaw(th);

    //send the transform
    odom_trans.header.stamp = current_time;
    odom_trans.transform.translation.x = x;
    odom_trans.transform.translation.y = y;
    odom_trans.transform.rotation = odom_quat;
    odom_broadcaster.sendTransform(odom_trans);

    //publish the message
    odom.header.stamp = current_time;
    odom.pose.pose.position.x = x;
    odom.pose.pose.position.y = y;
    odom.pose.pose.orientation = odom_quat;
```

```
    odom.twist.twist.linear.x = vxy;
    odom.twist.twist.angular.z = vth;
    odom_pub.publish(odom);
}
```

2.动态PID调节

2.1概述

底层提供了各个电机输入输出的，参见协议ROS机器人底盘(3)-通讯协议

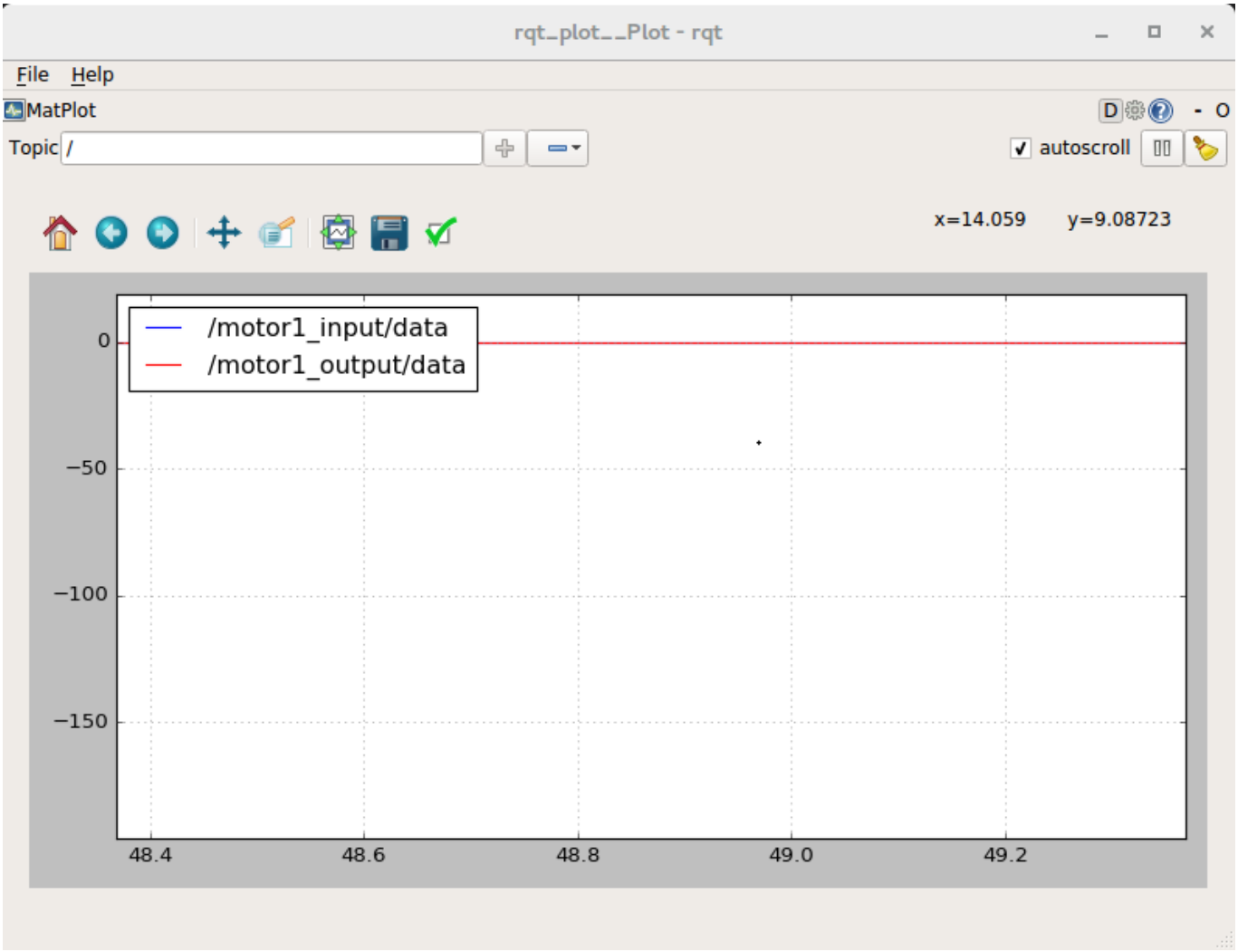
2.2配置

`params.yaml`打开`out_pid_debug_enable`

```
port: /dev/pibot
buadrate: 115200

base_frame: base_link

# topic
cmd_vel_topic: cmd_vel

#pid debug
out_pid_debug_enable: true
```

2.3PID曲线

运行`roslaunch pibot_bringup bringup.launch rosrun rqt_plot rqt_plot /motor1_input /motor1_output`即可展示出实时曲线

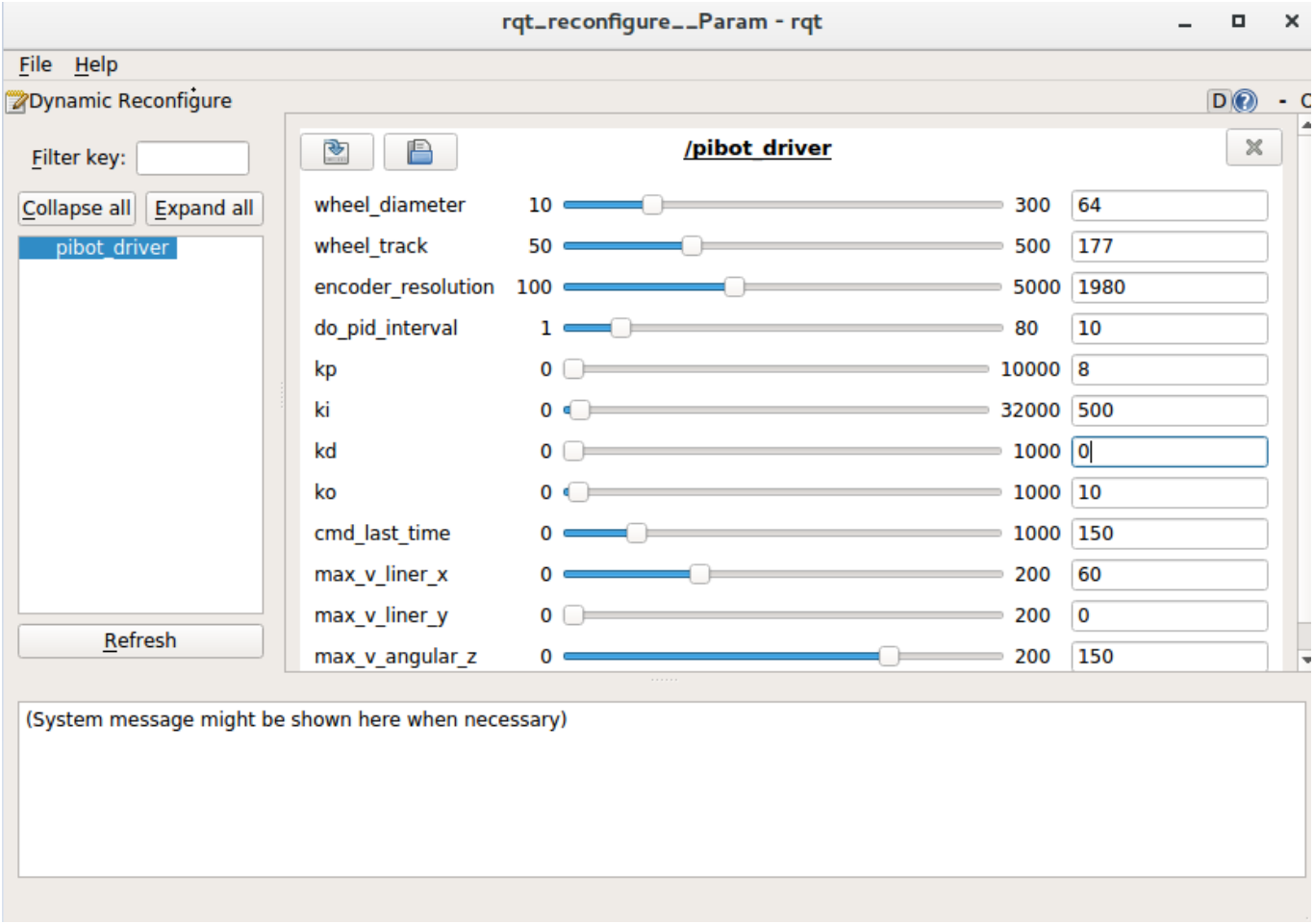


> 控制之前最好先架起小车

为了稳定给出输入直接向`cmd_vel`发消息

```
    x: 0.15
    y: 0.0
    z: 0.0
  angular:
    x: 0.0
    y: 0.0
    z: 0.0" -r 20
```

观察曲线变化输入输出基本一致输出振幅较小即可；否则打开配置页面 `rosrun rqt_reconfigure rqt_reconfigure`



调整PID参数，重复上一步操作直到输出较为稳定即可