

基于 zigbee 的飞行器数据互传方案

程鹏

1. 需求分析

多机器人系统之间要协同完成任务，首先要解决的是多个机器人之间的数据传输问题。对于我们的四旋翼飞行器系统也不例外，我们首先要实现多个飞行器之间的传输数据功能，才能够实现他们的分布式控制，比如，每个四旋翼可以获得邻居的高度，位置，速度等信息。在此，我只完成了他们之间的数据通信的功能，关于数据传输质量，以及针对数据丢失的处理并没有给出，希望能够在之后的工作中加以改进。

2. 系统设计

数据之间的通信我们采用 zigbee 协议。其优点是低功耗、低成本、安全、可靠，并具有智能化组网和信息路由的能力。

飞行器的飞控采用 pixhawk，自行设计的程序运行在 odroid 上。我们希望实现 odroid 读取 pixhawk 上的传感器的测量数据（如高度、GPS、速度等），然后通过 zigbee 模块发送到网络中的其他节点；同时，zigbee 模块能接收来自网络中其他节点的数据，然后传给 odroid。这种双向的通信模式，以及单个系统框图如图 2.1 所示。

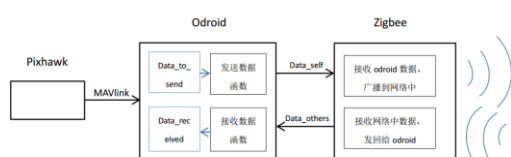


图 2.1 单个系统结构框图

整个系统则由多个这样的单个系统构成。在此单个系统中，我们需要编写的软件部分包括 Odroid 端的串口发送和接收函数，以及 Zigbee 端的串口读写函数和组网程序。

3. 数据互传

3.1. Zigbee 端

Zigbee 端的软件是基于 zigbee 协议栈的程序。Zigbee 协议栈中的每一层都有很多操作要执行，因此对于整个协议栈来说，会有很多的并发操作要执行。协议栈中的每一层都设计了一个事件处理函数，用来处理与这一层操作相关的各种事件。我们要编写串口的读写函数，就要在应用层修改相应的 Zigbee 协议栈代码。

Zigbee 协议栈在设计任务调度程序时，OSAL 只采用了轮询任务调度队列的方法进行任务调度管理。

当有串口数据输入时，产生一个 CMD_SERIAL_MSG 的事件，应用层的事件处理函数会检测到这个事件的发生，然后执行相应的处理函数（即把串口数据包里接收到的串口数据广播发送到网络中）。当接收到网络中的数据时，会产生一个 AF_INCOMING_CMD 的事件，应用层的事件处理函数会检测到这个事件的发生，然后执行相应的处理函数（即将接收到的数据通过串口发送出去，如果串口与 Odroid 相连，则会传给 Odroid）。

3.2. Odroid 端

Odroid 要实现两个功能，一个是接收 zigbee 模块从网络中获得的其他节点的数据，从而用于控制；另一个是读取 Pixhawk 内的传感器测量数据，发送给 Zigbee 的串口，然后发送给网络中的其他节点。

Python 中对串口的操作要用到一个 pySerial 的模块。发送的代码比较好写，只需要将读取到的数据发送出去，因为能始终确保有数据从 Pixhawk 中读到，程序会一直向串口写数据直到结束。而接收程序相对麻烦些，因为如果是阻塞的读串口，那当没有数据进来的时候，程序干不了其他任何事情，CPU 全都用来读串口了，因此，我们需要设定一定的数据读取时间，超过这个时间常数则停止读数，这个时间

常数由 timeout 参数设定。

其次，为了能确定一帧数据的结束，我们在每一帧的数据结尾加上回车字符（'\n'）。发送数据的时候，在 Zigbee 端协议栈的程序里加入回车字符，这样接收端就可以准确的识别每一帧的结尾，提高了数据传输质量。

3.3. Odroid 与 Pixhawk 通信

Odroid 与 Pixhawk 之间通过 MAVLink 协议通信，需要安装相应的软件包。

在 ubuntu 下安装 DroneKit，输入一下命令：

```
sudo apt-get install python-pip
python-numpy python-opencv
python-serial python-pyparsing
python-wxgtk2.8
```

```
sudo pip install droneapi
```

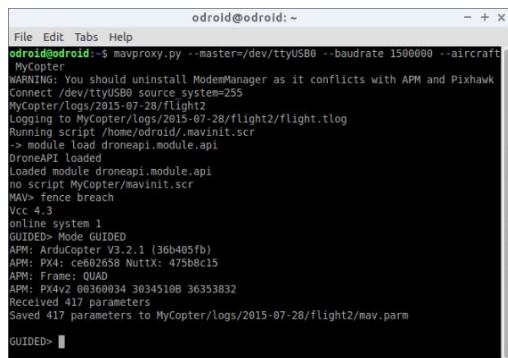
Dronekit 是 MAVProxy 中调用的一个模块，加载 Dronekit 最好的方式是把它加入到 mavinit.scr 文件中，输入如下命令：

```
echo "module load
droneapi.module.api" >>
~/mavinit.scr
```

打开 MAVProxy，运行命令：

```
mavproxy.py --master=/dev/ttyUSB0
--baudrate 1500000 --aircraft
MyCopter
```

其中，ttyUSB0 为连接 Pixhawk 和 Odroid 的串口号，需要查询获得。如图 3.1 所示，为运行成功的界面。



```
odroid@odroid:~$ mavproxy.py --master=/dev/ttyUSB0 --baudrate 1500000 --aircraft
MyCopter
WARNING: You should uninstall ModeManager as it conflicts with APM and Pixhawk
Connect /dev/ttyUSB0 source system=255
MyCopter/logs/2015-07-28/flight2
Logging to MyCopter/logs/2015-07-28/flight2/flight.tlog
Running script /home/odroid/mavinit.scr
-> module load droneapi.module.api
DroneAPI loaded
Loaded module droneapi.module.api
no script MyCopter/mavinit.scr
MAV> fence breach
Vcc 4.9
online system 1
GUIDED> Mode GUIDED
APM: ArduCopter V3.2.1 (30b405fb)
APM: PX4: ce002658 NutTX: 475b8c15
APM: Frame: QUAD
APM: PX4v2 00360034 30345108 36353832
Received 417 parameters
Saved 417 parameters to MyCopter/logs/2015-07-28/flight2/mav.parm
GUIDED> █
```

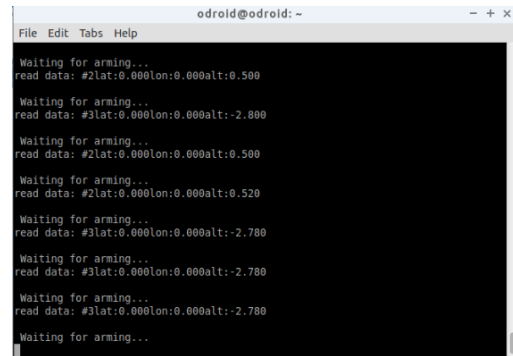
图 3.1 运行 MAVProxy 并加载 Dronekit 成功的界面

在 MAVProxy 窗口下运行.py 脚本的命令是：

```
api start xx.py
```

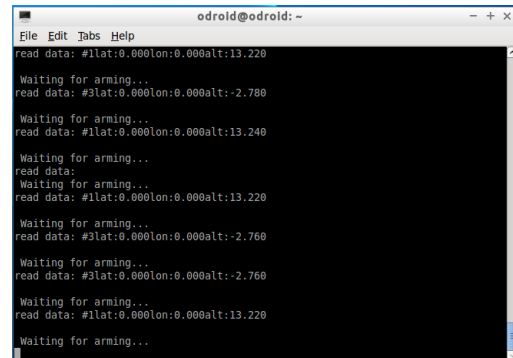
4. 实验结果

三个 zigbee 模块分别连接 Odroid 和 Pixhawk，分别向其他节点发送经度、纬度和高度数据，同时接收数据，并打印显示。图 4.1-图 4.3 分别显示了三个节点接收到其他节点发来的数据，并在 Odroid 终端内打印显示。



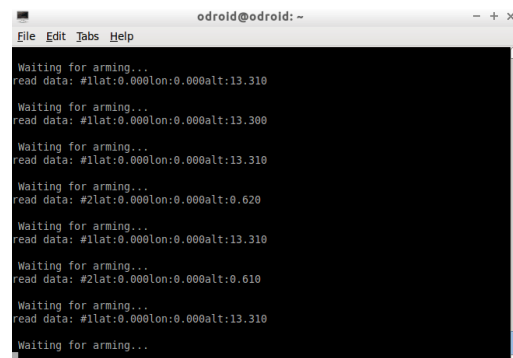
```
odroid@odroid:~$
Waiting for arming...
read data: #2lat:0.000lon:0.000alt:0.500
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.800
Waiting for arming...
read data: #2lat:0.000lon:0.000alt:0.500
Waiting for arming...
read data: #2lat:0.000lon:0.000alt:0.520
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.780
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.780
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.780
Waiting for arming...
```

图 4.1 #1 节点接收到其他节点的数据



```
odroid@odroid:~$
read data: #1lat:0.000lon:0.000alt:13.220
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.780
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.240
Waiting for arming...
read data:
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.220
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.760
Waiting for arming...
read data: #3lat:0.000lon:0.000alt:-2.760
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.220
Waiting for arming...
```

图 4.2 #2 节点接收到其他节点的数据



```
odroid@odroid:~$
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.310
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.300
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.310
Waiting for arming...
read data: #2lat:0.000lon:0.000alt:0.620
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.310
Waiting for arming...
read data: #2lat:0.000lon:0.000alt:0.610
Waiting for arming...
read data: #1lat:0.000lon:0.000alt:13.310
Waiting for arming...
```

图 4.3 #2 节点接收到其他节点的数据

实验结果表明，基于 Zigbee+Odroid+Pixhawk 的飞行器系统间成功的建立了通信连接并完成数据通信功能。