AGNSS 辅助定位

1. 学习目标

本次课程我们主要学习使用 jetson nano 和 GPS 模块和 agnss 服务器实现弱信号下位置信息读取解析。

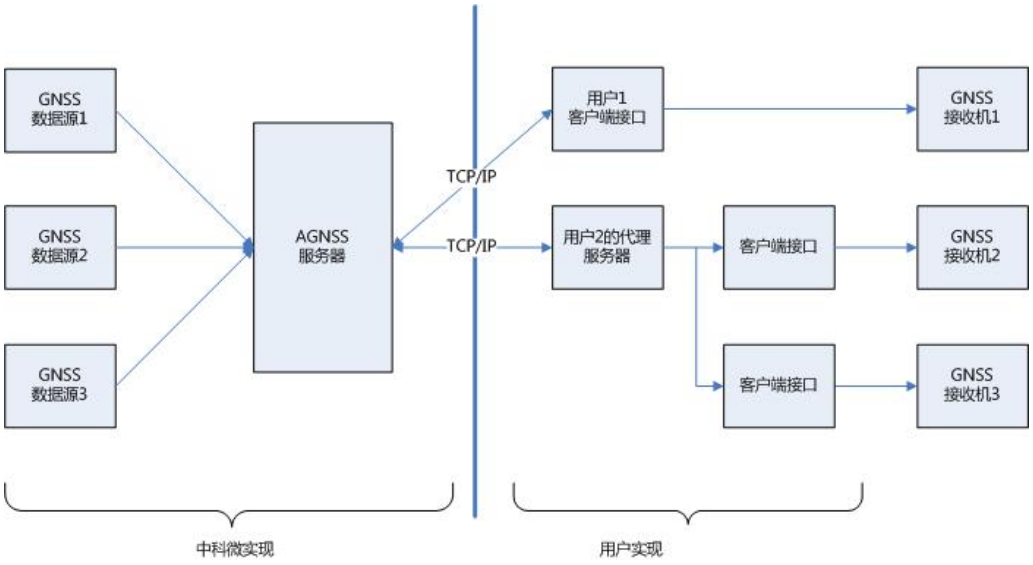
2. AGNSS 说明

2.1. 为什么要用 AGNSS

- 自主式GNSS接收机定位的条件包括：
  - 捕获并跟踪卫星信号，解析时间
  - 从卫星获取电文
- 强信号环境，自主式GNSS接收机可以在30秒左右冷启动定位；但是在弱信号环境，无外部辅助的接收机捕获卫星很慢，很难从卫星获取电文，因此需要很久才能定位，甚至无法定位。
- AGNSS可以为接收机提供定位必需的辅助信息，比如电文，粗略位置和时间。不管是在强信号还是弱信号环境，这些信息可以显著的缩短首次定位时间。

2.2. AGNSS 解决方案

- AGNSS服务器从多个GNSS数据源获取并管理AGNSS辅助信息。服务器时刻监听并响应客户端的AGNSS请求（需要用户名和密码）。
- 用户通过TCP/IP协议从AGNSS服务器获取辅助信息，获取到辅助信息可以直接传输给GNSS接收机。
- 用户也可以建立自己的代理服务器。



2.3. AGNSS 流程

- 连接AGNSS服务器

  –服务器的地址为121.41.40.95(域名：www.gnss-aide.com)

  –端口号为2621

- 发送AGNSS请求

  –请求语句：(用户名和密码字段是必需项)

  –user=pm@yahboom.com;pwd=yahboom;cmd=full;gnss=gps+bd;lat=60.0;lon=55.0;alt=0;

- 获取AGNSS辅助信息
- 把AGNSS辅助信息发送给接收机

2.4. AGNSS 请求参数

- 用户端发送请求到AGNSS服务器，请求语句的格式如下

  –请求语句是多组key=value;的组合，如：key=value;key=value;

- 示例：

  user=pm@yahboom.com;pwd=yahboom;cmd=full;gnss=gps+bd;lat=6

0.0;lon=55.0;alt=0;

- 具体的key和value定义如下表

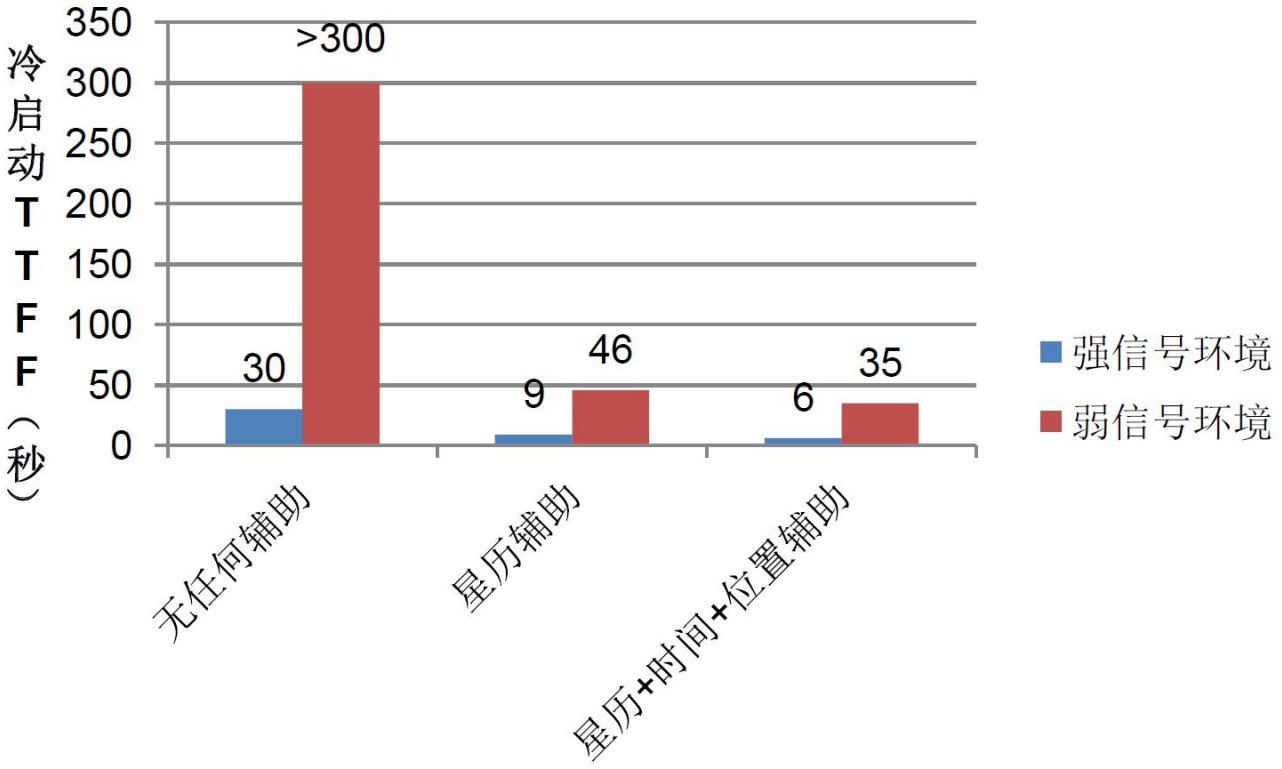
关键字 **(Key)**	取值 **(value)**	可选性	备注
user	字符串	必须	用户名。强烈建议用户名为一个有效的邮箱地址，重要的 AGNSS 服务器维护信息将会发送到该邮箱。
pwd	字符串	必须	用户密码
gnss	字符串	可选	用逗号隔开的 GNSS 列表，目前支持 GPS。有效的取值有：gps,bds,glo ”gnss=gps;”表示请求 GPS 辅助信息； gnss=gps,bds;”表示请求 GPS 和 BDS 辅助信息；
cmd	字符串	可选	full:全部信息，包括星历，估计的时间和位置 eph:仅提供星历信息 aid:辅助时间、位置等信息 此项若不填，默认为 full
lat	数值	可选	用户位置纬度的估计值。纬度的单位：度。取值范围是-90~90 度。两者位置辅助格式，经纬高格式和 ECEF 格式，二选一。有效的经纬高位置辅助格式是”lat=30;lon=120.3;alt=100;”三个字段都必须完整。
lon	数值	可选	用户位置经度的估计值。经度的单位：度。取值范围是-180~180 度。
alt	数值	可选	用户位置高度的估计值。单位:米。
x	数值	可选	用户位置（ECEF 坐标系下的 X,Y,Z）的估计值。单位:米。有效的 ECEF 位置辅助格式是”x=30000;y=1111120.3;z=3345100;”三个字段都必须完整。
y	数值	可选	用户位置（ECEF 坐标系下的 X,Y,Z）的估计值。单位:米。
z	数值	可选	用户位置（ECEF 坐标系下的 X,Y,Z）的估计值。单位:米。
pacc	数值	可选	用户位置的准确度。单位为米。

2.5. 服务器返回信息

- AGNSS服务器返回的数据示例：数据头+辅助数据内容
- 二进制数据就是GNSS接收机所需的辅助数据，这些二进制数据中都自带了数据校验。二进制数据格式参考中科微的接收机协议规范。
- 如果把数据头也发送给了GNSS接收机，不会对GNSS接收机产生影响。

2.6. AGNSS 性能对比

- 相比普通的独立式GNSS接收机，AGNSS接收机在TTFF性能上具有显著的提升，尤其是在弱信号条件下。



2.7. 注意事项

- 粗略位置辅助需要用户端通过其它方式获取，比如

  –GSM/GPRS/3G通信模块，这些模块都可以利用CELL ID的方式获取当前粗略的位置

  –WiFi等其它无线模块，也可以粗略定位

- 粗略位置的精度要求在15km以内，错误的位置辅助会影响接收机的性能
- 如果无法获取粗略位置，在AGNSS请求语句中忽略位置的字段（lat,lon,alt,x,y,z），接收机会自动选择历史定位的有效位置
- 没有必要把GNSS接收机自己输出的位置作为粗略位置

2.8. 什么时候需要 AGNSS

- 不需要每次开机都从服务器下载，节省流量

  –中科微的芯片内部有电池备份SRAM，以及永久备份FLASH，都可以自

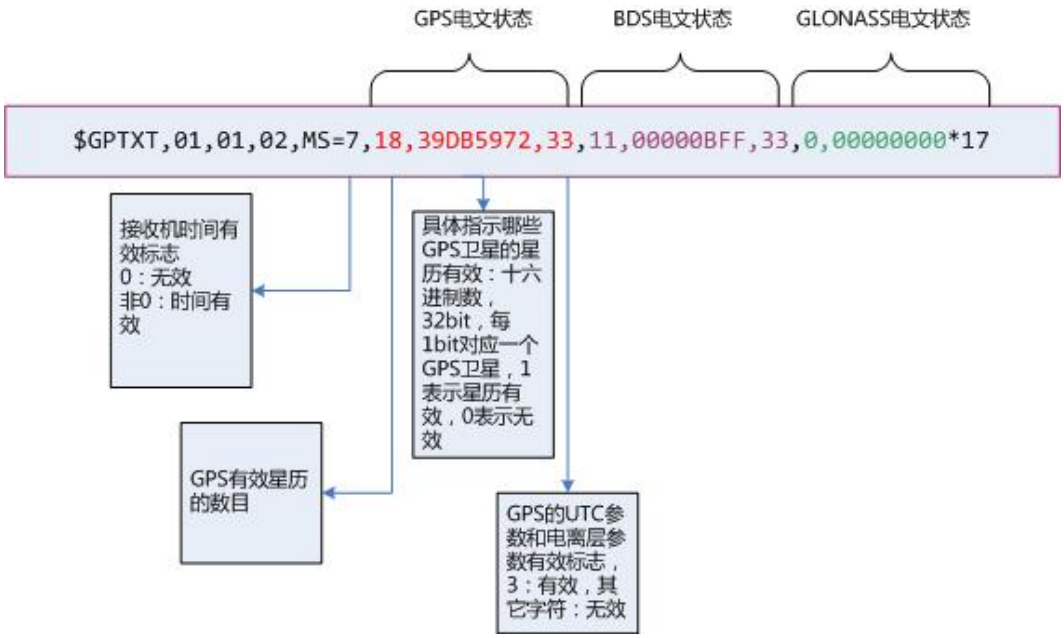
动保存接收到星历数据等

–芯片在正常工作中，不断的从卫星下载最新的星历数据

- 通过查询接收机的状态，决定是否需要从服务器下载AGNSS数据

–接收机可以输出电文状态语句（默认不输出，需要配置才输出）

2.9. 电文状态语句介绍



示例：

$GPTXT,01,01,02,MS=7,18,39DB5972,33,11,00000BFF,33,0,00000000*17

7:时间有效性标志。0 为无效，2 为外部输入，7 为有效。

18：星历有效的 GPS 卫星个数。

39DB5972：GPS 卫星星历有效标志（无符号整型数，十六进制显示。每颗卫星占一位，最低位为 1 号卫星，以此类推。1 为有效，0 为无效）

33：GPS 的 UTC 和 ION 信息有效标志（无符号短整型数，十六进制显示。前一位为 UTC 信息有效标志，后一位为 ION 信息有效标志。3 为有效，2 为过期，1 为不健康，0 为无效）

11：星历有效的 BDS 卫星个数。

00000BFF：BDS 卫星星历有效标志（无符号整型数，十六进制显示。每颗卫星占一位，最低位为 1 号卫星，以此类推。1 为有效，0 为无效）

33：BDS 的 UTC 和 ION 信息有效标志。信息规制同 GPS 的 UTC 和 ION 信息有效标志。

0：星历有效的 GLN 卫星个数。

00000000：GLN 卫星星历有效标志。与 GPS 卫星星历有效标志一致。

- 该语句输出的是当前接收机内部的时间+电文状态。
- 可以发送命令$PCAS03,,,,,,,,,,1*1F，每秒输出一次电文状态语句
- 可以发送命令$PCAS03,,,,,,,,,,0*1E，停止输出电文状态语句
- 注意：每条语句后面都必须\r\n结尾(0x0D,0x0A)，语句中有11个逗号
- 如果时间标志有效（非0），且有效星历数目很多（大于8个），就不必下载AGNSS星历了。

3. 课前准备

3.1. 接线

GPS 模块采用的是 UART 通讯或 USB 通讯，这里以 USB 通讯为例。

使用 type-c 线连接 jetson nano 和 GPS 模块，运行命令 ls /dev | grep 'ttyUSB' ，可以看到识别到 GPS 模块为 USB0

```
jetson@jetson-desktop:~$ ls /dev | grep 'ttyUSB'
ttyUSB0
```

3.2. 申请百度地图 ak

请看文档 百度地图 api 申请教程

4. 程序

本次课程的程序请参考：GPS-agnss.py

初始化 USB：

```
ser = serial.Serial("/dev/ttyUSB0", 9600)
```

资料 ak 需要填上自己申请的 ak 值，然后就能通过百度地图获得当前粗略的经纬度信息

```
ak = "填上自己申请的ak，请看文档 百度地图api申请教程"
baiduUrl = "http://api.map.baidu.com/location/ip?ak=%s&coor=bd0911" % (ak)
req = requests.get(baiduUrl)
content = req.text
baiduAddr = json.loads(content)
city = baiduAddr["content"]["address_detail"]["city"]
maplng = baiduAddr["content"]["point"]["x"]
maplat = baiduAddr["content"]["point"]["y"]
```

这里把百度地图获取到的粗略经纬度信息发送给服务器，我们使用的登录账号是亚博官方的账号，获取完成之后把整个包发送给模块

```
addr = "121.41.40.95"
port = 2621
message = "user=pm@yahboom.com;pwd=yahboom;cmd=full;gnss=gps+bd;lat=%s;lon=%s;" % (maplat,maplng)
socket.setdefaulttimeout(4)
client = socket.socket()
client.connect((addr, port))
client.send(message.encode())
reply_data = ""
print("GPS Agnss start")
while True:
    current_reply = client.recv(1024)
    current_reply = str(current_reply)
    if len(current_reply) == 0:
        break
    else:
        reply_data += current_reply
        #print(reply_data)
ser.write(reply_data)
print("GPS Agnss success")
```

位置信息获取和解析函数，下图中在位置信息中筛选出 GNGGA 开头的位置信息，然后解析出数据存到各个全局变量中。

```
def GPS_read():
        global utctime
        global lat
        global ulat
        global lon
        global ulon
        global numSv
        global msl
        global cogt
        global cogm
        global sog
        global kph
        global gps_t
        if ser.inWaiting():
            if ser.read(1) == b'G':
                if ser.inWaiting():
                    if ser.read(1) == b'N':
                        if ser.inWaiting():
                            choice = ser.read(1)
                            if choice == b'G':
                                if ser.inWaiting():
                                    if ser.read(1) == b'G':
                                        if ser.inWaiting():
                                            if ser.read(1) == b'A':
                                                #utctime = ser.read(7)
                                                GGA = ser.read(70)
                                                GGA_g = re.findall(r"\w+(?=,)|(?<=,)\w+", str(GGA))
                                                # print(GGA_g)
                                                if len(GGA_g) < 13:
                                                    print("GPS no found")
                                                    gps_t = 0
                                                    return 0
                                                else:
                                                    utctime = GGA_g[0]
                                                    # lat = GGA_g[2][0]+GGA_g[2][1]+'°'+GGA_g[2][2]+GGA_g[2][3]+'.'+GGA_g[3]+'\''
                                                    lat = "%.8f" % Convert_to_degrees(str(GGA_g[2]), str(GGA_g[3]))
                                                    ulat = GGA_g[4]
                                                    # lon = GGA_g[5][0]+GGA_g[5][1]+GGA_g[5][2]+'°'+GGA_g[5][3]+GGA_g[5][4]+'.'+GGA_g[6]+'\''
                                                    lon = "%.8f" % Convert_to_degrees(str(GGA_g[5]), str(GGA_g[6]))
                                                    ulon = GGA_g[7]
                                                    numSv = GGA_g[9]
                                                    msl = GGA_g[12]+'.'+GGA_g[13]+GGA_g[14]
                                                    #print(GGA_g)
                                                    gps_t = 1
                                                    return 1
```

```
def Convert_to_degrees(in_data1, in_data2):
    len_data1 = len(in_data1)
    str_data2 = "%05d" % int(in_data2)
    temp_data = int(in_data1)
    symbol = 1
    if temp_data < 0:
        symbol = -1
    degree = int(temp_data / 100.0)
    str_decimal = str(in_data1[len_data1-2]) + str(in_data1[len_data1-1]) + str(str_data2)
    f_degree = int(str_decimal)/60.0/100000.0
    # print("f_degree:", f_degree)
    if symbol > 0:
        result = degree + f_degree
    else:
        result = degree - f_degree
    return result
```

同样的方法获取了 GNVTG 的航向信息并解析。

```
elif choice == b'V':
    if ser.inWaiting():
        if ser.read(1) == b'T':
            if ser.inWaiting():
                if ser.read(1) == b'G':
                    VTG = ser.read(40)
                    VTG_g = re.findall(r"\w+(?=,)|(?<=,)\w+", str(VTG))
                    cogt = VTG_g[0]+'.'+VTG_g[1]+'T'
                    if VTG_g[3] == 'M':
                        cogm = '0.00'
                        sog = VTG_g[4]+'.'+VTG_g[5]
                        kph = VTG_g[7]+'.'+VTG_g[8]
                    elif VTG_g[3] != 'M':
                        cogm = VTG_g[3]+'.'+VTG_g[4]
                        sog = VTG_g[6]+'.'+VTG_g[7]
                        kph = VTG_g[9]+'.'+VTG_g[10]
                    #print(kph)
```

解析后的数据循环打印

```
if GPS_read():
    print("**********************")
    print('UTC Time:'+utctime)
    print('Latitude:'+lat+ulat)
    print('Longitude:'+lon+ulon)
    print('Number of satellites:'+numSv)
    print('Altitude:'+msl)
    print('True north heading:'+cogt+'°')
    print('Magnetic north heading:'+cogm+'°')
    print('Ground speed:'+sog+'Kn')
    print('Ground speed:'+kph+'Km/h')
    print("**********************")
```

5. 运行程序

终端输入 sudo python2 GPS-agnss.py 运行程序。

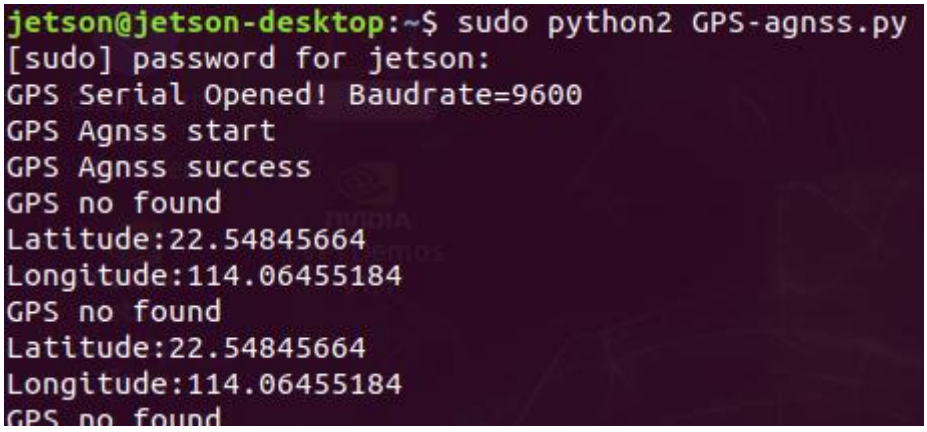
6. 实验现象

注意，辅助定位下 jetson nano 必须联网。

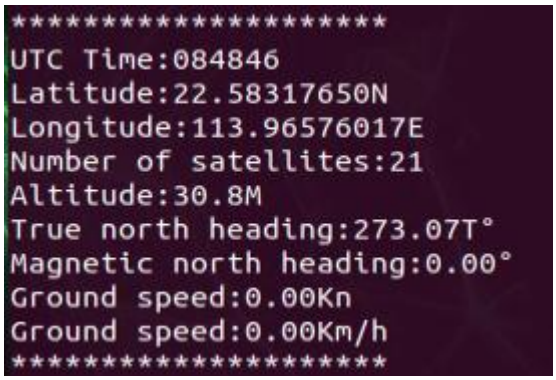
模块在弱信号下通电后，开始初始化 USB，初始化成功显示“GPS Serial Opened! Baudrate=9600”，否则显示“GPS Serial Open Failed!”，如果错误需要检查接线或 USB 端口。

之后显示"GPS Agnss start"开始将辅助定位信息发送给服务器，发送完成之后显示"GPS Agnss success"

在发送之后的一段时间内还未读取到 GPS 信号，此时显示"GPS no found"并且打印百度地图读取到的粗略经纬度信息。



过一段时间识别到 GPS 之后，识别会循环打印位置和航向信息。



按 Ctrl+C 退出信息读取。