GPS module parses location information

1. Learning Objectives

In this course, we mainly learn to use Jetson nano and GPS module to read and parse location information.

2. Preparation before class

The GPS module adopts UART communication or USB communication. Here we take USB communication as an example.

Use the type-c cable to connect the jetson nano and the GPS module, run the command ls /dev | grep 'ttyUSB', you can see that the GPS module is identified as USB0

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

3. Procedure

For the program of this course, please refer to: GPS.py

Initialize USB:

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

Location information acquisition and parsing function. In the following figure, the location information at the beginning of GNGGA is filtered out in the location information, and then the data is parsed and stored in each global variable.

```
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
        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))
                                                if len(GGA_g) < 15:
                                                    print("GPS no found")
                                                    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]+'\''
                                                    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]+'\''
                                                    ulon = GGA_g[7]
                                                    numSv = GGA_g[9]
                                                    msl = GGA_g[12]+'.'+GGA_g[13]+GGA_g[14]
                                                    #print(GGA_g)
                                                    return 1
```

In the same way, the heading information of GNVTG is obtained and parsed.

```
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)
```

The parsed data is printed in a loop

```
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("**********************")
```

4.Run code

Enter sudo python3 GPS.py in the terminal to run the program.

5. Experimental phenomenon

After the module is powered on, it takes about 32s to start, and then the serial port print status light on the module will continue to flash, and data can be received normally at this time.

After the program runs, it starts to initialize the USB. If the initialization is successful, it will display "GPS Serial Opened! Baudrate=9600", otherwise it will display "GPS Serial Open Failed!". If there is an error, you need to check the wiring or USB port, and then print the position and heading information cyclically.

```
jetson@jetson-desktop:~$ sudo python3 GPS.py
GPS Serial Opened! Baudrate=9600
**********************
UTC Time:025000
Latitude:22°34'98368"N
Longitude:11°35'89122"E
Number of satellites:19
Altitude:73.4M
True north heading:269.52T°
Magnetic north heading:0.00°
Ground speed:0.00Kn
Ground speed:0.00Km/h
**********************
**********************
UTC Time:025001
```

Press Ctrl+C to exit information reading.

```
**********************
UTC Time:025029
Latitude:22°34'98351"N
Longitude:11°35'89116"E
Number of satellites:17
Altitude:73.3M
True north heading:269.52T°
Magnetic north heading:0.00°
Ground speed:0.00Kn
Ground speed:0.00Km/h
**********************
^CGPS serial Close!
jetson@jetson-desktop:~$
```

Note that the module antenna needs to be outdoors, otherwise the GPS signal may not be searched, and "GPS no found" will be printed when no signal is found.