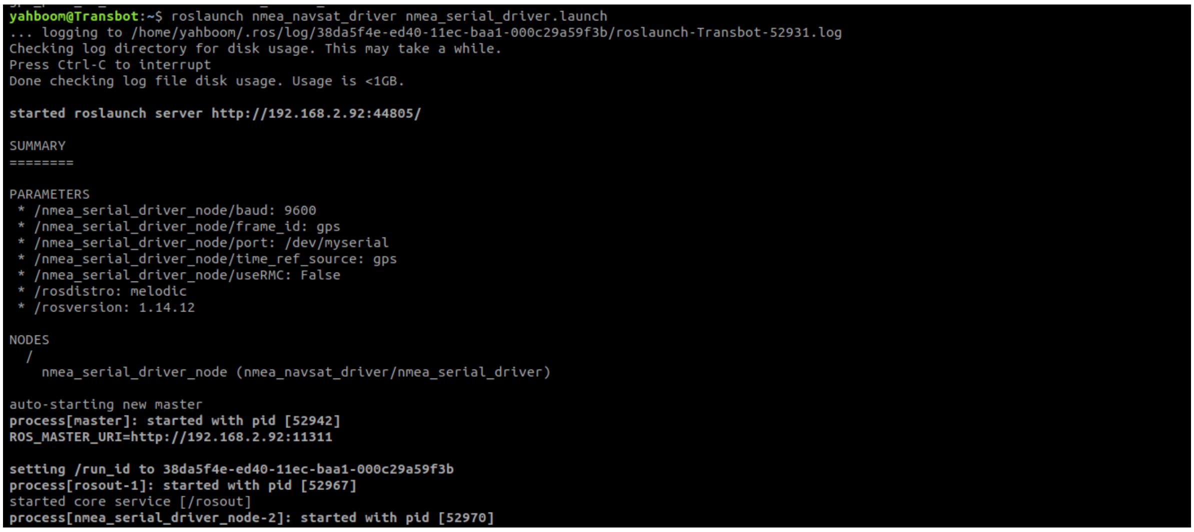
Read GPS module data

This function is to obtain longitude, latitude and altitude data by reading GPS module data and parsing GPS data through the terminal.

1. Read GPS data through the terminal

terminal input:

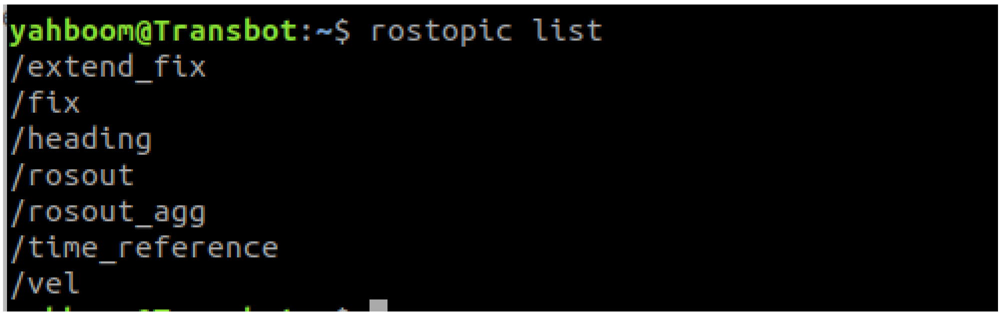
```
roslaunch nmea_navsat_driver nmea_serial_driver.launch
```



```
yahboom@Transbot:~$ roslaunch nmea_navsat_driver nmea_serial_driver.launch
... logging to /home/yahboom/.ros/log/38da5f4e-ed40-11ec-baa1-000c29a59f3b/roslaunch-Transbot-52931.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.2.92:44805/

SUMMARY
========

PARAMETERS
 * /nmea_serial_driver_node/baud: 9600
 * /nmea_serial_driver_node/frame_id: gps
 * /nmea_serial_driver_node/port: /dev/myserial
 * /nmea_serial_driver_node/time_ref_source: gps
 * /nmea_serial_driver_node/useRMC: False
 * /rosdistro: melodic
 * /rosversion: 1.14.12

NODES
  /
    nmea_serial_driver_node (nmea_navsat_driver/nmea_serial_driver)

auto-starting new master
process[master]: started with pid [52942]
ROS_MASTER_URI=http://192.168.2.92:11311

setting /run_id to 38da5f4e-ed40-11ec-baa1-000c29a59f3b
process[rosout-1]: started with pid [52967]
started core service [/rosout]
process[nmea_serial_driver_node-2]: started with pid [52970]
```

Then we look at the topic data, the terminal input:

```
rostopic list
```



```
yahboom@Transbot:~$ rostopic list
/extend_fix
/fix
/heading
/rosout
/rosout_agg
/time_reference
/vel
```

Regarding the data on these topics of GPS, here is an explanation.

topic	type	Description
/extend_fix	gps_common/GPSFix	GPSFix messages contain GPS satellite status and positioning information
/fix	sensor_msgs/NavSatFix	GPS location information
/time_reference	sensor_msgs/TimeReferencd	GPS time information
/vel	geometry_msgs/TwistStamped	GPS speed information

For each topic message type, please refer to the official website URL,

gps_common/GPSFix Documentation (ros.org)

sensor_msgs/NavSatFix Documentation (ros.org)

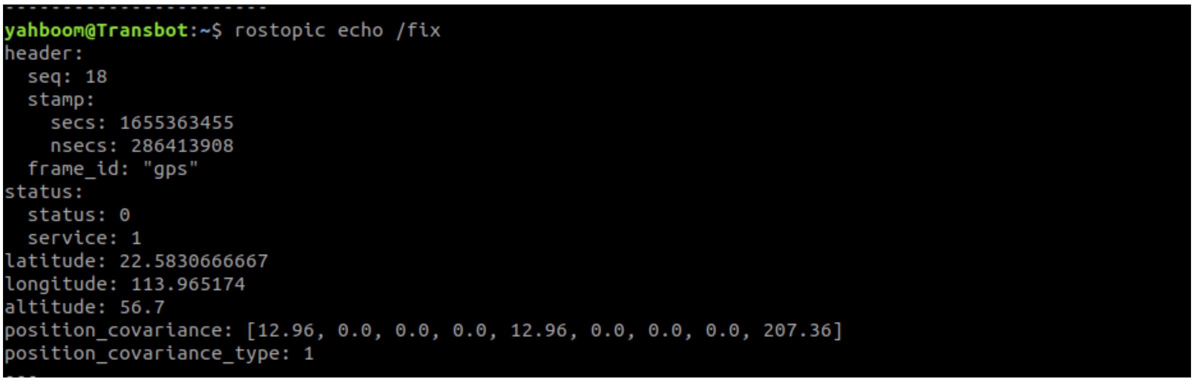
sensor_msgs/TimeReference Documentation (ros.org)

geometry_msgs/TwistStamped Documentation (ros.org)

We can print out these topic messages on the terminal, and what we get is the GPS data. Take printing /fix as an example, the terminal input

```
rostopic echo /fix
```

The terminal will print out the following data,



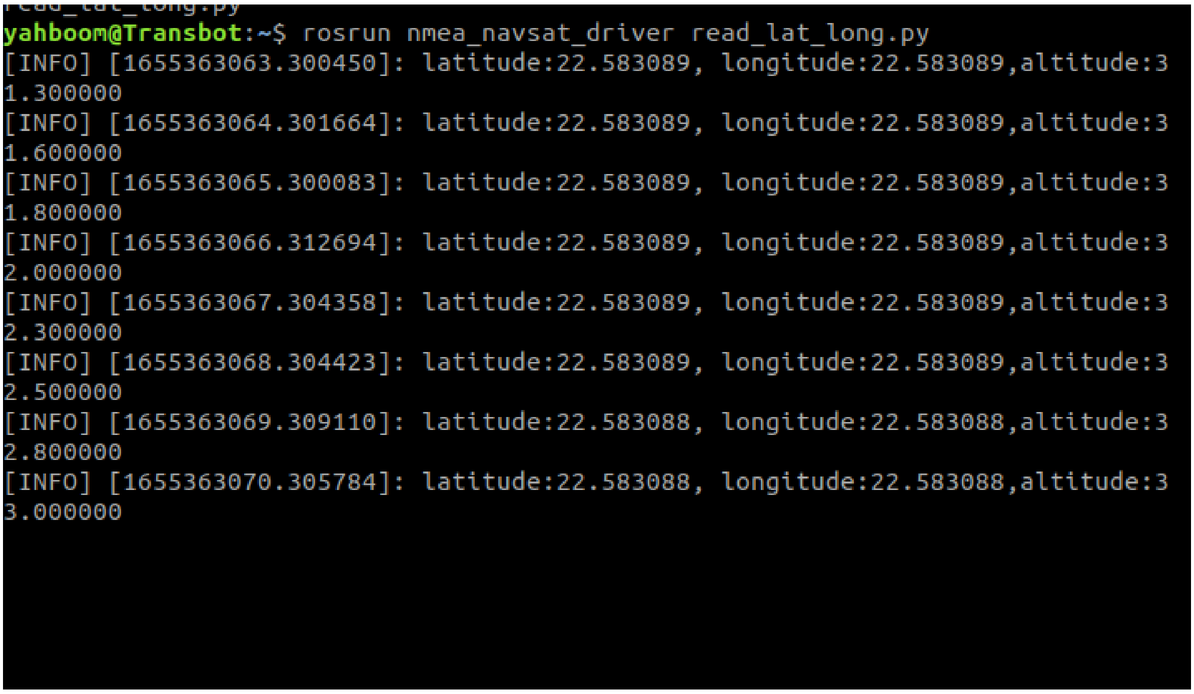
```
---------------------------
yahboom@Transbot:~$ rostopic echo /fix
header:
  seq: 18
  stamp:
    secs: 1655363455
    nsecs: 286413908
  frame_id: "gps"
status:
  status: 0
  service: 1
latitude: 22.5830666667
longitude: 113.965174
altitude: 56.7
position_covariance: [12.96, 0.0, 0.0, 0.0, 12.96, 0.0, 0.0, 0.0, 207.36]
position_covariance_type: 1
```

Among them, latitude, longitude, altitude represent latitude, longitude and altitude respectively.

2、Read the latitude, longitude and altitude of GPS data

terminal runs,

```
roslaunch nmea_navsat_driver nmea_serial_driver.launch
rosrun nmea_navsat_driver read_lat_long.py
```



```
read_lat_long.py
yahboom@Transbot:~$ rosrun nmea_navsat_driver read_lat_long.py
[INFO] [1655363063.300450]: latitude:22.583089, longitude:22.583089,altitude:3
1.300000
[INFO] [1655363064.301664]: latitude:22.583089, longitude:22.583089,altitude:3
1.600000
[INFO] [1655363065.300083]: latitude:22.583089, longitude:22.583089,altitude:3
1.800000
[INFO] [1655363066.312694]: latitude:22.583089, longitude:22.583089,altitude:3
2.000000
[INFO] [1655363067.304358]: latitude:22.583089, longitude:22.583089,altitude:3
2.300000
[INFO] [1655363068.304423]: latitude:22.583089, longitude:22.583089,altitude:3
2.500000
[INFO] [1655363069.309110]: latitude:22.583088, longitude:22.583088,altitude:3
2.800000
[INFO] [1655363070.305784]: latitude:22.583088, longitude:22.583088,altitude:3
3.000000
```

The data printed by the terminal is the latitude, longitude and altitude of the current GPS module, look at the source code, read_lat_long.py

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
import rospy
from sensor_msgs.msg import NavSatFix

def GPSCallback(msg):
    rospy.loginfo("latitude:%0.6f, longitude:%0.6f,altitude:%0.6f",
msg.latitude, msg.latitude,msg.altitude)
def GPS_subscriber():
    rospy.init_node('GPS_subscriber', anonymous=True)# ROS节点初始化
    rospy.Subscriber("/fix", NavSatFix, GPSCallback)
    rospy.spin()# 循环等待回调函数
if __name__ == '__main__':
    GPS_subscriber()
```

The program subscribes to the data of the /fix topic, then parses it in the callback function, and finally prints it to the terminal.