

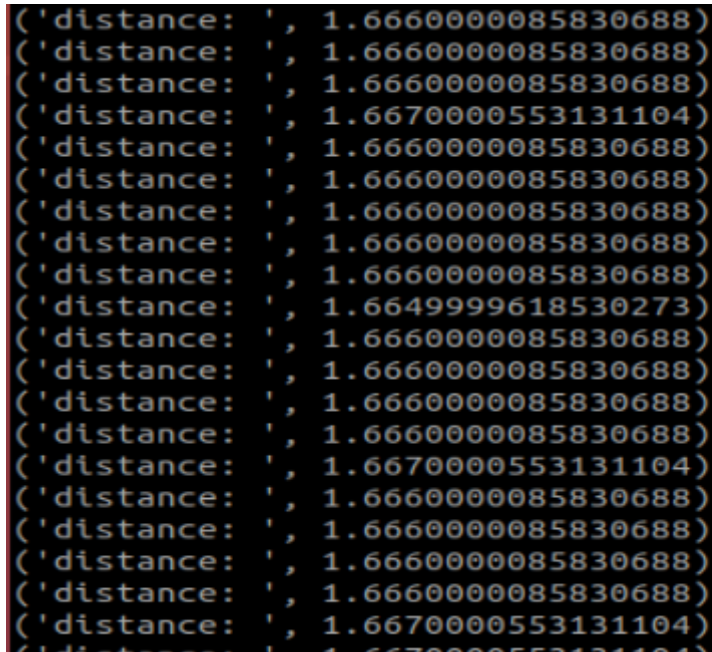
## 3、Get distance information\_ROS

### 1、Start

Input following command in terminal

```
roslaunch ydlidar_ros_driver SDM15.launch
roslaunch ydlidar_ros_driver Get_Distance.py
```

After run successfully , the distance will be printed at the terminal.



A terminal window with a black background and yellow text. It displays a series of 20 lines, each representing a distance measurement. The format is ('distance: ', 1.6660000085830688) for most lines, with some variations in the decimal part, such as 1.6670000553131104 and 1.6649999618530273.

Distance represents the distance measured by the module, unit: M.

### 2、Code

Code path (Installation package path: ~/software/library\_ws/src) :  
~/software/library\_ws/src/ydlidar\_ros\_driver/scripts/Get\_Distance.py

Note: Search according to the installation location of your function package, which is in the scripts under the function package directory.

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

def scancallback(scan_data):
    print("distance: ", scan_data.ranges[0])

def GetSDMData():
    rospy.init_node('Get_SDM_Data', anonymous=True) # ROS节点初始化
    rospy.Subscriber('/scan', LaserScan, scancallback, queue_size=1000)
    rospy.spin()
```

```
if __name__ == '__main__':  
    GetSDMData()
```

After running the launch file, a '/scan' topic data message will be published. Distance.py defines a subscriber to subscribe to this message.

```
rospy.Subscriber('/scan', LaserScan, scancallback, queue_size=1000)
```

In the callback function, we print the data of this message.

```
def scancallback(scan_data):  
    print("distance: ", scan_data.ranges[0])
```

We can see from the rostopic echo/scan that the distance information is stored in the ranges list. We can read the contents of this list. The subscript is 0 because the list has only one data.