

# GPS and Move\_Base Navigation

If you want to give a latitude and longitude, and then use `Move_Base` to navigate to the point, it is not possible to directly specify the target point, so you need to convert the latitude and longitude into the xy data of the `Move_Base` target point. This section explains how to realize the data conversion.

## 1. Start

Before starting, you need to successfully open `move_base`, the map can be imported into a blank map, and you must make sure that the latitude and longitude of the point you need to go to is on the map.

Otherwise the program will give a warning saying "**The goal sent to the navfn planner is off the global costmap. Planning will always fail to this goal**", which means that the path planning will not reach this place. There is no program to run `move_base` here, and `move_base` is successfully opened by default.

After successfully opening `move_base`, the terminal input.

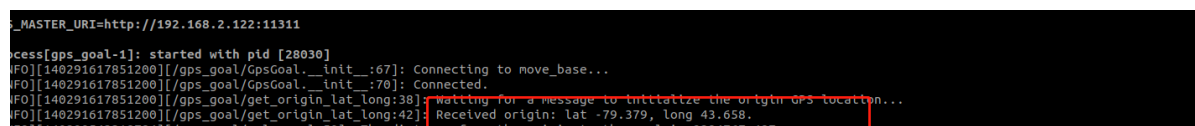
```
roslaunch gps_goal gps_goal.launch
```

- Given the initial location latitude and longitude (where am I?)

```
rostopic pub /local_xy_origin geometry_msgs/PoseStamped '{ header: { frame_id: "/map" }, pose: { position: { x: 43.658, y: -79.379 } } }' -1
```

Here is to publish a message, the topic name is `/local_xy_origin`, and the data x and y behind are reference values, which need to be assigned according to the actual situation.

After running, the terminal prints "Received origin: lat -79.379, long 43.658".

A terminal window screenshot showing the execution of a ROS launch file. The output includes messages from the gps\_goal process, such as "started with pid [28030]", "Connecting to move\_base...", and "Connected.". A red box highlights a warning message: "waiting for a message to initialize the origin gps location...". Below this, the terminal prints "Received origin: lat -79.379, long 43.658.".

- Given the latitude and longitude of the target point (where do I go?)

```
rostopic pub /gps_goal_fix sensor_msgs/NavSatFix "{latitude: 2.548, longitude: 11.06455}" -1
```

The data of the `/gps_goal_fix` topic message is published here, and the latitude and longitude data in it is for reference only, and needs to be given according to the actual situation.

After running, the terminal will print the following message.

```
[INFO][140032442627840][/gps_goal/calc_goal:50]: The distance from the origin to the goal is 9284767.487 m.
[INFO][140032442627840][/gps_goal/calc_goal:52]: The azimuth from the origin to the goal is -32.818 degrees.
[INFO][140032442627840][/gps_goal/calc_goal:59]: The translation from the origin to the goal is (x,y) 7802896.906, -5032067.881 m.
('Goal x: ', 7802896.905971293)
('Goal y: ', -5032067.881259187)
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:112]: Executing move_base goal to position (x,y) 7802896.90597, -5032067.88126, with 0 degrees yaw.
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:113]: To cancel the goal: 'rostopic pub -1 /move_base/cancel actionlib_msgs/GoalID -- {}'
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:117]: Initial goal status: _has_header
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:120]: This goal has been accepted by the simple action server
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:124]: Final goal status: ABORTED
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:127]: Failed to find a valid plan. Even after executing recovery behaviors.
```

```
[INFO][140032442627840][/gps_goal/calc_goal:50]: The distance from the origin to the goal is 9284767.487 m.
[INFO][140032442627840][/gps_goal/calc_goal:52]: The azimuth from the origin to the goal is -32.818 degrees.
[INFO][140032442627840][/gps_goal/calc_goal:59]: The translation from the origin to the goal is (x,y) 7802896.906, -5032067.881 m.
('Goal x: ', 7802896.905971293)
('Goal y: ', -5032067.881259187)
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:112]: Executing move_base goal to position (x,y) 7802896.90597, -5032067.88126, with 0 degrees yaw.
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:113]: To cancel the goal: 'rostopic pub -1 /move_base/cancel actionlib_msgs/GoalID -- {}'
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:117]: Initial goal status: _has_header
[INFO][140032442627840][/gps_goal/GpsGoal.publish_goal:120]: This goal has been accepted by the simple action server
```

We know from the print message that the calculated distance to the two points is **9284767.487 meters** (this is because the latitude and longitude we gave is too different), and the azimuth angle is **-32.818 degrees**; after conversion, it will print out (7802896.906, -5032067.881) this is the xy value of the target point in the move\_base map. This value obviously exceeds the range of our map, so it is impossible for path planning to reach here. This is why it was mentioned earlier. It is **necessary to ensure that the latitude and longitude of the origin and target point must be in the map**. After calculating the xy value, package the data and send it to move\_base for path planning.

## 2. The program source code gps\_goal.py

Code path: workspace/src/gps\_goal/src/gps\_goal/gps\_goal.py

The code is very simple, look directly at the cli\_main function,

```
gpsGoal = GpsGoal() # Create object
lat, long = DMS_to_decimal_format(lat, long) #Parse the latitude and longitude,
this part is mainly to parse the past data sent by the command line and the past
longitude and latitude sent by the topic
gpsGoal.do_gps_goal(lat, long, roll=roll, pitch=pitch, yaw=yaw)
```

**do\_gps\_goal** function

```
def do_gps_goal(self, goal_lat, goal_long, z=0, yaw=0, roll=0, pitch=0):  
    # Calculate goal x and y in the frame_id given the frame's origin GPS and a goal  
    # GPS location  
    x, y = calc_goal(self.origin_lat, self.origin_long, goal_lat, goal_long)  
    print("Goal x: ",x)  
    print("Goal y: ",y)  
    self.publish_goal(x=x, y=y, z=z, yaw=yaw, roll=roll, pitch=pitch)
```

The **calc\_goal** function calculates the xy value according to the latitude and longitude of the origin and the latitude and longitude of the target point

The **publish\_goal** function packs the xy value into data of type **MoveBaseGoal()** and sends it out