

# Odin1重定位地图获取手册

Odin1提供重定位功能，已实现与MANIFOLD手持扫描仪的数据联动。目前我们提供3个方案：

- 1.使用Odin采集olx数据，通过Mindcloud处理后使用“导出地图”功能导出bin文件用于Odin的重定位；
- 2.使用留形手持扫描仪（留形机，如：Q9000）采集lx数据，通过Mindcloud处理后使用“导出地图”功能导出bin文件用于Odin的重定位；
- 3.使用Odin SLAM模式生成bin文件用于Odin的重定位。详细操作如下。

## 1. Odin录制olx文件（选择1）

### 1.1 Odin录制olx文件（Odin固件版本要求0.10.0以上，驱动版本0.9.0以上）

1.1.1 修改/catkin\_ws/src/odin\_ros\_driver/config/control\_command.yaml文件中参数，打开recorddata: 1 选择custom\_map\_mode: 0 (odom模式)：

代码块

```
1 recorddata: 1 # 0: off; 1: on
2 ***
3 custom_map_mode: 0 # 0: Odometry mode 1: SLAM mode 2: Relocalization mode
```

### 1.1.2 运行驱动

代码块

```
1 cd ~/catkin_ws
2 # 以下以ros2为例，ros1请输入对应的指令
3 source install/setup.bash
4 ros2 launch odin_ros_driver odin1_ros2.launch.py
```

### 1.1.3 采集地图

采集完成后Ctrl+c结束驱动，即可在/catkin\_ws/src/odin\_ros\_driver/recorddata中找到对应的olx文件。将整个文件夹拷贝到装有MindCloud（版本 $\geq 0.2.8$ ）的电脑上进行第3步处理。

## 2. Q9000录制lx数据（选择2）

### 2.1 Q9000录制lx文件

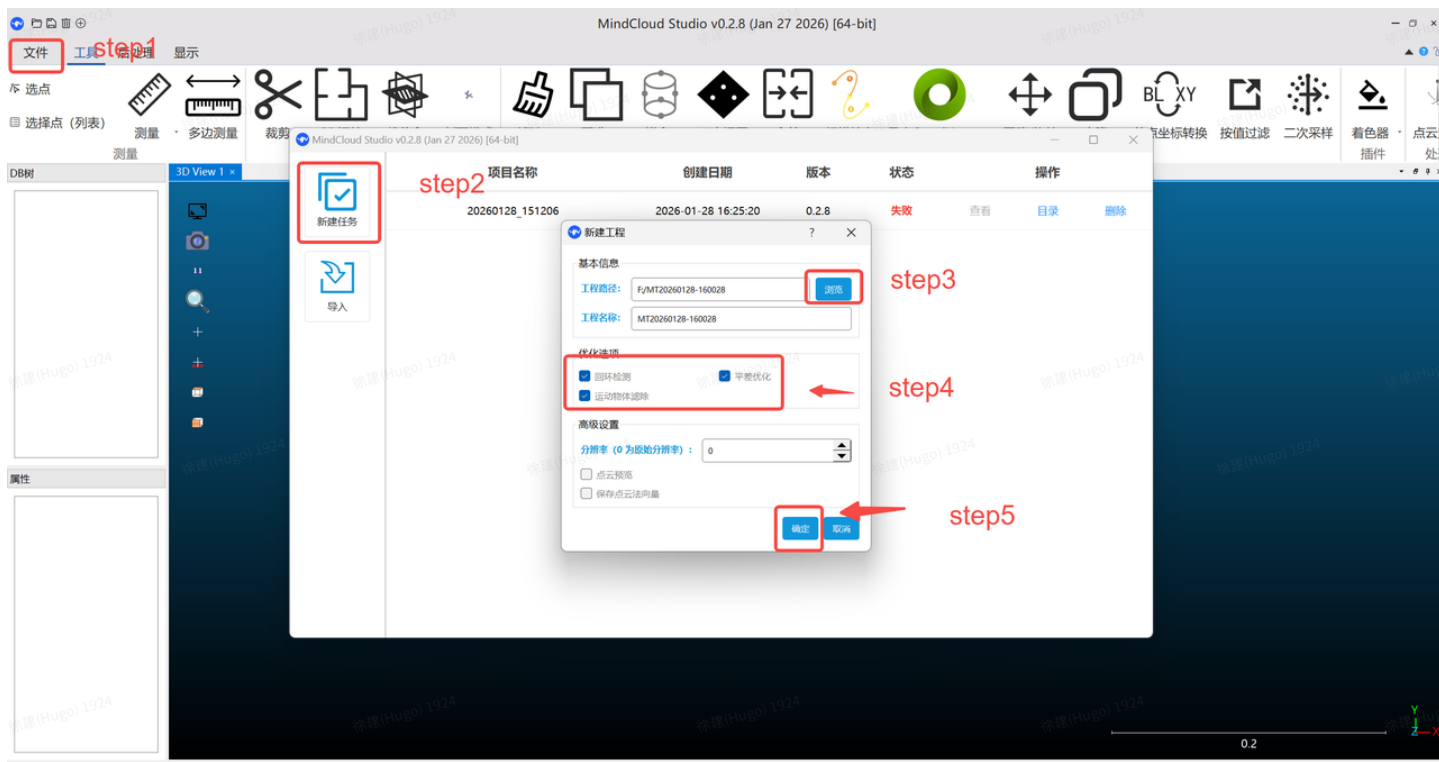
- 采集数据

- 将录制的lx文件拷贝到装有MindCloud（版本 $\geq 0.2.8$ ）的电脑上进行第3步处理

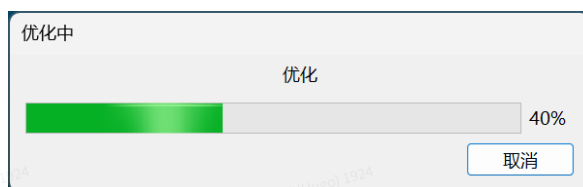
## 2.2 导入MindCloud Studio并保存pcd & bin文件（需要在windows系统）

## 3. MindCloud使用

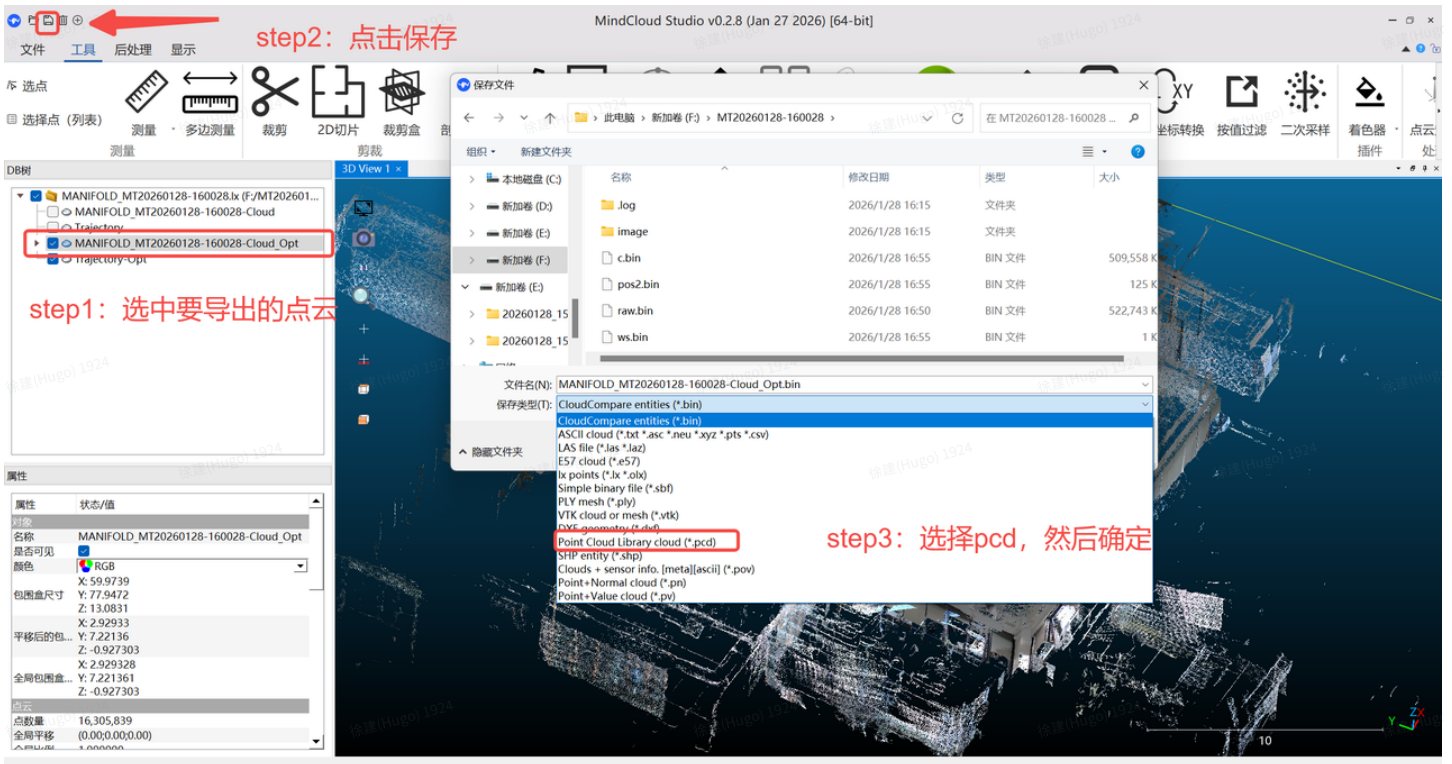
### 3.1 mindcloud处理 olx/lx数据



#### 3.1.1 等待处理完成



#### 3.1.2 保存Pcd点云图



### 3.1.3 保存bin地图



### 3.1.4 将生成的pcd点云地图和bin文件地图拷贝到ubuntu电脑上

## 3.2 pcd点云抽稀

- 由于原始点云很大如果想在rviz中显示，需要进行点云抽稀
- 使用如下代码，建议将pcd文件数据量抽稀至20M以内（保证rviz流畅运行即可）

代码块

```

1  #!/usr/bin/env python3
2  """
3  超大 PCD 降采样工具 (适合 RViz)
4  """
5
6  ##### 可修改区域 #####
7  INPUT_PCD = "/home/hugo/git/pcd_publisher/relocalization.pcd.pcd"      #
   原始 4.3G PCD
8  OUTPUT_PCD = "/home/hugo/git/pcd_publisher/relocalization_0.05.pcd"  # 输出
   压缩后
9  VOXEL_SIZE = 0.05                # 体素大小 (单位: 米)
10 #####
11
12 import open3d as o3d
13 import os
14
15 def main():
16     print("Loading PCD (this may take time)...")
17     pcd = o3d.io.read_point_cloud(INPUT_PCD)
18
19     print(f"Original points: {len(pcd.points):,}")
20
21     print(f"Downsampling with voxel size = {VOXEL_SIZE} m")
22     pcd_ds = pcd.voxel_down_sample(voxel_size=VOXEL_SIZE)
23
24     print(f"Downsampled points: {len(pcd_ds.points):,}")
25
26     print("Saving compressed PCD...")
27     o3d.io.write_point_cloud(
28         OUTPUT_PCD,
29         pcd_ds,
30         write_ascii=False,    # binary
31         compressed=True      # open3d 压缩
32     )
33
34     orig = os.path.getsize(INPUT_PCD) / 1024 / 1024
35     out = os.path.getsize(OUTPUT_PCD) / 1024 / 1024
36
37     print(f"File size: {orig:.1f} MB -> {out:.1f} MB")
38     print("Done.")
39
40 if __name__ == "__main__":
41     main()
42

```

- 抽稀后

```
hugo@hugo: ~/git/pcd_publisher
hugo@hugo:~/git/pcd_publisher$ ll -h
total 323M
drwxrwxr-x 3 hugo hugo 4.0K Jan 28 17:33 ./
drwxrwxrwx 13 hugo hugo 4.0K Jan 28 17:00 █/
drwxrwxr-x 2 hugo hugo 4.0K Jan 28 17:33 12/
-rw-rw-r-- 1 hugo hugo 1.3K Jan 28 17:35 pcd_downsample.py
-rw-rw-r-- 1 hugo hugo 2.5K Dec 31 16:58 pcd_publisher.py
-rwxr-xr-x 1 hugo hugo 53M Jan 28 17:09 relocalization.bin*
-rwxr-xr-x 1 hugo hugo 271M Jan 28 17:06 relocalization.pcd*
hugo@hugo:~/git/pcd_publisher$ python3 pcd_downsample.py
Loading PCD (this may take time)...
Original points: 16,305,839
Downsampling with voxel size = 0.05 m
Downsampled points: 850,475
Saving compressed PCD...
File size: 270.3 MB -> 12.9 MB
Done.
hugo@hugo:~/git/pcd_publisher$ ll -h
total 336M
drwxrwxr-x 3 hugo hugo 4.0K Jan 28 17:35 ./
drwxrwxrwx 13 hugo hugo 4.0K Jan 28 17:00 █/
drwxrwxr-x 2 hugo hugo 4.0K Jan 28 17:33 12/
-rw-rw-r-- 1 hugo hugo 1.3K Jan 28 17:35 pcd_downsample.py
-rw-rw-r-- 1 hugo hugo 2.5K Dec 31 16:58 pcd_publisher.py
-rw-rw-r-- 1 hugo hugo 13M Jan 28 17:35 relocalization_0.05.pcd
-rwxr-xr-x 1 hugo hugo 53M Jan 28 17:09 relocalization.bin*
-rwxr-xr-x 1 hugo hugo 271M Jan 28 17:06 relocalization.pcd*
hugo@hugo:~/git/pcd_publisher$
```

### 3.3 运行重定位同时发布pcd点云

#### 3.3.1 设置重定位模式，输入正确的bin文件路径：

代码块

```
1 custom_map_mode: 2 # 0: Odometry mode 1: SLAM mode 2: Relocalization
mode
2 custom_init_pos: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
3 relocalization_map_abs_path:
"/home/hugo/git/odin_0.8.0/map/relocalization.bin" # must be set for
Relocalization mode or will fail
```

#### 3.3.2 运行驱动

#### 3.3.3 使用如下代码将抽稀后的pcd发布到ros中

代码块

```
1
2 #!/usr/bin/env python3
3 """
4 PCD (XYZRGB) -> ROS2 PointCloud2
5 使用 open3d, 直接 python3 运行
6 """
7
8 ##### 可修改区域 #####
9 PCD_PATH = "/home/hugo/git/pcd_publisher/relocalization_0.05.pcd" # <<< 改这
里
10 FRAME_ID = "map"
11 TOPIC = "/pcd_points"
```

```

12  PUB_HZ    = 1.0
13  #####
14
15  import rclpy
16  from rclpy.node import Node
17
18  import numpy as np
19  import open3d as o3d
20  import struct
21
22  from std_msgs.msg import Header
23  from sensor_msgs.msg import PointCloud2, PointField
24  from sensor_msgs_py import point_cloud2
25
26  def rgb_to_float(r, g, b):
27      """RGB(uint8) -> packed float32"""
28      rgb_uint32 = (int(r) << 16) | (int(g) << 8) | int(b)
29      return struct.unpack("f", struct.pack("I", rgb_uint32))[0]
30
31  class PCDPublisher(Node):
32      def __init__(self):
33          super().__init__("pcd_rgb_publisher")
34
35          self.pub = self.create_publisher(PointCloud2, TOPIC, 10)
36          self.timer = self.create_timer(1.0 / PUB_HZ, self.timer_cb)
37
38          self.cloud_msg = self.load_pcd(PCD_PATH)
39          self.get_logger().info(f"Loaded PCD with RGB: {PCD_PATH}")
40
41      def load_pcd(self, path: str) -> PointCloud2:
42          pcd = o3d.io.read_point_cloud(path)
43
44          points = np.asarray(pcd.points, dtype=np.float32)
45          colors = np.asarray(pcd.colors, dtype=np.float32) # [0,1]
46
47          cloud_data = []
48          for i in range(points.shape[0]):
49              r = int(colors[i, 0] * 255)
50              g = int(colors[i, 1] * 255)
51              b = int(colors[i, 2] * 255)
52              rgb_f = rgb_to_float(r, g, b)
53
54              cloud_data.append((
55                  points[i, 0],
56                  points[i, 1],
57                  points[i, 2],
58                  rgb_f

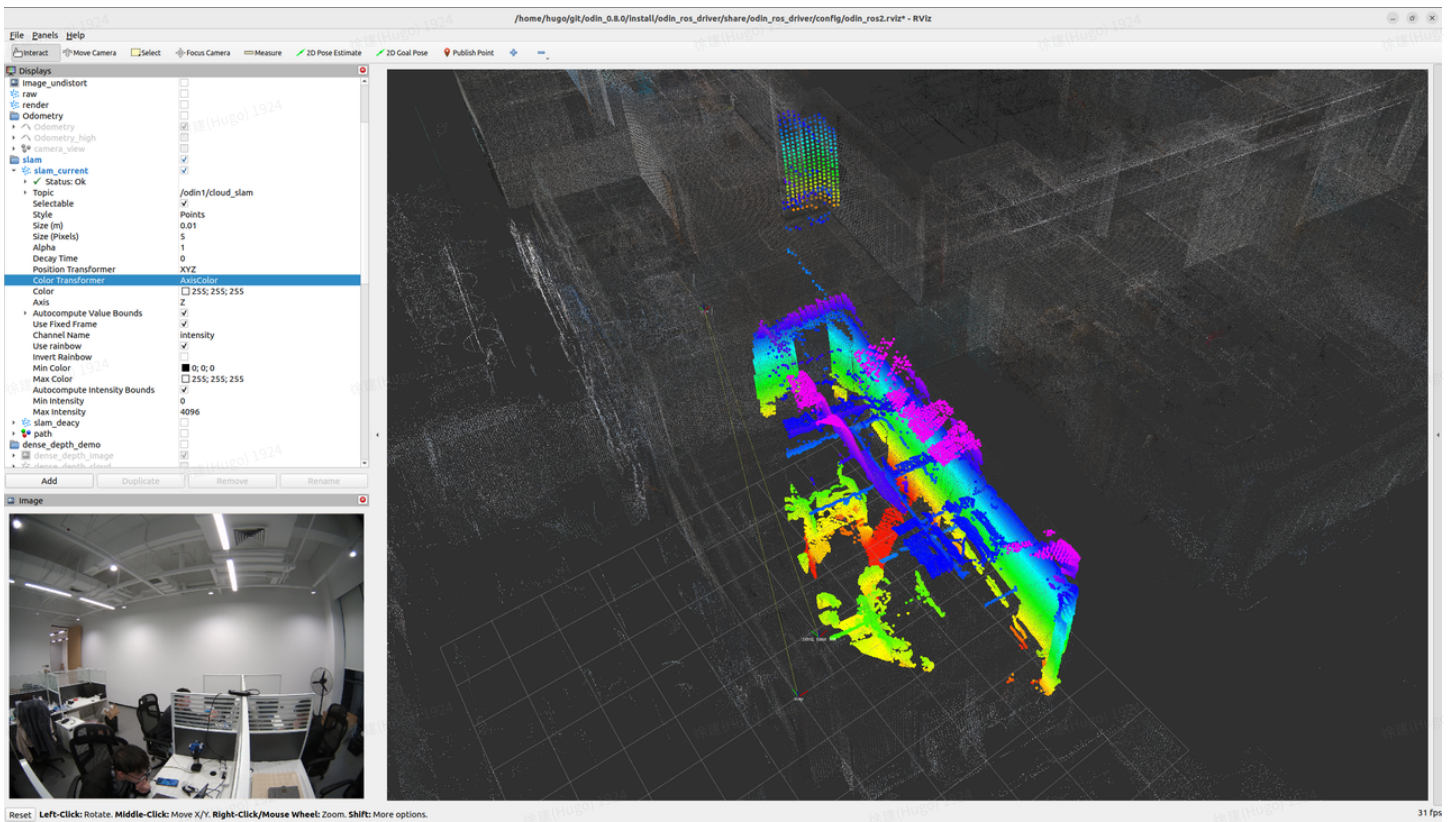
```

```

59         ))
60
61     header = Header()
62     header.frame_id = FRAME_ID
63
64     fields = [
65         PointField(name="x", offset=0, datatype=PointField.FLOAT32,
66 count=1),
67         PointField(name="y", offset=4, datatype=PointField.FLOAT32,
68 count=1),
69         PointField(name="z", offset=8, datatype=PointField.FLOAT32,
70 count=1),
71         PointField(name="rgb", offset=12, datatype=PointField.FLOAT32,
72 count=1),
73     ]
74
75     return point_cloud2.create_cloud(header, fields, cloud_data)
76
77 def timer_cb(self):
78     self.cloud_msg.header.stamp = self.get_clock().now().to_msg()
79     self.pub.publish(self.cloud_msg)
80
81 def main():
82     rclpy.init()
83     node = PCDPublisher()
84     rclpy.spin(node)
85     node.destroy_node()
86     rclpy.shutdown()
87
88 if __name__ == "__main__":
89     main()

```

### 3.3.4 效果如下:



## 4. 使用Odin录制重定位地图（选择3）

### 4.1 录制bin格式地图

- 设置control\_command.yaml文件中customer\_map\_mode: 1, recorddata: 0

代码块

```
1 recorddata: 0           # 0: off; 1: on
2 ***
3 custom_map_mode: 1     # 0: Odometry mode 1: SLAM mode 2: Relocalization mode
```

- 运行驱动进行数据采集

代码块

```
1 cd ~/catkin_ws
2 # 以下以ros2为例, ros1请输入对应的指令
3 source install/setup.bash
4 ros2 launch odin_ros_driver odin1_ros2.launch.py
```

- 采集地图结束后请勿直接Ctrl+c关闭驱动，需要新建终端，进入到驱动路径中，运行./set\_param.sh save\_map 1保存地图。

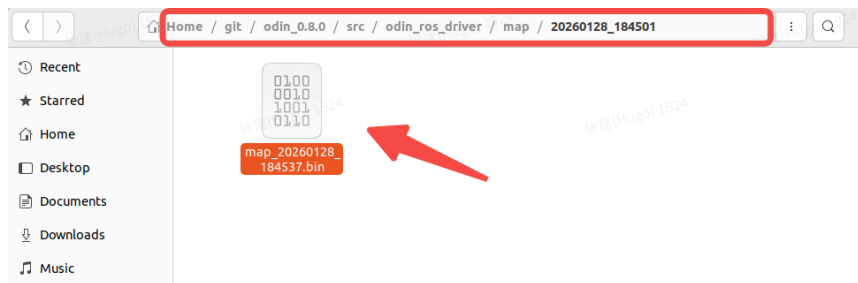
代码块

```
1 cd ~/catkin_ws/src/odin_ros_driver
```

- 2 ./set\_param.sh save\_map 1
- 3 # 等待驱动终端提示保存完成后再结束驱动

```
hugo@hugo: ~/git/odin_0.8.0
tialized save_map = 0
[host_sdk_sample-1] [INFO] [1769597102.298255969] [device_cb]: Command interface ready.
Use: echo 'set save_map 1' > /tmp/odin_command.txt
[host_sdk_sample-1] [2026:01:28:18:45:02][INFO][api.cpp:lidar_set_custom_parameter:1275
]: set custom parameter success: save_map (length: 4)
[host_sdk_sample-1] [2026:01:28:18:45:02][DEBUG][api.cpp:lidar_start_stream:740]: dtof_
subframe_odr: 328500
[host_sdk_sample-1] [2026:01:28:18:45:02][DEBUG][api.cpp:lidar_start_stream:743]: start
stream mode success....
[host_sdk_sample-1] Tct: -0.00455 -0.99988 -0.01472 0.0385
[host_sdk_sample-1] 0.00925 0.01468 -0.99985 -0.00914
[host_sdk_sample-1] 0.99995 -0.00468 0.00918 -0.00911
[host_sdk_sample-1] 0 0 0 1
[host_sdk_sample-1] [INFO] [1769597102.298726014] [device_cb]: Software connection succ
essful in 0 seconds
[host_sdk_sample-1] [INFO] [1769597102.298740681] [device_cb]: Device ready and streams
activated
[host_sdk_sample-1] [INFO] [1769597135.946245106] [command_processor]: Successfully set
save_map = 1
[host_sdk_sample-1] [INFO] [1769597137.493918230] [param_monitor]: Map is saved on dev
ice, now transferring to [/home/hugo/git/odin_0.8.0/src/odin_ros_driver/map/20260128_1845
01/map_20260128_184537.bin]
[host_sdk_sample-1] [INFO] [1769597137.543778056] [param_monitor]: map get start succes
s, now transferring...
```

```
hugo@hugo: ~/git/odin_0.8.0/src/odin_ros_driver
hugo@hugo:~/git/odin_0.8.0/src/odin_ros_driver$ ./set_param.sh save_map 1
Command sent: set save_map 1
Command file: /tmp/odin_command.txt
hugo@hugo:~/git/odin_0.8.0/src/odin_ros_driver$
```



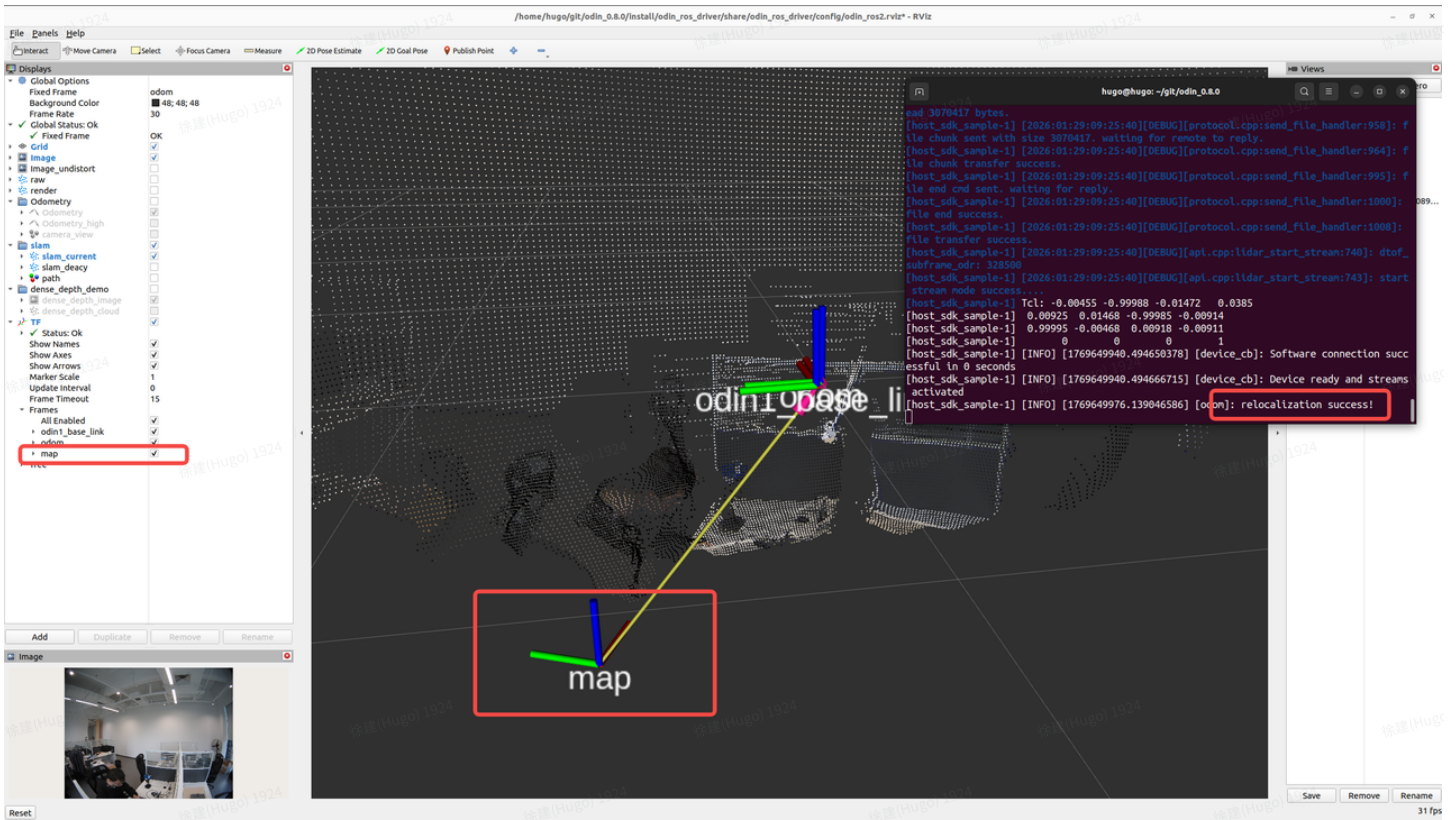
## 4.2 直接运行重定位

- 设置control\_command.yaml文件中customer\_map\_mode: 2, 配置bin文件所在路径:

代码块

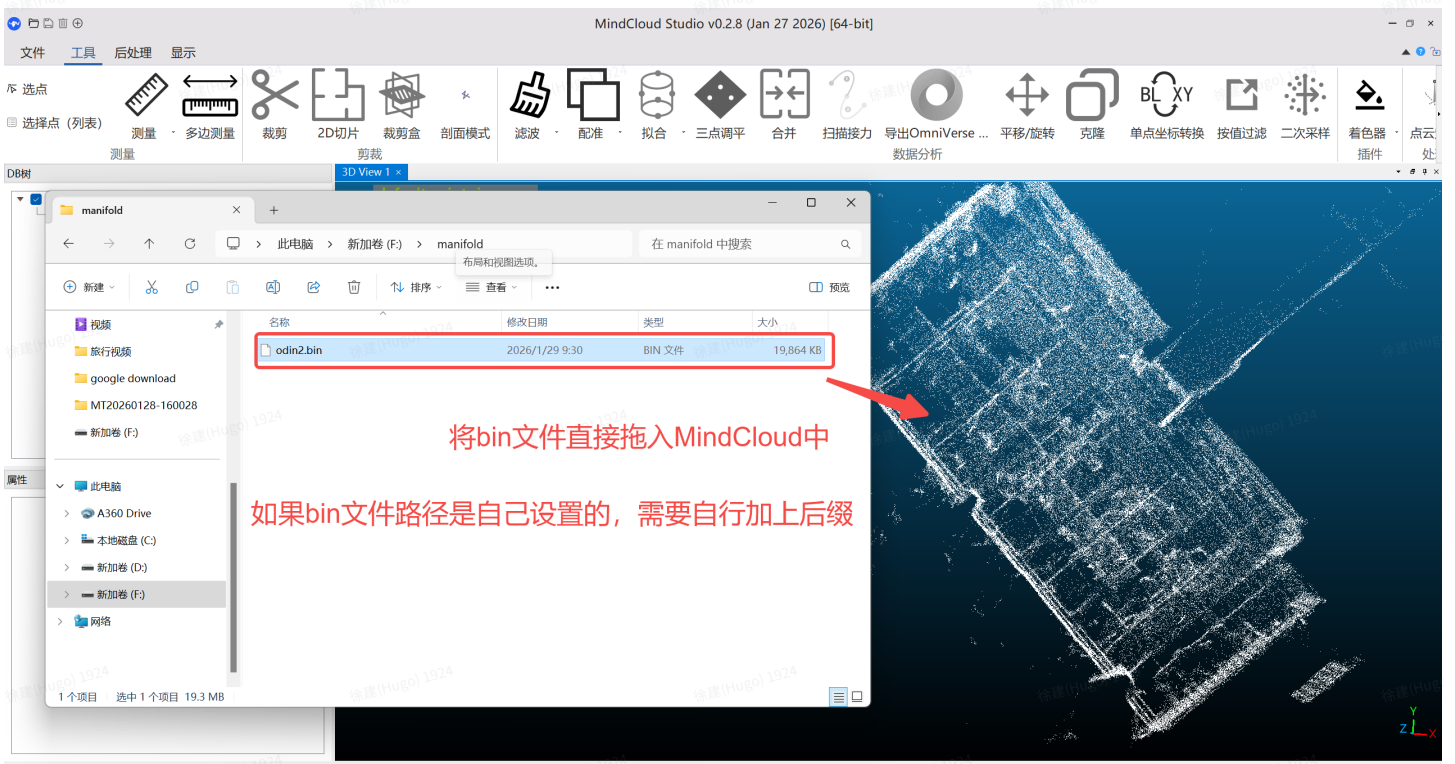
```
1 custom_map_mode: 2 # 0: Odometry mode 1: SLAM mode 2: Relocalization mode
2 custom_init_pos: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
3 relocalization_map_abs_path:
"/home/hugo/git/odin_0.8.0/src/odin_ros_driver/map/20260128_184501/map_20260128_
_184537.bin" # must be set for Relocalization mode or will fail
```

- 运行驱动
- 定位成功后显示如下: rviz显示map系, 驱动终端提示重定位成功!

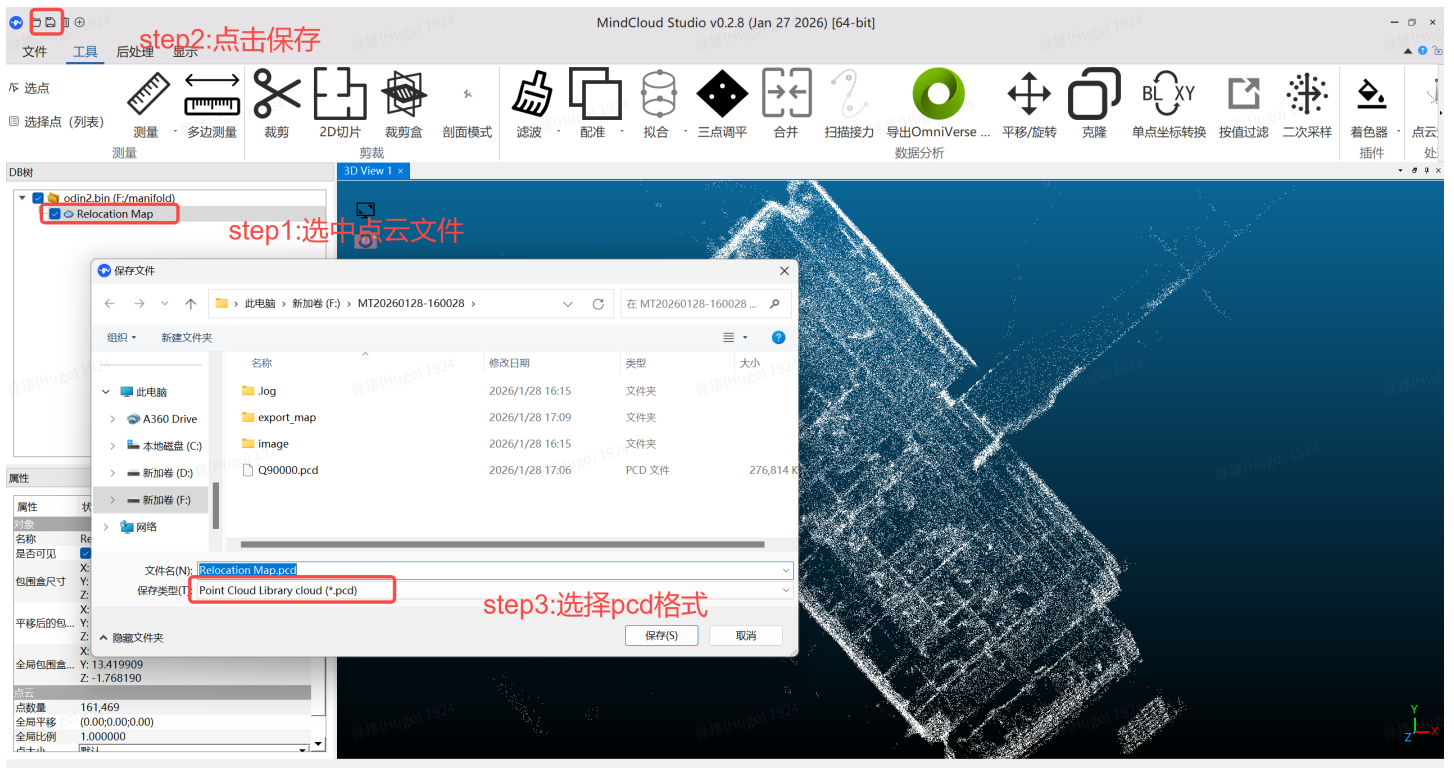


### 4.3 bin文件导出pcd

- 拷贝Odin导出的bin文件到安装Mindcloud0.2.8版本的PC上
- 将bin文件直接拖入MindCloud中



- 导出pcd格式点云



## 4.4 rviz显示pcd点云

- 将pcd拷贝到ubuntu（驱动所在）电脑上
- 同1.5一样进行加载但是要注意bin导出的pcd没有rgb信息，也可以依据自己需求书写发布代码

注：上述提及代码均为AI生成的demo，暂不技术支持。