大华 AGV 导航定位算法相关问题排查手册

先院导航定位算法组

2020.09.11

修改时间	修改内容	修改人
	文档合并	林辉、胡立志
	运动控制、标定	余冬冬、高炳舒
	反光柱、反光条、货架识别以及 点云预处理相关问题排查	胡立志 彭建建 石鹏
	建图问题文档	单泽泳、王坤
2020.9.11	附录相关文档	胡立志
2020.8.15	标定、建图和定位模块问题初稿	胡立志、林辉

附录

常见定位算法问题排查手册

一：激光外参标定(见标定文档)

1.1 标定环境要求：

拿到设备，第一件事情要先检查激光外参是否标定过，检查的途径为：

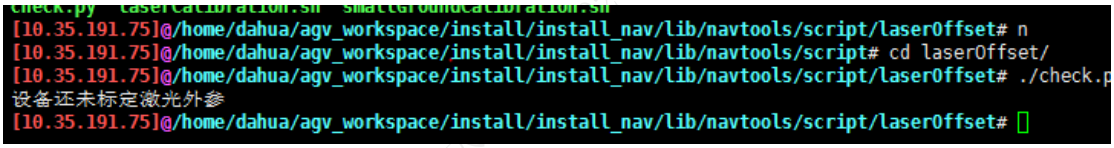
a) 在终端上登录设备,比如：

ssh dahua@10.35.191.75

b) 在终端输入 n 后，再输入指令：

cd laserOffset

执行该目录下的 check.py 文件，查看检验的结果，参考下图：



标定环境尽量选择地面平整、四周结构特征好的环境，选择标准如图 Fig1 所示：

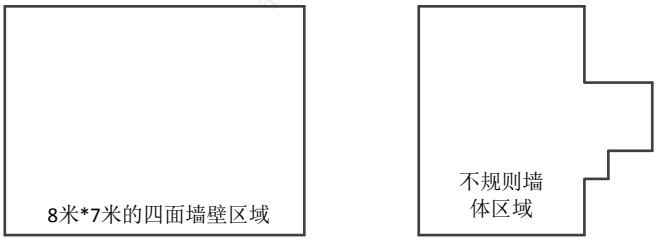


Fig1.环境结构标准图

默认标定的行走路线是 8 字形，标定区域的大小最好在 8m*8m 以上。稍微大一些有助于提高标定效果，标定的时候保证激光离墙面至少有 1.5m 以上的距离，减少测距误差，必要的情况下可以弄个挡板遮挡下。**现场环境选择的标定场景示例如 Fig2 所示。**

1.2 标定注意事项：

标定时环境尽量少变化且激光水平面上极少杂乱无章的物体，设备激光外参标定时需保证环境没有明显变化，比如人不能站在 AGV 前方区域乱走，，标定过程也不允许路线上有障碍。定区域尽量保证地面水平不打滑；标定时如果设备明显翘起则需在小车上放承重物，保证在运行中激光不会因小车加减速导致激光上下晃动

1.3 标定实施：

xshell 登录设备，sudo 后，执行如下命令即可：
/home/dahua/agv_workspace/install/install_nav/lib/navtools/script/laserOffset/laserCalibration.sh

Fig.2 现场环境选择的标定环境示例

二、建图注意事项：

2.1 建图参数文件检查

建图之前先检查建图参数文件，参数文件名 mapper_params.yaml。
先检查 use_scan_matching:true，该参数表示建图是否使用激光数据；务必确保打开状态,设置为 true；
若使用反光柱或者反光条辅助定位，则对应的参数名 use_cylinder_reflector(反光柱)或者 use_stripe_reflector(反光条)确保打开，设置为 true;其中，利用反光柱辅助建图时，需要根据现场使用的反光柱半径修改参数：cylinder_reflector_radius。比如现场实际使用的反光柱半径为 0.06m，则该参数对应的数值就为 0.06。

反射强度根据对应的激光型号来设置，修改对应的参数 laser_intensity_threshold。其中：
北洋激光反射强度：10000；sick 激光反射强度：900；倍加福反射前度：1000；

根据场地的大小以及是否适合建立回环，需要修改回环检测参数：

loop_search_maximum_distance，该参数默认设置为 10m。

2.2 建议的建图路线(具体见建图文档)

当现场需要使用二维码和激光融合定位算法，建立地图时，注意先要将设备推到二维码并朝向为 0 度。(其中二维码上数字的方向为零度方向)。

打开向下扫二维码相机的指令：

rosservice call /dm0/setDropFrame “enable: false”，返回 0，表示打开成功；

```
[10.35.191.31]@/home/dahua# rosservice call /dm0/setDropFrame "enable: false"
result: 0
```

然后在终端打开 d，推动小车，将小车推到二维码上，能查看小车在二维码的相对位置。

2.3.地图检查：

对于现场使用的大地图，一般肉眼很难观测出地图细节建立得好不好，基于当前现有手段，需要记录现场环境的激光、里程计数据，然后利用定位算法仿真来对比(后期会使用地图评价工具来评价)。

登陆设备，记录一组激光和里程计数据。先打开里程计数据：

rostopic pub /set_odom_freq std_msgs/Int8 “data: 110” -1

```
[10.35.191.31]@/home/dahua# rostopic pub /set_odom_freq std_msgs/Int8 "data: 110" -1
publishing and latching message for 3.0 seconds
```

然后使用命令 rostopic echo /odom，确认里程计是否打开。

```
[10.35.191.31]@/home/dahua# rostopic echo /odom
header:
  seq: 10468
  stamp:
    secs: 1598266053
    nsecs: 138301516
  frame_id: "odom"
child_frame_id: "base_link"
pose:
  pose:
    position:
      x: -0.0889257181279
      y: -0.08250436131
      z: 0.0
    orientation:
      x: 0.0
      y: 0.0
      z: 0.001120181468
      w: 0.999999372597
  covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

然后在终端输入 o，看一下当前设备的位置：

```
---
timestamp:
  secs: 1604910658
  nsecs: 909169067
x: -0.0587540566921
y: -0.283590465784
angle: 91.5437316895
slipAngle: 0.0
v_l: 0.0
v_a: 0.0
valid: True
score: 100
---
```

再使用如下命令记录激光和里程计数据：

rosbag record /scan /odom –O ***.bag

```
[10.35.191.73]@/home/dahua# rosbag record /scan /odom -O 0824_-0.05_-0.28.bag
[ INFO] [1604910830.984131764]: Subscribing to /odom
[ INFO] [1604910830.999002232]: Subscribing to /scan
[ INFO] [1604910831.014066583]: Recording to 0824_-0.05_-0.28.bag.
```

***自己手动输入的 bag 文件名称，以日期+当前设备位姿命名！

比如 0824_-0.05_-0.28.bag

a) 检查地图中是否有明显的杂点，见图 3，对于这种轮廓里面的杂点需要擦除；

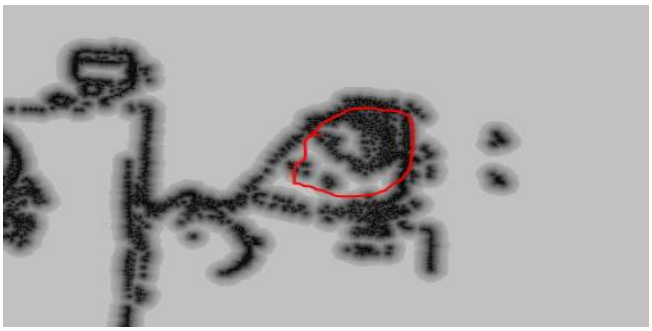


Fig3 包含杂点地图

b) 地图重影问题以及地图细节，将数据、地图以及外参文件发给定位小组，定位小组离线仿真来观察地图细节是否足够好，以及重影中，该删除的是哪条边。

三、定位问题

定位日志在/var/robot/log/nav/alglog/loc 目录下，对应的定位日志为 enml_loc。

```
[10.35.191.31]@/var/robot/log/nav/alglog/loc# ls
enml_loc  enml_loc.1  enml_loc.2  enml_loc.3
```

3.1 重定位问题

通过平台调用重定位时，在地图界面上点一下，就会以该点坐标为中心，搜索范围为 4x4m 的搜索框，在该搜索框中搜索范围内搜索出最优位置。请务必在搜索出最优位置后，在平台上点确认按钮！(这里需要插图说明！)

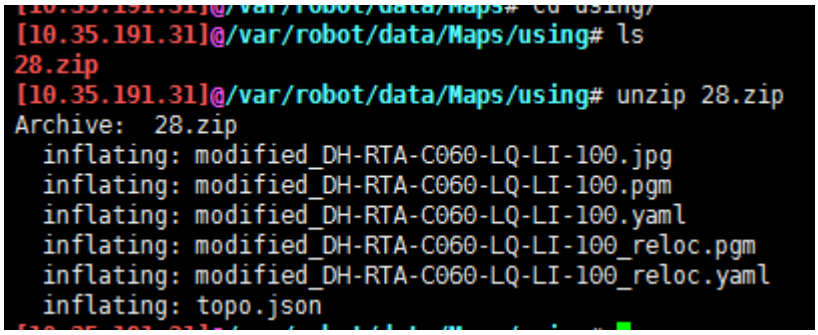
通过终端调用的重定位命令如下：

rosservice call /nav_monitor/set_init_pose "{pose_x: 0, pose_y: 0, pose_theta: 0, width: 0, height: 0, isFromNav: false}"，单位是毫米！

在定位日志中搜索的关键字是 relocalizationService！

3.1.1 重定位耗时问题:

现在使用的是多分辨率地图重定位算法，登入设备，进入地图所在的目录，一般在/var/robot/data/Maps/using 目录下！解压缩地图文件，查看地图是否带有 reloc 后缀的地图文件。有该文件，重定位耗时和内存占用都会大大缩短。(拥有后缀 reloc 地图，前提是扫图工具版本要正确。老的平台工具不支持带后缀 reloc 的地图，从而重定位耗时都很慢)



调用重定位给的初值和搜索范围对应的日志是：

[EnML]: relocalizationService called with req = [54.198, -23.099, 0.000, 4.000, 4.000]

重定位耗时对应的日志是:

[EnML]: relocalizationService return with searched_pose = [54.118, -23.099, 3.141], searched_score = 0.741, fine_searched_pose = [54.158, -23.099, 3.137], fine_searched_score = 0.722, matched_pose = [54.163, -23.096, 3.137], matched_score = 0.674, sonsume time 1303.765 ms

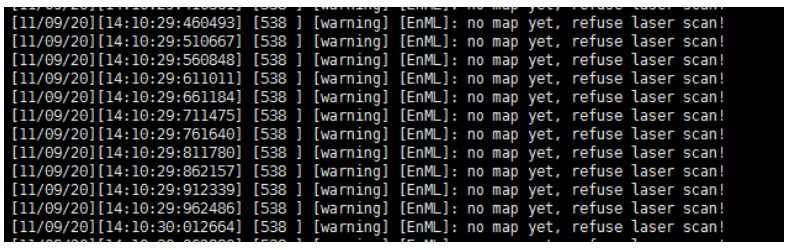
3.1.2 重定位失败问题：

重定位失败主要三个因素：

因素一：与给的初值有关，如果初值给的不对，那么重定位的结果必然不对；查看重定位的初值的关键字为：

[EnML]: relocalizationService called with req = [54.198, -23.099, 0.000, 4.000, 4.000]，查看初值是否给的准确，需要对比调用重定位之前的设备的日志去对比！

因素二：没有地图，直接拒绝重定位；对应的关键字为：



因素三：地图因素。

当前环境相对于建立地图时的环境变化较大，导致重定位后的得分很低导致定位失败；

或者地图很空旷，重定位的得分很低，始终低于设置的重定位得分阈值；或者当前环境有很大的对称性，导致重定位后，方向相反；这种情况也是需要录取激光数据，发给定位小组，进行离线仿真对比分析；比如：

案例 1：

对于上面美的现场的地图文件，红色表示机器人的前进方向，B 这边的区域环境变化量特别大，所以在扫图的时候，就会把 B 这块区域进行擦除；地图中 B 区域没有什么特征点，此时就要让设备对着区域 A 调用重定位，才能提高重定位的成功率，如图 4 所示；

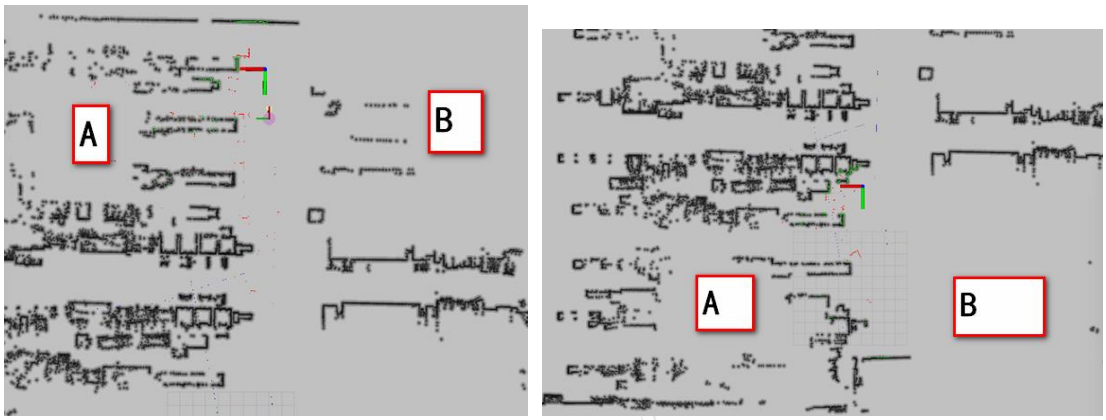


Fig4 重定位地图的设备朝向

案例 2：

对于维信项目，长走廊的情形，设备的朝向如图 5 的 1 所示，看到上方凸起的位置，有助于重定位成功；如果是 2 的朝向，对着一堵墙，此时设备沿着走廊的方向偏移，看到的都是一堵墙，不利于重定位；

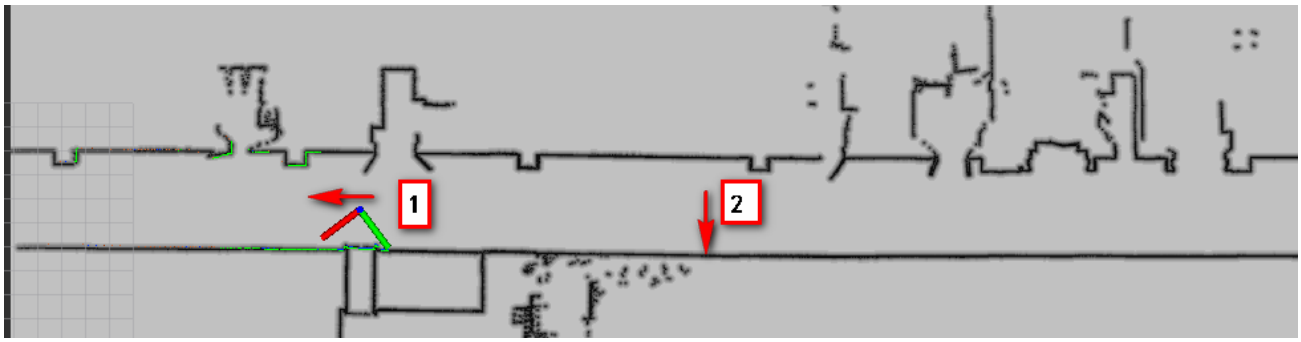


Fig5. 长走廊环境地图

案例 3：

对于熊猫现场空旷的环境，设备在位置 1 的朝向，能看到较多特征，有助于重定位；而在位置 2 和位置 3 的朝向，能看到的特征很少，重定位的评分也会很低，不利于重定位。

Fig6. 空旷环境地图

因素四：

当前设备启动很慢，一般按开机键到完全启动需要 4 分钟到 5 分钟，需要等待设备完全启动后才能调用重定位；如果在启动过程中调用重定位，则大概率会重定位失败！查询该问题的方法是在定位日志中，按照 a)找到调用重定位的关键字，然后查看在重定位过程中是否有线速度！如下图所示。

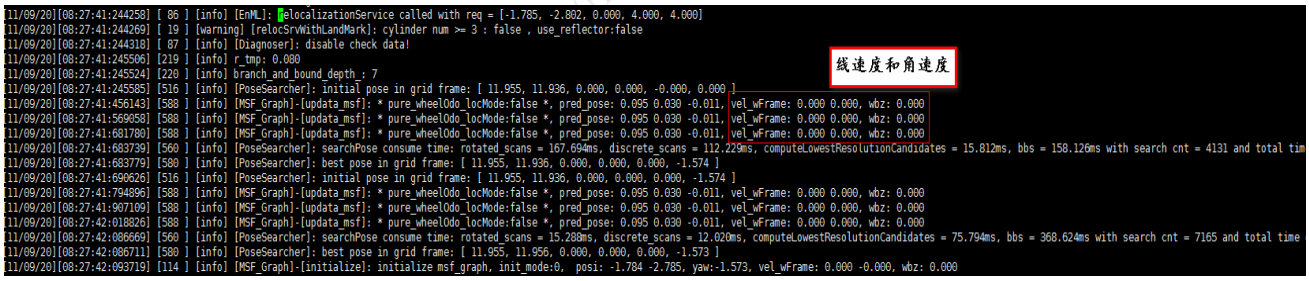


Fig7:重定位过程中移动设备关键日志

3.2 定位丢失问题：

3.2.1 重定位失败上报定位丢失

重定位时，重定位得分始终低于重定位成功的阈值(地牛叉车项目：0.4，AGV 项目 0.2)，此时会上报定位丢失，重定位得分查看如下关键字。

[EnML]: relocalizationService return with searched_pose = [54.118, -23.099, 3.141], searched_score = 0.741, fine_searched_pose = [54.158, -23.099, 3.137], fine_searched_score = 0.722, matched_pose = [54.163, -23.096, 3.137], matched_score = 0.674, sonsume time 1303.765 ms。如果重定位成功，会有关键日志输出 set poseEstimation_Right_Flag TRUE !

3.2.2 定位过程中上报的定位丢失

定位过程中上报的定位丢失主要由三个方面引起：

因素一：

地图变化太大，导致激光与地图匹配得分很低，导致综合定位评分低；或者地图太空旷，特征环境特征建得不全，如下图所示，图 1 中右边的墙在建图的时候没有建出来，正确的建图如图 2；此时查看的定位算法日志中的关键字为 loc_matchQuality：0.3；

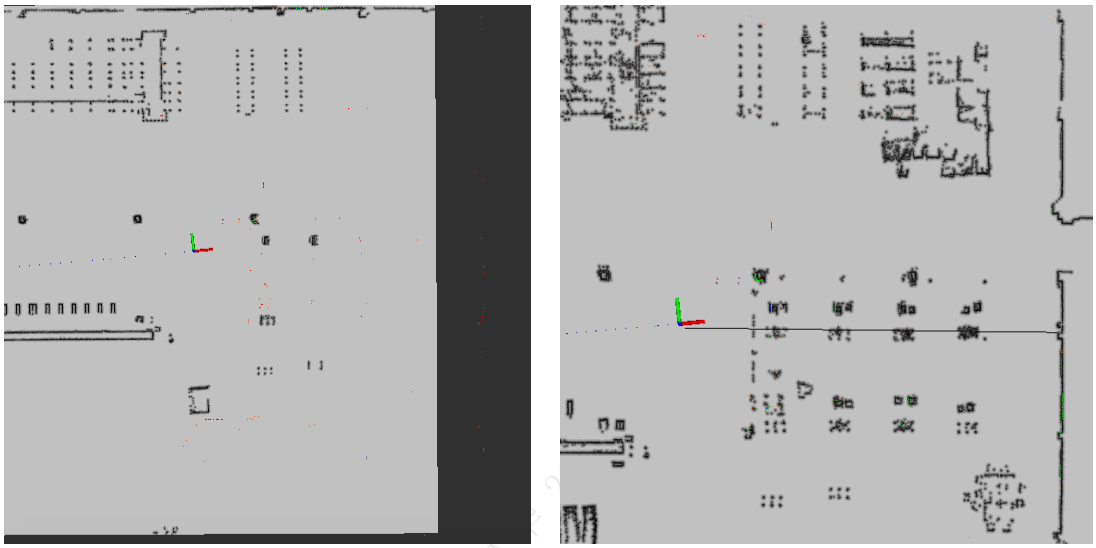


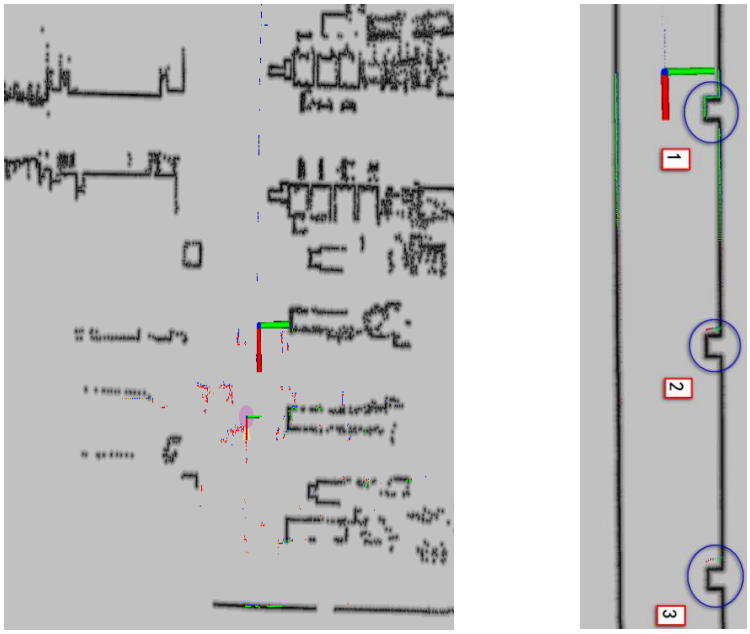
Fig8:环境空旷地图

因素二：

相邻时刻里程计变化量和对应相邻时刻激光与地图匹配结果的变化量差异大，2s 内检测到 10 次，则会上报定位丢失。搜索的关键字为：BadStamp_eva_by_Ekf_Match_ num：10，同时有 Pose estimation is BAD!日志输出。出现该类问题一般主要引起的原因环境变化大(如图 Fig.9.a)或者建立的地图本身尺度误差大(如图 Fig.9.b)，导致相邻时刻激光点云与先验地图匹配的结果变化量与对应时刻里程计变化量有明显差异；但小概率也可能是里程计跳变，导致里程计变化量相对激光匹配的变化量差异较大。

因素三：

激光与先验地图的匹配结果和激光与 submap 匹配的结果差异是否很大，如果 5s 内检测到 8 次，则会上报定位丢失。搜索的关键字为：BadStamp_eva_by_localSlam_ num:8，同时有 Pose estimation is BAD!日志输出。这部分因素很少被触发！



a) 大变化环境地图　　b)尺度不对地图

Fig9:大环境变化与尺度差异大地图

3.3 脱轨以及到点精度不够等问题：

该问题需要对照**导航组的日志**进行分析，进入/var/robot/log/nav,

执行 nav-parse 命令，然后打开 task 文件；

上报脱轨的关键字：0x3207！查找报脱轨的时间点；该条任务到达目标点的关键字为：taskid，然后看 waypoint 关键字，看路径通过的道路节点坐标。

涉及到激光二维码混合导航的场景，有可能扫不到二维码上报脱轨，需要保证标定的二维码坐标的准确性，此时打开 odom 文件，搜索关键字：

```
L:Info|M:[loc]|m: dm adjust from laser->08000100 0.029 0.008 0.015
```

其中 08000100 是激光切换二维码的二维码 ID，坐标 0.029,0.008 是切换二维码时，设备相对二维码的偏移量，主要关注横向坐标，尽量保证横向坐标在 1cm 以内。

而从定位算法本身来看，该问题需要查看定位算法的**预测结果**和**匹配结果**差异是否过大，

查看定位算法日志(enml_loc)的关键字为：

里程计预测位姿：befor_opt

激光匹配的位姿：lidarLoc

局部地图匹配的位姿：cur_lo

多传感器融合的位姿 opt_result！

举个栗子：

定位日志中，befor_opt: 16.649 -1.052 1.573 为预测值，lidarLoc: 16.655 -1.041 1.569 为匹配值，opt_result: 16.649 -1.052 1.573 为融合的结果。

[add_lo_edge]: lo_2_lo last_lo: 16.670 -1.032 1.569 cur_lo: 16.665 -1.036 1.567 为 scan 与局部地图匹配前一时刻和当前时刻匹配的结果。

对着脱轨的时间看对应时间前后的日志，先看融合结果 opt_result，看名义上坐标是否就走偏了，特别要注意看下角度，匹配的角度是否就有偏差；接着继续查看预测位姿 befor_opt 和匹配结果 lidarLoc 是否差异大！

如果环境变化很大，激光匹配和里程计预测的结果差异大就比较正常；日志中，环境变化大的参考依据日志关键字是：loc_matchQuality,该关键字表示激光与先验地图的匹配程度。

```
eva_Lidarloc_success_:true, eva_lidarLoc_Score_:0.898, 0.898, loc_matchQuality:0.742
cur_lo: 98.757 -260.544 -2.273
```

下面两图分别是激光匹配与融合结果轨迹，图 1 是激光匹配结果和融合结果有差异横向偏差比较大，而图 2 激光匹配和融合结果就很贴合，定位稳定性好，精度高。

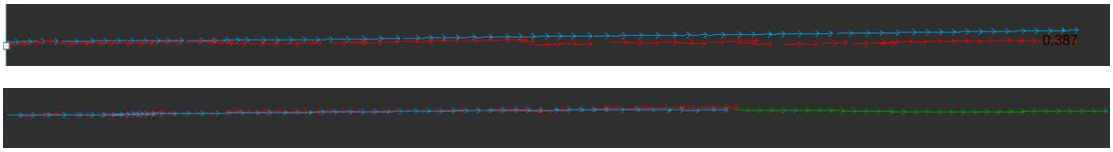


Fig10.激光匹配结果与多传感器融合结果对比图

3.4 日志显示工具

看日志嫌麻烦，可以利用日志显示工具，根据日志显示定位轨迹；

3.4.1 日志脚本分析工具：

执行的脚本文件为 log_exact.py 和 draw.py

使用方法：

a. 先截取出问题前后一段时间的日志；

b. 打开 log_exact.py 脚本，分别将 before_opt(里程计预测)和 lidLoc 关键字填入 re1 和 out 所在行，提取出来 before_opt 和 lidLoc 两个文件。

c. 打开 draw.py,open 中写入 before_opt 和 lidLoc 两个文件所在的目录，就能画出两条轨迹进行对比啦

3.4.2 日志分析动态显示工具(enml_logvisual_plugin 模块)：

a) 进入 resource 目录，打开 logvisual_param.json 文件：

```
"logvisual":
{
    "map_file": "/home/robot/bag/600kg/xiongmao/1029final"
},
```

将上述 map_file 文件名替换成对应的项目的地图；

b) 进入 plugins 下的 enml_logvisual_plugin 目录，启动 view_log.launch 文件；

c) 运行可执行文件(plugin_logvisualize_test)+日志文件

```
robot@robot:~/na_NA/Trunk/build$ ./plugin_logvisualize_test
/home/robot/bag/600kg/xiongmao/enml_loc
```

d) 常用的功能：按空格键，播放日志；字母 d,前进；字母 a,后退；字母 w，放大播放日志的倍数；字母 s，缩小播放日志的倍数；

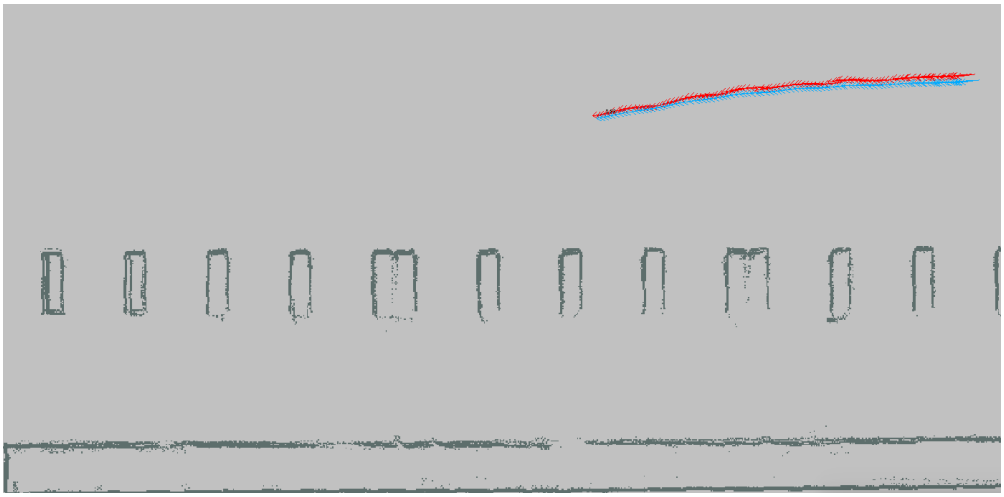
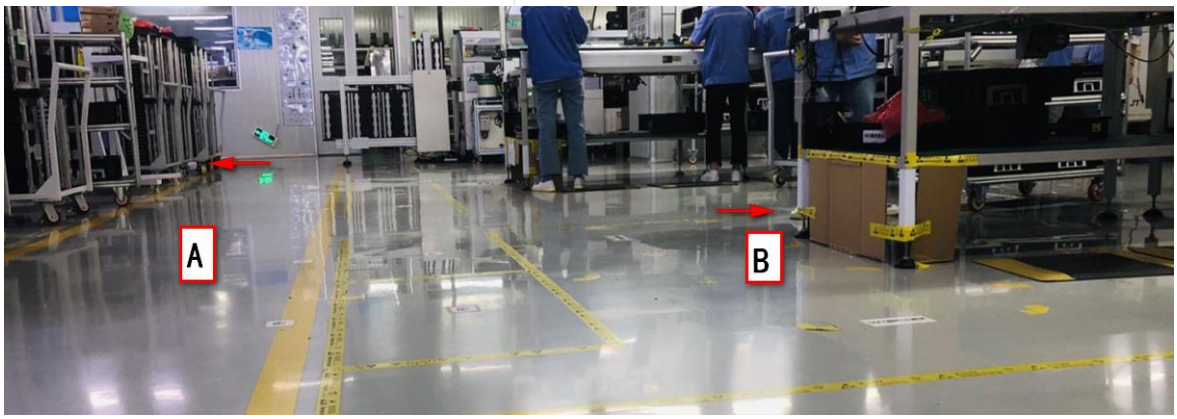


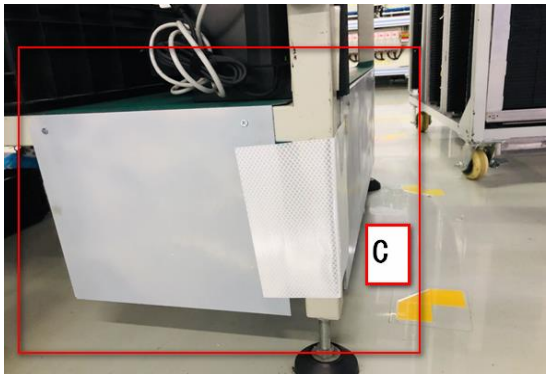
Fig11.日志动态显示示意图

3.5 注意事项和实施意见：

案例 1：

类似美的现场(申洲现场)，如下图一所示，两边都是环境变化非常大的场景，一边是完全变化的(A 方向)，另一边是部分变化，而不变的只有货架腿(B 方向)，此时可以在货架腿之间人为添加钢板，补充特征，并在钢板上添加反光纸，使用反光轮廓定位的方法。如下图三的 C 所示，反光轮廓地图建下图四。





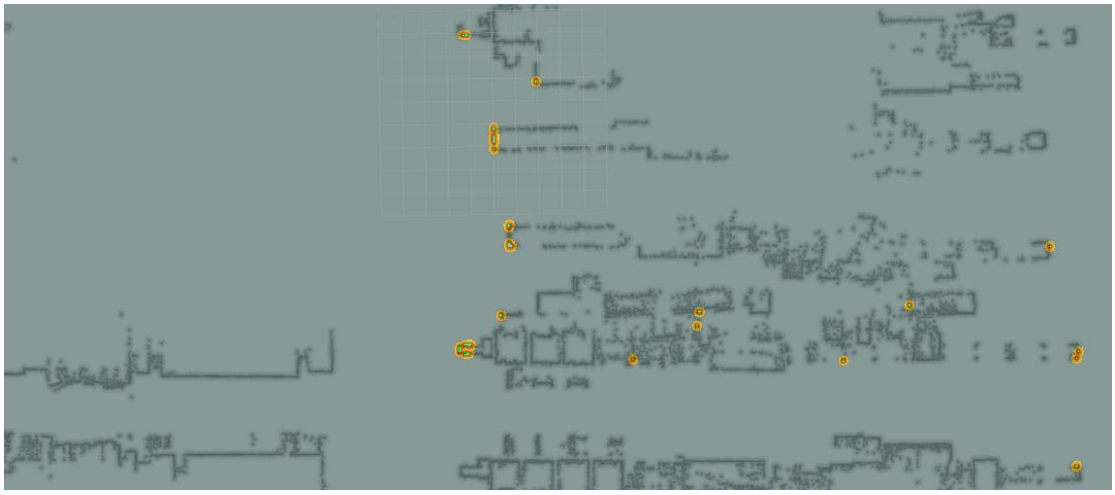


Fig12. 一边环境完全变化，另一边环境不变或者部分变化地图

案例 2：

类似长走廊环境(维信现场，三安现场)，建图的时候最好就要添加人工特征，保证建立的走廊尺度没有问题；沿着走廊方向没有前进方向的约束，此时可以添加反光柱或者反光条。如下图所示，在 A 位置是充电桩，有比较高的到点精度要求点位。此时利用反光条约束前进方向。反光条部署的间隔至少要有 35cm 及以上，且反光条的间隔不要等间距，要错开，连续布置两条以上的反光条。下图中的绿色点就是反光条！

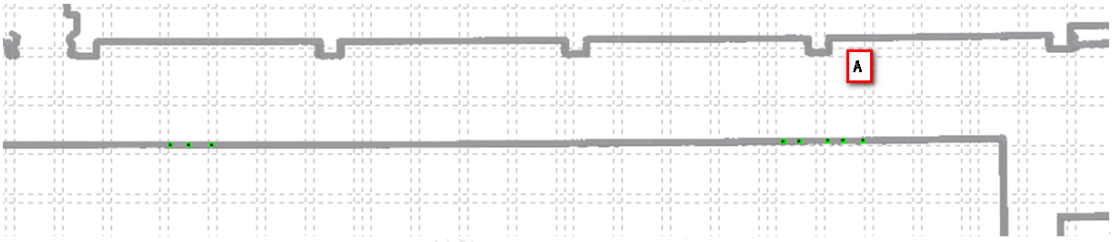


Fig13.长走廊环境地图

案例 3：

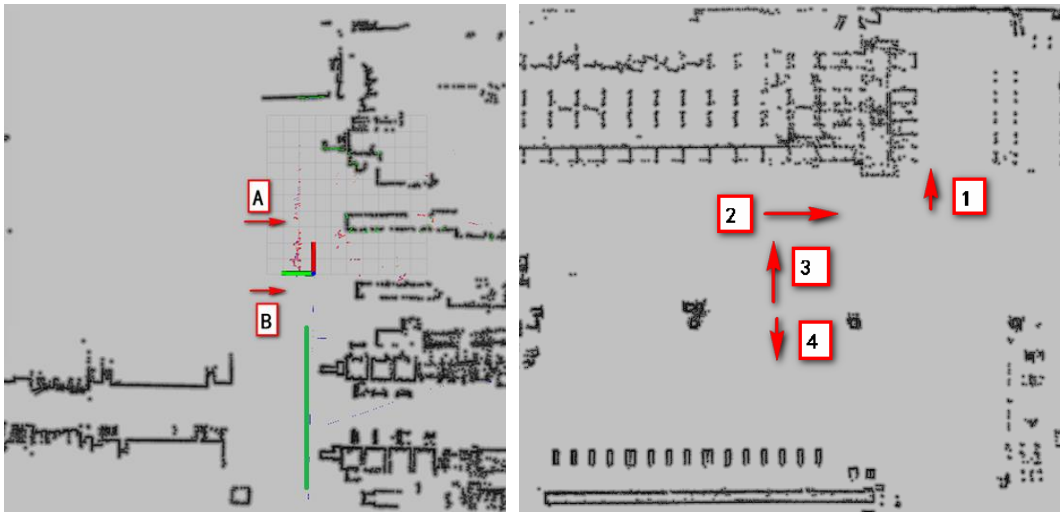
针对凯旺项目，这件这块区域两边都是变化场景，压根就不要使用激光导航，这块区域就要使用纯二维码导航！

Fig14：两边环境完全变化场景

案例 4：

导航时，设置的路线要能最大化激光传感器的使用价值。如下图所示：

图 a 中，设备的激光传感器要对基台方向(如 A、B 的朝向)后退到主干线上(主干道如图中绿色线所示)，然后在主干道上旋转九十度，沿着主干道到下一个基台位置。图 2 中，设备从 1 后退到 2，再对着特征后退到 3，在 3 处做原地旋转，朝着 4 方向，对着明显特征继续前进，这样能保证激光的稳定性和精度相对更好，更高。



a) 变化环境　　b) 空旷环境

Fig15：后退路线设置地图

3.6 反光柱、反光条辅助定位相关日志

3.7 货架识别关键字

四：运动控制问题

附录：

附录 1.配置主客机：

a.登录设备，比如：ssh dahua@172.16.204.103;
然后 vi /etc/hosts,查看设备的名字。
b.打开本地电脑的终端，首先 ifconfig,查看本地电脑的 ip。比如 192.16.204.106.
然后 vi /etc/hosts,添加主机 ip 和名字以及客机名字和 ip；其中主机为设备，客机为本地电脑。
比如：
192.16.204.106　　dahua-E40-70
172.16.204.103　　ubuntu #设备 ip 和名称

c.vi ~/.bashrc
继续添加主客机
export ROS_IP=192.16.204.106　#本地电脑 ip
export ROS_MASTER_URI=http://172.16.204.103:11311　#主机 ip
最后 source ～/.bashrc 就可以直接运行 rviz 看点云、地图和运动轨迹啦！

附录 2.定位模块主要查看的话题名称：

a.长期点云、短期点云：
/enml_offline_ltf_cloud(长　绿色)　和 /enml_offline_ltf_cloud(短　红色)
b.先验地图：
/enml_costmap
c.设备位置：
/enml_broadcast_pose
d.反光柱相关话题：
/cluster

附录 3.导航相关常用命令：

3.1.打开里程计：
rostopic pub /set_odom_freq std_msgs/Int8 "data: 110" -1
3.2.打开下视相机：
rosservice call /dm0/setDropFrame "enable: false"
然后按 d 指令，当设备经过二维码时，就能查看设备相对于二维码的偏移量！
3.3.调用重定位(单位是毫米)：
rosservice call /nav_monitor/set_init_pose "{pose_x: 0, pose_y: 0, pose_theta: 0, width: 0, height: 0, isFromNav: false}"
3.4.查看以及设置定位模式(注意：二维码模式下，是不允许调用重定位的)
a.设置定位模式，一般主要设置两种 mode:2 表示激光模式　mode:3 表示二维码模式
rosservice call /nav/set_navlocalization_mode "mode: 0
fromInternal: false"
b.查看定位模式：
rosservice call /nav/get_navlocalization_mode "{}"
3.5.截取下视相机拍摄的二维码图片，使用 snap 命令，保存的相机图片在 /var/robot/log/nav/snapshoot/目录下！

附录 4.改大定位精度阈值：

当前平台、调度和导航都会判断到点精度，当到点精度不在他们设置的阈值范围内，就认为

设备不在点上，从而拒绝后面的路径规划，出现卡死不动的现象。
而平台又对每个点是统一的到点精度要求，在一些项目中，到点精度没必要这么高，此时就需要修改到点精度阈值，需要修改三个位置：
4.1.平台和调度的到点精度阈值修改，通过客户端进行修改，电脑需要下载 navicat 软件，登入该平台。账号和密码分别为：
用户名：wms
密码：CCshenda889
进入 wms 下的 agv_model 目录下的，查找关键字 rptsNodePrecision 和 nodePrecision,这两个参数的值都要修改，默认是 30mm，根据现场直接更改，一般更改为 50mm。
4.2.导航组的到点精度阈值修改：
roscd nav_config，进入该目录下，找到 nav.yaml 文件，打开该文件，
修改 startPositionToleration 参数，默认值为 0.035m，一般更改为 0.055m

附录 5.agv 调度平台的使用和地图编辑工具的使用：

AGV 调度平台的账号和密码为：
用户名：admin
密码：DHRTA@2018
5.1 通过平台的配置管理->区域管理添加新的区域，然后加载地图，然后添加设备！如果出问题再加载一次地图；
5.2 通过配置管理->任务配置，就可以添加任务类型，比如两点移动， 负载运行，搬运货架等任务，然后通过任务管理->任务池，就可以添加任务，发任务让设备运动起来！
5.3 地图编辑工具直接打开地图文件，可以编辑拓扑点，也可以查看地图的拓扑关系。首先打开已有地图，然后点击预览地图；

接着就能查看拓扑地图以及拓扑地图中各个点的 id 和对应坐标。