

# 定位与导航实验报告

Wugang Meng

Xiang Feng

哈尔滨工业大学(威海)

版本: 0.1

日期: 2021年5月17日

## 摘要

本文介绍了《\*\*智能感知系统》项目中的定位与建图实验、自主导航实验的理论基础、详尽的软件执行流程和具体参数。为了本次实验的可复现性，除提供程序源码外还详述了本次实验的测试大纲。包括但不限于本次实验的测试环境，软件硬件条件以及实验的具体步骤和已达到的实验指标，并体现了实验结果的有效性。

**关键词:** 定位与建图，自主导航，试验大纲

## 1 定位与导航系统程序设计与实现

### 1.1 自适应蒙特卡洛定位系统

自适应蒙特卡洛定位 (AMCL, Adaptive Monte Carlo Localization) 是一套适用于二维机器人运动的定位算法。其主要的算法原理源于算法1, 算法2, 和算法3算法的结合与实现。Motion Model 和 Sensor Model，用于 AMCL 中运动的预测和粒子权重的更新。本课题内部集成的 AMCL，功能上添加了随机角度、随机位置和角度的初始化支持，以便于竞技比赛中的快速部署。

AMCL 算法的设计详细参数如所表1所示：

### 1.2 全局路径规划系统

全局路径规划（简称全局规划）是导航系统中运动规划的第一个步骤，在给定目标位置后，根据感知的全局代价地图搜索得到一条无碰撞的最短路径（即一系列离散的坐标点），然后作为输入传递给局部轨迹规划模块控制机器人的具体运动。核心规划节点的执行流程如下：

1. 初始化 Actionlib Server，构建可视化的 publisher。
2. 读取参数。
3. 创建 tf listener, global\_costmap 对象。
4. 创建具体算法的 planner 对象。
5. 回调队列在主线程开始回调，同时线程 Actionlib Server Callback 也开始回调。

规划线程执行流程和 Actionlib Server Callback 线程执行流程如图2和图3所示。

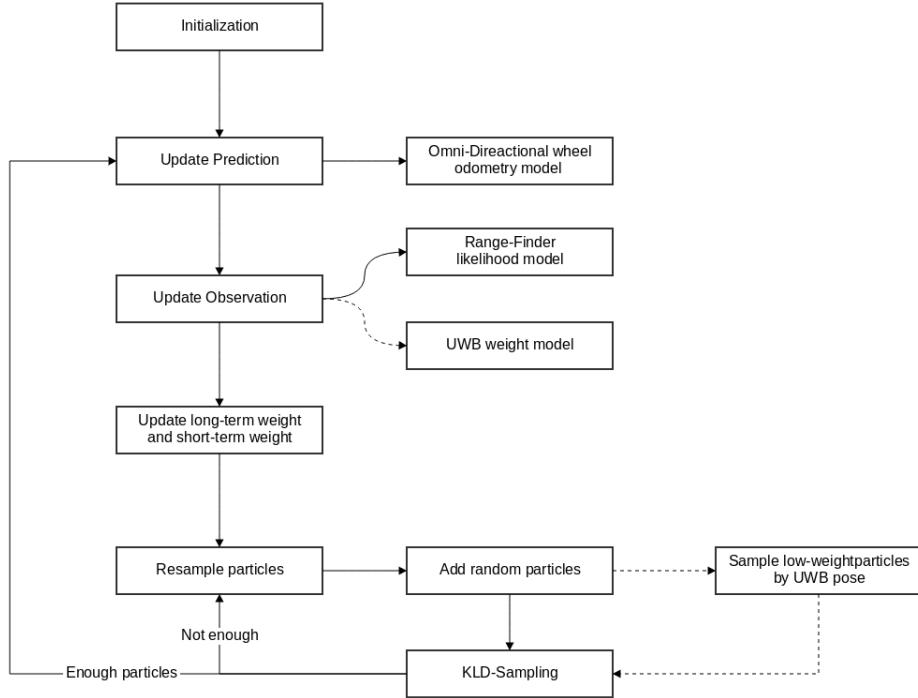


图 1：自适应蒙特卡洛定位算法流图

<b>Input:</b> $X_{t-1}, u_t, z_t, m$
<b>Output:</b> $X_t$
1 $\bar{X}_t = X_t = \emptyset$
2 <b>for</b> $m = 1$ <b>to</b> $M$ <b>do</b>
3     sample $x_t^{[m]} \sim p(x_t   u_t, x_{t-1}^{[m]}, m)$
4     $w_t^{[m]} = p(z_t   x_t^{[m]}, m)$
5     $\bar{X}_t = \bar{X}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
6 <b>end</b>
7 <b>for</b> $m = 1$ <b>to</b> $M$ <b>do</b>
8     draw $i$ <b>with probability</b> $\propto w_t^{[i]}$
9     add $x_t^{[i]}$ <b>to</b> $X_t$
10 <b>end</b>

Algorithm 1: 蒙特卡洛定位算法

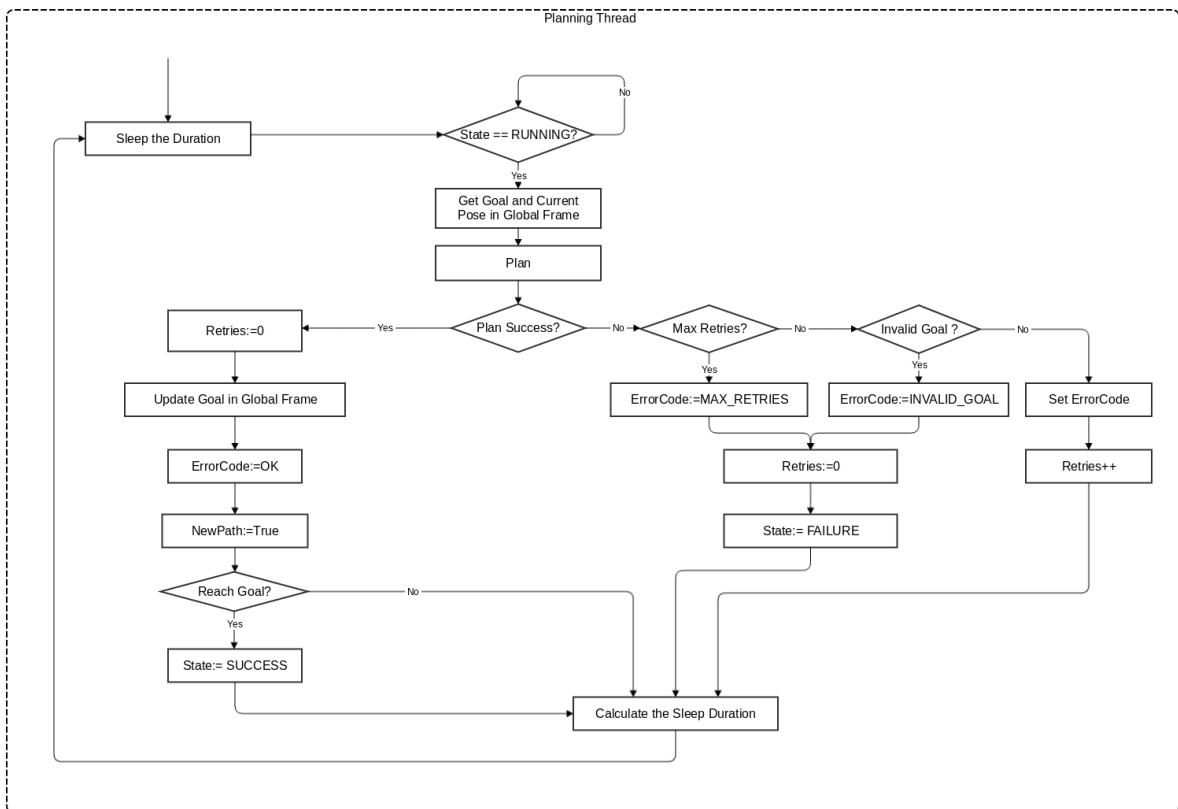
```

Input:  $X_{t-1}, u_t, z_t, m$ 
Output:  $X_t$ 

1 static  $w_{slow}$   $w_{fast}$ 
2  $\bar{X}_t = X_t = \emptyset$ 
3 for  $m = 1$  to  $M$  do
4    $sample x_t^{[m]} \sim p(x_t | u_t, x_{t-1}^{[m]}, m)$ 
5    $w_t^{[m]} = p(z_t | x_t^{[m]}, m)$ 
6    $\bar{X}_t = \bar{X}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$ 
7    $w_{avg} += \frac{1}{M} w_t^{[m]}$ 
8 end
9  $w_{slow} += \alpha_{slow} (w_{avg} - w_{slow})$ 
10  $w_{fast} += \alpha_{fast} (w_{avg} - w_{fast})$ 
11 for  $m = 1$  to  $M$  do
12   draw  $i$  with probability  $\propto w_t^{[i]}$ 
13   add  $x_t^{[i]}$  to  $X_t$ 
14 end

```

**Algorithm 2:** 增强蒙特卡洛定位算法



**图 2:** 全局规划线程执行流程

```

Input:  $\mathcal{X}_{t-1}, u_t, z_t, m, \varepsilon, \delta$ 
Output:  $\mathcal{X}_t$ 

1  $X_t = \emptyset$ 
2  $M = 0, M_\chi = 0, k = 0$ 
3 for all  $b$  in  $H$  do
4    $b = \text{empty}$ 
5 end
6 while  $M < M_\chi \|\| M < M_{\chi min}$  do
7   draw  $i$  with probability  $\propto w_t^{[i]}$ 
8    $x_t^{[m]} = \text{smaple\_Motion\_model}(u_t, x_{t-1}^{[m]})$ 
9    $w_t^{[m]} = \text{measurement\_model}(z_t, x_t^{[m]}, m)$ 
10   $X_t+ = \langle x_t^{[m]}, w_t^{[m]} \rangle$ 
11  if  $x_t^{[m]}$  falls in empty bin  $b$  then
12     $k+ = 1$ 
13     $b = \text{non - empty}$ 
14    if  $k > 1$  then
15       $M_\chi := \frac{k-1}{2\varepsilon} \{1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}} z_{1-\delta}\}$ 
16    end
17  end
18   $M+ = 1$ 
19 end

```

**Algorithm 3:** 库尔贝科-莱布勒散度采样蒙特卡洛定位算法

表 1: AMCL 设计参数

参数名称	参考数值
粒子滤波器的最小粒子数	50
粒子滤波器的最大粒子数	200
滤波器更新的位移阈值	0.1
滤波器更新的旋转阈值	0.5
机器人运动模型	全向轮模型
里程计模型的误差参数	0.05
雷达的最小有效测距距离	0.15m
雷达的最大有效测距距离	8m
雷达的波束	90°
忽略波束的障碍物距离阈值	20cm
重采样周期	不重采样
$\alpha_{slow}$	0.001
$\alpha_{fast}$	0.1
$\varepsilon$	0.05
$1 - \delta$	0.99

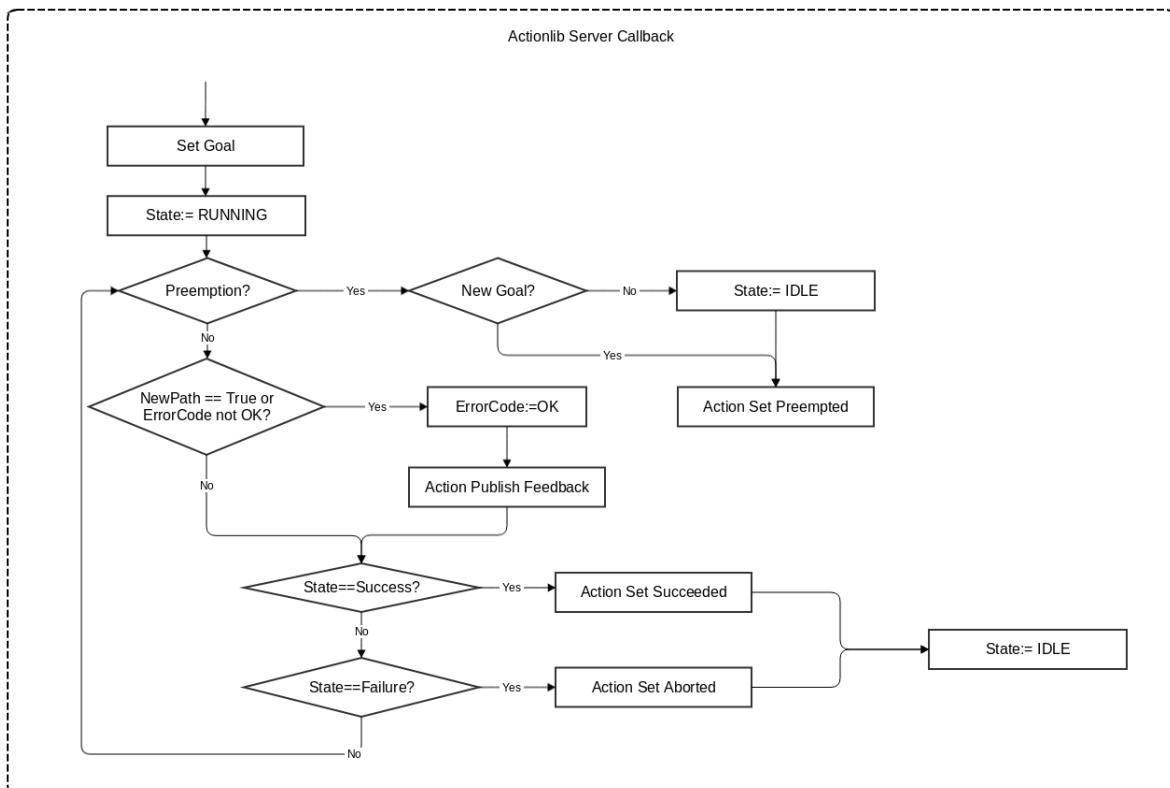


图 3: Actionlib Server Callback 线程执行流程

## 2 测试概要

### 2.1 测试目标

进行本实验的目的是验证毫米波传感器能否与机器人操作系统（ROS）环境中流行的地图和导航库一起使用，这是许多机器人技术人员所熟悉的。本实验将 Octomap 服务器和 move\_base 库与 TI 的 mmWave ROS 驱动程序软件包以及 DJI RoboMaster 底盘库一起使用，以与 TI mmWave 传感器接口。本实验支持使用 IWR1443ISK ES1.0 EVM 或 IWR1443ISK ES2.0 EVM。借助 TI 驱动程序和ROS 社区的软件，工程师可以快速，轻松地评估机器人导航和避开物体。本实验室采用四个传感器一起使用。通过使用四个传感器，该机器人可以具有 360° 视野，因此该机器人能够检测周围的物体，以便更好地进行地图绘制和导航。

### 2.2 测试环境

#### 2.2.1 场地环境

本实验采用 DJI RMUC 的核心比赛场地，也被称为“战场”。如图所示<sup>4</sup>战场是一个长为 28 米、宽为 15 米的区域，内部为木质结构，表面贴地胶（厚度 3mm），主要包含基地区、高地区、资源岛区、补给区和飞行区等。战场外围有上边沿距离战场地面高度为 2.4 米的黑色钢制围挡。全文描述的所有场地道具的尺寸误差均在 5% 以内。场地说明图纸尺寸参数单位为 mm。

#### 2.2.2 硬件设备

移动作战平台基于模块化设计，分为动力模块，感知模块，控制模块和计算模块。所有模块支持快拆结构，各模块可单独编程与调试使用。

##### 1). 动力模块

动力模块是承载和安装机器人动力系统及其附属部件的机构。如图<sup>5</sup>所示使用麦克纳姆轮，支持全向运动。动力系统由 RoboMaster M3508 P19 无刷直流减速电机（图<sup>6a</sup>）和 RoboMaster C620 电子调速器组成（图<sup>6b</sup>）。C620 无刷电机调速器采用 32 位定制电机驱动芯片，配合磁场定向控制（FOC）技术，实现对电机转矩的精确控制。M3508 直流无刷减速电机是专为中小型移动平台和机器人等量身打造的高性能伺服电机，可搭配 C620 电调实现正弦驱动，相比传统方波驱动具有更高的效率、机动性和稳定性。本产品减速箱减速比约为 19:1。

##### 2). 感知模块

感知模块应组装一个远距离分辨率联模组 mmWave-CAS EVM 如图<sup>8</sup>和四个 mmWave EVM 如图<sup>9</sup>所示。根据 EVM 的版本，安装可能会略有不同。在所示示例中，用于安装 mmWave EVM 的支架是 3D 打印的。对于本演示而言，至关重要的是，必须每隔 90 度将 mmWave EVM 安装在机器人周围，与机器人中心等距。在所示示例中，将 12V 至 5V 转换器安装在顶板中心下方，而 USB 分配器则放置在顶板下方。四个 EVM 使用 USB 分配器连接到计算模块。

##### 3). 控制模块

控制模块使用 RoboMaster 开发板 C 型（STM32F427）作为主控板，开发板主控芯片为

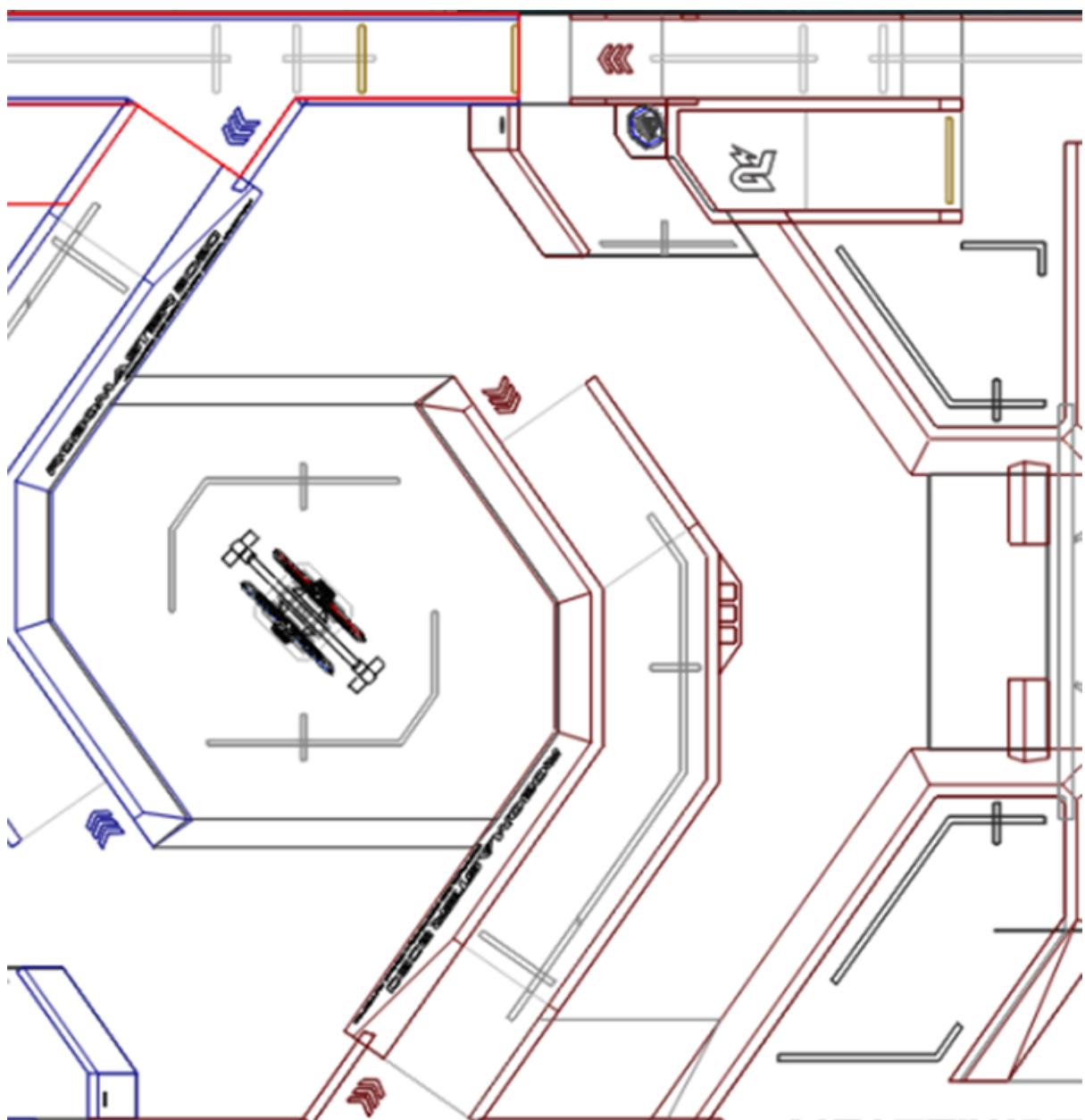
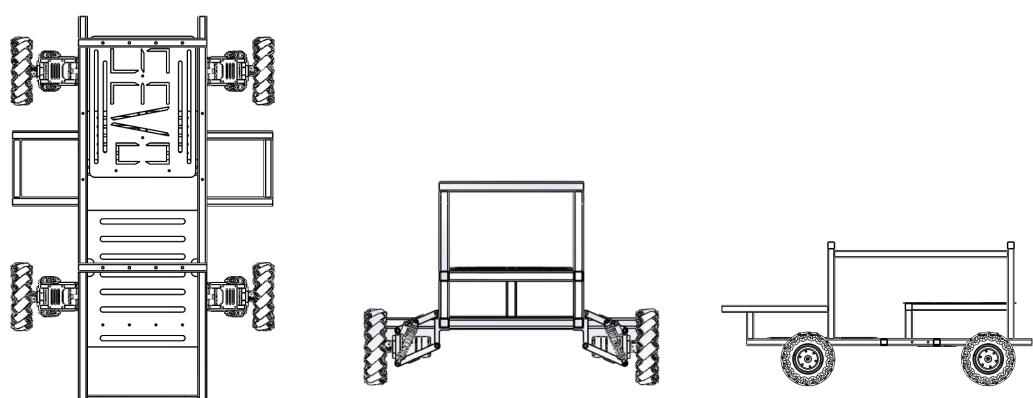


图4：测试场地渲染图

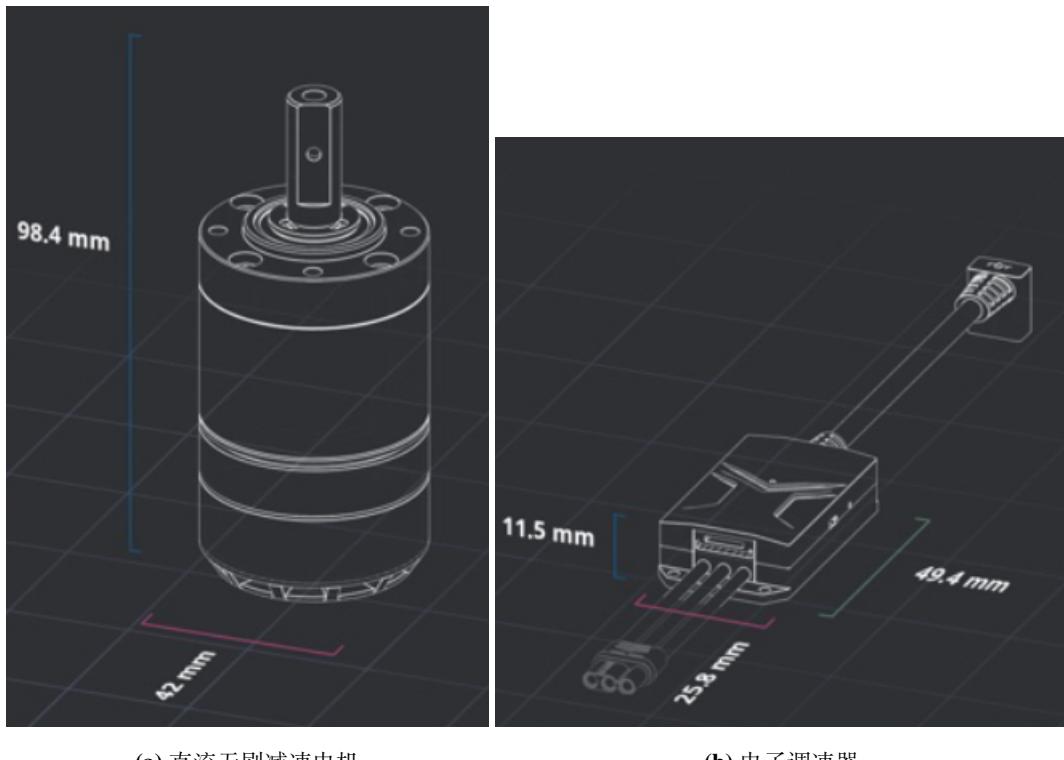


(a) 俯视图

(b) 主视图

(c) 侧视图

图5：动力模块设计图



(a) 直流无刷减速电机

(b) 电子调速器

图 6: 电机与电子调速器设计图

STM32F427IHI6，拥有丰富的扩展接口和通信接口包括 12V/5v/3.3v 电源接口，CAN 接口，UART 接口、可变 PWM 接口、SWD 接口，板载 IMU 传感器，可配合 RoboMaster 出品的 M3508、M2006 直流无刷减速电机、UWB 模块以及计算模块等产品使用，亦可配合 DJI 飞控 SDK 使用，配件丰富。

#### 4). 计算模块

计算模块使用的是英特尔酷睿 i7-8550U 处理器，功率范围 5-60W，具备优秀的处理能力和响应速度的同时仅有 200g 左右的超轻重量。计算模块详细参数见表2。

表 2: 计算模块规格参数

参数名称	参考数值
重量	205g
尺寸	91 × 61 × 35 mm
处理器	i7-8550U
内存	8GB 64bit, DDR4 2400MHz
SATA-SSD	256GB
网络	千兆以太 RJ-45 接口
USB	USB3.0(Type A)×2, USB3.0(Micro-B)×1
I/O	UART ×1
功率	5-60w
电源	15.2-27.0V 电源接口 ×2
工作温度	-25 至 45°C



图 7: 计算模块



图 8: mmWave-CAS EVM

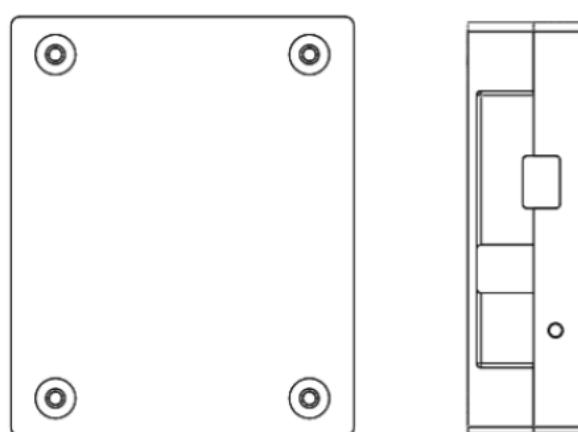


图 9: mmWave EVM

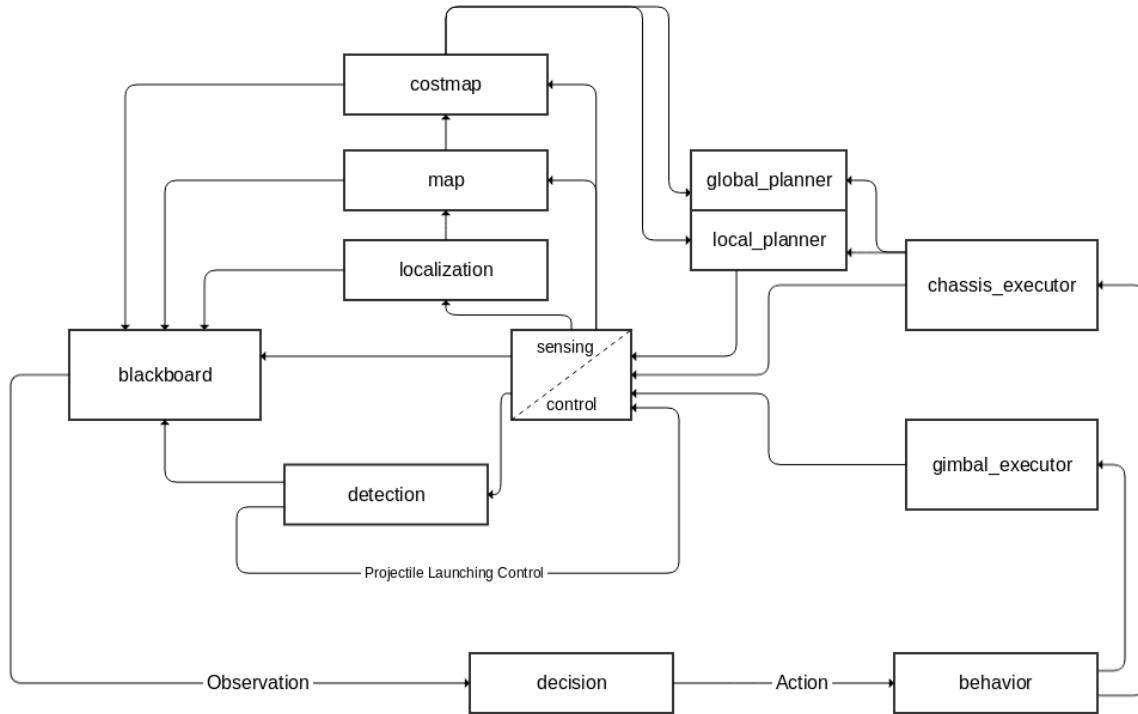


图 10: 架构与模块介绍

### 2.2.3 软件平台

软件平台以机器人传感器 → 感知 → 决策 → 规划 → 控制 → 执行器的环路进行架构，不同模块具体以 ROS Package 的形式维护，模块和其数据流如下图所示。

#### 1). 传感器、控制器与执行器

中心模块集成传感器模块（雷达、相机、IMU 等）、嵌入式控制平台（执行实时任务，如闭环控制和数据采集与预处理）与执行器（电机等），负责 sensing 和 control 两大任务，具体 ROS Package 为包含嵌入式 SDK 的 roborts\_base，相机的 roborts\_camera 以及相关传感器驱动包。

#### 2). 感知部分

感知部分包括机器人定位、地图的维护和抽象、目标识别与追踪等。localization 模块负责机器人定位，是导航系统中强依赖的节点，其主要通过获得一系列的传感器信息，通过特定算法的处理，最终得到机器人坐标系到地图坐标系的变换关系，也就是机器人在地图中的位姿。map 模块负责机器人地图维护，目前采用 ROS 开源 Package map\_server。costmap 模块负责代价地图维护，集成了静态地图层，障碍物层和膨胀层，主要用于运动规划部分，不单纯针对规划使用。detection 模块负责目标识别和追踪，主要利用 mmWave-CAS EVM 对拍摄到的点云信息进行物体的聚类与跟踪。

#### 3). 任务调度与决策部分

任务调度与决策部分包括调度感知输入模块和调度规划执行输出模块的接口，以及决策的核心框架。decision 模块为机器人决策框架，以行为树(BehaviorTree)为决策框架。blackboard 模块调度各种模块的感知任务获取信息，behavior 模块集成了离散动作空间的各种动作或

行为。`executor` 模块是 `behavior` 模块的依赖，其包含底盘和云台内不同模块内不同抽象程度的机器人任务委托接口（例如调度底盘运动规划执行）。

#### 4). 运动规划部分

运动规划部分是运动规划功能模块，由决策部分中 `chassis_executor` 模块来调度完成导航。`global_planner` 模块负责机器人的全局路径规划，全局路径规划（简称全局规划）是导航系统中运动规划的第一个步骤，在给定目标位置后，根据感知的全局代价地图搜索得到一条无碰撞的最短路径（即一系列离散的坐标点），然后作为输入传递给局部轨迹规划模块控制机器人的具体运动。全局路径规划 A Star 算法，是一种静态路网中求解最短路最有效的方法。公式表示为：

$$f(n) = g(n) + h(n)$$

其中  $f(n)$  是节点  $n$  从初始点到目标点的估价函数， $g(n)$  是在状态空间中从初始节点到  $n$  节点的实际代价， $h(n)$  是从  $n$  到目标节点最佳路径的估计代价。在实际的全局路径规划中，静态网路由全局代价地图提供。`local_planner` 模块负责机器人的局部轨迹规划模块，局部路径规划模块根据机器人的里程计信息，雷达的感知信息，结合全局路径规划给出的最优路径，计算出机器人当前能够避免与障碍物碰撞的最优速度。局部路径规划的相关算法参考 Time Elastic Band，算法建立了轨迹执行时间的目标方程，在满足与障碍物保持安全距离，运动时遵守动力学约束的前提下，优化机器人实际运行轨迹，最终求出机器人可执行的速度。

### 2.3 测试方案

在重量和尺寸检测方面，分别称量设备的重量和尺寸。在功耗测量方面，通过稳压限流模块精准控制单一设备功耗。在建图精度测试方面，通过定位时间戳与室内定位基站作为参考，比较测量误差。在定位精度方面，在图4场地进行测试，通过小车车载 LPS 定位装置与 UWB 定位装置，完成规定路径的行驶，通过比较 UWB 与 LPS 的定位航迹，计算 LPS 的定位误差。

### 2.4 条件与限制

本次实验等比例模拟了某区域的城市环境、地面材料及战车尺寸、质量等参数，力图复现一个真实且残酷的战场环境。但仍受以下客观条件限制。

1. 本实验在室内封闭进行，因而无法使用无人机设备进行空中拍摄，故仅提供侧轴视角和 POV 视角测试视频；
2. 因为 UWB、LPS、IMU 和里程计都存在误差，因此定位与建图结果在此误差基础上会存在一定且无法消除的偏移；
3. 高分辨识别雷达（图8）和建图与导航雷达（图9）采用解耦合设计，高分辨识别雷达仅作目标识别之用不参与建图与导航过程；

### 3 测试计划

#### 3.1 测试项目

完成设备重量检测、功耗测量、建图精度测试、定位精度测试等内容，为后续实验提供指标参数与参考依据。测试目标包括高分辨雷达，建图雷达，导航基准为 UWB 接收机。

#### 3.2 测试技术流程

##### 3.2.1 硬件指标测量

直接对设备测量即可。测量项目包括尺寸，质量，设备功耗等。

##### 3.2.2 软件指标测试

首先连接 STM32 设备的虚拟串口，lsusb 可以查看 Vendor 和 Product 的 ID，然后创建并配置/etc/udev/rules.d/roborts.rules 文件：

```
KERNEL=="ttyACM*", ATTRS{idVendor}=="0483", ATTRS{idProduct}=="5740",
MODE=="0777", SYMLINK+="serial_sdk"
```

同理配置雷达。然后重新加载并启动 udev 服务，可能需要重新插拔设备后生效。

```
SUBSYSTEM=="tty", ATTRS{idVendor}=="10c4", ATTRS{idProduct}=="ea70",
SYMLINK+="mmWave_%s{serial}_%E{ID_USB_INTERFACE_NUM}"
```

将北向传感器插入远程笔记本电脑，然后在命令行中键入以下内容：

```
ls -l /dev/ | grep mmWave
```

应出现两个符号链接，一个对应命令端口，一个对应数据端口。重复步骤，插入每个传感器，并在相应的方向启动文件中更改 command\_port 和 data\_port 字段。配置多个相同型号的雷达的会麻烦一些，由于 Vendor 和 Product 的 ID 是一样的，因此要查看每个数据端口具体的特性。

```
udevadm info --attribute-walk --name=/dev/ttyACM*
```

##### 3.2.3 环境感知指标测量

首先如图11中所示，装配整车四片导航雷达和前置摄像头，并如图12所示，将战车放置在图中起点位置。并在图所示场地中红线位置设置障碍物。红点位置设置 LPS 基站，并与战车定位系统同步。将定位值输入电脑中；启动远程控制程序：

```
roslaunch mmwave_launchers my_teleop.launch
```

打开一个新的终端窗口，将 ssh 连接到计算模块并运行以下命令打开所有雷达：

```
roslaunch mmwave_launchers sensor_bringup.launch
roslaunch mmwave_launchers sensor_left.launch
```



图 11：整车装配图

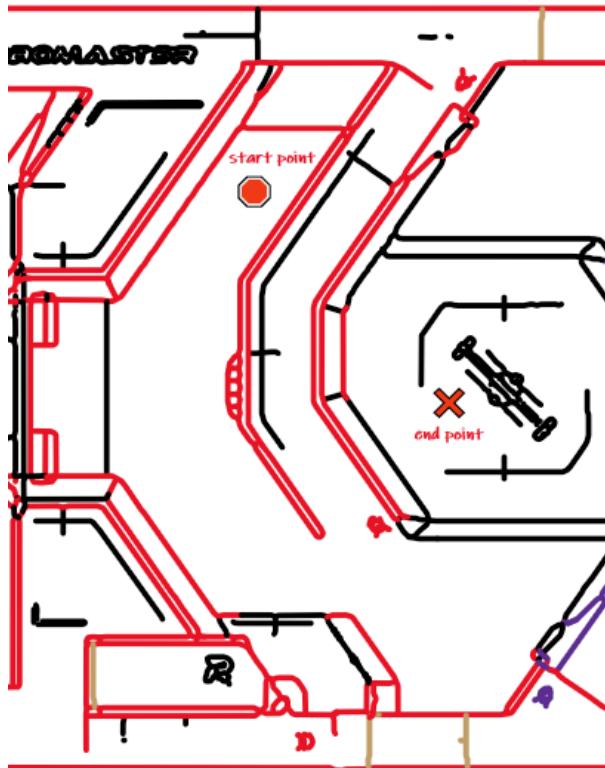


图 12: 实验场配置草图

```
roslaunch mmwave_launchers sensor_right.launch
roslaunch mmwave_launchers sensor_back.launch
```

聚焦远程控制窗口，遥控战车完成建图。

### 3.2.4 自主导航指标测量

首先如图13中所示，获取环境感知实验得到的雷达建图结果。

```
rosrun octomap_server octomap_server_node <path to map>.bt
```

打开导航程序。

```
roslaunch mmwave_launchers crl_navigation.launch
rosrun my_plugin smooth_vel
```

为导航栈提供初始姿态估计，开始导航。选择 2D Pose Estimate（沿屏幕顶部），然后单击战车在地图中的位置，并沿其面对的方向拖动。看到战车出现在起点时，点击后立即释放。再通过选择 2D Nav goal 并单击导航目标到位置并沿其面对的方向拖动，（如图14所示）为战车指定一个导航目标。释放安全遥控后，应该开始导航到它的目标。

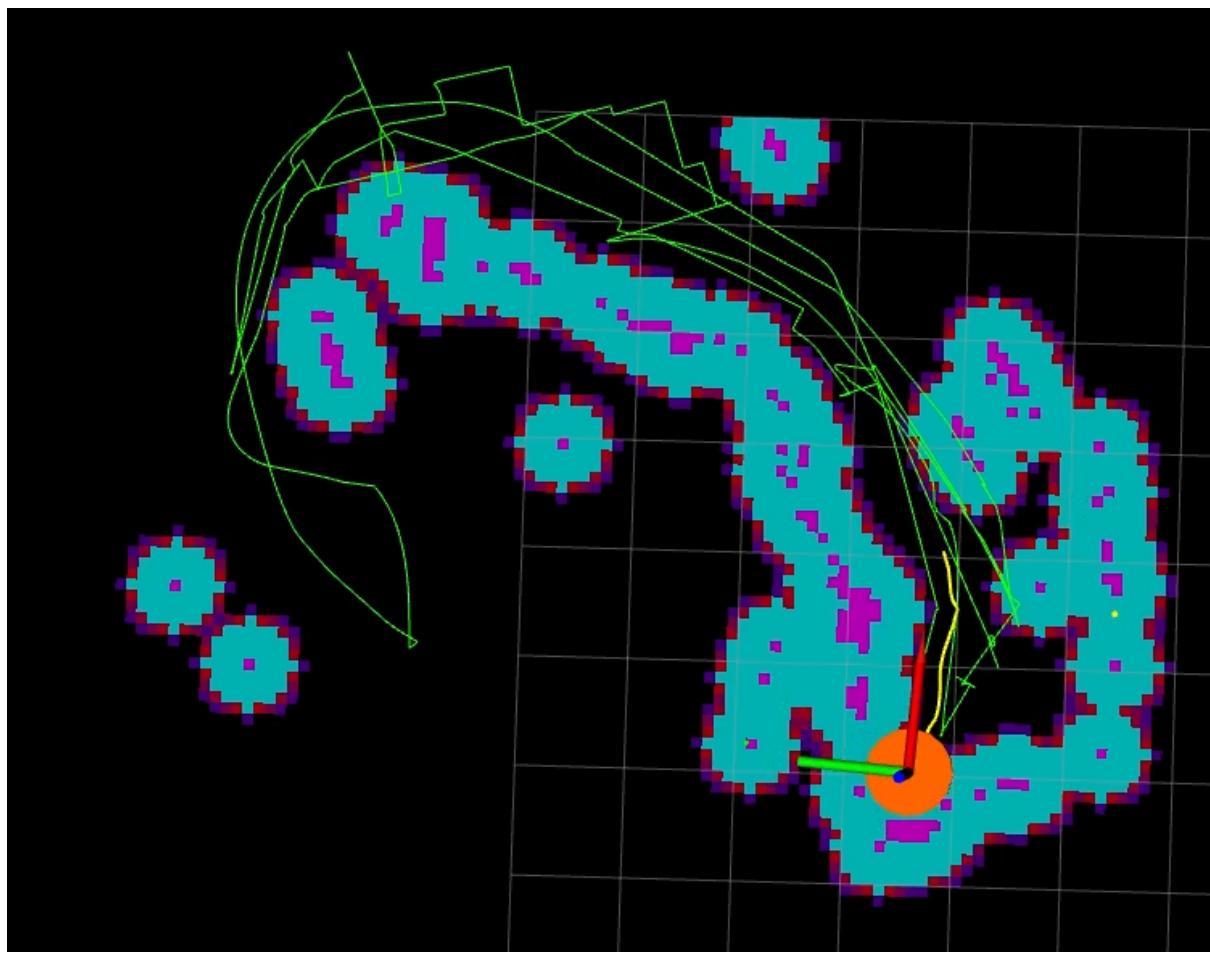


图 13: 雷达建立地图

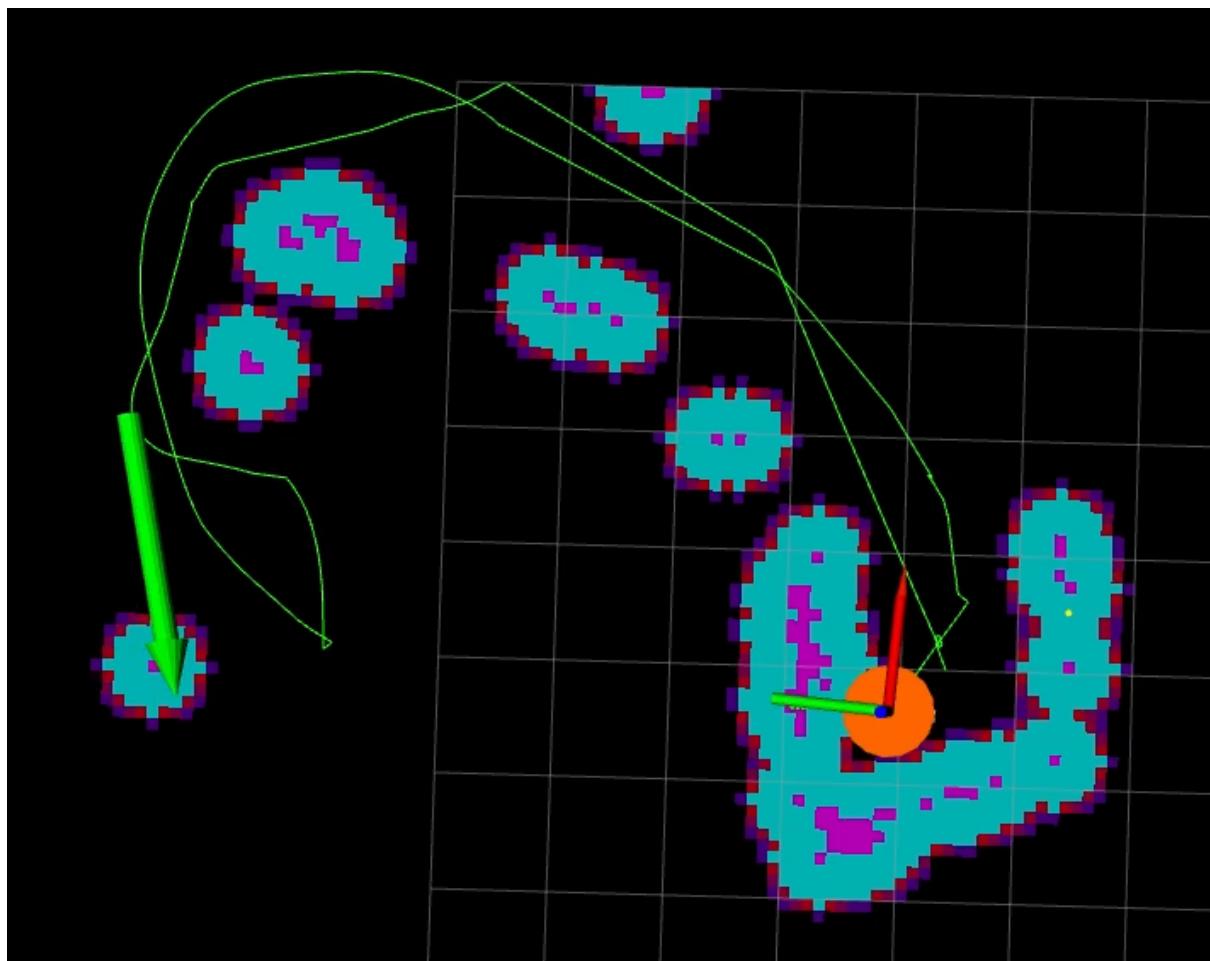


图 14: 提供导航目标



图 15: 高分辨雷达质量测量



图 16: 环境感知雷达质量测量

## 4 测试结果

### 4.1 重量检测

通过对雷达与设备称重，如图15高分辨所示，高分辨雷达重量 1.170kg，达到指标要求。如图16所示，环境感知雷达重量 0.1kg，达到指标要求。

### 4.2 尺寸检测

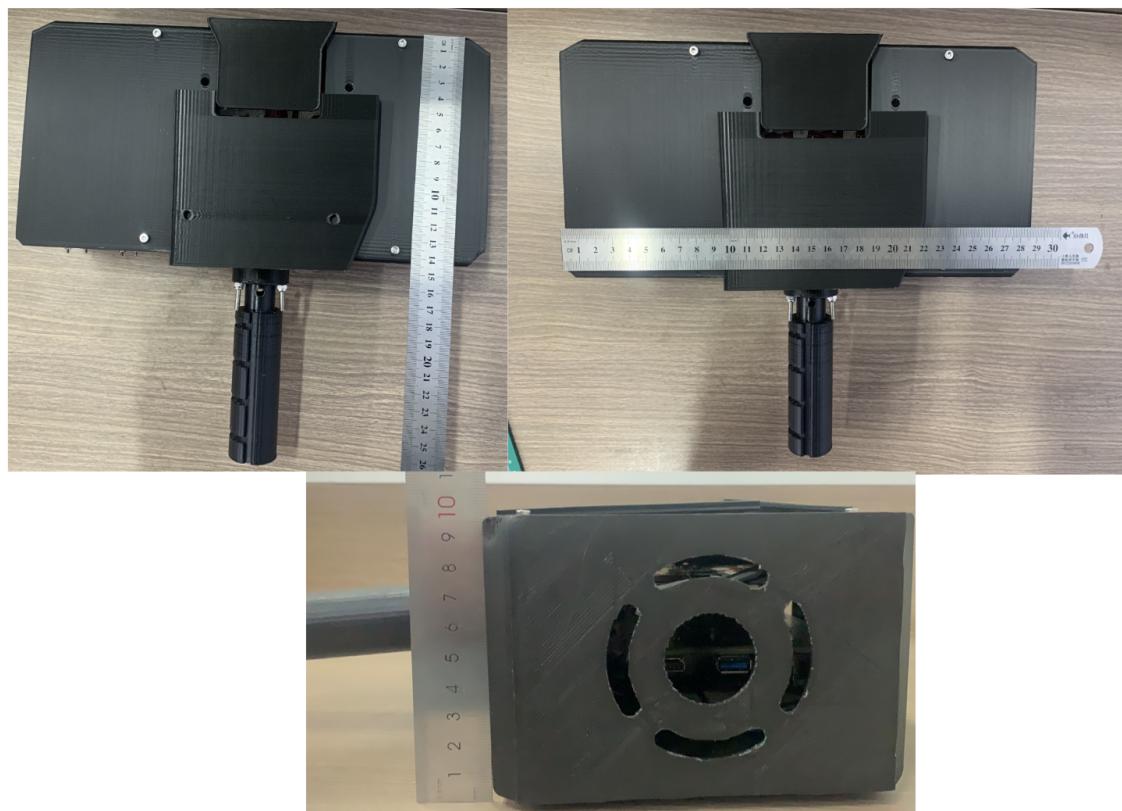


图 17: 高分辨雷达尺寸测量

通过对雷达与设备尺寸测量，如图17所示，高分辨雷达体积  $97mm \times 285mm \times 143mm$ ，达到指标要求。如图18所示，环境感知雷达体积  $15mm \times 70mm \times 80mm$ ，达到指标要求。

### 4.3 功率检测

通过对环境感知雷达利用稳压限流模块驱动，如图所示，工作功率 2.5W，达到指标要求。

### 4.4 环境感知检测

通过对在建图结束时刻与结束时所保存的地图文件与场地设计文件以及 LPS 基站和车载里程计信息交叉比对，并取对误差取绝对值，求得同时定位与建图误差；通过实验，其测距误差如图19所示，误差在 1mm 到 10mm 之间，达到指标要求的 10mm 以内。

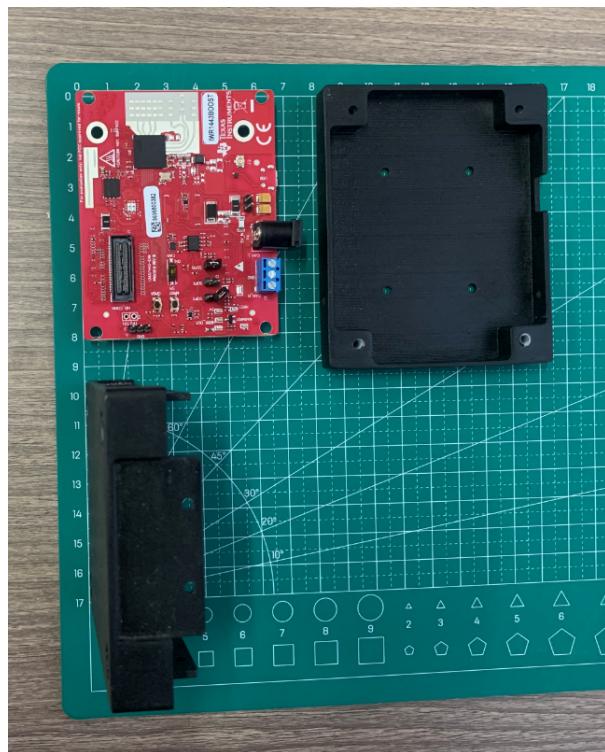


图 18: 环境感知雷达尺寸测量

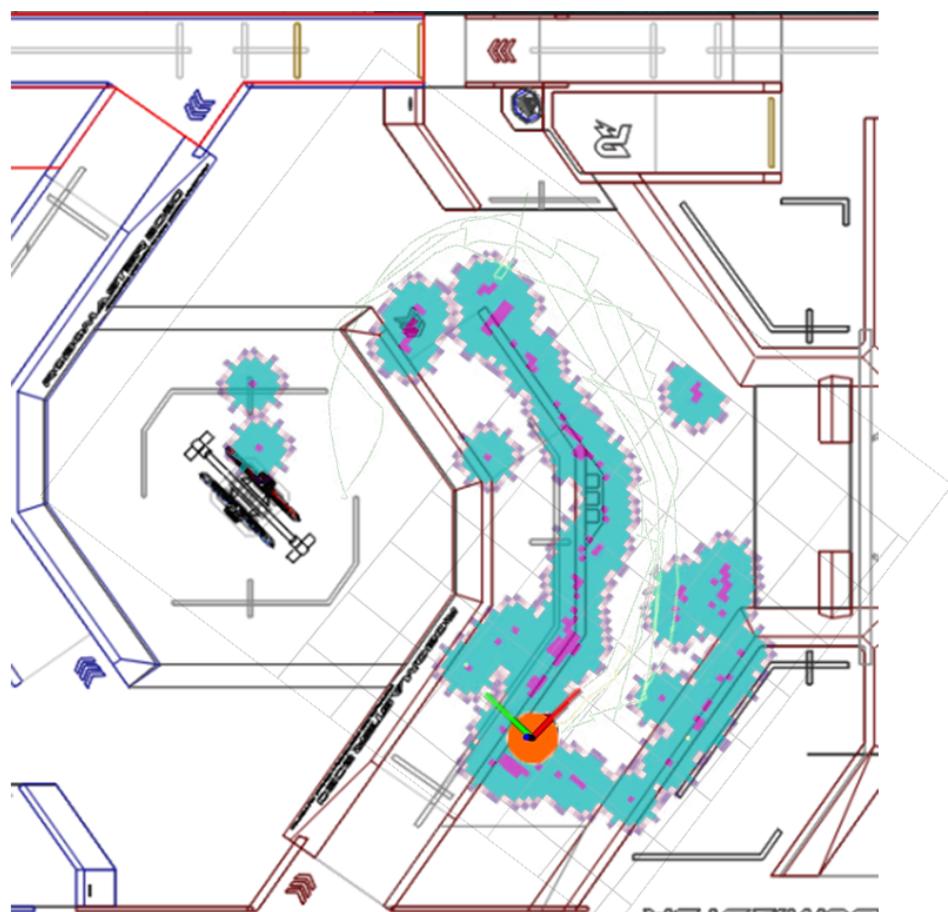


图 19: 环境感知检测结果

## 4.5 自主导航检测

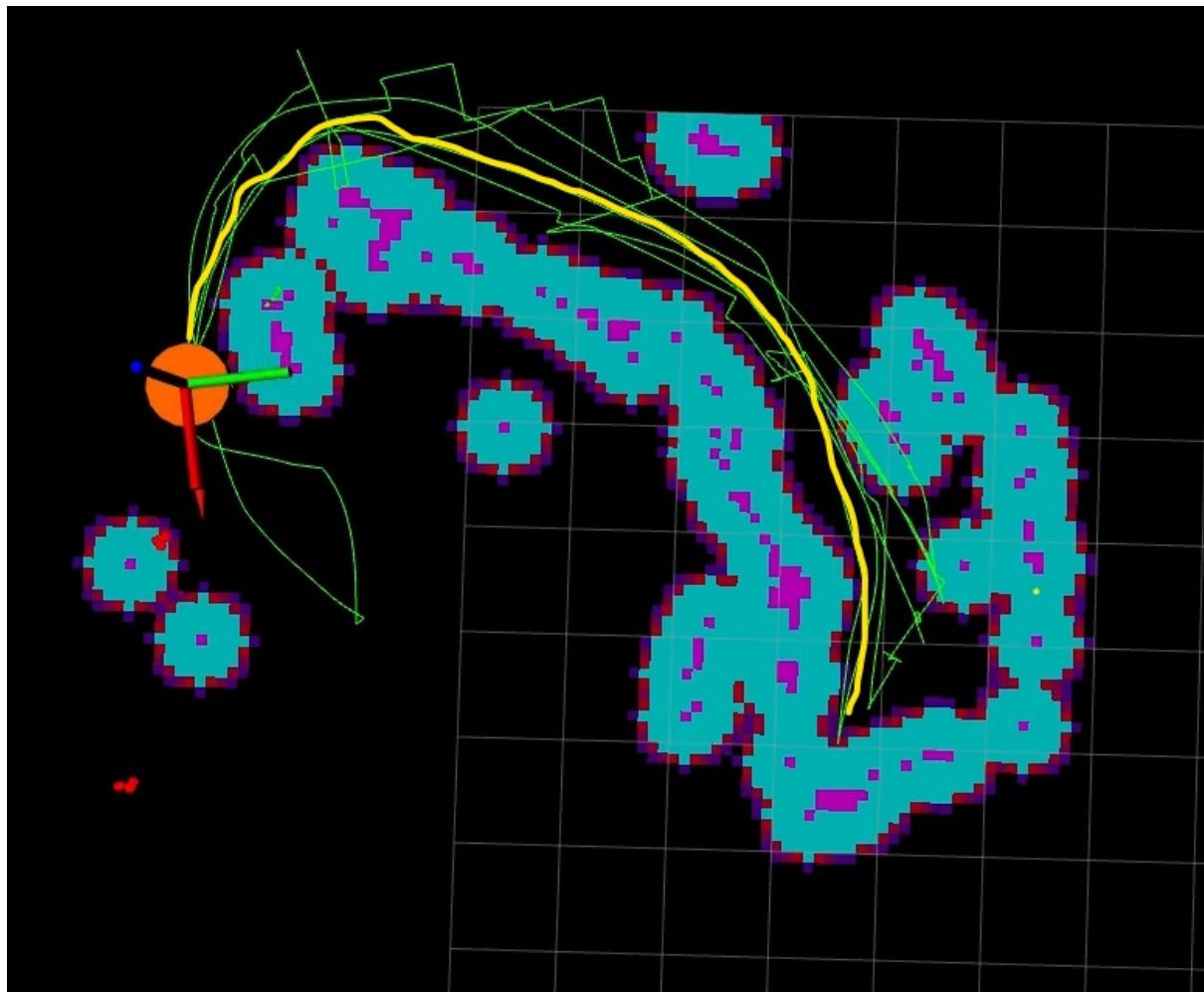


图 20: 自主导航检测结果

图11为实验战车，车上包括环境感知设备与 LPS 设备，场地见图4。按照图14配置车辆后释放战车。战车再运行过程中，黄色线条为实时规划路径。图20可以看出战车在环境感知设备的辅助下无碰撞的完成了避障自主导航。