评分：_____

机器人入门项目设计 II

飞龙绣球期末个人报告

学　院　粤港机器人学院

年　级　黄埔二期

组　别　第三组

姓　名　岑兴设

学　号　3116000892

老　师　陈玮、黄之峰、李东

2018 年 07 月

目　录

1　项目概述

1.1 题目背景及意义

第十七届全国大学生机器人大赛 ROBOCON 以“飞龙绣球”为主题，灵感来自我国广西壮族人民的一项传统体育游戏“抛绣球”。这种游戏历史悠久，最早出现于 2000 年前绘制的花山壁画上，但当时是用青铜铸制的兵器，用于甩投，称为“飞砣”，多在作战和狩猎中应用。后来人们将飞砣改制成绣花囊，互相抛接娱乐。到了宋代，逐渐演变成为壮族男女青年表达爱情的媒介。现“抛绣球”仍在广西百色、柳州、南宁、河池等地区流传。每逢春节、三月三、中秋等传统佳节举行的歌圩中，壮族青年相邀集聚村边、地头、河畔，分成男女双方，互相引亢呼唤，以表达亲切问候。抛球的同时，以对歌相互询问，歌词内容包括理想、情操、农事问答，谈古论今，十分广博。

1.2 规则解读

1.每场比赛由两队在 3 分钟内进行。每队有两台机器人：一台手动机器人和一台自动机器人，或两台自动机器人。只有一台自动机器人可以投掷绣球。

2.比赛场地分为 3 个区域，有两队的活动区和 NC 区（见图册中的图 1.1）。比赛场地外不得放置任何传感器。

3.每一队的活动区中有启动区、放球区、投掷区、手动区。自动机器人启动后可在本队所有区域活动。

4.NC 区中放置着立柱、彩环、金环和金杯。

5.比赛开始前，10 个彩球和 5 个金球放在放球区中。

6.比赛开始后，手动机器人拿起彩球并交给自动机器人。

7.自动机器人收到彩球后，进入 TZ1 或 TZ2 并向彩环投掷彩球。如果绣球成功穿过圆环，得分。

8.在从 TZ1 和 TZ2 至少各投掷一个彩球并成功穿过彩环后，手动机器人才能去拿起金球。

9.从手动机器人那里收到金球后，自动机器人可以进入 TZ3 并向金环投掷金球。如果金球成功穿过金环，得分。

10.如果某队投掷的金球穿过金环后直接落在金杯上且未弹出（不考虑飘带和系绳），该队获胜，比赛结束。这种胜利称为“飞龙”。

11.如果两支参赛队均未实现“飞龙”且比赛时间已到 3 分钟，比赛就结束。比赛结束时得分较高的参赛队获胜。两队得分相同时，按 3.7 判定胜负。

1.3 总方案规划

1.手动车应实现功能：利用导轮贴壁运动，利用全场定位模块实现点对点移动，绣球的抓取和交接。

2.自动车应实现功能；利用巡线模块沿白线运动，利用全场定位模块实现点对点移动，上层平台的旋转，绣球的投射。

3.整体流程图

(a)启动阶段

自动车按下图路线进入 TZ1 区的边缘，并在 TZ1 区边缘处等待手动车与自动车合体并交接球体。

(b)取球阶段

手动车与自动车合体完成，并伸出机械臂与自动车进行交接，这时自动车上将会有两个球。彩球 1 用来投射 TZ1，彩球 2 用来投射 TZ2。

(c)投射 TZ1 阶段

自动车沿 TZ1 白线区域前进，并利用车下方的十字进行姿态矫正。经过矫正后，开始抛投。抛投事件是否成功依靠视觉进行判断。如果抛射不成功，退回取球阶段。成功则进入移动阶段。

(d)投射 TZ2 阶段

自动车沿下图的轨迹前进，并利用车下方的十字进行姿态矫正。经过矫正后，开始抛投。抛投事件是否成功依靠视觉进行判断。如果抛射不成功，退回取球阶段。成功则进入取球阶段并且一次取出两个金球。

(e)投射 TZ3 阶段

自动车沿 TZ2 的白线前进，并利用车下方的十字进行姿态矫正。经过矫正后，开始抛投。抛投事件是否成功依靠视觉进行判断。如果抛射不成功，退回。

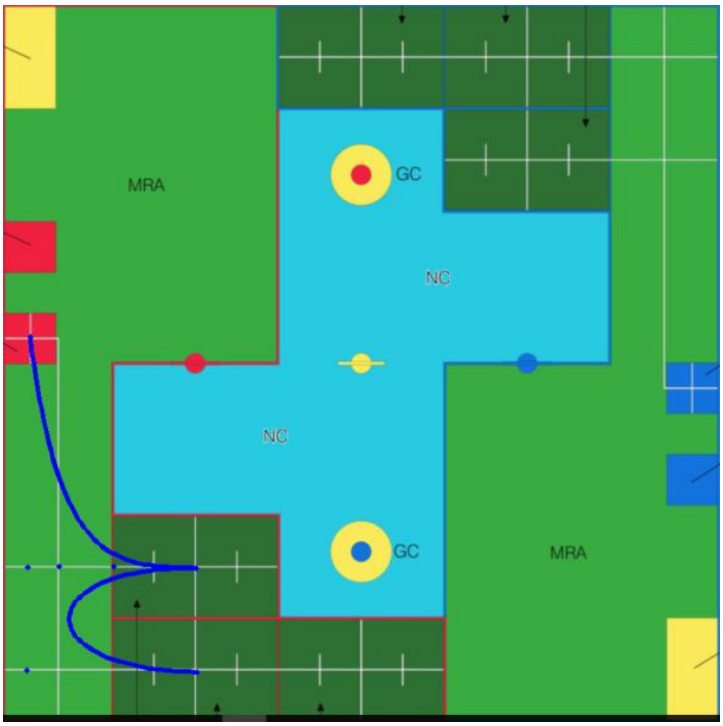


图 1.1 B 组自动车整体移动方案

2　电控部分方案设计

2.1 电控部分实施流程

在自动机器人的电控设计上我们小组使用探索者 STM32F407ZGT6 作为主控板，并通过 CAN 通讯或者串口通讯等与其他模块相接。下面为杂方案自动机器人的电控流程图。

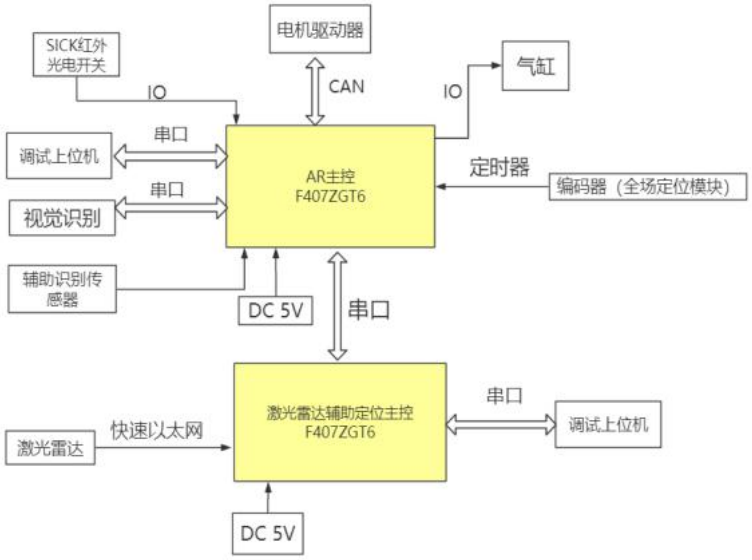


图 2.1 电控部分整体流程图

2.2 雷达

激光雷达是以发射激光束探测目标的位置、速度等特征量的雷达系统。其工作原理为向目标发射探测信号木激光束亦本然后将接收到的从目标反射回来的信号木目标回波亦与发射信号进行比较本作适当处理后本就可获得目标的有关信息本如目标距离、方位、高度、速度、姿态、甚至形状等参数本从而对飞机、导弹等目标进行探测、跟踪和识别。在比赛中如果能使用到这样精度高的设备那么对于我们的定位是大有裨益的，而定位是我们比赛中重要的核心功能。

2.2.1 雷达整体方案

因为在实践时我们的组员发现，全场定位的精度任待改进，误差在 0.2 左右所以我们组打算采用雷达的方案进行全场定位模块的矫正。在关键位置实现自动机器人的微调将雷达安装在机器人最高点，利用雷达检测场上我方赛区的两根柱子来确定机器人的位置。由赛场图可以算出我方自动机器人主要活动区域面积是 7*7M 的正方形。

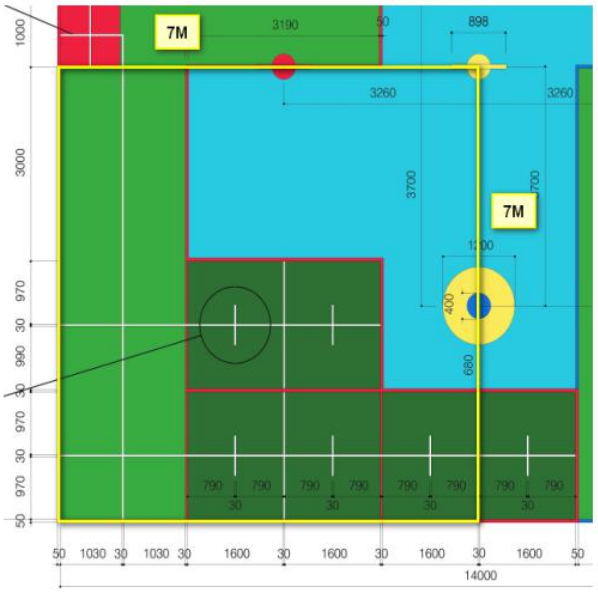


图 2.2 2018RC 场地图 1

从机器人启动到比赛结束需要检测的最远距离是正方形的对角线，经过计算为 9.8M

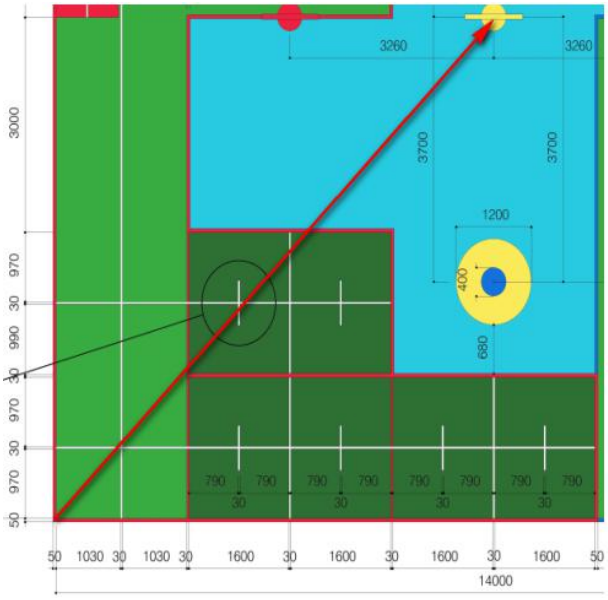


图 2.3 2018RC 场地图 2

雷达能够检测到它距离两根柱子的距离。由三角形相似定理，三边已知就能确定一个三角形。柱子位置已知，则能够知道机器人在世界坐标系中的位置。

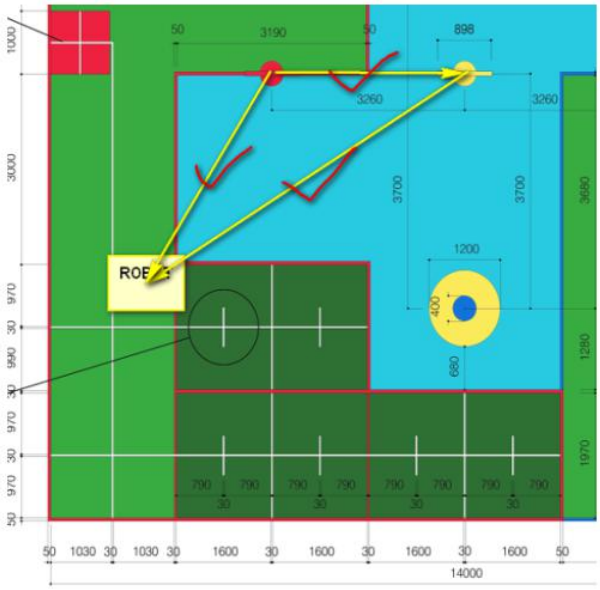


图 2.4 2018RC 场地图 3

2.2.2 雷达选择

柱子直径 60CM，若以十米的最远距离计算，要求雷达的最大分辨率为 360*60/(10000*3.1415*2)=0.3038 下面以 EAI G4 为参照

项目	最小值	典型值	最大值	单位	备注
测距频率	4000	9000	9000	Hz	每秒测距 9000 次
扫描频率	5	7	12	Hz	软件调速
测距范围	0.10	-	16	m	测距频率=4KHz 时
	0.22	-	16	m	测距频率=8KHz 时
	0.26	-	16	m	测距频率=9KHz 时
扫描角度	-	0~360	-	Deg	-
测距分辨率	-	<0.5	-	mm	测距范围<2m
		<实际距离的 1%			测距范围>2m
角度分辨率	0.26	0.28	0.30	Deg	扫描频率 为 7Hz 时

图 2.5 EAI G4 参数表

各情况下角度分辨率均小于 0.34，故能够检测到柱子。以测距分辨率来说，在这一款雷达十米的距离扫描，至少能够采集到一个柱子的数据包，近处扫描应该能采集到三四个。并且 EAI G4 采用 TTL 电平，利用串口进行通信。因为这款雷达运行模式固定为全方位扫描并将所有数据输出，STM32 需要对接收到的数据包检查，并筛选出一定距离范围的数据包。并计算发送给上位机。

2.2.3 雷达方案总结

优点:

1. 避免了检测赛场围栏时，被轮子与全场定位模块遮挡的风险。并且如果将雷达放置在底盘的话就会因为地面不平导致雷达扫描的数据有误。在比赛中如果某一模块没在相应的情况下发挥作用会使得我们的比赛出现很大的隐患。但针对柱子检测的方案可以在整个场地上使用，这保证了系统的稳定
性。

2. 检测距离只要求达到 10M，能够选择雷达的空间变大。激光雷达的价格是昂贵的，但是如果是检测柱子的方案，我们就能节约部分的资金。

缺点：

计算量大。如果只交给 STM32F4 运算的话，可能单片机的运算能力是无法负担的，这导致要需求高计算量的机器。而在比赛现场如果是因为发生意外的话，非单片机的重启时间需要谨慎考虑。启动较慢的仪器会带来隐患，所以这部分要细细斟酌。同时因为激光雷达的底层代码是不开源的，这会给相应的操作者带来很多程序方面的困难。

2.3 投射机构

投射机构是我们这次比赛的另外一个核心技术。在比赛的场上是以投球算分数的，如果我们的投射机构十分完善我们就能在比赛场上取得巨大的优势。在该技术上我们首先进行了数学的建模，并对该模型进行了机构方面的改进。以实现精准地投球。并且根据模型我们选择了合适参数的电机以实现我们的功能。

2.3.1 模型计算

在球体运动的过程中，绳子可视为刚体结构。基于加速度分析直观的可以得到绳结的加速度的大小与方向与抛射臂相同，进而球体必与抛射臂呈 90 度角。基于上述分析，当电机开始减速时，球体将会围绕结点做圆周运动直到结点无法提供向心力。可是我们需要明白的是，减速时绳子不再是刚体模型。所以我们组提出了更换抛射结构，换为可控抛射机构，比如使用舵机控制爪子的方式进行抛射。下面对可控抛射方案进行计算由上述可知，当向心加速度由舵机控制的爪子提供时，球体必与抛射臂呈现 90 度。假设要
在 45 度进行抛射，初速度为 12m/s 则有抛射机构需要在与水平面呈 45 度时停止运动，则球体抛射的抛射角也为朴朵度。由于球体与抛射臂的速度一致，故需要抛射臂在 12m/s 时松开爪子。

2.3.2 电机选型

电机按照工作电源种类来分的话，分为直流电机和交流电机。直流电机又分为无刷直流电动机和有刷直流电动机。无刷直流电动机和有刷直流电动机。有刷直流电动机又可划分为永磁直流电动机和电磁直流电动机。我们组将模型中计算得到的抛射的出射速度，出射角度继续分析得到参数，并就这些参数进行了与商家沟通等步骤最终找到了合适的电机类型。下面是我们组的抛射机构电机的选型 T-motor u8 lite KV150 搭配 ALPHA 60A LV 电调（1 套）参数如下：

KV值	150	最大电流（180s）	29.7 / 26.5
电机重量（含线）	239g	最大功率（180s）	712.8 / 1272
空载电流（@18V）	1A	电调型号	AIR 40A、ALPHA 60A LV FLAME 60A HV、ALPHA 60A HV
相间内阻	85±5mΩ	4轴整机重量（28CF）	8kg
匹配螺旋桨	28" -30" / 22"	6轴整机重量（28CF）	12kg
工作额定电压	6S Lipo / 12S Lipo	8轴整机重量（28CF）	16kg

图 2.6 电控部分抛射机构电机

仅仅有一个合适的电机是不足够的我们还要选择一款相配套的投射机构驱动器，下面是我们的选择的投射机构的驱动器。Elmo guitar 35/48（GUI-35/48）参数如下：

4.10. Communications

Specification	Details
RS-232	**Signals:** • RxD , TxD , Gnd • Full duplex, serial communication for setup and control. • Baud Rate of 9,600 to 57,600 bit/sec.
CAN	**CAN bus Signals:** • CAN_H, CAN_L, CAN_GND • Maximum Baud Rate of 1 Mbit/sec. **Version:** • DS 301 V4.01 **Layer Setting Service and Protocol Support:** • DS 305 **Device Profile** (drive and motion control): • DS 402

图 2.7 电控部分投射机构驱动器

2.3.3 底盘控制

在比赛时，机器人的移动灵活性，与精确性是我们的另一关键技术。而底盘的控制需要一款合适的电机，这款电机需要满足最大电流不超过朴朴杶的电压，并且满足可方便调节电压，提供较大转矩，适于频繁地调整转速的负载的特点。就此我们针对这个情

况进行了计算得到合适的电机参数，并根据参数进行电机的选择。

2.3.3.1 模型计算

我们根据自动机器人在场上的需要的表现，反向推出电机达到这种状态所需要的扭矩与转速。

$$v=2m/s.a=1m/s^2.m=25kg,r=80mm,u=0.13 \tag{2.1}$$

$$F_a=ma=25N \tag{2.2}$$

$$F_f=mg=0.13*25*9.8=31.85N \tag{2.3}$$

$$F=F_f+F_a=56.85N \tag{2.4}$$

$$F'=F/3=18.95N \tag{2.5}$$

$$P=F'v=37.9w \tag{2.6}$$

$$T=F'r=18.95*0.08=1.516Nm \tag{2.7}$$

由上解得扭矩为 0.177*26=4.602Nm>T

转速为 7580/26=291.5r/min

计算的所需机器人轮子转速是 238.8r/min<291.5r/min

2.3.3.2 电机选型

根据上面的计算我们综合对比其他的电机，我们得出杽杆朴杏电机及其配套物资符合我们的需求。故我们选择购买一套 RE40，减速比 26 减速箱与 500 线编码器作为我们投射机构的驱动装置。下面为电机的参数表

图 2.8 电控部分电机参数表

值得注意的是这款电机不仅满足了我们的需求，并且性价比也是很高的。

3 我的工作

3.1 工作目标

使用雷达纠正全场定位的累计误差，完成机器人定位的任务。完成控制组其他比较简单的任务。

了解雷达的类型和工作原理，在全球范围内的雷达产出公司中比较雷达性能、价格和性价比，选择适合比赛的雷达，给购买提出意见。了解与雷达通讯的方式、通讯协议和物理特性等。使用雷达测量的数据，计算出车体的位置和姿态，实现机器人的定位。了解控制气缸和气瓶气压的方式，选择适当的气压检测传感器，制定气压检测和控制方案。

3.2 工作详述

3.2.1 雷达选型

在笔者使用雷达之前，学院没有使用雷达的经验。我们需要从雷达的选型开始。指导老师李冬推荐了一篇题为“全球 12 家顶级激光雷达公司解读：谁有机会撼动 Velodyne”介绍雷达主要产出公司的微信推文，节选截图如下。

图 3.1 文章节选图片

而该文章主要介绍的是用于汽车自动导航的、检测范围大于 100M 的、大多数为 64

线、32 线或 16 线的非常昂贵、性能最优秀的雷达，不适合用于场地只有 14M*14M，主要分析二维平面运动的比赛。受限于当时的阅历，本人只在淘宝上搜索了曾经听说过的 EAI 雷达，进行了一些初步的比较：

该雷达检测半径 16M，覆盖 14*14M 赛场四条边，能够检测更多有用的点。售价 2000 元，花费不高。高度 41mm，比较小，容易装在车体的任意位置。波长为 785nm 红光/红外光，比较容易受红光干扰。

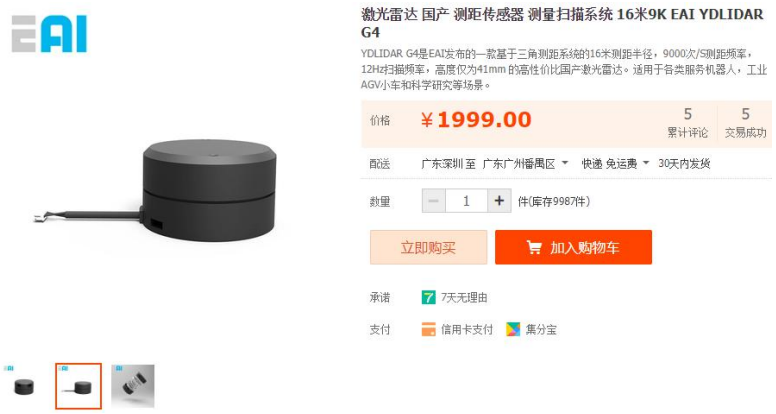


图 3.2 EAI G4 淘宝售卖网页

淘宝上的彬川、思岚科技等一些雷达检测距离均在 10M 或 10M 以下，检测到的信息有所减少。另外，当时的方案主要检测 9M 以及 9M 以下的区域。10M 检测距离的雷达可能在 9M 的地方就不稳定了。

图 3.3 彬川雷达

所以第一次粗略的选型，笔者偏向于选购 EAI 的 YDLIDAR G4 作为比赛使用的雷达。将选型的结果上报给黄老师审核，老师指出 YDLIDAR G4 的扫描频率太低，要求选购扫描时间更短的雷达，并向我们推荐了德国的西克公司生产的雷达。国产雷达基本没有高频率扫描的雷达，只能在国外的企业寻找。当时我们收到消息，称华南理工大学机器人战队使用的雷达是一家叫北阳的公司出产的。于是笔者第二次到北阳和西克的官

方网页上寻找合适的雷达，并做比较。

型号	TiM3xx系列	S100系列	LMS4xx系列	TIM100-3010200	LMS1xx系列	LMC1xx系列
应用领域	Indoor / Outdoor	Indoor	Indoor	Indoor	Indoor Outdoor Security	Indoor Security Outdoor
角度分辨率	1° 0.33°	0.5° 1°	0.1° ...1.0°，可选	1°	0.25° 0.5°	0.25° 0.5°
开启角度	270°	270°	70°	200°	270°	270°
扫描范围	0.05~4 0.05~10	0 m ... 10 m	0.7 m ... 3 m	0.05 m ... 3 m	0.5~20m 0.5~40m 0.5~50m	0.5~20m
扫描频率	15m	25 Hz	230 Hz ... 500 Hz	14.5 Hz	25 Hz / 50 Hz	50 Hz
反射率为 10 %时的扫描范围	2 m / 8 m	4.5 m	3 m	1.5 m	18 m / 30 m	18 m
ETHERNET	是		是		是	是
USB	是					
串行		是	是		是	是
IO-LINK				是		
CANOPEN		是				
CAN					是	是

图 3.4 西克雷达性能比较

Product name	Scanning Laser Range Finder			
Model	UST-10LX	UST-10LX-H01	UST-20LX	UST-20LX-H01
Supply voltage	DC 12V/DC 24V (operation range 10 to 30V ripple within 10%)			
Supply current	150mA or less (during start up 450mA is necessary.)			
Light source	Laser semiconductor (905nm), Laser class 1(IEC60825-1:2007)			
Detection range	0.06m to 10m (white Kent sheet) 0.06m to 4m (diffuse reflectance 10%) Max. detection distance : 30m		0.06m to 20m (white Kent sheet) 0.06m to 8m (diffuse reflectance 10%) Max. detection distance : 60m	
Accuracy	±40mm*1			
Repeated accuracy	σ<30mm*1			
Scan angle	270°			
Scan speed	25ms (Motor speed 2400rpm)			
Angular resolution	0.25°	0.125°	0.25°	0.125°
Start up time	Within 10 sec (start up time differs if malfunction is detected during start up)			
Input	IP reset input, photo-coupler input(current 4mA at ON)			
Output	Synchronous Output, photo coupler open collector output 30VDC 50mA MAX.			
Interface	Ethernet 100BASE-TX			

图 3.5 北阳常用雷达性能比较

西克的雷达扫描距离普遍比较远，扫描频率也很高。性能总体上来说比北阳要优越，但是价格也比较昂贵。并且比赛还要求购买备用雷达，我们的经费比较紧张。北阳的雷达价格便宜，性能也能够满足老师的要求，是不错的选择。

两家公司的雷达基本都满足扫描频率的要求，扫描半径因型号的不同而相异，选择范围比较宽。结合方案、场地和经费考虑后，我们决定购买北阳的 UST-10LX，售价约一万人民币。后来在购买备用雷达时，因为经销商北京友科莱公司暂时没有存货，需要

花费半个月的时间从日本进口，而备战 ROBOCON 没有那么多的时间，所以我们花费大约 18000 元购买了扫描距离更长，其他参数一样的 UST-20LX 作为备用雷达。

3.2.2 雷达主要参数解读

UST-10LX 主要参数如下图所示

Product name	Scanning Laser Range Finder
Model	UST-10LX
Supply voltage	12VDC/24VDC (Operation range 10 to 30V ripple within 10%)
Supply current	150mA or less (during start up 450mA is necessary.)
Light source	Laser semiconductor (905nm) Laser class 1 (IEC60825-1:2007)
Detection range	0.06m to 10m (white Kent sheet) 0.06m to 4m (diffuse reflectance 10%) Max. detection distance : 30m
Accuracy	±40mm (*1)
Repeated accuracy	σ< 30mm (*1)
Scan angle	270°
Scan speed	25ms (Motor speed 2400rpm)
Angular resolution	0.25°
Start up time	Within 10 sec (start up time differs if malfunction is detected during start up)
Input	IP reset input, photo-coupler input (current 4mA at ON)
Output	Synchronous Output, photo coupler open collector output 30VDC 50mA MAX.
Interface	Ethernet 100BASE-TX

图 3.6 UST-10LX 主要参数

3.2.2.1 扫描半径

UST-10LX 最大扫描半径(Detection range)是 30M，在反射率为 10%的情况下扫描距离降低为 4M。影响反射率的主要因素有两个：物体的颜色和表面的光滑程度。

雷达的红外光反射受物体颜色和表面形状所影响。越光滑、颜色越明亮，红外光反射率越高。白石膏能够 100%反射红外光，而黑色泡沫橡胶只能反射 2.4%。

图 3.7 白石灰和黑色泡沫橡胶

3.2.2.2 扫描范围

扫描范围(Scan angle)是指雷达能够扫描的角度，UST-10LX 和 UST-20LX 都只能扫描 270°的范围，西克公司的一些雷达和一些国产雷达能够扫描 360°的范围。范围不是

越大越好，要看是否符合应用需求。

3.2.2.3 扫描频率

扫描频率(Scan speed)是指一秒钟内的扫描次数，也可以说是扫描一次所用的时间。UST-10LX 扫描一次用时 25ms。前 24ms 扫描获取数据，最后一毫秒编码和输出数据。西克公司的一些雷达有更高的扫描频率，国内的雷达还达不到这样的速度。

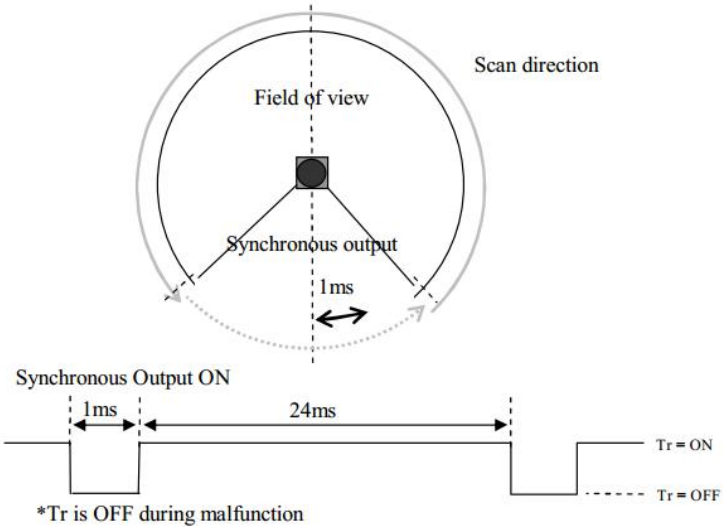


图 3.8 雷达扫描时序图

3.2.2.4 角度分辨率

角度分辨率(Angular resolution)为 0.25°是指两两光线之间的角度是 0.25°。雷达能否扫描到直径 6cm 的柱子，能够获得多少个点，就是根据这个参数来计算的。

3.2.2.5 通讯接口

为了完成短时间内数据大量传输的任务，UST-10LX 采用了以太网通讯，最大传输速度能够达到 100M/s，通信接口是以太网接口 RJ45。由于雷达引出的网线太短，笔者便使用 RJ45 延长接头和另一根网线将它延长。

图 3.9 RJ45 延长接头

雷达其他的通讯方式还有串口通信（多为国内雷达所使用）、CAN 通信、IO-LINK、

CANOPEN 和 USB 等。

3.2.2.6 光线波长

UST-10LX 的光源(Light source)波长为 905nm。可见光波长为 380~780nm。使用 905nm 的光源能够有效避免可见光光源的干扰，避免雷达传感器错误的检测。另外比赛规则要求使用的激光强度须在一级或以下，UST-10LX 的激光等级是一级，符合要求。

3.2.2.7 其他

UST-10LX 供电电压为 12~24V。UST-10LX 接电源的接插件如下图所示，是一种非常小的接插件，比成年人的指甲盖还要小几分。据友科莱的工程师介绍，这是一种国外使用的接插件，国内没有生产。我们在使用时，将提供电源的正负极线挑出来，接经过降压的 12V 电源。

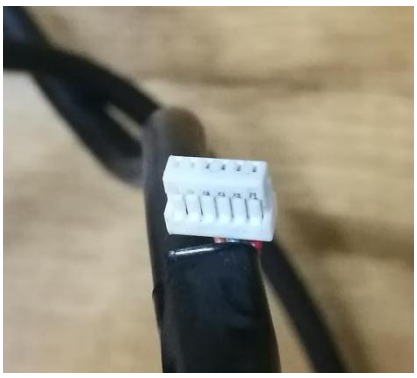


图 3.10 供电接插件

UST-10LX 启动时间小于 10s。当雷达正在启动时，雷达前方的蓝色指示灯闪烁。

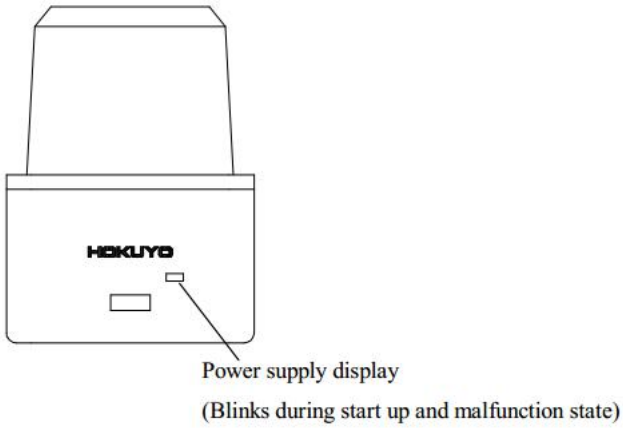


图 3.11 雷达指示灯位置

UST-10LX 输出的数据是步数和步数对应的距离。雷达总共有 1081 步，从第 0 步开始。控制雷达时，要向雷达发送检测开始的步数和结束的步数。

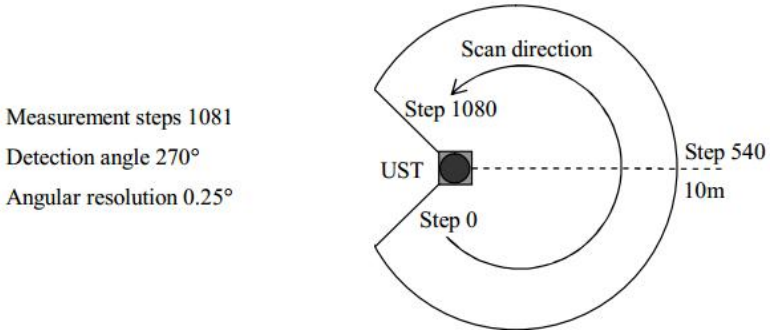


图 3.12 扫描方式

3.2.3 制定定位方案

定位方案在上文机器人控制部分做了大体的描述，下面进行更细致的描述。

3.2.3.1 角点定位

在制定方案的初期，黄老师建议检测如下图所示的边角来纠正全场定位的累计误差。

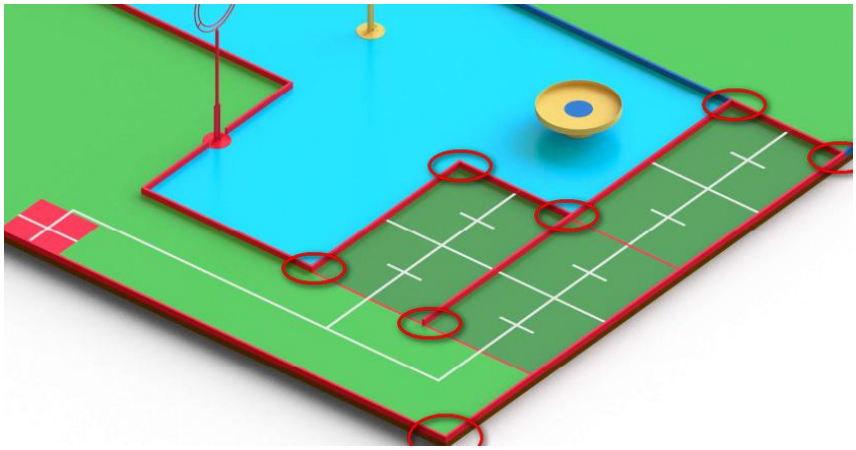


图 3.13 场地上的角点

在模拟将雷达装在车体底部，车体运动采集雷达扫描角点的数据时，因为位置比较低，没能创造让雷达在其水平面上平动的条件，遂没有完整采集。转而使用上文提到的检测柱子定位的方案。在了解了雷达的性能参数后，笔者猜测这样的检测方案同样是不太稳定的。因为雷达高度为 7CM，边界的高度为 10CM，而雷达上半部分用来发射红外线，下半部分接收，雷达发射的红外线不一定平行于地面，在水平方向上有一个正负 1°的波动，再加上车体底盘的晃动，所以雷达的扫描平面会有一个不规则的波动，有很大可能会扫描到底板或者从 10CM 边界上方射过去。那么采集到的数据处理起来就比较复杂。

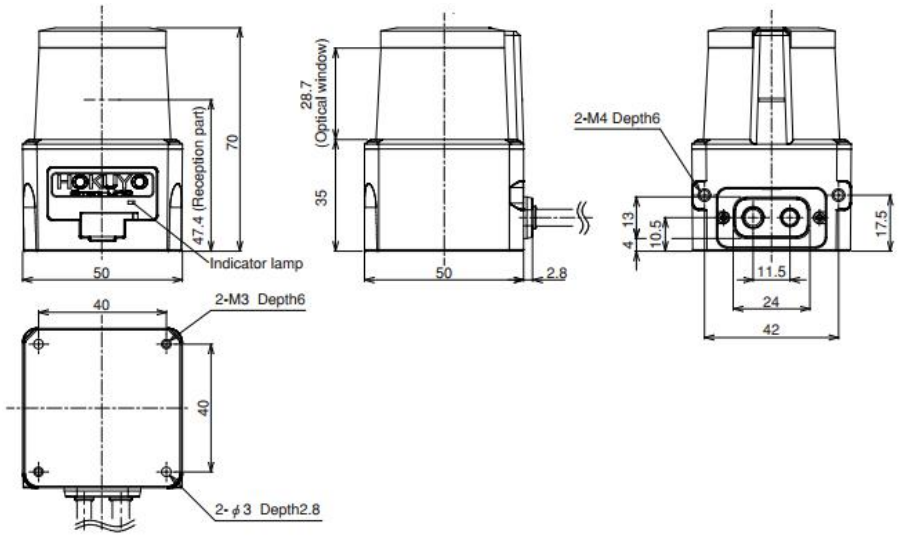


图 3.14 雷达尺寸

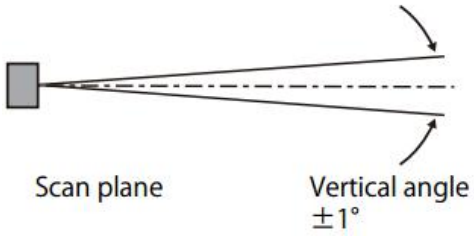


图 3.15 雷达光束偏离垂直方向示意图

A 组在这个方案上有过尝试，大体思路是采集角点两边各几个点，连起来成为两条直线，计算并判断这两条直线是否垂直。若垂直，那么两条直线的交点就是角点。否则，反之。但他们采用的霍夫圆变换算法在计算过程中出现了内存不足的情况，最后以探索者内存不足的失败告终。

3.2.3.2 柱子定位

笔者提议采用检测柱子实现定位的方案。柱子在场地中的位置是绝对的。在从场地上方往下看的二维平面中，两根柱子可以近似成两个固定的点。根据两个固定点，利用三角定位原理，实现机器人位置和姿态的确定。雷达检测柱子的效果如下图所示，圆圈圈起来的即为金环和彩环的立柱。虽然从图上看上去非常细小，但是最细的时候也被两条红外线检测到了。并且，柱子的信息在车体移动过程中都非常的清晰。这就为利用柱子定位提供了非常有利的条件。

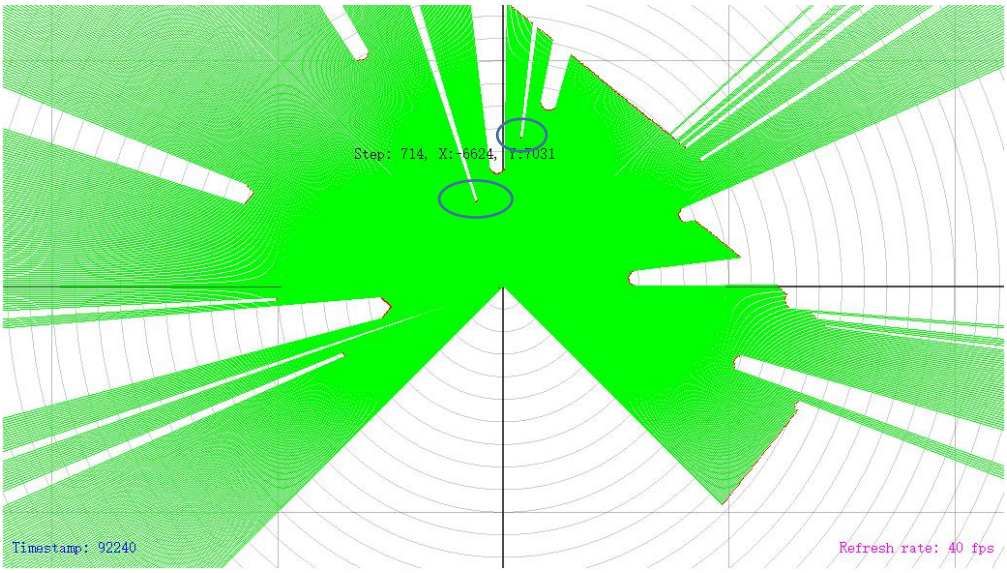


图 3.16 雷达检测柱子效果图

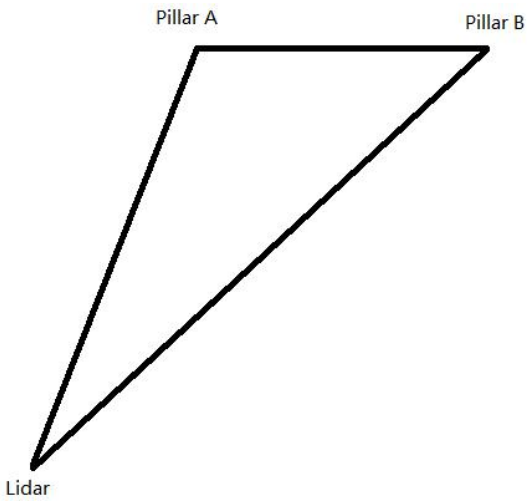


图 3.17 柱子定位示意图

两个柱子之间的距离已知，通过雷达测得雷达分别到两根柱子之间的距离和两个柱子到了雷达之间形成夹角，利用正弦定理得到 PillarB 的角的正弦值

$$\frac{a}{\sin A}=\frac{b}{\sin B}$$

进一步可以得到雷达的 Y 轴坐标，再利用直角三角形的两边平方和等于第三边的平方性质，即可得到雷达在世界坐标系中的位置。

计算出 PillarB 的 step 和 540step 之间的夹角，再考虑 PillarB 的角的大小，就能够确定机器人在二维平面上的姿态。

3.2.4 了解通讯协议

3.2.3.1 雷达地址

UST10-LX 采用 100M 高速以太网(Ethernet 100BASE-T)通信。HOKUYO 提供了两份通讯协议手册。一份比较简单，描述比较笼统(官方编号 C-42-04076)；另一份比较复杂，描述比较详细，并带有很多帮助理解的配图(官方编号 C-42-03886)。下面笔者做详细介绍。

TCP/IP 在通讯中被使用，雷达最初的 IP 地址、MAC 地址、子网掩码和端口号为

```
IP address       : 192.168.0.10
Subnet mask      : 255.255.255.0
Default gateway  : 192.168.0.1
Port number      : 10940(fixed)
```

图 3.18 雷达默认地址参数

IP 地址可以通过 HOKUYO 提供的程序修改。UST-10LX 现行使用的 IP 地址是：192.168.1.11

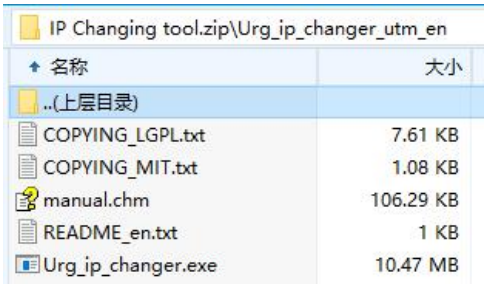


图 3.19 修改 IP 地址的工具

3.2.3.2 通讯格式

UST-10LX 通讯时使用 ASCII 标准。控制雷达的一般过程是控制器发送指令给雷达，雷达再返回指令要求的数据。指令都有统一的格式，如下图所示。

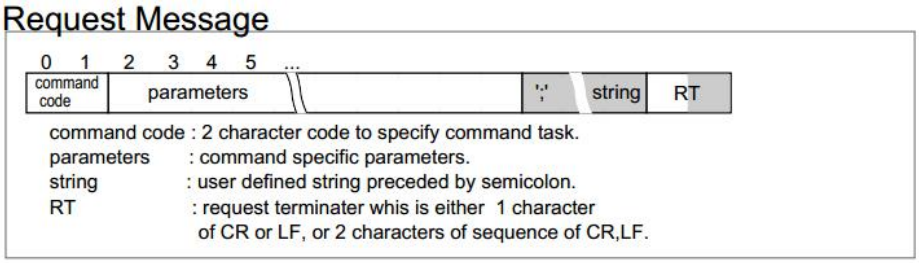


图 3.20 雷达请求指令格式

发送的第 0 和第 1 个字节是指令的代码，即两个字母；从第二个指令开始则是指令特殊的参数；结束后是由使用者定义的可以有也可以没有的字符串；最后是结束符，由

两个 0x0a 或两个 0x0d 组成，即两个回车符或两个换行符。我们在项目中没有定义字符串，也没有定义字符串的必要。

雷达响应发送回来的消息也有特定的格式，如下图所示

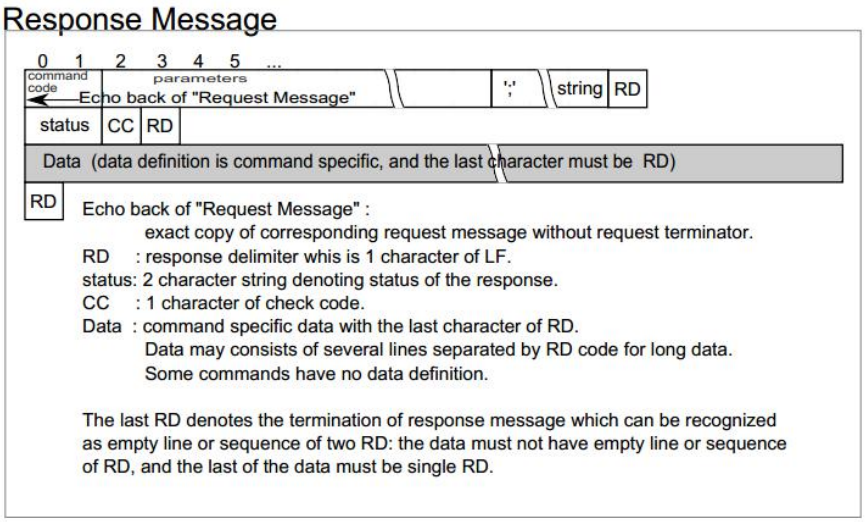


图 3.21 雷达响应信息格式

雷达的响应信息会把发过去的指令、参数和由使用者定义的字符串原原本本的返回来，即响应信息的第一、第二和第三种数据，后面跟着一个 RD，即一个回车符或换行符，这是第一行数据；第二行数据是 status，即雷达的状态信息，后跟一个 CC，即校验位，再跟一个 RD；从第三行开始是根据命令不同而返回响应的数据，最后也是由一个 RD 结束；最后再加一个 RD。以上就组成了一个雷达的响应信息。

3.2.3.3 雷达状态

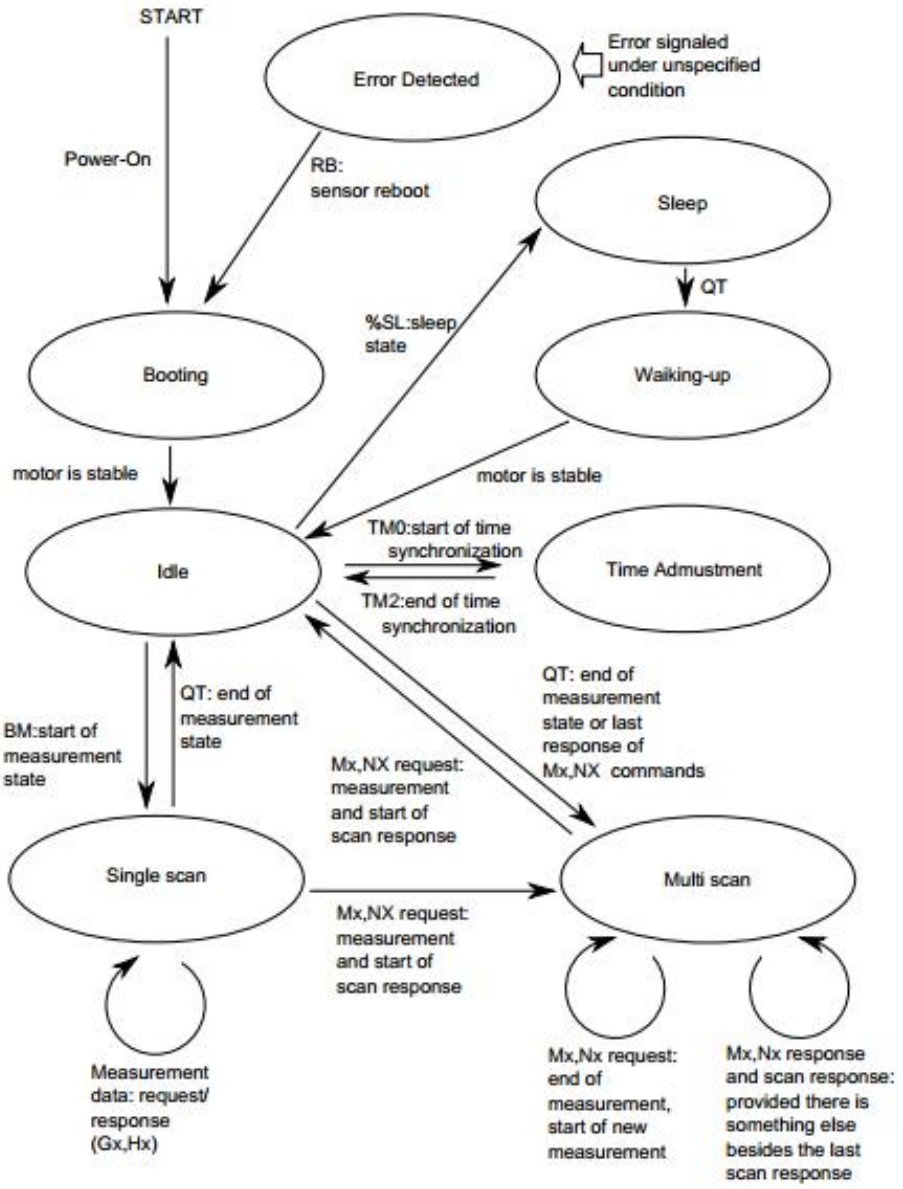


图 3.22 雷达状态转换图

雷达相应状态的解释如下：

Booting state and Waking-up state

The transitional state by the time when the scanner becomes ready for measurement. When the scanner becomes ready, the state switches to the next state automatically.

Standby state

The sensor is not performing measurement but ready for measurement. The laser is not lighted (activated).

Single scan state

The sensor is taking measurement data for all the available steps. If the BM command is received in the standby state, the sensor switches to measurement state. As the sensor is measuring all the measurement area with the laser activated in this state, the sensor

returns the latest measurement data when it receives the measurement data acquisition command.

Multi scan state

Measurement starts and the scan response cycle is in operation. The sensor enters into this state when it receives the measurement data acquisition with continuous scanning command in standby state or in single scan state. In this state, the sensor performs scanning for the specified scan area and the number of scans and returns the scan response message for each scan. If the operation parameters (the scan area and the number of scans) are changed by another continuous scanning command, the sensor resumes measurement with the changed parameters. The sensor switches to the standby state when the specified number of scans are finished or when the sensor receives the scan stop command.

Time synchronization state

A special state to synchronize with the internal timer of the sensor. The sensor switches to this state after receiving the time synchronization commands. While in this state, responses to time queries are sent with
the minimum delay.

Sleep state

The state to realize low power mode. The scanner is stopped, and the laser is not lighted.

Abnormal condition state

An abnormal sensor condition was found and the sensor switches from any arbitrary state to this one.

Unstable state

If normal measurements cannot be taken due to some interference, the sensor is retrying to complete the requested operation during some time (from a few seconds to tens of seconds), otherwise it goes to the abnormal condition state. The sensor remains temporally in this state and the sensor status is reported as such, and afterward moves into the abnormal condition state.

雷达一共有八个状态，从供上电开始，雷达无时无刻不处在其中一个状态。状态与状态之间的转换有明确的转换指令。只有在相应的状态下才能做相应状态的工作。比如雷达正处在 Idle 空闲状态，如果发送 GD 测量一次距离的命令，雷达不会测量。它会返回响应信息，status 是 10，即当前状态不接受 GD 命令。正确的做法是先发送 BM 命令，待雷达响应后，即转换到 Single scan，再发送 GD 命令，雷达便能正常工作。

雷达返回命令的含义如下图所示。

Error-abnormal-state (code 0L)
The sensor is in the abnormal condition state and therefore requests cannot be received. The code that shows the type of abnormal condition can be checked by the command to obtain the sensor state.

Error-unstable (code 0M)
The sensor is in the unstable state and therefore requests cannot be received. The code that shows the type of abnormal condition can be checked by the command to obtain the sensor state.

Error-command-not-defined (code 0E)
The command specified in the request message is not defined (unknown).

Error-command-not-supported (code 0F)
The command specified in the request message is not supported in the current sensor.

Error-denied (code 10)
The command specified in the request message cannot be received in the current sensor state.

Error-user-string-long (code 0G)
The length of the user defined string is too long.

Error-user-string-character (code 0H)
The user defined string has a problem.

Error-command-short (code 0C)
The length of the request message is shorter than expected, according to the existing definition.

Error-command-long (code 0D)
The length of the request message is longer than expected, according to the existing definition.

Error-parameter (code 01,02,03.04,05,06,07)
One of the parameters specified in the request message has a problem.

图 3.23 雷达 status 数字代表的含义

3.2.3.4 6 位编码和三字符编码

雷达通讯使用 ASCII 标准。为了压缩数据，减少通信信道的拥挤，雷达采用的编码方式是把 6 位二进制距离数值转换为 ASCII 码再加上 0x30 传送。

```
0x1a + 0x30 = 0x4a
              'J'
```

图 3.24 6 位编码示意图

如上图所示，如果有一个值 26，十六进制为 0x1a，加上 0x30 后为 0x4a。根据 ASCII 值，即为大写字母“J”。

雷达经过测量得到一个距离数值后，会将距离的二进制每六位分为一组，每一组根据上述方法进行编码。UST-10LX 的距离值最后会被编为 3 个字符，三个字符代表一个距离数值，这种编码方式成为三字符编码。相对应的，如果编码后形成 2 个字符，则称为二字符编码。

```
1234 = 0100 1101 0010 = 010011 010010
                        0x13   0x12

0x13 + 0x30 = 0x43      0x12 + 0x30 = 0x42
              'C'                     'B'
```

图 3.25 二字符编码示意图

如上图所示，如果要把距离数值 1234 编码发送。先将 1234 转换成二进制，再分成六位一组，即 0x13 和 0x12。0x13 加上 0x30 等于 0x43，ASCII 值代表字符“C”。同理得到字符“B”。所以得到 1234 的编码“CB”。编码时，高六位形成的 ASCII 码在前面，低

位的在后面。

3.2.3.5 校验码

为了检验发送与接收的过程中有没有出现错误，雷达提供了检验码(CC,Check Code)用于检测判断是否出错。将检验码覆盖范围的所有字符相加，得到的数值取二进制低六位再加上 0x30 成为覆盖范围的校验码。举例如下

```
'A'    'B'    'C'    '0'    '1'    '2'
0x41 + 0x42 + 0x43 + 0x30 + 0x31 + 0x32 = 0x159
0x19 + 0x30 = 0x49
              'I'
```

图 3.26 校验码生成示意图

上图中要求算出 ABC012 的校验码。先将各个字符华为对应的 ASCII 码数值，再相加得到 0x159，取其二进制低六位，得到 0x19，再加上 0x30 得数值 0x49，即 ASCII 码的大写字母“I”。在项目中，因为 UST-10LX 的稳定性很高，基本没有出现传输错误的问题，所以笔者没有利用校验位来检验数据的正确与否。

3.2.3.6 BM：激活雷达激光

控制雷达第一个必须了解的指令是 BM。该命令的作用是激活雷达激光，让雷达转换到单次测量(Single scan)状态。指令格式和响应格式如下

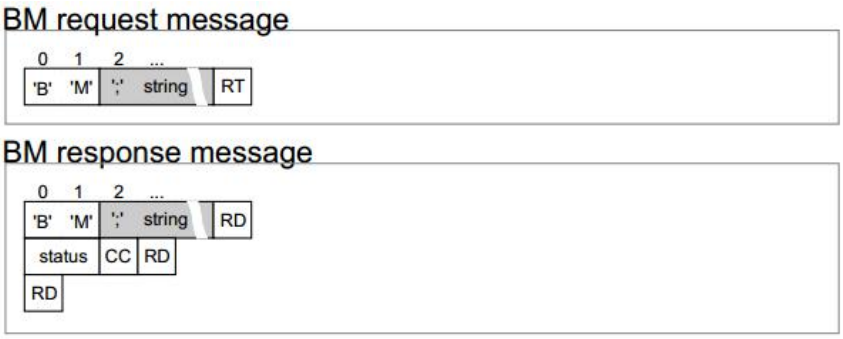


图 3.27 BM 指令与响应格式

请求指令先是字符“B”“M”，后是结束符，非常简单。响应格式第一行和请求指令相同，第二行返回的是雷达的状态，后面跟一个校验位和结束位。返回的状态有以下三种情况。

Status	Description
“00”	Normal. The sensor is in measurement state and the laser was lighted.
“01”	The laser was not lighted due to unstable or abnormal condition.
“02”	The sensor is already in measurement state and the laser is already lighted.

图 3.28 BM 指令中 Status 的含义

“00”表示一切正常，激活成功；“01”表示因为雷达处于不稳定状态导致激光没有激

活；“02”表示激光事先已经激活。做项目时可以根据反馈的雷达状态判断雷达的工作情况。

3.2.3.7 GD：单次测量

将雷达转换到指定状态后，要控制雷达扫描指定范围。项目中笔者使用的是 GD 单次测量命令，顾名思义就是控制雷达进行单次测量。其他某个指令能够实现连续测量的功能。根据比赛需求，我们进行一次测量后都要改变雷达测量的方位。因为单次测量命令能够实现单次测量方位控制，所以该指令是适合完成任务需求的。GD 指令的响应比 BM 要略微复杂。指令格式如下

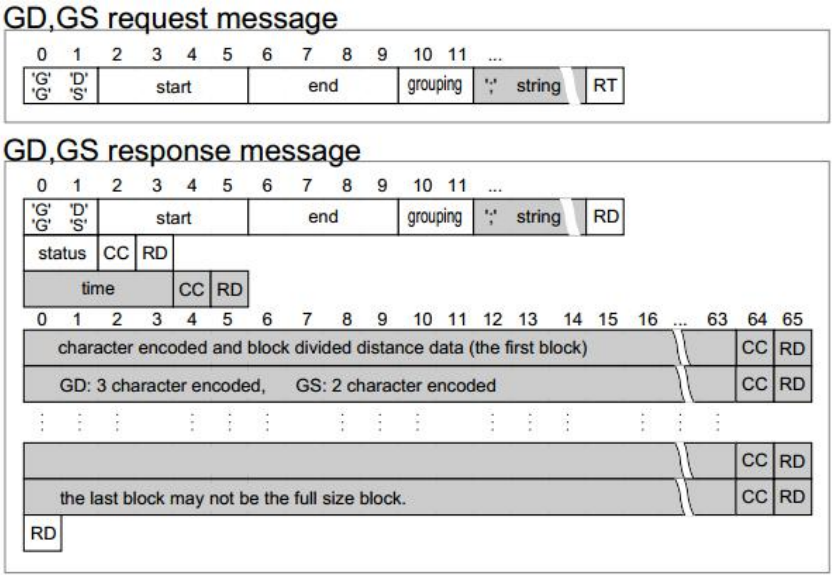


图 3.29 GD 指令与响应格式

请求指令最前面两个字节是“G”“D”，接下来的四个字符是开始的 step，后面四个是结束的 step。第 10、11 个字符是分组，即将多少条光线合并为一条，此项默认为 0。只有该项为 0 时，才能收集到全部 1081steps 的数据。再后面的则是使用者自定义的字符串和结束位。

响应第一行和请求指令是一样的。第二行和 BM 命令一样，返回的是雷达的状态，只不过状态信息的含义有变，如下图所示，50~98 的 status 含义参照上文提过的状态码含义。

● Status:

01 --- Starting Step has non-numeric value.
02 --- End Step has non-numeric value
03 --- Cluster Count has non-numeric value.
04 --- End Step is out of range
05 --- End Step is smaller than Starting Step.
10 --- Laser is off.
50~98 --- Hardware trouble (such as laser, motor malfunction etc.)

图 3.30 雷达指令的 Status 代表的含义

第三行是时间戳，表示雷达内部的时间，到达一定值会自动清零。第三行最后一个位仍旧是结束位。

从第四行开始就是距离数据了，采用的是上文提到的三字符编码。比较特别的是，每 64 个字符之间，都会插入一个校验位校验前 64 个字符是否出错，还有一个结束符。直到最后一组数据不足 64 个字符，再插入一个校验位和一个结束符。最后再加一个结束符。

3.2.5 在探索者 STM32F407ZGT6 上实现通讯

项目中，我们使用探索者 STM32F407ZGT6 与雷达进行通讯实现控制。探索者 STM32F407ZGT6 是由广州星翼有限公司开发的一块供初学者使用的开发板，并提供一些技术支持。笔者使用其中的 RAW 编程 LWIP 客户端实验与雷达进行连接。

LWIP 是由瑞典计算机科学院 Adam Dunkels 等开发的用于嵌入式系统的开源 TCP/IP 协议栈（与 uIP 系同一作者），LwIP 的含义是 Light Weight IP 协议，是在嵌入式网络领域比较有名的软件。该协议在许多地方，诸如卫星、石油装置、飞机、赛车、远程航船、电视接收装置中得到了广泛应用，产品涉及 ABB、Altera、BMW、Cisco、Ericsson、GE、HP、Xilinx 等有名公司。该协议栈可以移植到操作系统上，也可以在无操作系统下独立运行。LwIP 实现的重点是在保持 TCP 协议主要功能的基础上减少对 RAM 的占用，它只需要十几 KB 的 RAM 和 40K 左右的 ROM 就能够运行，这使 LwIP 协议栈适合在低端的嵌入式系统中使用。笔者项目中使用的版本是 1.4.1。LwIP 的详细信息和下载网址是：

http://download.savannah.gnu.org/releases/lwip/，LwIP 主要特性如下：

1. ARP 协议，以太网地址解析协议
2. IP 协议，包括 IPv4 和 IPv6，支持 IP 分片与重装，支持多网络接口下数据转发
3. ICMP 协议，用于网络调试与维护

4. UDP 协议，用户数据报协议
5. TCP 协议，支持 TCP 拥塞控制，RTT 估计，快速恢复与重传等
6. 提供三种用户编程接口方式：raw/callback API、sequential API、BSD-style socket API
7. DNS,域名解析
8. SNMP，简单网络管理协议
9. DHCP，动态主机配置协议
10. AUTOIP，IP 地址自动配置
11. PPP，点对点协议，支持 PPPoE

以太网 PHY 层芯片使用的是 LAN8720。LAN8720 是低功耗的 10/100M 以太网 PHY 层芯片，I/O 引脚电压符合 IEEE802.3-2005 标准。LAN8720 支持通过 RMII 接口与以太网 MAC 层通信，内置 10-BASE-T/100BASE-TX 全双工传输模块，支持 10Mbps 和 100Mbps。LAN8720 可以通过自协商的方式与目的主机最佳的连接方式（速度和双工模式）。支持 HP Auto-MDIX 自动翻转功能，无需更换网线即可将连接更改为直连或交叉连接。LAN8720 的主要特点如下：

1. 高性能的 10/100M 以太网传输模块
2. 支持 RMII 接口以减少引脚数
3. 支持全双工和半双工模式
4. 两个状态 LED 输出
5. 可以使用 25M 晶振以降低成本
6. 支持自协商模式
7. 支持 HP Auto-MDIX 自动翻转功能
8. 支持 SMI 串行管理接口
9. 支持 MAC 接口

LAN8720 功能框图如下图所示

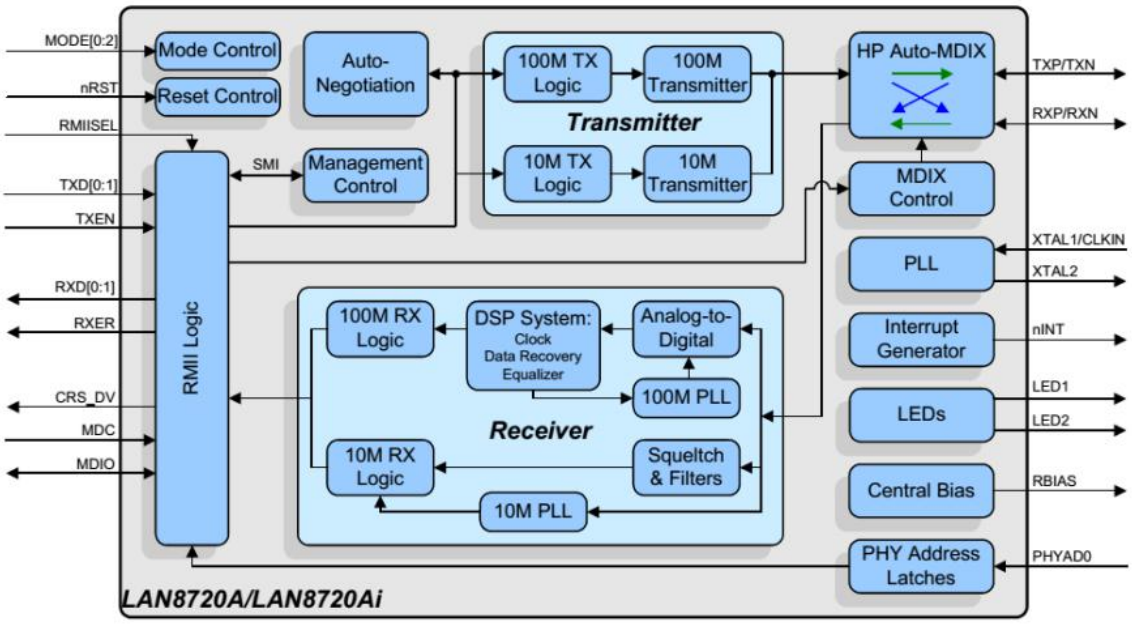


图 3.31 LAN8720 功能框图

为了实现数据的高速传送，STM32 采用了 DMA 直接访问内存的方式将网口输出的数据直接放置到内存中。直接存储器访问 (DMA) 用于在外设与存储器之间以及存储器与存储器之间提供高速数据传输。可以在无需任何 CPU 操作的情况下通过 DMA 快速移动数据。这样节省的 CPU 资源可供其它操作使用。DMA 控制器基于复杂的总线矩阵架构，将功能强大的双 AHB 主总线架构与独立的 FIFO 结合在一起，优化了系统带宽。两个 DMA 控制器总共有 16 个数据流（每个控制器 8 个），每一个 DMA 控制器都用于管理

一个或多个外设的存储器访问请求。每个数据流总共可以有多达 8 个通道（或称请求）。每个通道都有一个仲裁器，用于处理 DMA 请求间的优先级。 DMA 主要特性是：

1. 双 AHB 主总线架构，一个用于存储器访问，另一个用于外设访问

2. 仅支持 32 位访问的 AHB 从编程接口

3. 每个 DMA 控制器有 8 个数据流，每个数据流有多达 8 个通道（或称请求）

4. 每个数据流有单独的四级 32 位先进先出存储器缓冲区 (FIFO)，可用于 FIFO 模式或直接模式：

— FIFO 模式：可通过软件将阈值级别选取为 FIFO 大小的 1/4、 1/2 或 3/4

— 直接模式：每个 DMA 请求会立即启动对存储器的传输。当在直接模式（禁止 FIFO）下将 DMA 请求配置为以存储器到外设模式传输数据时， DMA 仅会将一

个数据从存储器预加载到内部 FIFO，从而确保一旦外设触发 DMA 请求时则立即传输数据。

5. 通过硬件可以将每个数据流配置为：

— 支持外设到存储器、存储器到外设和存储器到存储器传输的常规通道

— 也支持在存储器方双缓冲的双缓冲区通道

6. 8 个数据流中的每一个都连接到专用硬件 DMA 通道（请求）

7. DMA 数据流请求之间的优先级可用软件编程（4 个级别：非常高、高、中、低），在软件优先级相同的情况下可以通过硬件决定优先级（例如，请求 0 的优先级高于请求 1）

8. 每个数据流也支持通过软件触发存储器到存储器的传输（仅限 DMA2 控制器）

9. 可供每个数据流选择的通道请求多达 8 个。此选择可由软件配置，允许几个外设启动 DMA 请求

10. 要传输的数据项的数目可以由 DMA 控制器或外设管理：

— DMA 流控制器：要传输的数据项的数目是 1 到 65535，可用软件编程

— 外设流控制器：要传输的数据项的数目未知并由源或目标外设控制，这些外设通过硬件发出传输结束的信号

11. 独立的源和目标传输宽度（字节、半字、字）：源和目标的数据宽度不相等时，DMA 自动封装/解封必要的传输数据来优化带宽。这个特性仅在 FIFO 模式下可用

12. 对源和目标的增量或非增量寻址

13. 支持 4 个、 8 个和 16 个节拍的增量突发传输。突发增量的大小可由软件配置，通常等于外设 FIFO 大小的一半

14. 每个数据流都支持循环缓冲区管理

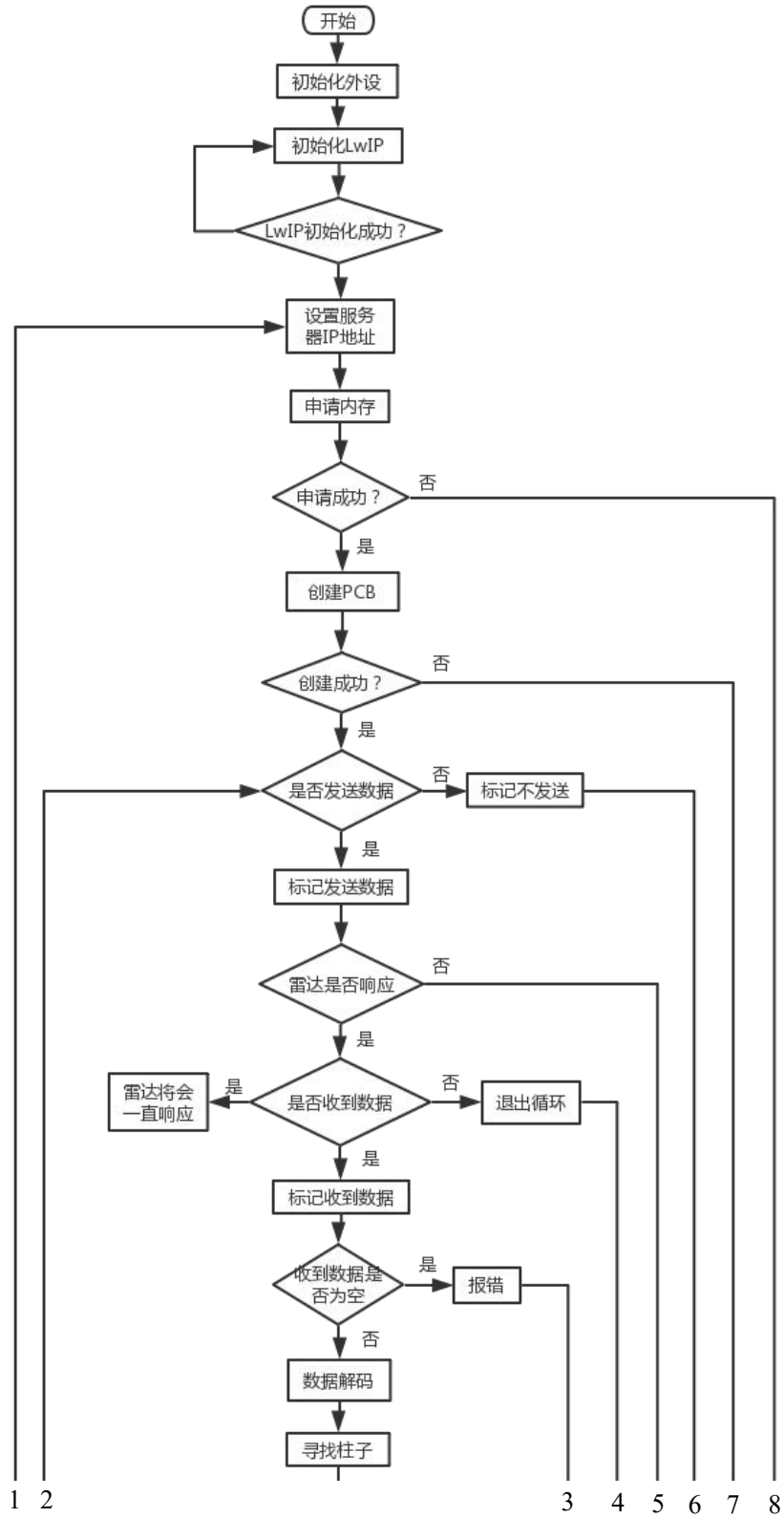
15. 5 个事件标志（DMA 半传输、 DMA 传输完成、 DMA 传输错误、 DMA FIFO 错误、直接模式错误），进行逻辑或运算，从而产生每个数据流的单个中断请求

3.2.6 SCIP 解码

SCIP 即 Sensor Communication Protocol，是 HOKUYO 使用的传感器通讯协议，UST-10LX 便是采用该协议。前文所述雷达特性就是 SCIP 的详细叙述。

在细分讲解各部分代码之前，笔者先介绍雷达检测整个程序的流程图，如下图所示。

程序整体为一个大循环，大循环里有许多为满足要求的小循环和分支结构。循环检测满足了雷达需要不断定位的要求。雷达程序里有许多判断检测到的数据是否是柱子的判断语句。采集到的数据必须经过严格的检查才能输出给底盘作为定位信息。详情见后文分析。



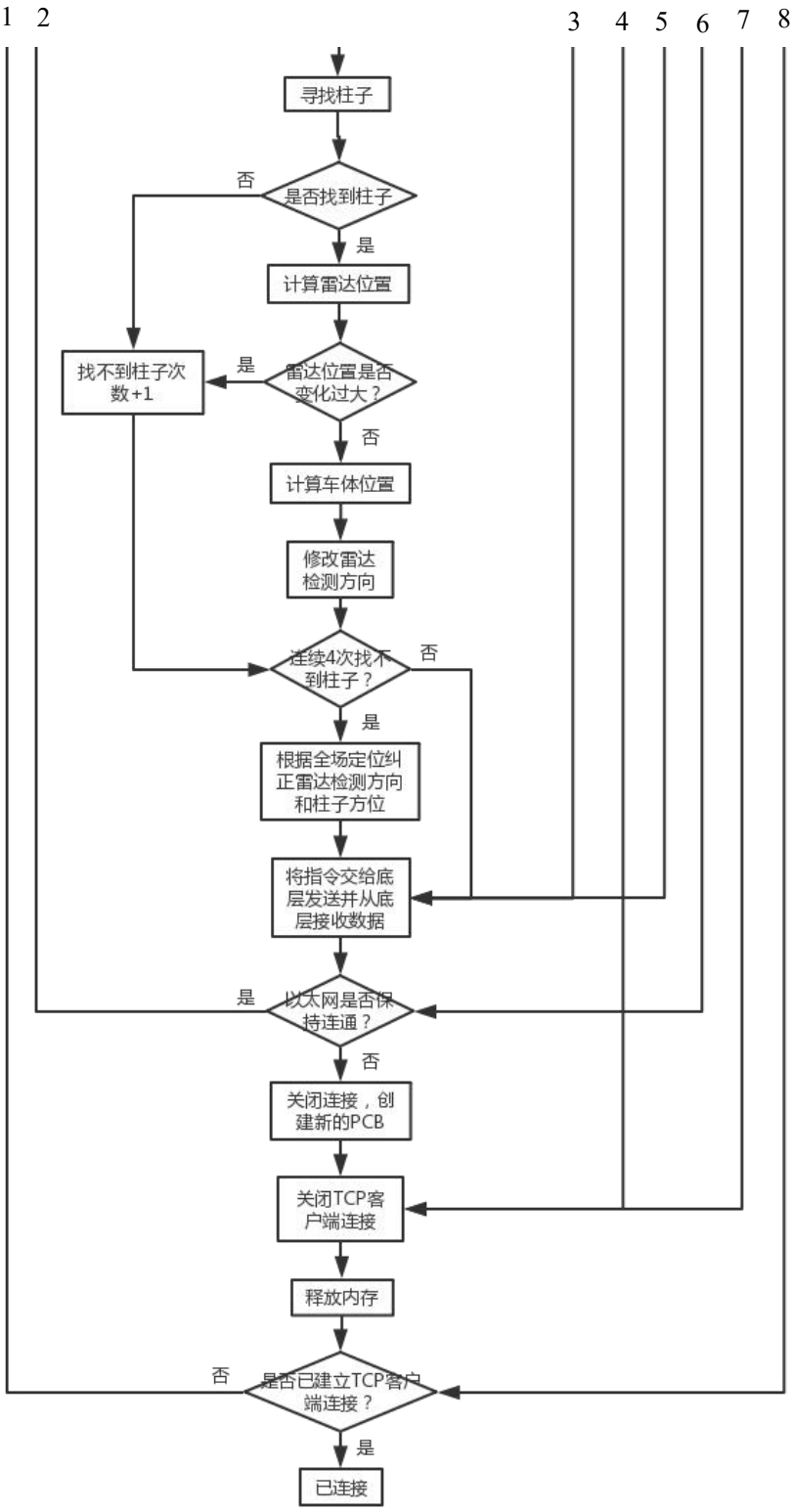


图 3.32 雷达程序流程图

我们使用正点原子的探索者 STM32F4ZGT6 实现了与雷达的通信，控制机器人定位的代码也是在这块开发板上实现的。从雷达接收到编码的数据后，需要将其解码还原为距离信息，以供数据处理需要。下面的功能实现由笔者独立完成。

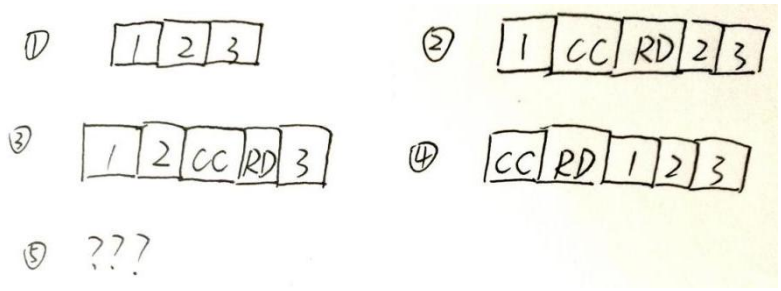


图 3.33 解码示意图

如上图片示意那样，笔者采用每三个三个字符组成一个距离信息的方式，向后找字符。因为数据里掺杂有校验位和结束位，情况复杂，所以分成了以上四种情况来解码。另外可能有笔者考虑不到的情况，增加了第五种特殊情况，即四种情况之外的情况程序都会报错。但经过一个项目的验证，解码函数从来没有报错，说明前面四种情况已经是解码数据时的所有情况，并且运行稳定。需要解码的距离多时候都有上百个，为了加快解码速度，该函数大量使用了位移运算代替乘除法运算，大大加快了解码速度。伪解码实现代码如下。datachar 是处理采集到数组字符的第某个字符，temp 用来存放暂时用到的量。

```
datachar=0;
temp=0;
datachar=23;
while(recieve[datachar]!='\0')
      if(recieve[datachar+1]!='\n'&&
      recieve[datachar+2]!='\n'&&
      recieve[datachar+3]!='\n')
      temp=(recieve[datachar]-0x30)<<12;
      temp+=(recieve[datachar+1]-0x30)<<6;
      temp+=(recieve[datachar+2]-0x30);
      Distance[i]=temp;
      i++;
      datachar+=3;
      continue;

      else if
      (recieve[datachar+1]!='\n'&&
      recieve[datachar+2]=='\n'&&
      recieve[datachar+1]!='\n')
      temp=(recieve[datachar]-0x30)<<12;
      temp+=(recieve[datachar+3]-0x30)<<6;
      temp+=(recieve[datachar+4]-0x30);
      Distance[i]=temp;
```

```
    i++;
    datachar+=5;
    continue;

    else if
    (recieve[datachar+1]!='\n'&&
    recieve[datachar+2]!='\n'&&
    recieve[datachar+3]=='\n')
    temp=(recieve[datachar]-0x30)<<12;
    temp+=(recieve[datachar+1]-0x30)<<6;
    temp+=(recieve[datachar+4]-0x30);
    Distance[i]=temp;
    i++;
    datachar+=5;
    continue;

    Else if
    (recieve[datachar+1]=='\n'&&
    recieve[datachar+2]!='\n'&&
    recieve[datachar+3]!='\n')
    datachar+=2;
    continue;

    Else if
    (recieve[datachar+1]=='\n'&&
    recieve[datachar+2]=='\n')
    Break;

    Else
    printf("Dispose Data Error");
endwhile
```

3.2.7 寻找柱子

将数据解码后，得到各个 steps 的距离信息。笔者编写了一个函数来寻找柱子。柱子在上文雷达检测柱子的上位机视图中，两根柱子遮挡雷达的红外线非常明显的显示了出来。笔者利用这一特点，将找到落差作为找到柱子的特点。当发现一个落差大于设定值时，则认为找到了柱子的边缘点。找到柱子的第一个落差后，继续计算后续若干距离点与第一个的距离差。距离差在一定范围内则认为这些点都是照射到柱子的点。点的多少就表示着柱子的粗细。寻找柱子的流程图如下。

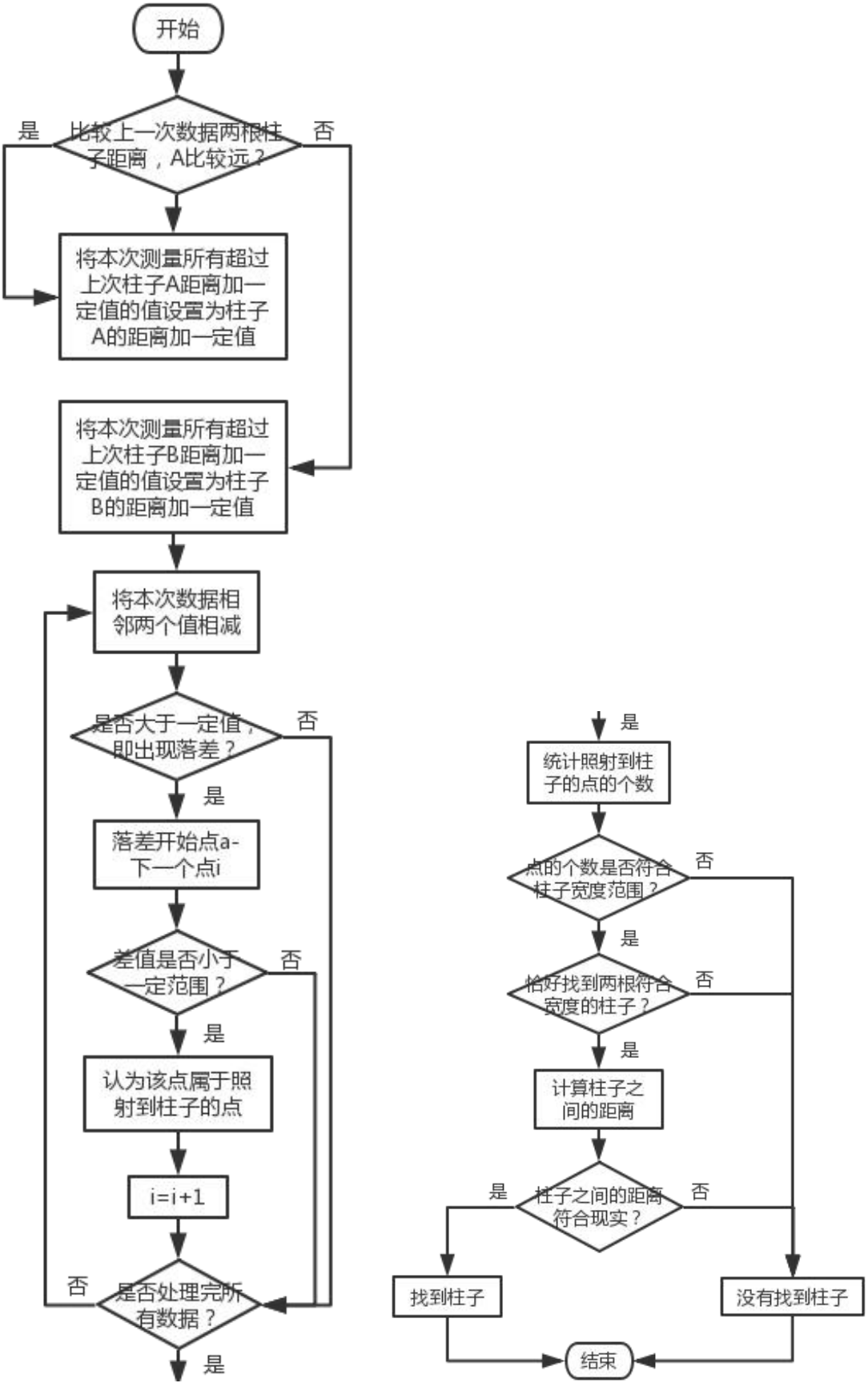


图 3.34 寻找柱子函数流程图

为了最大程度的减少干扰，笔者编写了两个寻找柱子的函数。一个作用在机器人刚启动和启动不久的一段距离。功能实现伪代码如下。变量 j 用来推进第一个落差点和后续照射到柱子第点的比较，objwid 即 object width，用来记录柱子占的点的个数，pillarwidA 和 pillarwidB 用来记录柱子的宽度。fallstep 是记录落差 step 的一个数组，numfall 用来记录落差的个数，BOUNDARY 代表最大的距离值，要在该值范围内寻找柱子。THRESHOLD_A 和 THRESHOLD_B 是某个用来判定的值，distancelen 代表数组长度。Distance[]就是要处理的数组。

```
j=2;
objwid=1;
pillarwidA=0;
pillarwidB=0;

numfall=0;
memset(fallstep,0,20);
i=distancelen-1;
while (i>=0)
      if(Distance[i]>BOUNDARY)
            Distance[i]=BOUNDARY;
       Endif
       i--;
endwhile

for(i=distancelen-1;(i-1)>=0;i--)
      temp=(Distance[i]-Distance[i-1]);
      if(temp>THRESHOLD_A)
            fallstep[numfall]=i;
            numfall++;
      endif
endfor
if(numfall==3)
      i=fallstep[0];
      temp=abs((int)(Distance[i-1]-Distance[i-j]));
      while (temp<THRESHOLD_B)
            objwid++;
            j++;
            temp=abs((int)(Distance[i-1]-Distance[i-j]));
      Endwhile
      stepA=i;
      pillarwidA=objwid;
      j=2;
```

```
    objwid=1;

    i=fallstep[2];
    temp=abs((int)(Distance[i-1]-Distance[i-j]));
    while (temp<THRESHOLD_B)
        objwid++;
        j++;
        temp=abs((int)(Distance[i-1]-Distance[i-j]));
    Endwhile
    stepB=i;
    pillarwidB=objwid;

    if(pillarwidA>2&&pillarwidA<10&&pillarwidB>2&&pillarwidB<10)
        oldstepA=stepA;
        oldstepB=stepB;

        if(pillarwidA&1)
            PillarDstA=Distance[stepA-(pillarwidA>>1)-1]+30;
        else
            PillarDstA=((Distance[stepA-(pillarwidA>>1)]+Distance[stepA-(pillarwidA>>1)-1])>>1)+3
0;

        if(pillarwidB&1)
            PillarDstB=Distance[stepB-(pillarwidB>>1)-1]+30;
            realstepB=stepB-(pillarwidB>>1)-1;
        endif
        else
            PillarDstB=((Distance[stepB-(pillarwidB>>1)]+Distance[stepB-(pillarwidB>>1)-1])>>1)+3
0;
            realstepB=stepB-(pillarwidB>>1);
        endelse

        if((pillarwidA&1)&&(pillarwidB&1))
            PillarAgl=((stepA-(pillarwidA>>1)-1)-(stepB-(pillarwidB>>1)-1))*0.25;
        else if(((pillarwidA&1)==0)&&(pillarwidB&1))
            PillarAgl=((stepA-(pillarwidA>>1))-(stepB-(pillarwidB>>1)-1))*0.25;
        else if((pillarwidA&1)&&((pillarwidB&1)==0))
            PillarAgl=((stepA-(pillarwidA>>1)-1)-(stepB-(pillarwidB>>1)))*0.25;
        else if(((pillarwidA&1)==0)&&((pillarwidB&1)==0))
            PillarAgl=((stepA-(pillarwidA>>1))-(stepB-(pillarwidB>>1)))*0.25;
        anglerad=PillarAgl/180*PI;
        pillarsdst2=PillarDstA*PillarDstA+PillarDstB*PillarDstB-2*PillarDstA*PillarDstB*cos(anglera
d);
```

```
            pillarsdst=(int)sqrt(pillarsdst2);

            if(pillarsdst>MIN_PLADST&&pillarsdst<MAX_PLADST)
                find=1;
                oldPillarDstA=PillarDstA;
                oldPillarDstB=PillarDstB;
                if(LidarY>START_TRACK_MIN&&LidarY<START_TRACK_MAX)
                    MODE=1;
                    lstepA=oldstepA+startstep;
                    lstepB=oldstepB+startstep;
                    numfall=0;
                    memset(fallstep,0,20);
                Endif
                Else
                    find=0;
                    numfall=0;
                    memset(fallstep,0,20);
                endelse
        endif
        Else
            find=0;
            numfall=0;
            memset(fallstep,0,20);
        endelse
endif
Else
        find=0;
        numfall=0;
        memset(fallstep,0,20);
endelse
```

该函数的特点是能够在最初的一段时间，即使有人的遮挡，也能够继续检测，不跑崩或者输出错误数据。人的遮挡主要来自于上场准备时，雷达开机后人的不规律运动。这有遮挡到雷达的可能。另外，该扫描模式能防止我方手动车从启动到拿球的干扰。在第一个寻找柱子的模式中，判断找到的是柱子的条件非常严格。该函数没有使用全场定位的辅助纠正，但输出的数据必须是正确的。而第二个函数则少一个判断条件，即使找不到，仍能通过全场定位提供的位置重新确定扫描方位重新找到柱子。该函数的效果非常好，计算非常稳定。

第二个寻找柱子的函数是在离出发点较远的时候使用的。该函数能够在找丢柱子或雷达被恶意遮挡的时候，利用全场定位提供的数据重新找柱子，并继续跟踪柱子。与全

场定位相辅相成，充分发挥了雷达定位的绝对性和全场定位的稳定性。在前期调试时，该函数并不能达到非常稳定的程度，所以写了上文提到的函数。撰写该报告时，第二个寻找柱子的函数已经非常完善，只要没有恶意干扰，就能够非常稳定的找到柱子。除了雷达偶尔检测失误，返回的数据有干扰外，基本每一帧都能找到场地的柱子。代码实现如下。J、objwid、pillarwidA 和 pillarwidB 与上文的 j、objwid、pillarwidA 和 pillarwidB 作用一致，dontfind 是表示没有找到落差或只找到一个落差的标志量。

```
j=2;
objwid=1;
pillarwidA=0;
pillarwidB=0;
dontfind=0;

if(oldPillarDstA>oldPillarDstB)
    temp=oldPillarDstA+TRACK_DST;
else
    temp=oldPillarDstB+TRACK_DST;

for(i=distancelen-1;i>=0;i--)
    if(Distance[i]>temp)
    Distance[i]=temp;
    endif
endfor

for(i=distancelen-1;(i-1)>=0;i--)
    temp=(Distance[i]-Distance[i-1]);
    if(temp>THRESHOLD_A)
        temp1=Distance[i-1]-Distance[i-j];
        temp1=abs((int)temp1);
        while (temp1<THRESHOLD_B)
            objwid++;
            j++;
            temp1=Distance[i-1]-Distance[i-j];
            temp1=abs((int)temp1);
        Endwhile
        if(objwid>2&&objwid<12)
            if(difline)
                stepA=i;
                pillarwidA=objwid;
                difline=0;
            endif
```

```
                else
                    stepB=i;
                    pillarwidB=objwid;
                    break;
                endelse
            endif
            j=2;
            objwid=1;
        endif
        if(i==1)
            dontfind=1;
        endif
endfor

if(dontfind)
        find=0;
        dontfind=0;
        return;
endif

if(pillarwidA>2&&pillarwidA<15&&pillarwidB>2&&pillarwidB<15)
            oldstepA=stepA;
            oldstepB=stepB;
            lstepA=oldstepA+minstep;
            lstepB=oldstepB+minstep;

        if(pillarwidA&1)
            PillarDstA=Distance[stepA-(pillarwidA>>1)-1]+30;
        Else
            PillarDstA=((Distance[stepA-(pillarwidA>>1)]+Distance[stepA-(pillarwidA>>1)-1])>>1)+30;

        if(pillarwidB&1)
            PillarDstB=Distance[stepB-(pillarwidB>>1)-1]+30;
            realstepB=stepB-(pillarwidB>>1)-1;
        else
            PillarDstB=((Distance[stepB-(pillarwidB>>1)]+Distance[stepB-(pillarwidB>>1)-1])>>1)+30;
            realstepB=stepB-(pillarwidB>>1);
        endelse

        if((pillarwidA&1)&&(pillarwidB&1))
            PillarAgl=((stepA-(pillarwidA>>1)-1)-(stepB-(pillarwidB>>1)-1))*0.25;
        else if(((pillarwidA&1)==0)&&(pillarwidB&1))
            PillarAgl=((stepA-(pillarwidA>>1))-(stepB-(pillarwidB>>1)-1))*0.25;
```

```
    else if((pillarwidA&1)&&((pillarwidB&1)==0))
        PillarAgl=((stepA-(pillarwidA>>1)-1)-(stepB-(pillarwidB>>1)))*0.25;

    else if(((pillarwidA&1)==0)&&((pillarwidB&1)==0))
        PillarAgl=((stepA-(pillarwidA>>1))-(stepB-(pillarwidB>>1)))*0.25;

    anglerad=PillarAgl/180*PI;
    pillarsdst2=PillarDstA*PillarDstA+PillarDstB*PillarDstB-2*PillarDstA*PillarDstB*cos(anglerad);
    pillarsdst=(int)sqrt(pillarsdst2);

    if(pillarsdst>MIN_PLADST&&pillarsdst<MAX_PLADST)
        find=1;
        oldPillarDstA=PillarDstA;
        oldPillarDstB=PillarDstB;
    endif
    else
        find=0;
        return;
    endelse
endif
else
    find=0;
    return;
endelse
```

找柱子是整个程序里最重要的部分，也是最容易出问题的部分。笔者调试的大部分时间都花在了编写调试找柱子的函数上面。函数的参数要选取得非常适当，才能在教六建筑柱产生干扰的情况下准确的找到柱子。

3.2.8 计算机器人位姿和滤波

计算机器人的位姿和滤波涉及三个函数。第一个先计算雷达的位置和姿态，再转换成底盘控制所需要的机器人的坐标系和位置，第三个函数是视觉组同学写的滤波函数，使用卡尔曼算法进行滤波。

计算雷达的位置和姿态的方法在上文讲解利用柱子进行三角定位时解释过，在这里不再解释。

得到雷达的位置后，需要将雷达的位置转换成机器人的位置。机器人的坐标原点在启动点，雷达的坐标原点在金色柱子。X 轴和 Y 轴加减一个值便能得到机器人的坐标原点。

除了坐标系不同之外，需要格外注意雷达的安装位置并不是车体中心，也就是说雷达的坐标并不是车体中心坐标，如下图所示。此处还应有转换。但现阶段的我们对移动旋转车体的坐标变换理解还不够透彻，也只是笼统的进行了一些加减，使雷达的坐标转换为车体中心的坐标。

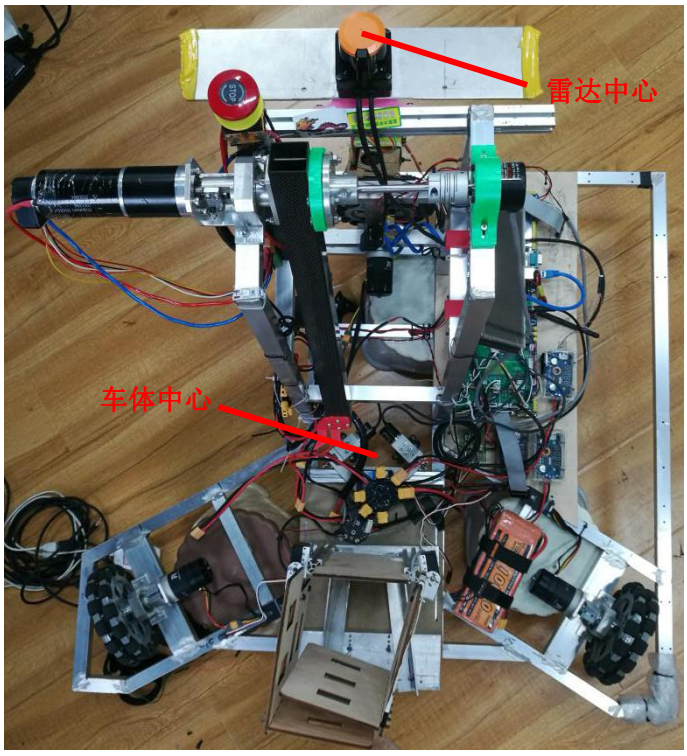


图 3.35 雷达中心与车体中心不重合

雷达检测输出的坐标波动是比较大的，特别是在需要停下的点，车体抖动非常明显。这主要是雷达反馈的数据波动比较大的原因。所以我们联系视觉组的同学帮忙滤波。滤波是由视觉组曹睿同学独自完成的，笔者并不清楚其中道理，不便做过多阐述。从使用效果上看，调试过程中，出现了波动变小，但是输出的坐标延迟的情况。这种情况相当于定位失败，没有跑出理想的轨迹。修改了一些参数后，做到了实时快速输出，但也牺牲了一些把输出的数据“撸平”的性能。

计算雷达坐标和计算机器人坐标的函数比较简单，滤波函数并非笔者编写，这里便不贴出代码。详情可在文末附录中查看。

3.2.9 异常处理

异常处理程序的流程图如下图所示。

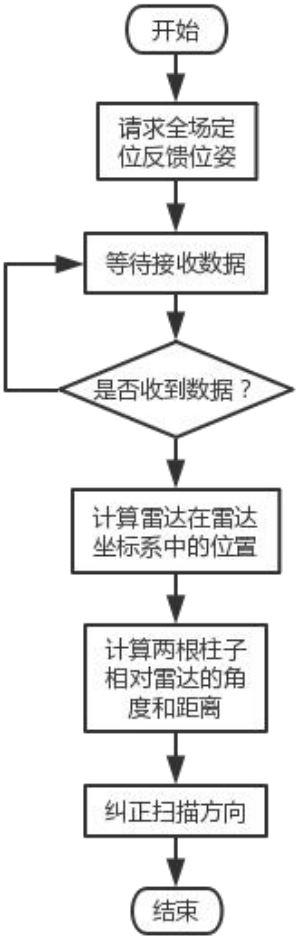


图 3.36 异常处理流程图

异常处理是仅次于寻找柱子的第二重要的任务。当雷达找不到柱子时，异常就发生了。异常处理函数通过全场定位反馈的坐标重新计算柱子的位置，修改雷达测量方向和测量范围的宽度，能够再次找到柱子。增加了异常处理功能的雷达定位程序大大提高了稳定性。不管车体在什么位置，什么姿态，只要雷达能够照射到柱子，不被车体遮挡，就能得到绝对世界坐标系下的绝对位置。即使对手做恶意干扰，机器人仍能正常定位。代码实现如下。LostRobotX、LostRobotX 代表从全场定位接收到的 xy 坐标，lidarx、lidary 表示雷达坐标系下雷达的坐标，thirdA、thirdB 表示雷达到两根柱子的距离，A、B 表示柱子与雷达所成的线与 Y 轴形成的夹角。Deltax、deltay 表示因为车体航向角偏离产生的雷达坐标修正值，RAgl 表示从全场定位接收到的航向角。LIDAR_X、LIDAR_Y、LIDAR_Z 表示全场定位发送到本地存储的 X、Y 和航向角值。

```
LostRobotX=0;
LostRobotY=0;
```

```
lidarx=0;
lidary=0;
thirdA=0;
thirdB=0;
A=0;
B=0;
RAgl=0;
deltax=0;
deltay=0;

LIDAR_X=0;
LIDAR_Y=0;
SEND_LIDAR_REQUEST_MSG(CAN1, 8);
while(1)
      if(LIDAR_X!=0&&LIDAR_Y!=0)
            break;
Endif
endwhile
LostRobotX=LIDAR_X;
LostRobotY=LIDAR_Y;
RAgl=LIDAR_Z;

deltax=365*sin(RAgl/180*PI);
deltay=365*(1-cos(RAgl/180*PI));
oldLidarX=(int)(LostRobotX+X_DIFFERENT+deltax);
oldLidarY=(int)(LostRobotY-Y_DIFFERENT+deltay);
oldLdaThreAgl=LIDAR_Z;
lidarx=(int)(LostRobotX+X_DIFFERENT+deltax);
lidary=(int)(LostRobotY-Y_DIFFERENT+deltay);

realAX=abs((int)(lidarx-pillarsdst));
thirdA2=realAX*realAX+lidary*lidary;
thirdA=(int)sqrt(thirdA2);
cosA=(float)lidary/thirdA;
A=acos(cosA)/PI*180;
oldPillarDstA=thirdA;
int pd = pillarsdst;
if((lidarx-pd)>0)
      LidarstepA=540-(int)(A/0.25)-(int)(RAgl/0.25);
endif
else
      LidarstepA=540+(int)(A/0.25)-(int)(RAgl/0.25);
endelse
```

```
realBX=abs((int)lidarx);
thirdB2=lidarx*lidarx+lidary*lidary;
thirdB=(int)sqrt(thirdB2);
cosB=(float)lidary/thirdB;
B=acos(cosB)/PI*180;
oldPillarDstB=thirdB;
if(lidarx>0)
    LidarstepB=540-(int)(B/0.25)-(int)(RAgl/0.25);
endif
else
    LidarstepB=540+(int)(B/0.25)-(int)(RAgl/0.25);
endelse
    lstepA=LidarstepA;
    lstepB=LidarstepB;
```

需要特别注意的是，车体的旋转会影响纠正雷达的距离和角度，有可能导致纠正后找不到柱子。

3.2.10 扫描方向跟随柱子移动

该功能实现方式的流程图如下

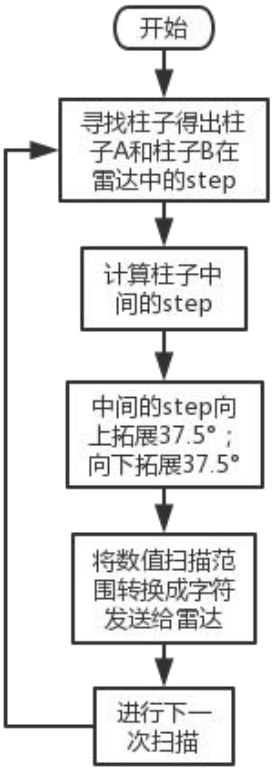


图 3.37 纠正扫描方向流程图

为了尽量减少我方手动车和对方手动车干扰，以及其他未知的干扰，笔者将雷达的

扫描范围限定在了 75°，75°的扫描范围必须时刻都对准两根柱子。这一部分代码没有编写在寻找柱子的函数中，而是编写在了雷达死循环的主体里。根据这次检测到的柱子在雷达的 step 位置，将 75°扫描范围纠正到柱子的正中间；下次找到柱子时，再进行这一过程，便能够完成扫描方向跟随柱子移动的任务。实现代码如下，lstepA 和 lstepB 代表这次扫描得到的柱子在雷达视野中的 step。tcp_client_sendbuf2 是用来修改和发送指令的字符数组。

```
midstep=(lstepA+lstepB)>>1;
maxstep=midstep+(ALL_SCAN_STEP>>1);
minstep=midstep-(ALL_SCAN_STEP>>1);

if(minstep<SCAN_MIN)
      minstep=SCAN_MIN;
      maxstep=SCAN_MIN+ALL_SCAN_STEP;
endif
if(maxstep>SCAN_MAX)
      maxstep=SCAN_MAX;
      minstep=SCAN_MAX-ALL_SCAN_STEP;
endif

tcp_client_sendbuf2[6]=(char)(maxstep/1000+0x30);
tcp_client_sendbuf2[7]=(char)(maxstep/100+0x30);
tcp_client_sendbuf2[8]=(char)((maxstep%100)/10+0x30);
tcp_client_sendbuf2[9]=(char)(maxstep%10+0x30);

tcp_client_sendbuf2[2]=(char)(minstep/1000+0x30);
tcp_client_sendbuf2[3]=(char)(minstep/100+0x30);
tcp_client_sendbuf2[4]=(char)((minstep%100)/10+0x30);
tcp_client_sendbuf2[5]=(char)(minstep%10+0x30);
```

3.2.11 气瓶气压控制

设计机器人之初，投射绣球有使用气缸的需求。要使投射稳定，则气缸的气压要稳定。在赛场上，机器人不能拖着气泵或者充气管子移动，只能使用放在车上的压力源。为了使连接气缸的气瓶在气缸运动数次，气体流失一部分后仍有要求的稳定气压，我们需要用一个气压更高的气瓶连接着连接气缸的气瓶。当连接气缸的气瓶压力下降时，让高气压气瓶给它补给气压，便能实现连接气缸气瓶气压的稳定。我们形象的称这种控制气压的方案为“子母瓶”。

控制气压的方案有了，我们需要去寻找合适的检测气压传感器。第一次找传感器，

笔者在天猫上找了下图这款气压传感器

图 3.38 气压传感器

该传感器可输出反映气压的电流、电压，也可以采用 RS485 协议输出气压信息，输出途径比较丰富。

图 3.39 传感器的输出方式

当时笔者的想法是购买电压输出的气压传感器，加上一个 ADC 转换模块转换成数字量便能够给控制器处理。

将寻找的结果汇报给黄老师后，黄老师建议购买大公司靠谱的气压传感器。笔者便第二次寻找传感器。黄老师推荐 SMC 的气压传感器，但最后没有写成方案，机器人也采用了电机投射，控制气压的方案就此作罢。

4 工作总结

4.1 建立数学模型

在做项目过程中，笔者本身暴露了许多问题。其中最严重的问题就是没有建立数学模型的意识，没有充分利用数学模型控制雷达的运转。因为没有明确好数学模型，头脑里没有清晰的思路，导致一个很小的逻辑 bug 要浪费大量的时间去解决。例如修改雷达扫描方向跟随柱子的那部分代码，当扫描范围小于雷达能够扫描的最小值时（不能扫描的地方被车体遮挡），笔者直接将最小的 step 赋值给 GD 指令，并没有修改数学模型。这样做得结果是程序运行不起来，运行就崩溃。所以良好的数学模型意识是很重要的。

4.2 编程经验

笔者这个学期才开始学习控制、写代码，编程经验非常欠缺。主要表现在 C 语言知识基本忘记、遇到奇怪的 bug 无能为力等方面。开始编写程序时，必须要有一本 C 语言学习书籍在手边，笔者都忘了 for 循环结构怎么运行、int 型是有符号还是无符号等等这类知识。在项目中，笔者因为想整型除以整型得到浮点型数据而浪费了大量时间。

4.3 良好的调试方法

调试不能只使用 ST-Link，使用 ST-Link 的 KEIL 窗口只能显示一个数据。对于刷新速度快的程序，它显示不出所有值。像 robocon 这样移动范围大而迅速的车体，我们需要一款无线的，而且能记录程序每次运行状态的工具——蓝牙串口。无线的特性解放了拖着有线移动的束缚，而快速的输出速度能记录每一次的运行结果。从那一点点蛛丝马迹中，我们很容易知道 bug 出现在哪里。