第十七届全国大学生机器人大赛

RoboMaster 2018 机甲大师裁判系统

规范手册

V1.2 2018.01

ROBOMASTER

免责声明

在使用之前，请仔细阅读本声明，一旦使用，即被视为对本声明全部内容的认可和接受。请严格遵守手册、产品说明和相关的法律法规、政策、准确安装和使用该产品。在使用产品过程中，用户承诺对自己的行为及因此而产生的所有后果负责。因用户不当使用、安装、改装造成的任何损失，大疆™ 创新（ DJI™ ）将不承担法律责任。DJI 和 RoboMaster™ 是深圳市大疆创新科技有限公司及其关联公司的商标或注册商标。本文出现的产品名称、品牌等，均为其所属公司的商标或注册商标。本产品及手册，包括与裁判系统配合使用的 RoboMaster Client、RoboMaster Assistant、RoboMaster Server 软件及 DJI WIN 驱动程序，为大疆创新版权所有。未经许可，不得以任何形式修改、复制、翻印或传播。本文档及本产品所有相关的文档最终解释权归 DJI 所有。所有内容，以最新版本号手册为准。

产品使用注意事项

1. 使用前请检查机器人端裁判系统监控装置已安装正确且牢固。
2. 使用前请确保连线正确。
3. 使用前请检查零部件是否完好，如有部件老化或损坏，请及时更换新部件。

阅读提示

符号说明

⚠ 重要注意事项　　操作、使用提示　　词汇解释、参考信息

前置参考阅读

1. RoboMaster 裁判系统用户手册
2. 裁判系统各模块说明书

修改日志

日期	版本	改动记录
2018.01.08	V1.0	首次发布
2018.01.10	V1.1	修改工程机器人的装甲模块 ID 设置规范（P21）
2018.01.18	V1.2	修改 42mm 测速模块枪管尺寸（P15）

目录

裁判系统使用规范说明

为了保证 RoboMaster2018 机甲大师赛的比赛公平公正，机器人对抗结果的评判完全由电子裁判系统自动执行。各参赛队必须严格遵守使用规范的各个事项，正确安装裁判系统，如果违反使用规范则无法通过赛前检录，后果由参赛队自行承担。

机器人裁判系统配置

模块数量 / 机器人类型	主控模块（A 版）	大装甲模块	小装甲模块	相机图传模块（发送端）	场地交互模块	测速模块	定位模块
步兵机器人	1	0	4	1	1	1（17mm）	1
哨兵机器人	1	2	0	0	0	1（17mm）	1
英雄机器人	1	4	0	1	1	1（17mm）和 1（42mm）	1
空中机器人	1（主控模块 B 版）	0	0	1	0	1（17mm）	1
工程机器人	1	0	4	1	1	0	1

安装规范

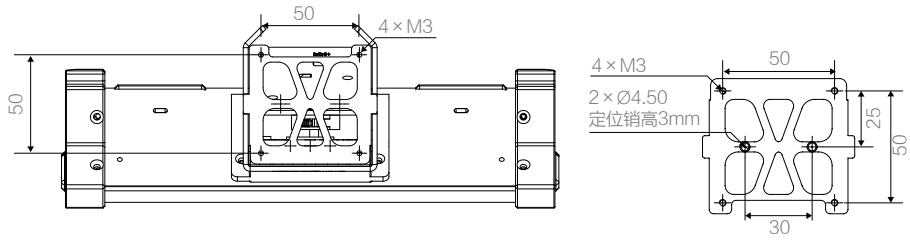
裁判系统是由 RM2018 组委会提供，可记录机器人在比赛中被攻击的情况，如血量值、弹丸发射速度、底盘功率，并将实时信息发送到对应操作间电脑以及裁判系统服务器，自动判定比赛胜负，确保比赛的公平性。参赛队设计的机器人需保留好机械和电气接口以便安装裁判系统。

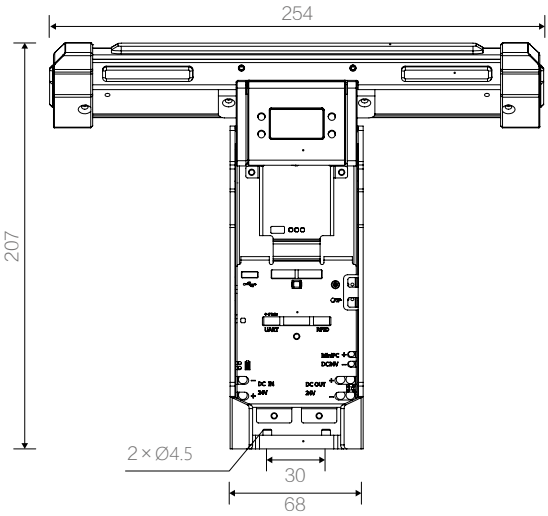
主控模块

安装

步兵机器人、英雄机器人、工程机器人：

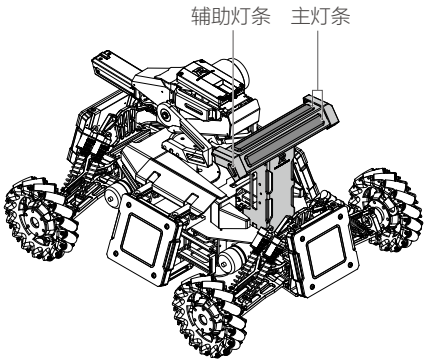
1. 参考主控模块尺寸，在机器人底盘预留安装孔位。主控模块固定座可以进行上下翻转，其中一面有两个距离 30mm，直径为 4.5mm 的定位销，如下图所示，可根据需求选择使用。





单位：mm

2. 使用 4 颗 M3 螺丝固定主控模块至底盘。
3. 主控模块的安装，必须保证左右辅助灯条的连线与地面平行；必须保证至少从一个方向直视机器人时，可以完整看到顶部主灯条的状态。

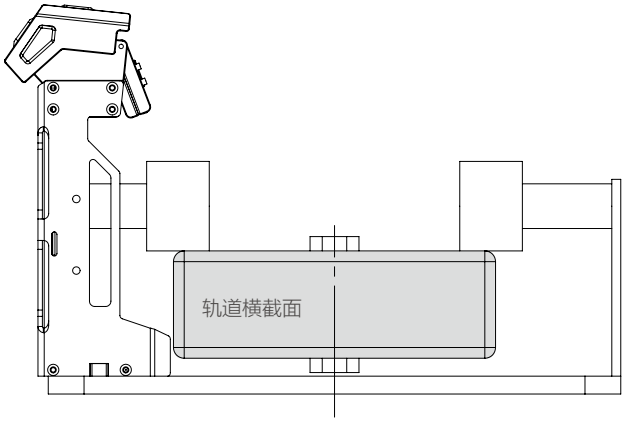


⚠ • 安装步兵机器人的主控模块时，主灯条部分要高于装甲模块。

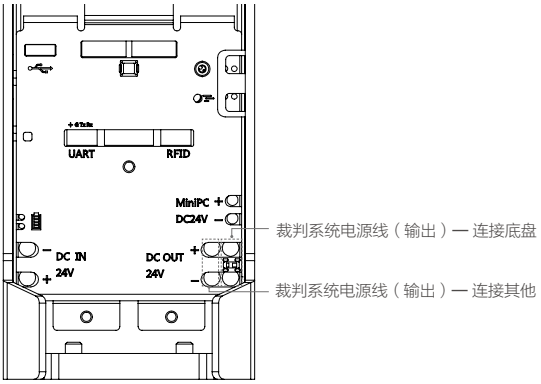
哨兵机器人：

安装步骤与上述步兵机器人一致，但安装位置有所不同。

哨兵机器人挂载在轨道上运行，应确保安装完成后，主控模块在轨道一侧，主控模块灯条部分必须在轨道上表面以上位置，主控模块灯条部分不计入总体尺寸约束。



⚠ • 电流大于 10A 的用电设备可以直接由机器人的电池供电，通过继电器进行控制。继电器必须通过“裁判系统电源线（输出）- 连接其他”接口供电，确保机器人死亡后，裁判系统可以切断“裁判系统电源线（输出）”所连接的所有设备电源，否则视为作弊处理。



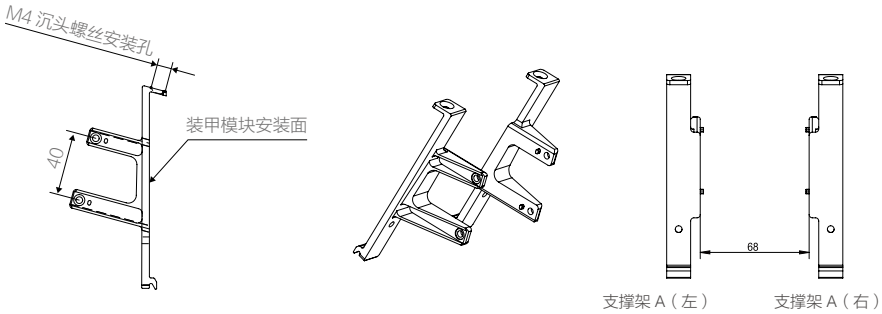
装甲模块

说明

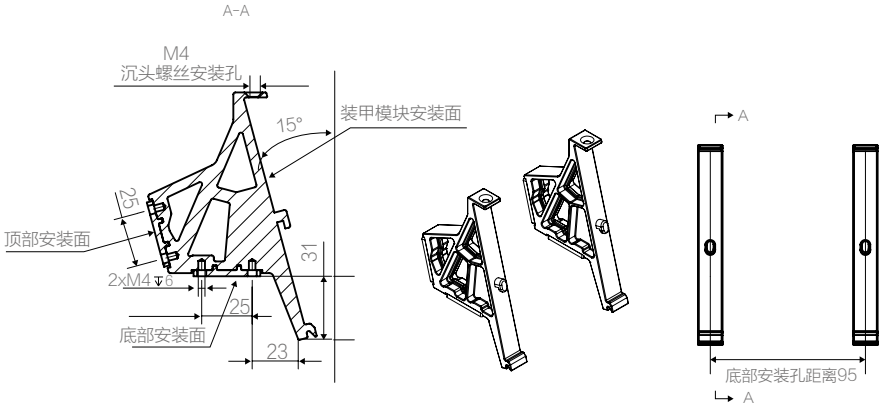
装甲模块需要通过装甲支撑架安装至机器人。

装甲支撑架包含两种，装甲支撑架 A 和装甲支撑架 B。

装甲支撑架 A，有左右之分，如下图所示：

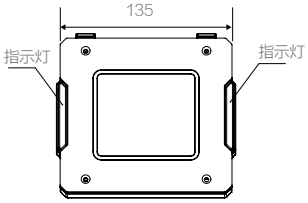


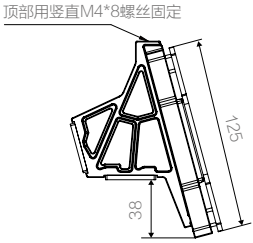
装甲支撑架 B 如下图所示：



装甲模块分为大装甲与小装甲，步兵机器人和工程机器人安装小装甲模块，英雄机器人和哨兵机器人安装大装甲模块。所有机器人装甲安装方式均为侧面安装。

小装甲模块如图所示：

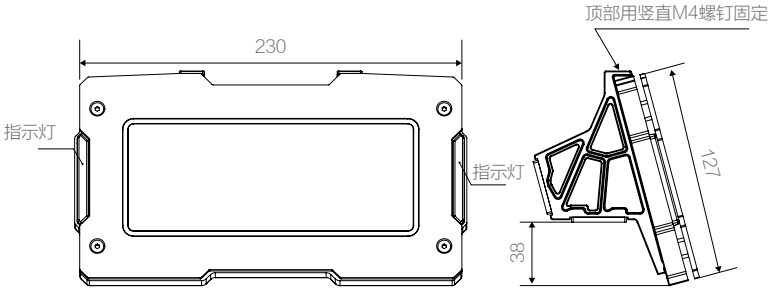




使用底部安装面进行安装

单位：mm

大装甲模块如图所示：



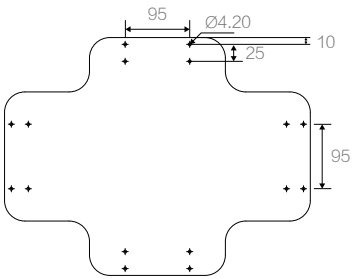
单位：mm

图中标注对于支撑架 A 和 B 同样适用，上图以支撑架 B 为例。

安装步骤

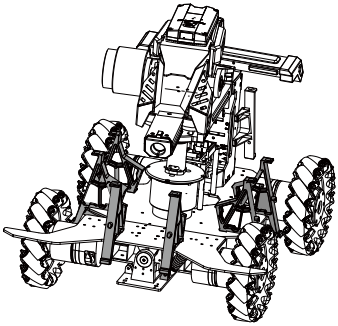
步兵机器人、工程机器人、英雄机器人：

1. 按照下图尺寸，在底盘预留安装孔位，四个安装孔位大小位置保持一致。

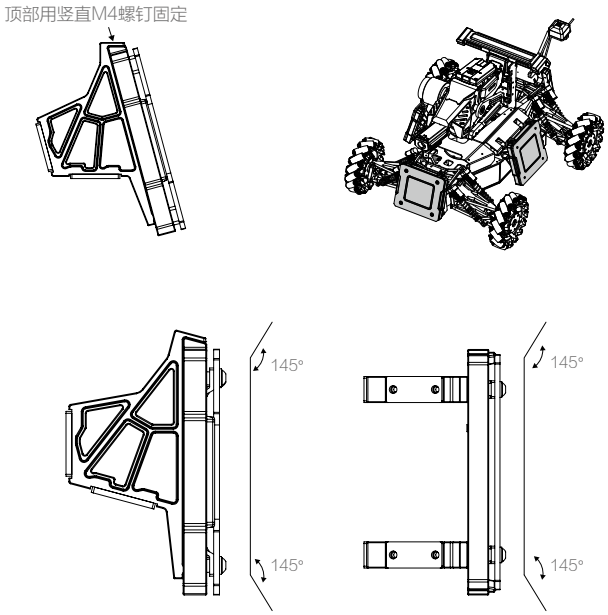


单位：mm

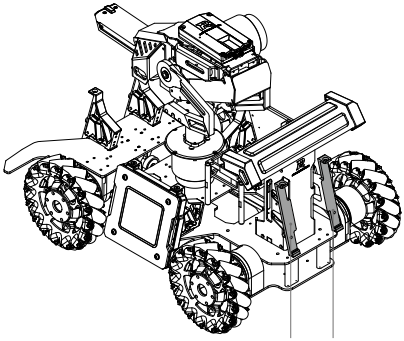
2. 使用 M4 螺丝固定支撑架 B 至三个面的底盘上。



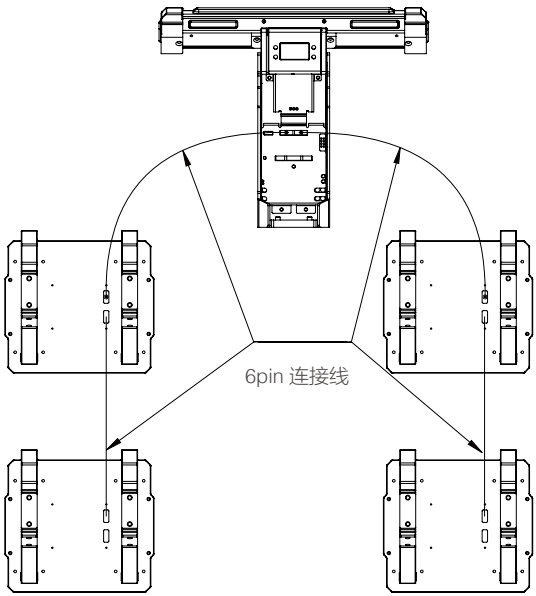
3. 安装装甲模块至支撑架 B，并使用 M4 螺丝固定，装甲支撑架顶部螺纹孔不和支撑架顶面垂直，在正确安装支撑架的情况下顶面螺纹孔与水平面垂直。装甲模块受攻击面 145° 内不得被遮挡。



4. 选装：可使用主控模块两侧的 M4 螺栓将装甲支撑架 A 固定至机器人后部。此时需先安装装甲模块至支撑架 A，再把安装好的整体安装到主控模块上。安装时需区分支撑架 A（左）和支撑架 A（右），注意支撑架的安装方向以保证其与主控模块紧固贴合。



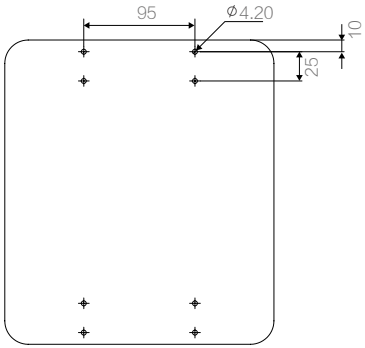
5. 使用包装内提供的 6pin 连接线串联各装甲模块至主控面板的装甲模块接口。装甲模块的两个 6pin 接口为等效接口，连接时建议将装甲模块均分串联至主控面板上的两个接口，以均分该接口的电流。



⚠ • 步兵机器人侧面装甲模块下边沿距离地面高度必须在 60mm - 150mm 范围内。
• 工程机器人侧面装甲模块下边沿距离地面高度必须在 50mm - 400mm。
• 英雄机器人侧面装甲模块下边沿距离地面高度必须高于 400mm，任意两个装甲模块的下边沿在 Z 轴方向的高度差不超过 50mm。

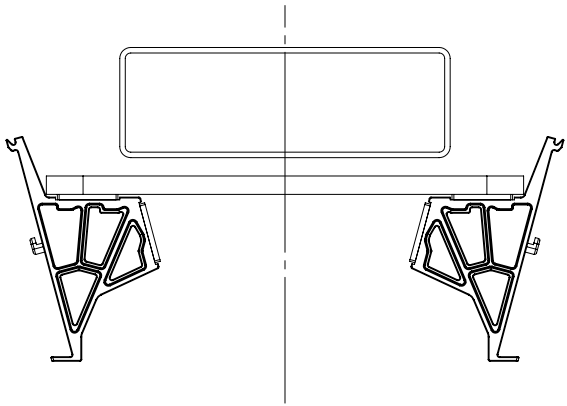
哨兵机器人：

1. 按照下图尺寸，在底盘预留安装孔位，四个安装孔位大小位置保持一致。

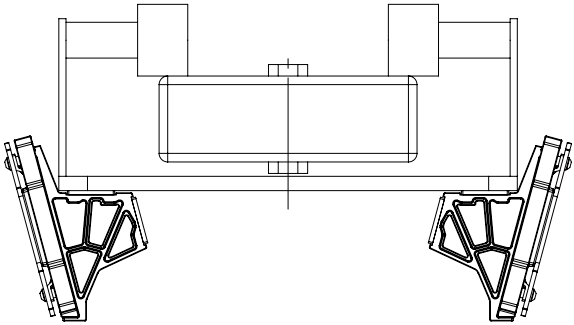


单位：mm

2. 使用 M4 螺丝固定支撑架 B 至底盘。注意螺纹孔在底部。



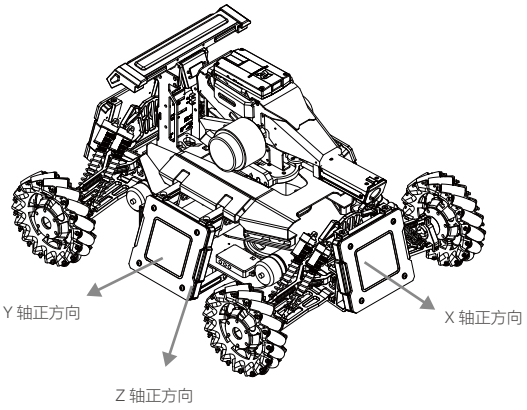
3. 安装大装甲模块至支撑架，并使用 M4 螺丝固定，装甲支撑架顶部螺纹孔不和支撑架底面垂直，在正确安装支撑架的情况下底面螺纹孔与水平面垂直。装甲模块受攻击面 145° 内不得被遮挡。



⚠ • 哨兵机器人在哨兵轨道上表面以下的最大尺寸不超过 450mm、在哨兵轨道上表面以上的最大尺寸不超过 150mm（以上两个尺寸限制都为机器人任意时刻时的尺寸限制）。哨兵机器人挂载在哨兵轨道直线段上时，哨兵大装甲模块的长边需要与哨兵轨道直线段平行；装甲模块的上沿在哨兵轨道上表面所在平面正负 100mm 的范围内。装甲模块所受打击面与赛场地面所在水平面成 75 度夹角，装甲受打击面法线指向战场地面。

安装规范及要求

下文中的讨论中，机器人机体坐标系是标准的 X，Y，Z 笛卡尔坐标系，坐标原点为机器人的质量中心，如下图所示：



机器人本身的运动学方程须建立以笛卡尔坐标系为参考的机体坐标系下。如果参赛机器人使用非笛卡尔坐标系建立运动学模型，则机体坐标系定义为：机器人最大口径的发射机构初始状态下射出弹丸的方向向量投影到 XY 平面作为 X 轴，根据 X 轴和指向地心的 Z 轴按照右手定则生成 Y 轴，原点为机器人的质量中心。

侧面安装

机器人进行侧面安装时装甲模块的受力面和支撑架必须稳固连接。装甲模块的支撑架底部连接面必须与 XY 平面平行，使得装甲模块受力面所在平面的法向量所在直线与 Z 轴负方向所在直线的锐角夹角为 75° 。装甲模块不含指示灯的两条边与 XY 平面保持平行。装甲模块安装好之后必须具备良好的刚性。定义一块安装好的装甲模块受力面所在平面的法向量（与 Z 轴负方向夹角为锐角）在 XY 平面上的投影为该装甲模块的方向向量。4 块装甲模块的方向向量的单位向量必须分别等于机器人机体坐标系的正 X 轴、负 X 轴，正 Y 轴，负 Y 轴（方向向量和对应坐标轴向量之间的角度误差不能超过 5° ）。机器人本身的运动学方程也必须建立在上述作为参考机体坐标系下。装甲模块的安装方式必须与机器人本身的结构特性或者运动学特性共享同一个参考坐标系。X 轴上安装的装甲模块几何中心点连线与 Y 轴上安装的装甲模块几何中心点连线要互相垂直，且连线穿过机器人的几何中心，X,Y 轴的装甲模块允许偏离几何中心正负 50mm。

机器人变形

原则上，比赛开始后，任何一个装甲模块均不能主动地相对于机器人整体的质量中心发生移动。如果参赛机器人因为机器人结构设计需求导致机器人具有可变形特性，则对于装甲模块的要求如下：

1. 任何时候，任何一个装甲模块不可相对于机器人整体的质量中心发生连续、往复的快速移动，短时间移动速度不能超过 0.5m/s。
2. 对于工程机器人来说，变形前后侧面装甲下沿距离地面高度必须在 50mm - 400mm 范围内。
3. 对于英雄机器人来说，变形前后任一装甲模块下沿距离地面高度必须在 400mm 以上。四块侧面装甲模块整体的几何中心点和任一发射机构处于水平时发射管中轴线所在的水平面之间的相对位置在比赛中不能发生变化。
5. 对于哨兵机器人来说 , 变形前后任一装甲模块的上边线必须在哨兵轨道上表面所在平面上下 100mm 的高度上。装甲板相对于轨道平面高度不允许变化；也不得与将机器人挂载在轨道上的结构发生相对水平移动。

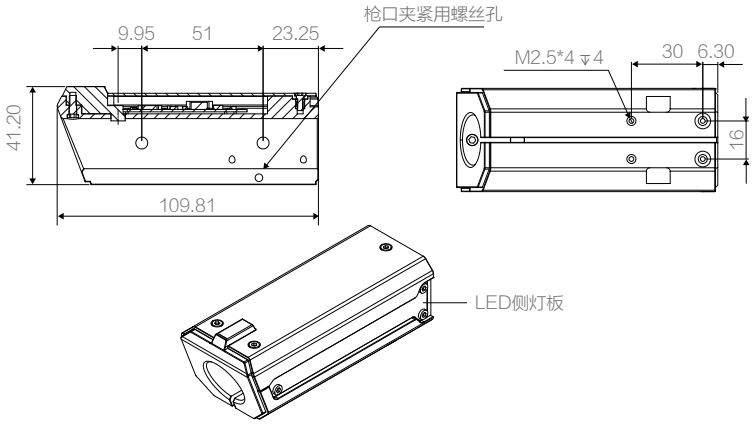
⚠
- 自行设计的保护装甲，不能与 RM2018 组委会提供的装甲模块有任何接触。
- 请勿对 RM2018 组委会装甲模块进行任何修改和装饰。
- 根据机器人自身情况合理连线，保证连接稳固，保护线材避免受损。

测速模块

说明

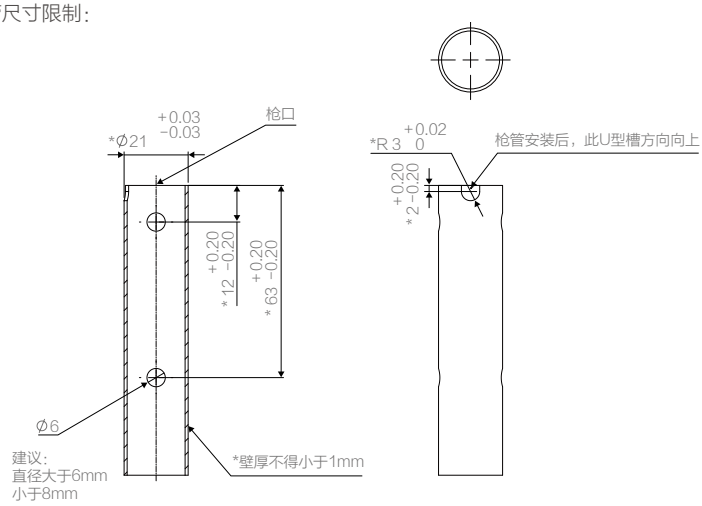
测速模块有两种类型，分别是 17mm 与 42mm 测速模块。

17mm 测速模块：



单位：mm

17mm 枪管尺寸限制：

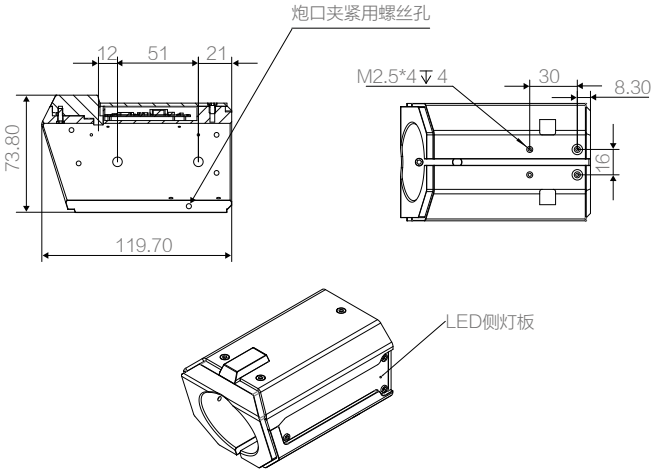


单位：mm

17mm 枪管要求：

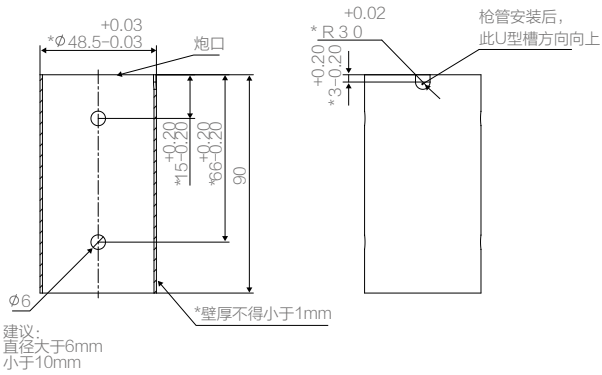
1. 枪管长必须大于 90mm。
2. 加 * 号为参赛选手需要重点掌控尺寸。
3. 保证光电管不被遮挡。
4. 禁止使用透明材料。

42mm 测速模块：



单位：mm

42mm 枪管尺寸限制：



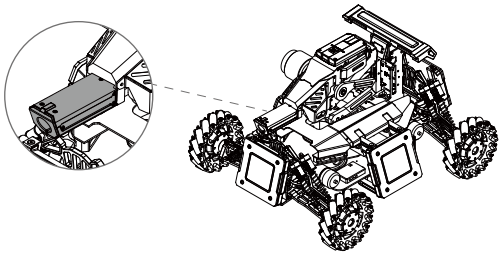
单位：mm

42mm 枪管要求：

1. 枪管长必须大于 90mm。
2. 加 * 号为参赛选手需要重点掌控尺寸。
3. 保证红外对管不被遮挡。
4. 禁止使用透明材料。

安装步骤（以 17mm 测速模块为例）

1. 把测速模块套在枪管上，使圆柱形台阶对齐枪管的 U 形槽，连线一端朝向主控模块。
2. 使用 M3 螺丝穿过测速模块后部的螺丝孔以夹紧枪管。
3. 连接测速模块与主控面板上测速接口的航空插头。安装完成后的效果图如下所示：

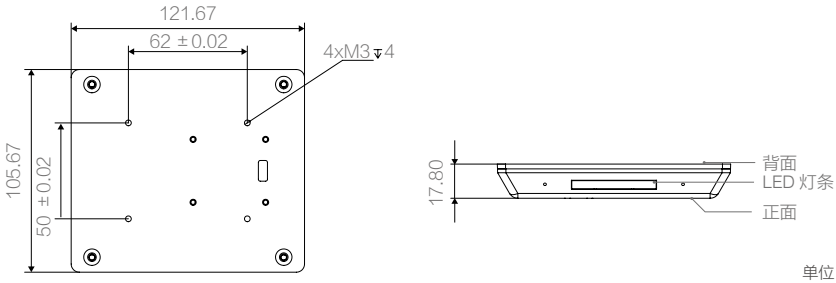


⚠
- 提供 4×M2.5 的螺纹孔，可以安装 RM 激光瞄准器或者自备的激光。自备激光功率必须小于 35mW，并且激光颜色要求为红色。
- 切勿使用双眼直视激光，建议操作中佩戴护目镜。
- 切勿遮挡红外对管的安装孔位。否则会导致测速模块自检不过。
- 注意测速模块要固定牢固，确保使用过程中测速模块和枪口不能发生相对移动。

场地交互模块

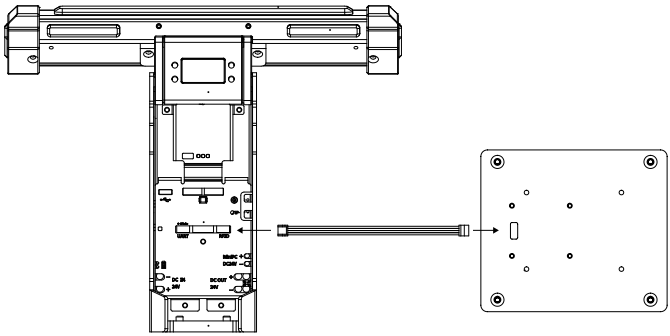
安装

1. 参考场地交互模块结构尺寸和安装接口在底盘预留安装孔位。

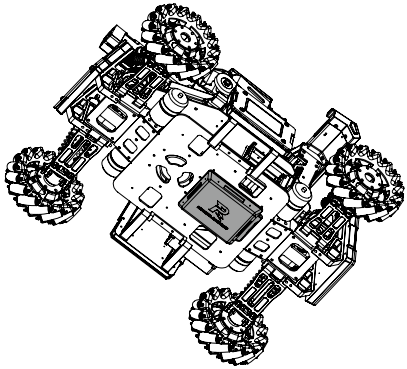


单位：mm

2. 使用包装内提供的 4pin 连接线将裁判系统场地交互模块连接到裁判系统主控模块的主控面板上的 RFID 接口。



3. 使用 M3 螺丝固定裁判系统场地交互模块至底盘，安装时切勿压到连接线，并注意保持交互模块与地面有适当的距离。

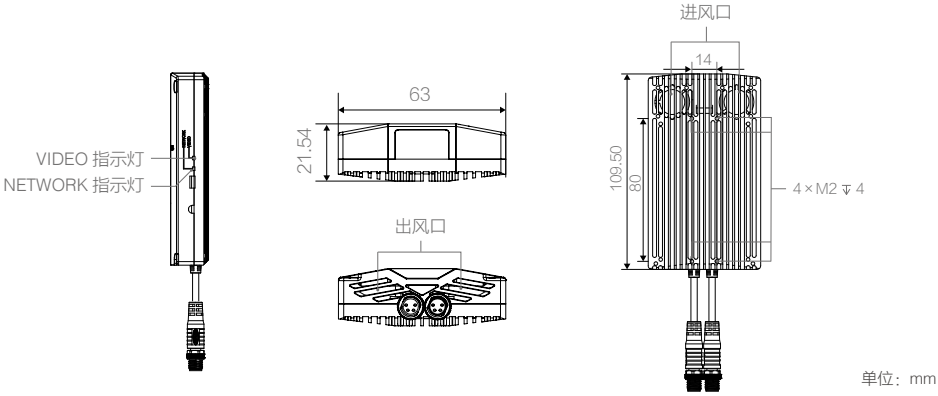


⚠
- 确保场地交互模块有 Logo 的面安装后没有金属遮挡，并且确保没有 Logo 的面安装后没有电流干扰（如电机线、RM 中心板）。安装后实际检测距离以测试为准，如果有效检测距离缩短，请检查安装是否合理。
- 工程机器人使用机器人治疗卡，治疗卡两面均不能有金属遮挡。金属遮挡会减少场地交互模块的有效检测距离，甚至无法检测。
- 赠送的 IC 卡为机器人治疗卡。

相机图传模块

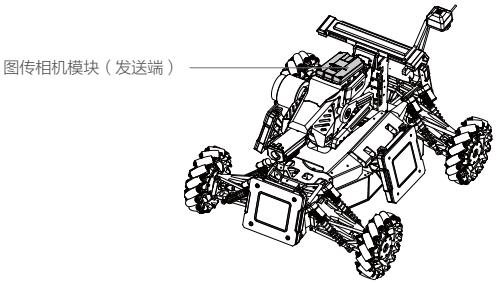
发送端安装

1. 参考发送端结构尺寸和安装接口在所需位置预留安装孔位。



单位：mm

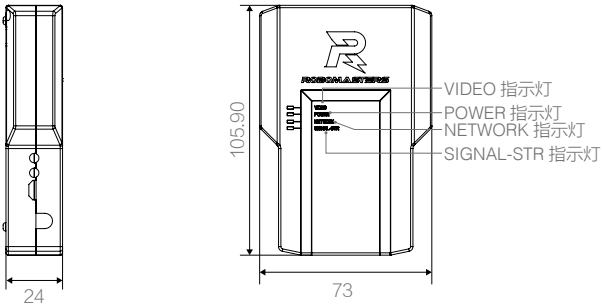
使用 4 颗 M2 螺丝固定发送端至适当位置。安装位置不能遮挡相机图传模块的进风口与出风口；相机图传模块的天线在模块顶部，因此顶部不能有任何金属遮挡。不按要求安装，会导致图传模块工作异常。



2. 发送端的航空插头与主控面板上图传接口的航空插头相连接。

接收端安装

相机图传模块的接收端可以使用配送的安装夹进行固定。固定的位置可以是显示器或者其它支撑物，需要保证固定位置离地高度不低于 1m, 且没有金属遮挡，具体的安装位置，可以通过查看接收图像质量确认。接收端模块如下图所示：

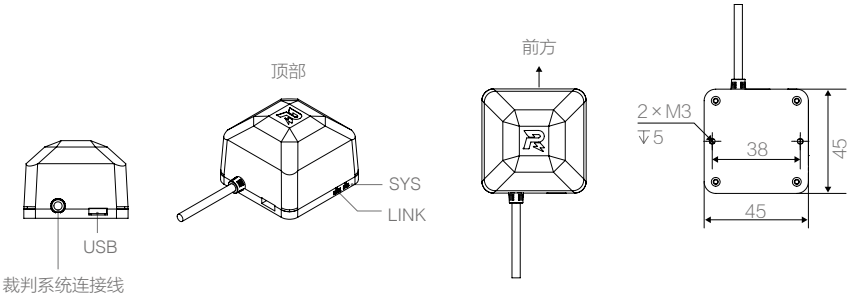


单位：mm

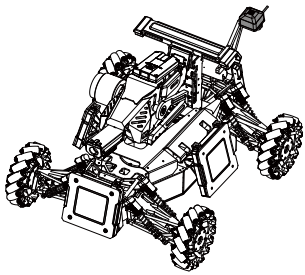
定位模块

安装

1. 参考定位模块尺寸在特定位置预留安装孔位。



2. 使用 2 颗 M3 螺丝固定定位模块至特定位置。定位模块的前方必须与机器人的前方保持一致，并且顶部朝上水平安装。



3. 使用包装内的航空插头对接线连接定位模块至相机图传模块的航空插头。

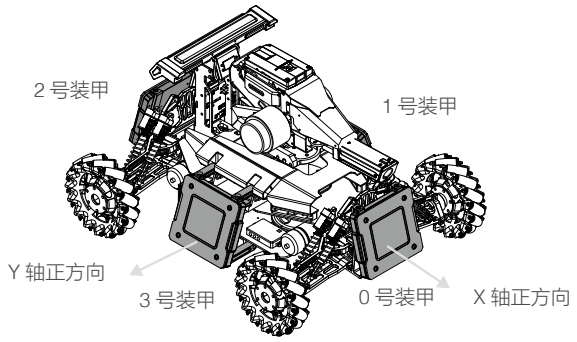
⚠
- 相机图传模块（发送端）、测速模块、定位模块的航空插头和主控模块上的航空插头均为等效接口。
- 定位基站固定在场地四周围栏的顶部，在机器人运动过程中，须全程保证定位模块和各个基站之间的直线连线中间不能有自身遮挡。推荐将定位模块安装在最高点。
- 安装位置离电机、相机图传模块、带磁性或运行过程中会产生强烈磁场的部件距离推荐在 20cm 以上，最短不能少于 10cm。
- 哨兵机器人的定位模块不计入机器人总体尺寸约束。

功能概述及使用规范

装甲模块 ID 设置规范

裁判系统各模块通信使用 CAN 网络，因此每个模块均需要有唯一的 ID 设置，才能保证正常的通信。裁判系统在生产的过程中，对装甲模块均设置了一个默认的 ID 号。因此在第一次连接多块装甲模块，或者重新更新了新装甲模块后，需要对装甲模块进行 ID 设置，ID 设置的步骤如下：

1. 在裁判系统交互主页面下，长按“确认按键”，进入裁判系统“功能页面”。
2. 在“功能页面”下，短按“上下翻按键”选中“System Setup”选项，进入“系统设置页面”。
3. 在“系统设置页面”，短按“上下翻按键”选中“Armor ID Setup”，然后短按“确认按键”将进入装甲 ID 设置模式，选中“Armor ID Reset”，进入装甲 ID 重置状态，此时装甲指示灯以一定频率闪烁（如果机器人 ID 为红方，则红灯闪烁，如果机器人 ID 为蓝方，则蓝灯闪烁）。
4. 以一定力度依次敲击装甲模块，装甲指示灯停止闪烁，表明该装甲 ID 设置成功。首次被敲击的装甲 ID 编号为 0，并且装甲 ID 编号根据敲击的先后顺序依次递增。完成以上操作后，可以通过查询装甲模块的版本号来确认装甲 ID 设置是否成功。如果读取的有效装甲模块数和实际安装的装甲模块数量相同，则表示装甲模块的 ID 设置成功。机器人装甲模块的 ID 编号有严格要求，标识如下图所示：



步兵机器人、英雄机器人和工程机器人：

参考“装甲模块”的安装规范，根据装甲模块安装要求建立的机器人坐标系，X 轴正方向的装甲模块 ID 设置为 0；Y 轴负方向的装甲模块 ID 设置为 1，X 轴负方向的装甲模块 ID 设置为 2，Y 轴正方向的装甲模块 ID 设置为 3。即进入装甲 ID 设置模式后，依次敲击 X 轴正方向、Y 轴负方向、X 轴负方向、Y 轴正方向的装甲模块以完成机器人所有的装甲模块 ID 设置。（侧面的装甲模块 ID 规范设置，主要是在选手的操作页面中提示机器人受攻击的方向信息，同时裁判系统数据输出“实时血量变化信息”也是以此为参考依据）

哨兵机器人：

哨兵机器人只有左右两个装甲，朝向基地一侧的装甲 ID 设置为 0，另一个装甲 ID 设置为 1。

⚠ • 比赛前一定要提前设置好装甲模块的 ID，如果设置异常，裁判系统自检过程中，会检测不到设置错误的装甲模块 ID 号，在比赛过程中会判断为装甲模块离线，自动扣除机器人血量。

功率监测

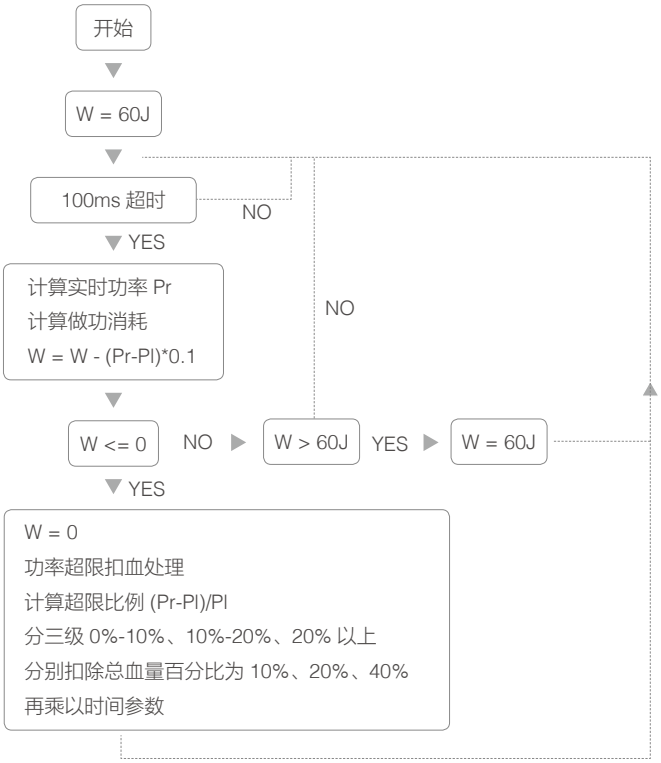
承载和安装机器人动力系统及其附属部件的部分为机器人底盘，可使机器人产生水平方向运动。机器人的底盘功率会被裁判系统持续监控，超出功率后系统会触发惩罚机制，扣除机器人的血量。底盘功率是指负责机器人产生水平方向运动的动力系统的功率，不包含完成特殊任务时使用的动力系统的功率，例如活动上层机械结构等功能性动作。

根据 RoboMaster2018 比赛规则，对各种类的机器人底盘功率输出做了以下限制：

机器人种类	功率上限
步兵机器人	80
哨兵机器人	不限制
英雄机器人	120
空中机器人	不限制
工程机器人	不限制

底盘功率超限的扣除机器人血量值由底盘功率超限比例而定。超限比例的计算公式是：(Pr － PI)/PI, 其中 Pr 代表瞬时底盘输出功率， PI 代表 RoboMaster 比赛规定的限定功率值，具体数值参考上表。如果底盘功率超限值小于或者等于 10%，则扣除机器人上限血量的 10%，如果底盘功率超限值大于 10% 且小于 20%（包括 20%），则扣除机器人上限血量的 20%，如果底盘功率超限大于 20%，则扣除机器人上限血量的 40%。

考虑到机器人在运动过程中，很难做到瞬时输出功率的控制，因此 RM2018 组委会在软件上限定了一个缓冲能量 W， 其值等于 60 焦耳。裁判系统做底盘功率检测的频率是 10HZ，整个检测以及扣除机器人血量的逻辑如下图所示：



举例说明: 以步兵 80W 限制功率为例。假如机器人以 140W 的功率持续输出，那么 1S 后会消耗掉 60J 的能量。在下一个 100ms 的检测周期，计算得到的超限比例 (140-80)/80=75%. 超过限定功率 20%，扣除血量值等于 1500*40%*0.1 = 60。

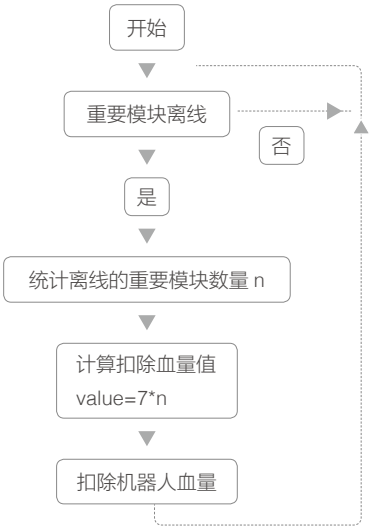
模块状态监测

裁判系统各模块正常工作，是保证比赛公平公正的前提条件。因此，裁判系统在上电启动之后，会对各个模块进行自检。根据对比赛局势的影响程度，对各模块进行了重要程度的两级分类：重要模块与一般模块。

重要模块：装甲模块、测速模块、主控模块、WIFI 模块

一般模块：场地交互模块、相机图传模块、定位模块

重要模块会影响比赛的公平性，如装甲模块，一旦出现异常，将无法检测到敌方攻击。正式比赛中，裁判系统以 2HZ 的频率对各模块进行自检，一旦检测到重要模块离线，将会自动扣除机器人相应血量值。扣血计算方式如下流程图所示:



一般模块离线本身会对己方不利，因此裁判系统不做任何处理，只会通过裁判系统主控模块进行辅助灯黄色闪烁进行提醒。

鉴于裁判系统各模块在比赛过程中的重要性，RM2018 组委会对裁判系统做了充分的测试，各模块均不易损坏，各参赛选手不能私自拆解，更改裁判系统任何部分。

裁判系统自检屏蔽方式，以屏蔽所有模块为例：

⚠ • 屏蔽模块功能只在平时调试时开放，正式比赛前的检录过程中，工作人员会给机器人烧录无法屏蔽这些功能的比赛专用裁判系统固件。

• 为了保证比赛的公平性，参赛选手在比赛的检录环节，有权要求 RM2018 组委会对损坏的裁判系统模块进行维修或者更换。经过检录环节后，裁判系统的功能正常与否将由参赛选手自行负责。

灯条状态说明

状态	主控主灯条	主控前方辅助灯条	主控两侧辅助灯条	RFID 灯	装甲灯	17mm/42mm 测速模块灯
默认状态	显示血量	显示机器人颜色（红 / 蓝）	有等级：周期 N 次闪烁（N 为等级） 无等级：红 \ 蓝常亮	红 / 蓝常亮	红 / 蓝常亮	显示热量
模块升级	绿色显示进度条			/	/	/
找车指令	所有血条绿色闪烁			/	/	/
非比赛中重要模块离线	黄灯常亮	黄灯常亮	黄灯常亮	/	/	/
非比赛中一般模块离线	/	黄色闪烁	黄色闪烁	/	/	/
比赛中重要模块离线	/	黄色常亮	/	/	/	/
上电自检	红 / 蓝自检进度条	/	/	/	/	/
复活	彩灯从中间往两边滚动	/	/	/	/	/
回血	绿色格子从左往右滚动	/	/	/	/	/
热量 / 防御 buff	/	/	/	绿色常亮	/	/
攻击加成	/	白色闪烁	白色闪烁	"一级：黄色常亮 二级：紫色常亮"	/	/
被 17mm 弹丸打击	/	/	/	/	相应装甲闪烁	/
被 42mm 弹丸打击	闪烁	闪烁	闪烁	/	相应装甲闪烁	/
死亡	熄灭	红 / 蓝常亮	红 / 蓝常亮	白色常亮	熄灭	淡红 / 淡蓝常亮
罚下	熄灭	熄灭	熄灭	白色常亮	熄灭	白色常亮

赛前 20s 系统自检

在三分钟比赛准备阶段结束后，会进入 20 秒的裁判系统自检阶段，自检结束后才正式开始比赛。自检过程中，比赛服务器会自动检测客户端连接状态，比赛机器人无线连接状态，机器人模块状态，场内道具状态等。若状态不符合开始比赛需求，如客户端离线，机器人离线，场内道具离线等，比赛自检倒计时将会暂停，待修复好故障设施后，由裁判恢复自检，自检倒计时继续。进入 20s 系统自检时，比赛服务器会恢复所有机器人血量，确保正式比赛开始时，所有机器人为满血状态。20s 裁判系统自检阶段，机器人操作手（除空中机器人操作手）只能待在比赛操作间，空中机器人操作手只能待在空中机器人操作间，在这期间选手可以检查用于比赛使用的电脑鼠标、键盘是否功能正常。

比赛信息接口

为了方便参赛选手更好的做自动控制以及获得比赛的实时状态，裁判系统设计了一路 UART 输入输出口，输出比赛中机器人以及比赛战场的部分数据；同时可以透传参赛选手自定义的部分数据，显示在参赛选手的操作 UI 上。输出的信息包含比赛剩余时间、机器人剩余血量、底盘输出的实时电流电压值，显示在裁判系统客户端操作界面的 UI 上等；开放的上行数据接口，保留了三个浮点型数据和一个无符号字符型数据，参赛选手可以根据“裁判系统接口协议说明”进行实现，上行数据最终会在对应的操作手页面呈现，其中浮点型数据会显示具体数值，无符号字符型数据会按位显示为状态指示。

比赛地理围栏

使用目的

防止除比赛以外的机器人接入比赛系统，干扰比赛的正常进行。

原理简介

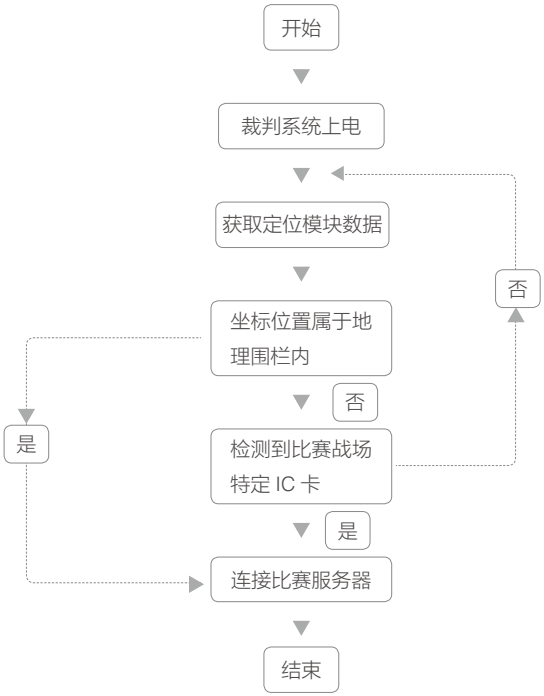
地理围栏以比赛战场作为划分，战场范围内属于地理围栏内部，其余均属于地理围栏外部。机器人只有处在地理围栏内部，才能接入比赛服务器。判断机器人是否属于地理围栏内部，依赖裁判系统的两个模块，分别是定位模块和场地交互模块。

1. 定位模块实现地理围栏区域的判断

   根据“裁判系统安装”章节中的定位模块描述，机器人上安装的定位模块，通过与比赛战场周围安装的定位模块基站进行通信，可以计算获得机器人在战场中的相对位置。根据此位置信息可以准确判断机器人属于地理围栏内部或者外部，如果属于地理围栏以内，则机器人上安装的裁判系统会自动接入比赛服务器。注意，定位模块只有严格按照安装说明进行安装，才能保证相对于比赛战车相对位置计算的准确性。

2. 场地交互模块实现地理围栏区域的判断

   针对此项功能，制作了一批比赛战场特定 IC 卡。比赛战场特定 IC 卡包含地理围栏相关信息，场地交互模块只要检测到此比赛战场特定 IC 卡信息，机器人上安装的裁判系统就会自动接入比赛服务器。此方法只适合在比赛开始前，三分钟准确阶段内或者比赛开始前 20S 裁判系统自检阶段内使用，期间如果发现战场内的机器人没有连接比赛服务器，裁判人员就会使用比赛战场特定 IC 卡使得机器人正常接入比赛服务器。裁判系统利用地理围栏功能接入比赛服务器的处理流程图如下：



⚠ • 地理围栏功能依赖于定位模块，如果安装不正确，在比赛过程中，机器人将会出现意外断电重启。若 15s 内重新接入比赛系统，机器人会恢复上一次血量继续比赛。若超过 15s 该定位模块无法生成正确的定位数据，则会因为读取不到定位模块提供的位置信息而无法接入比赛系统，比赛结束时，服务器只能将机器人剩余血量值判定为 0。

离线模式

裁判系统现分为离线模式和在线模式。裁判系统未连接到服务器时，此时为离线模式。离线模式方便用户在没有服务器连接时依然可以进行调试和使用。在离线模式中，裁判系统可以自动检测弹丸的攻击伤害值，监测机器人的底盘功率，监测枪口热量及射速限制等。但在离线模式时，机器人无法升级，无法获取比赛信息。

在线模式

裁判系统连接到服务器时，此时为在线模式。正式比赛的模式也是在线模式。在线模式中，机器人会受到比赛机制的影响。在线模式和离线模式可自动切换。

附 录

裁判系统接口协议说明

通信协议格式

FrameHeader(5-Byte)	CmdID(2-Byte)	Data(n-Byte)	FrameTail(2-Byte, CRC16)

FrameHeader 格式

域	偏移位置	大小（字节）	详细描述
SOF	0	1	数据帧起始字节，固定值为 0xA5
DataLength	1	2	数据帧内 Data 长度
Seq	3	1	包序号
CRC8	4	1	帧头 CRC8 校验

命令码 ID

简介

命令码	功能说明
0x0001	比赛机器人状态，10Hz 频率周期发送
0x0002	伤害数据，收到伤害时发送
0x0003	实时射击数据，发射弹丸时发送
0x0004	实时功率和热量数据，50Hz 频率周期发送
0x0005	实时场地交互数据，检测到 RFID 卡时 10Hz 周期发送
0x0006	比赛结果数据，比赛结束时发送
0x0007	获取到 buff，激活机关后发送一次
0x0100	参赛队自定义数据，用于显示在操作界面

详细说明

比赛机器人状态 (0x0001)

字节偏移	大小	说明
0	2	当前阶段剩余时间，单位 s
2	1	当前比赛阶段 0：未开始比赛 1：准备阶段 2：自检阶段 3：5s 倒计时 4：对战中 5：比赛结算中
3	1	机器人当前等级
4	2	机器人当前血量
6	2	机器人满血量
8	1	位置、角度信息有效标志位 0：无效 1：有效
9	4	位置 X 坐标值
13	4	位置 Y 坐标值
17	4	位置 Z 坐标值
21	4	枪口朝向角度值

结构体定义：

```
typedef __packed struct
{
  uint8_t  validFlag
  float    x;
  float    y;
  float    z;
  float    yaw;
}positon_t;
typedef __packed struct
{
  uint16_t   stageRemianTime;
  uint8_t    gameProgress;
  uint8_t    robotLevel;
  uint16_t   remainHP;
  uint16_t   maxHP;
```



```
  positon_t  position;
}extGameRobotState_t;
```

伤害数据 (0x0002)

字节偏移	大小	说明
0	1	0-3bits：若变化类型为装甲伤害时，标识装甲 ID 0x0：0 号装甲（前） 0x1：1 号装甲（左） 0x2：2 号装甲（后） 0x3：3 号装甲（右） 0x4：4 号装甲（上 1） 0x5：5 号装甲（上 2） 其他保留 4-7bits：血量变化类型 0x0：装甲伤害（受到攻击） 0x1：模块掉线 0x2：弹丸超速 0x3：弹丸超频 0x4：枪口超热量 0x5：底盘超功率

结构体定义：

```
typedef __packed struct
{
  uint8_t  armorType  : 4;
  uint8_t  hurtType   : 4;
}extRobotHurt_t;
```


实时射击信息 (0x0003)

字节偏移	大小	说明
0	1	弹丸类型 1：17mm 弹丸 2：42mm 弹丸
1	1	弹丸射频
2	4	弹丸射速
6	4	保留

结构体定义：

```
typedef __packed struct
{
  uint8_t  bulletType;
  uint8_t  bulletFreq;
  float    bulletSpeed;
  float    reserved;
}extShootData_t;
```

实时功率热量数据 (0x0004)

字节偏移	大小	说明
0	4	底盘输出电压
4	4	底盘输出电流
8	4	底盘输出功率
12	4	底盘功率缓冲
16	2	17mm 枪口热量
18	2	42mm 枪口热量

结构体定义：

```
typedef __packed struct
{
  float    chassisVolt;
  float    chassisCurrent;
  float    chassisPower;
  float    chassisPowerBuffer;
  uint16_t shooterHeat0;
  uint16_t shooterHeat1;
}extPowerHeatData_t;
```


场地交互数据（0x0005）

字节偏移	大小	说明
0	1	卡类型 0：攻击加成卡 1：防御加成卡 2：红方加血卡 3：蓝方加血卡 4：红方大能量机关卡 5：蓝方大能量机关卡
1	1	卡索引号，可用于区分不同区域

结构体定义：

```
typedef __packed struct
{
  uint8_t  cardType;
  uint8_t  cardIdx;
}extRfidDetect_t;
```

比赛胜负数据（0x0006）

字节偏移	大小	说明
0	1	比赛结果 0：平局 1：红方胜 2：蓝方胜

结构体定义：

```
typedef __packed struct
{
  uint8_t  winner;
}extGameResult_t;
```

Buff 获取数据（0x0007）

字节偏移	大小	说明
0	1	Buff 类型 0：攻击加成 1：防御加成 2：获得大能量机关
1	1	加成百分比（比如 10 代表加成 10%）

结构体定义：

```
typedef __packed struct
{
  uint8_t  buffType;
  uint8_t  buffAddition;
} extGetBuff_t;
```

参赛队自定义数据（0x0100）

字节偏移	大小	说明
0	4	自定义数据 1
4	4	自定义数据 2
8	4	自定义数据 3
12	1	自定义数据 4

结构体定义：

```
typedef __packed struct
{
  float  data1;
  float  data2;
  float  data3;
  uint8_t mask;
}extShowData_t;
```

CRC 校验代码示例

```
//crc8 generator polynomial:G(x)=x8+x5+x4+1
const unsigned char CRC8_INIT = 0xff;
const unsigned char CRC8_TAB[256] =
{
0x00, 0x5e, 0xbc, 0xe2, 0x61, 0x3f, 0xdd, 0x83, 0xc2, 0x9c, 0x7e, 0x20, 0xa3, 0xfd, 0x1f, 0x41,
0x9d, 0xc3, 0x21, 0x7f, 0xfc, 0xa2, 0x40, 0x1e, 0x5f, 0x01, 0xe3, 0xbd, 0x3e, 0x60, 0x82, 0xdc, 0x23,
0x7d, 0x9f, 0xc1, 0x42, 0x1c, 0xfe, 0xa0, 0xe1, 0xbf, 0x5d, 0x03, 0x80, 0xde, 0x3c, 0x62, 0xbe, 0xe0,
0x02, 0x5c, 0xdf, 0x81, 0x63, 0x3d, 0x7c, 0x22, 0xc0, 0x9e, 0x1d, 0x43, 0xa1, 0xff, 0x46, 0x18, 0xfa,
0xa4, 0x27, 0x79, 0x9b, 0xc5, 0x84, 0xda, 0x38, 0x66, 0xe5, 0xbb, 0x59, 0x07, 0xdb, 0x85, 0x67, 0x39,
0xba, 0xe4, 0x06, 0x58, 0x19, 0x47, 0xa5, 0xfb, 0x78, 0x26, 0xc4, 0x9a, 0x65, 0x3b, 0xd9, 0x87, 0x04,
0x5a, 0xb8, 0xe6, 0xa7, 0xf9, 0x1b, 0x45, 0xc6, 0x98, 0x7a, 0x24, 0xf8, 0xa6, 0x44, 0x1a, 0x99, 0xc7,
```



```
0x25, 0x7b, 0x3a, 0x64, 0x86, 0xd8, 0x5b, 0x05, 0xe7, 0xb9,
0x8c, 0xd2, 0x30, 0x6e, 0xed, 0xb3, 0x51, 0x0f, 0x4e, 0x10, 0xf2, 0xac, 0x2f, 0x71, 0x93, 0xcd, 0x11,
0x4f, 0xad, 0xf3, 0x70, 0x2e, 0xcc, 0x92, 0xd3, 0x8d, 0x6f, 0x31, 0xb2, 0xec, 0x0e, 0x50, 0xaf, 0xf1,
0x13, 0x4d, 0xce, 0x90, 0x72, 0x2c, 0x6d, 0x33, 0xd1, 0x8f, 0x0c, 0x52, 0xb0, 0xee, 0x32, 0x6c, 0x8e,
0xd0, 0x53, 0x0d, 0xef, 0xb1, 0xf0, 0xae, 0x4c, 0x12, 0x91, 0xcf, 0x2d, 0x73, 0xca, 0x94, 0x76, 0x28,
0xab, 0xf5, 0x17, 0x49, 0x08, 0x56, 0xb4, 0xea, 0x69, 0x37, 0xd5, 0x8b, 0x57, 0x09, 0xeb, 0xb5, 0x36,
0x68, 0x8a, 0xd4, 0x95, 0xcb, 0x29, 0x77, 0xf4, 0xaa, 0x48, 0x16, 0xe9, 0xb7, 0x55, 0x0b, 0x88, 0xd6,
0x34, 0x6a, 0x2b, 0x75, 0x97, 0xc9, 0x4a, 0x14, 0xf6, 0xa8,
0x74, 0x2a, 0xc8, 0x96, 0x15, 0x4b, 0xa9, 0xf7, 0xb6, 0xe8, 0x0a, 0x54, 0xd7, 0x89, 0x6b, 0x35,
};
unsigned char Get_CRC8_Check_Sum(unsigned char *pchMessage,unsigned int dwLength,unsigned
char ucCRC8)
{
unsigned char ucIndex;
while (dwLength--)
{
ucIndex = ucCRC8^(*pchMessage++);
ucCRC8 = CRC8_TAB[ucIndex];
}
return(ucCRC8);
}
/*
** Descriptions: CRC8 Verify function
** Input: Data to Verify,Stream length = Data + checksum
** Output: True or False (CRC Verify Result)
*/
unsigned int Verify_CRC8_Check_Sum(unsigned char *pchMessage, unsigned int dwLength)
{
unsigned char ucExpected = 0;
if ((pchMessage == 0) || (dwLength <= 2)) return 0;
ucExpected = Get_CRC8_Check_Sum (pchMessage, dwLength-1, CRC8_INIT);
return ( ucExpected == pchMessage[dwLength-1] );
}
/*
** Descriptions: append CRC8 to the end of data
** Input: Data to CRC and append,Stream length = Data + checksum
** Output: True or False (CRC Verify Result)
*/
void Append_CRC8_Check_Sum(unsigned char *pchMessage, unsigned int dwLength)
{
unsigned char ucCRC = 0;
if ((pchMessage == 0) || (dwLength <= 2)) return;
```



```
ucCRC = Get_CRC8_Check_Sum ( (unsigned char *)pchMessage, dwLength-1, CRC8_INIT);
pchMessage[dwLength-1] = ucCRC;
}

uint16_t CRC_INIT = 0xffff;
const uint16_t wCRC_Table[256] =
{
0x0000, 0x1189, 0x2312, 0x329b, 0x4624, 0x57ad, 0x6536, 0x74bf,
0x8c48, 0x9dc1, 0xaf5a, 0xbed3, 0xca6c, 0xdbe5, 0xe97e, 0xf8f7,
0x1081, 0x0108, 0x3393, 0x221a, 0x56a5, 0x472c, 0x75b7, 0x643e,
0x9cc9, 0x8d40, 0xbfdb, 0xae52, 0xdaed, 0xcb64, 0xf9ff, 0xe876,
0x2102, 0x308b, 0x0210, 0x1399, 0x6726, 0x76af, 0x4434, 0x55bd,
0xad4a, 0xbcc3, 0x8e58, 0x9fd1, 0xeb6e, 0xfae7, 0xc87c, 0xd9f5,
0x3183, 0x200a, 0x1291, 0x0318, 0x77a7, 0x662e, 0x54b5, 0x453c,
0xbdcb, 0xac42, 0x9ed9, 0x8f50, 0xfbef, 0xea66, 0xd8fd, 0xc974,
0x4204, 0x538d, 0x6116, 0x709f, 0x0420, 0x15a9, 0x2732, 0x36bb,
0xce4c, 0xdfc5, 0xed5e, 0xfcd7, 0x8868, 0x99e1, 0xab7a, 0xbaf3,
0x5285, 0x430c, 0x7197, 0x601e, 0x14a1, 0x0528, 0x37b3, 0x263a,
0xdecd, 0xcf44, 0xfddf, 0xec56, 0x98e9, 0x8960, 0xbbfb, 0xaa72,
0x6306, 0x728f, 0x4014, 0x519d, 0x2522, 0x34ab, 0x0630, 0x17b9,
0xef4e, 0xfec7, 0xcc5c, 0xddd5, 0xa96a, 0xb8e3, 0x8a78, 0x9bf1,
0x7387, 0x620e, 0x5095, 0x411c, 0x35a3, 0x242a, 0x16b1, 0x0738,
0xffcf, 0xee46, 0xdcdd, 0xcd54, 0xb9eb, 0xa862, 0x9af9, 0x8b70,
0x8408, 0x9581, 0xa71a, 0xb693, 0xc22c, 0xd3a5, 0xe13e, 0xf0b7,
0x0840, 0x19c9, 0x2b52, 0x3adb, 0x4e64, 0x5fed, 0x6d76, 0x7cff,
0x9489, 0x8500, 0xb79b, 0xa612, 0xd2ad, 0xc324, 0xf1bf, 0xe036,
0x18c1, 0x0948, 0x3bd3, 0x2a5a, 0x5ee5, 0x4f6c, 0x7df7, 0x6c7e,
0xa50a, 0xb483, 0x8618, 0x9791, 0xe32e, 0xf2a7, 0xc03c, 0xd1b5,
0x2942, 0x38cb, 0x0a50, 0x1bd9, 0x6f66, 0x7eef, 0x4c74, 0x5dfd,
0xb58b, 0xa402, 0x9699, 0x8710, 0xf3af, 0xe226, 0xd0bd, 0xc134,
0x39c3, 0x284a, 0x1ad1, 0x0b58, 0x7fe7, 0x6e6e, 0x5cf5, 0x4d7c,
0xc60c, 0xd785, 0xe51e, 0xf497, 0x8028, 0x91a1, 0xa33a, 0xb2b3,
0x4a44, 0x5bcd, 0x6956, 0x78df, 0x0c60, 0x1de9, 0x2f72, 0x3efb,
0xd68d, 0xc704, 0xf59f, 0xe416, 0x90a9, 0x8120, 0xb3bb, 0xa232,
0x5ac5, 0x4b4c, 0x79d7, 0x685e, 0x1ce1, 0x0d68, 0x3ff3, 0x2e7a,
0xe70e, 0xf687, 0xc41c, 0xd595, 0xa12a, 0xb0a3, 0x8238, 0x93b1,
0x6b46, 0x7acf, 0x4854, 0x59dd, 0x2d62, 0x3ceb, 0x0e70, 0x1ff9,
0xf78f, 0xe606, 0xd49d, 0xc514, 0xb1ab, 0xa022, 0x92b9, 0x8330,
0x7bc7, 0x6a4e, 0x58d5, 0x495c, 0x3de3, 0x2c6a, 0x1ef1, 0x0f78
};
/*
** Descriptions: CRC16 checksum function
```



```
** Input: Data to check,Stream length, initialized checksum
** Output: CRC checksum
*/
uint16_t Get_CRC16_Check_Sum(uint8_t *pchMessage,uint32_t dwLength,uint16_t wCRC)
{
Uint8_t chData;
if (pchMessage == NULL)
{
return 0xFFFF;
}
while(dwLength--)
{
chData = *pchMessage++;
(wCRC) = ((uint16_t)(wCRC) >> 8) ^ wCRC_Table[((uint16_t)(wCRC) ^ (uint16_t)(chData)) & 0x00ff];
}
return wCRC;
}

/*
** Descriptions: CRC16 Verify function
** Input: Data to Verify,Stream length = Data + checksum
** Output: True or False (CRC Verify Result)
*/
uint32_t Verify_CRC16_Check_Sum(uint8_t *pchMessage, uint32_t dwLength)
{
uint16_t wExpected = 0;
if ((pchMessage == NULL) || (dwLength <= 2))
{
return __FALSE;
}
wExpected = Get_CRC16_Check_Sum ( pchMessage, dwLength - 2, CRC_INIT);
return ((wExpected & 0xff) == pchMessage[dwLength - 2] && ((wExpected >> 8) & 0xff) ==
pchMessage[dwLength - 1]);
}

/*
** Descriptions: append CRC16 to the end of data
** Input: Data to CRC and append,Stream length = Data + checksum
** Output: True or False (CRC Verify Result)
*/
void Append_CRC16_Check_Sum(uint8_t * pchMessage,uint32_t dwLength)
{
```



```
uint16_t wCRC = 0;
if ((pchMessage == NULL) || (dwLength <= 2))
{
return;
}
wCRC = Get_CRC16_Check_Sum ( (U8 *)pchMessage, dwLength-2, CRC_INIT );
pchMessage[dwLength-2] = (U8)(wCRC & 0x00ff);
pchMessage[dwLength-1] = (U8)((wCRC >> 8)& 0x00ff);
```

⚠ • Uart 通信配置，波特率 115200，数据位 8，停止位 1，校验位无，流控制无。

RM2018 组委会

邮　　箱：robomaster@dji.com

官方论坛：http://bbs.robomaster.com

官方网站：http://www.robomaster.com

电　　话：075536383255（周一至周五 10:00-19:00）

地　　址：广东省深圳市南山区西丽镇茶光路 1089 号集成电路设计应用产业园 2 楼 202

微信　　微博

RoboMasterNews　　RoboMaster