

2017 RoboMaster 夏令营

技术报告

第一组

2017.8

目 录

| | |
|--------------------|----|
| 1. 机械部分..... | 1 |
| 1.1 设计动机..... | 1 |
| 1.2 设计需求..... | 1 |
| 1.3 设计方案..... | 1 |
| 1.3.1 底盘机构..... | 1 |
| 1.3.2 云台设计..... | 1 |
| 1.3.3 取弹机构设计..... | 2 |
| 1.4 方案的优点与不足..... | 4 |
| 2. 嵌入式部分..... | 5 |
| 2.1 整体方案..... | 5 |
| 2.2 运动学解算方法..... | 6 |
| 2.2.1 运动学模型..... | 6 |
| 2.3 云台与底盘控制方案..... | 7 |
| 2.3.1 底盘控制方案..... | 7 |
| 2.3.2 云台控制方案..... | 16 |
| 2.4 串口高效读取..... | 28 |
| 2.5 难点与不足..... | 30 |
| 3. 算法部分..... | 31 |
| 3.1 开发环境介绍..... | 31 |
| 3.1.1 硬件环境..... | 31 |
| 3.1.2 算法环境..... | 31 |
| 3.2 整体技术方案概述..... | 32 |
| 3.2.1 技术原理介绍..... | 32 |
| 3.3 算法整体框架设计..... | 32 |
| 3.4 算法功能模块说明..... | 33 |
| 3.4.1 定位算法..... | 34 |
| 3.4.1 导航算法..... | 34 |
| 3.4.2 跟踪射击..... | 34 |
| 3.4.3 单兵逻辑..... | 34 |
| 3.4.4 多兵作战..... | 35 |
| 3.4.5 控制..... | 35 |
| 3.5 测试结果..... | 35 |
| 3.6 可优化方案..... | 35 |
| 4 夏令营感想、总结..... | 36 |

1. 机械部分

1.1 设计动机

2017 年 Robomaster 夏令营的比赛规则主要是参赛的两支队伍在核心比赛场地“战场”内进行全自动战术对抗。同时空中机器人可以从停机坪上取子弹并给地面机器人补充子弹。

1.2 设计需求

官方已经提供了一台步兵机器人以及一台空中机器人 M100。机械设计上主要的需求是完成空中机器人挂载的取弹机构以及相应的接弹机构。

1.3 设计方案

1.3.1 底盘机构

底盘我们没有做较大的改动，主要考虑到工作量的问题，另外在现有的前联合悬挂，后轮无悬挂的方式，虽然联合悬挂占用较大的空间，但是也可以接受。不过其减震效果较差，通过性还不错。

1.3.2 云台设计

卡弹问题困扰我们很久，最终我们采用一种补救措施，当出现卡死的时候，我们进行了反转。为了能进行反转，我们需要保证在末端子弹不掉进去。

提高子弹一致性，我们主要从云台稳定性着手。对于 pitch 来说，现有的结构重心偏心严重，这对于电机来说是很不好的。因此我们加了一根皮筋，使得 pitch 方向上负载较小，这样在调节过程中点击有足够的裕量来进行调节。

1.3.3 取弹机构设计

取弹机构设计：

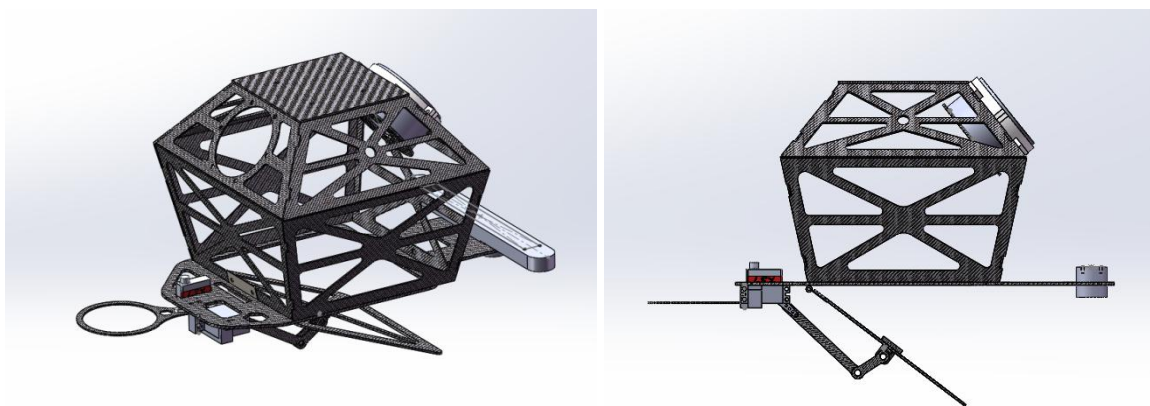


图 1.1 涵道式取弹机构三维模型

我们采用涵道的方式进行取弹，通过涵道产生的负压将子弹吸气。底部加一个舵机，通过连杆结构实现门的开闭，从而实现子弹的释放。为了实现取弹的时候能在较大的范围内取足够的弹量，我们还加了一个用舵机驱动的摆杆来摆动管道口。

如图，整个密封室主要采用碳骨架拼接，表面贴上胶布进行密封。初始在试验中发现在试验中吸弹效果比较差，初步认定是密封问题，因此，我们对其内部流场进行了数值模拟。左下图中是计算的结果，发现其内部流场非常混乱，当底部门关的不是很死的时候就很难产生较强的负压。

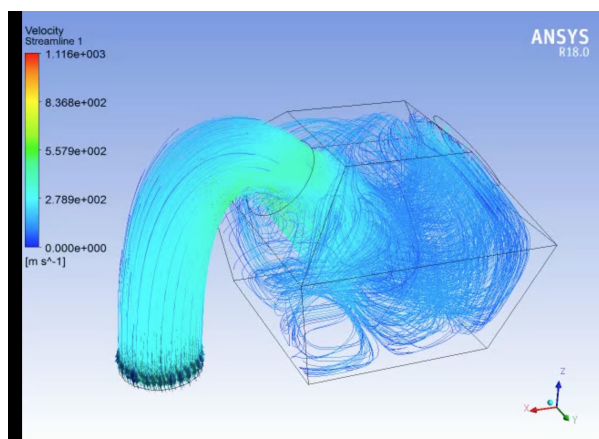


图 1.2(a) 原设计的流场分布

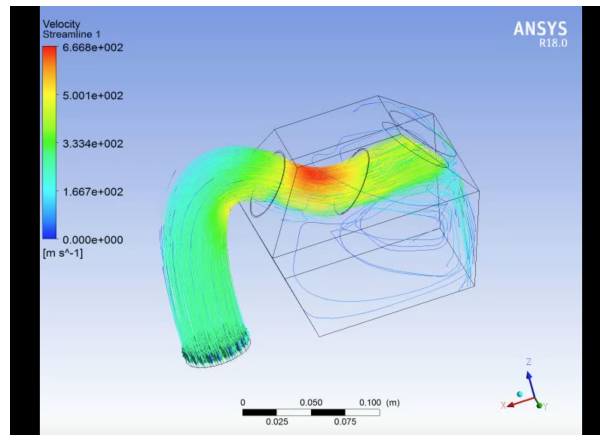


图 1.2(b) 新设计的流场分布

于是我们将其内部流场进行改善，如右图，我们将管道入口靠近涵道口，这样其内部流场好了很多，进行试验发现效果好了很多。这也是我们最终采用的结果。

整个发射机构重量 350g，装弹量约为 200 发，经过实验，飞机可以带动这个负载，并且飞行较为稳定。

接弹机构

交接时，应尽可能对飞机的定位要求小，对投弹的弹道分布要求低。因此应尽可能将接弹部分得尺寸做大且尽量简洁。

应使得在初始尺寸不超标的情况下，尽可能多的利用展开尺寸。联想到伞，扇。伞能使展开尺寸更大，故选伞结构。

连杆的设计：控制量包括初始尺寸，展开尺寸，移动端行程。初始尺寸尽量利用，展开后投影面积尽量大（应大于 400x400）

传动方式选择：输出的运动类型：直线运动。同步带（偏载），气缸（冲击太大且需要添加气路），丝杠螺母副（较合适）

由此设计了下图所示的多连杆机构。其空间利用率较高：收缩投影尺寸（210x210），展开投影尺寸（500x500）。选用 maxon 的 A-Max22 电机减速比 19:1，导程 14 的丝杠，展开所需时间约 0.7 秒。

在制作过程中遇到一些的问题。面材料尝试了多种材料，包括不同厚度的塑料膜，摇粒绒等。最终综合考虑弹性和可折叠性，选择摇粒绒。出现子弹弹出问

题，主要原因是连杆的刚度之间的不够，通过转而改变布的布置。另外出现问题，飞机风力太大，造成布面下压。解决：在布上扎洞。减小风力对布面造成的影响

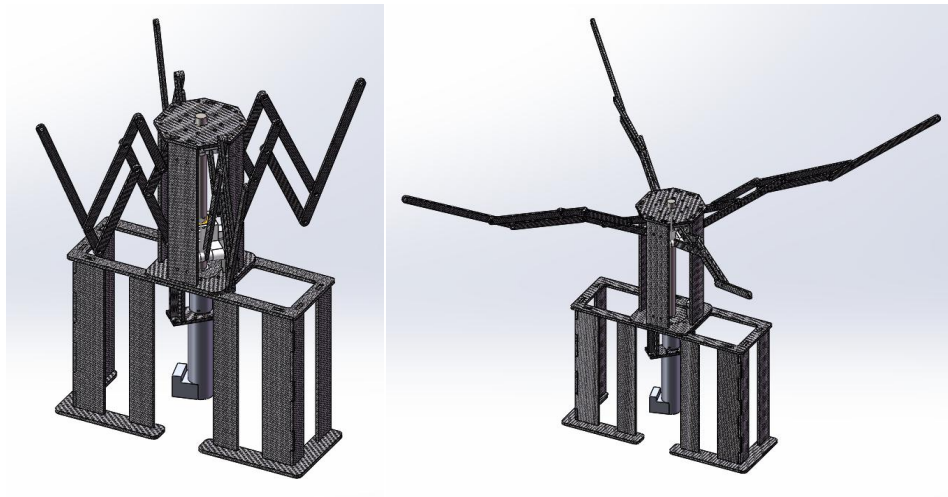


图 1.3 接弹机构三维模型

陆地机器人和空中机器人对接的机构怎么设计的

1.4 方案的优点与不足

使用涵道进行取弹存在一个较大的问题就是效率较低，取弹时间要 6-8s。相对于机械爪来说要慢很多。采用伞结构装弹和会存在拆装复杂的问题，当装了很多弹时，还容易烧毁电机。

2. 嵌入式部分

2.1 整体方案

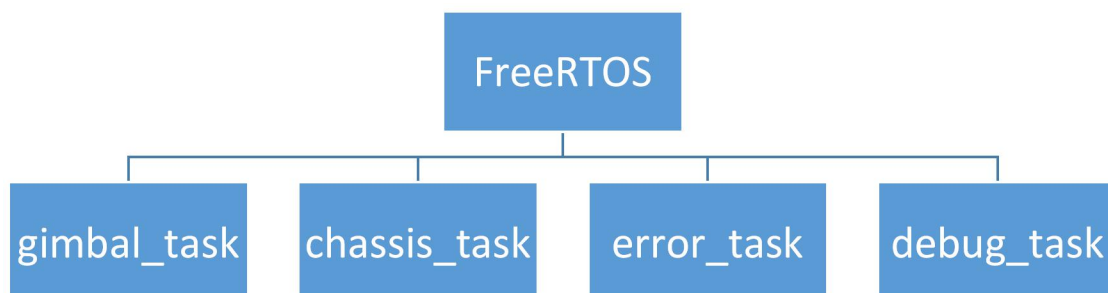


图 2.1 FreeRTOS 的代码框架

嵌入式采用 HAL 库基于 FreeRTOS 的代码框架，通过开启 4 个线程来进行控制与通讯，分别为 `error_task`, `gimbal_task`, `chassis_task` 以及 `debug_task`。`Error_task` 进行错误检测，`Gimbal_task` 进行云台的控制，`Chassis_task` 进行底盘的控制，`Debug_task` 负责调试数据的传输。

在底盘的控制线程中，我们采用传统的运动学解算方法对麦克纳姆轮进行分析解算，再通过 pid 控制方法实现各轮子力矩到速度的控制，通过 pid 调节实现位置角度等的闭环控制，读取 uwb 数据与麦轮解算进行一阶互补滤波，实现位置信息的融合输出。

在云台的控制线程中，我们通过读取视觉传感器的数据，将其作为位置控制环作用在云台控制上，能够有效的实现云台的跟踪闭环功能。通过对子弹射击进行重力模型的弹道校准，实现云台的高效准确打击。

在调试的线程中，我们通过 wifi 串口将调试数据发送到 PC 端进行交互，采用国内匿名飞控的上位机通讯协议，可以实现调试数据的波形显示与数据保存，同时，也方便进行 PID 参数的调节。

在错误检测的线程中，我们通过间隔时间检测的方法进行 can 总线上各电机的离线检测与遥控器检测，并在离线时进行蜂鸣器警报。

2.2 运动学解算方法

2.2.1 运动学模型

Mecanum 轮的轮体周围等距的遍布鼓形的小辊子，这些辊子能够自由旋转且外廓线与轮子的理论圆周相重合。Mecanum 轮有三个方向的自由度：绕主轮轴的转动和沿辊子轴线垂线方向的平动和绕辊子与地面接触点的转动。当轮子向一个方向上主动移动的时候，在另外一个方向上轮子也具有被动自由移动的效

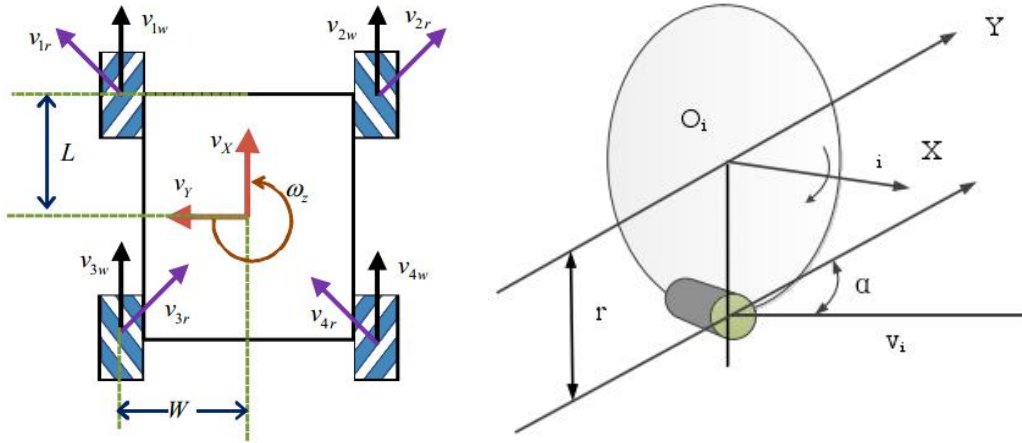


图 2.2 Mecanum 轮布局与力学模型

果。在实际工程中辊子轴与轮轴的夹角大小一般取 45° 角以保证多个轮子之间运动的均衡性。如图 2.2 为四轮移动机器人的平面布局图，其中轮子中的斜线表示轮子上辊子的轴线方向，相同方向的轮子对角放置，而两个前轮和两个后轮之间的轮子方向相反 [4]。车体关于自身的纵向的抽线程对称的关系。以机构系统的中心点 O 为原点，建立系统的自身的机器人运动坐标系，图中 R 为轮子的半径， L, W 分别表示机器人机构的长度和宽度的一半。

$$v_x = \frac{R_w}{4} (\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4)$$

$$v_y = \frac{R_w}{4} (-\dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 - \dot{\theta}_4)$$

$$w_z = \frac{R_w}{4(L+W)} (-\dot{\theta}_1 + \dot{\theta}_2 - \dot{\theta}_3 + \dot{\theta}_4)$$

对应的嵌入式代码为：


```

void Body_Speed_Calculate(int16_t Wheel1_Speed, int16_t Wheel2_Speed,
int16_t Wheel3_Speed, int16_t Wheel4_Speed)
{
int Body_Speed_x, Body_Speed_y, Body_Speed_z;
Body_Speed_x=(Wheel1_Speed+Wheel2_Speed+Wheel3_Speed+Wheel4_Speed)*r/4;
Body_Speed_y=Wheel1_Speed-Wheel2_Speed-Wheel3_Speed+Wheel4_Speed*r/4;
Body_Speed_z=(-Wheel1_Speed+Wheel2_Speed-Wheel3_Speed+Wheel4_Speed)*r/4*(a+b);
}

```

若 Mecanum 轮不存在打滑现象，则可根据式计算得到四个轮子的速度。

$$\begin{bmatrix} V_{1W} \\ V_{2W} \\ V_{3W} \\ V_{4W} \end{bmatrix} = K \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} 1 & -1 & -(L+W) \\ 1 & -1 & (L+W) \\ 1 & 1 & -(L+W) \\ 1 & -1 & L+W \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ \omega_z \end{bmatrix}$$

2.3 云台与底盘控制方案

2.3.1 底盘控制方案

2.3.1.1 动力学模型

战车的动力学建模采用拉格朗日函数法，从系统总体出发，用广义坐标确定系统位置，用动能和势能表述对象的运动量和互相作用，是一种力的动态平衡，不必考虑系统内部互相作用，适用于复杂系统的动力学建模。

战车一直处于平面，所以系统的势能为零。拉格朗日函数 L 如下

$$L = E_k = \frac{1}{2}m(\dot{x}^2 + \dot{y}^2) + \frac{1}{2}J_z\dot{\theta}_z^2 + \frac{1}{2}\sum_{i=1}^4 J_w\dot{\theta}_w^2$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \frac{\partial L}{\partial q_j} + \frac{\partial \phi}{\partial q_j} = Q_j q$$

其中， m 为战车的质量， E_k 为平台的动能， J_z 和 J_w 分别为平台和轮子的转动惯量。

一般的，运动方程表示为式的形式，

$$M\ddot{\theta} + D\dot{\theta} = T$$

其中

$$M = \begin{bmatrix} GH & -H & H & G-H \\ -H & GH & G-H & H \\ H & G-H & GH & -H \\ G-H & H & -H & GH \end{bmatrix}$$

$$G = \frac{mR^2}{8}, \quad H = \frac{IR^2}{16(L+W)}, \quad GH = G + H + I_\omega; \quad J_z = \frac{1}{12}m(L^2 + W^2)$$

从而运动方程如下：

$$\ddot{\theta} = -M^{-1}D\dot{\theta} + M^{-1}T$$

由上式可知，因此状态空间模型 $V(s)$ 表示为

$$\begin{aligned} \dot{x} &= A_c x + B_c u \\ y &= C_c x_c \end{aligned}$$

其中， $A_c = -M^{-1}D, B_c = nM^{-1}, C_c = I_4 \in R^{4 \times 4}, x = \dot{\theta}_w, u = \tau_m \in R^4, y = x$,

对此，采用一种解耦控制的方法来提高系统的精度

2.3.1.2 电机模型

比赛所用电机的控制内环中电流环已由电调完成，所以电机模型方面没有深究。一般电机控制变量往往是电压，具体的模型如下所示



图 2.3 电机与车轮实物图片

根据无刷直流力矩电机的模型，可以得到：

$$\begin{cases} U = R_a i_a + E_g \\ E_g = K_e n \\ M = K_T i_a \\ n = 9.55\omega \end{cases}$$

公式中， U 表示电机电枢两端的电压， R_a 表示电机电枢的等效电阻， E_g 表示电机反电动势， i_a 表示流过电枢的电流。 n 为电机旋转角速度（ rad/s ）， M 为电机转矩（ $\text{N}\cdot\text{m}$ ）， K_T 为电机转矩系数（ $\text{N}\cdot\text{m/A}$ ）， K_e 为反电动势系数（ V/r/min ）。

实际过程中电机实际曲线

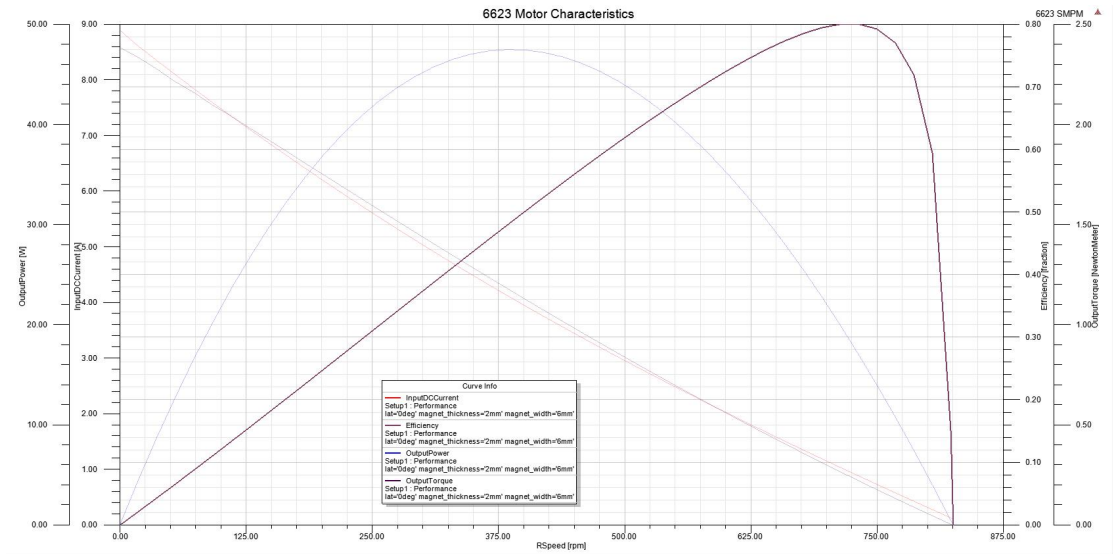


图 2.4 实际过程中电机实际曲线图

可得公式 $K_e: 0.274 \text{ Vs/rad}$, $K_V: 35 \text{ RPM/V}$, $K_t: 0.268 \text{ Nm/A}$

虽然，电流的产生往往存在一定的瞬态过程；但是 RM 比赛电机在控制过程内部存在一个电流环，无法控制电机的电压。所以在仿真分析的过程中用一个一阶惯性环节来近似电机模型。（一）控制模型的化简

状态空间模型 $V(s)$ 表示为：

$$\begin{aligned} \dot{x} &= A_c x + B_c u \\ y &= C_c x \end{aligned}$$

其中， $A_c = -M^{-1}D$, $B_c = nM^{-1}$, $C_c = I_4 \in R^{4 \times 4}$, $x = \dot{\theta}_w$, $u = \tau_m \in R^4$, $y = x$,

具体参数如下

$R = 0.075 \text{m}$, $L = 0.205 \text{m}$, $W = 0.185 \text{m}$, $m = 18.5 \text{Kg}$, $D = 18.5 (\text{Nms/rad})$,

$n = 19$; $J_w = 0.01665 \text{kgm}^2$

$$J_z = \frac{1}{12} m (L^2 + W^2) = \frac{1}{12} * 18.5 * (0.41^2 + 0.37^2) = 0.47 \text{ kgm}^2$$

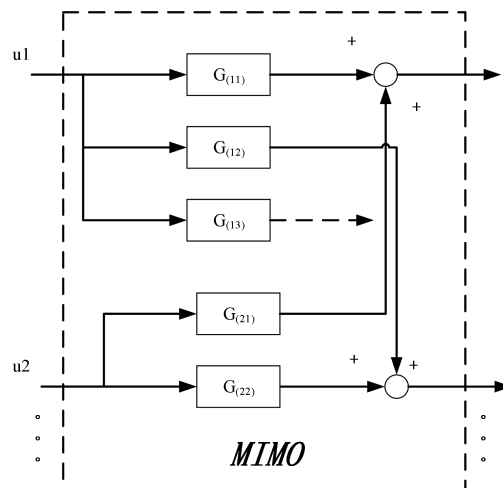
代入

$$G = \frac{mR^2}{8} = \frac{14 * 0.05^2}{8} =, \quad H = \frac{J_z R^2}{16(L+W)} = \frac{2 \text{ kg} \cdot \text{m}^2 * 0.05^2 \text{ m}^2}{16 * (0.7 \text{ m})} =, \quad GH = G + H + I_\omega;$$

用一个 4x4 的传递函数矩阵来描述上述 MIMO 系统,将 $V(s)$ 转化成传递函数的形式即

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} & G_{14} \\ G_{21} & G_{22} & G_{23} & G_{24} \\ G_{31} & G_{32} & G_{33} & G_{34} \\ G_{41} & G_{42} & G_{43} & G_{44} \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix}$$

其中 G_{ij} 表示从输入 i 到输出 j , 即传递函数的框图如下 (取 2 个控制通道作为例子)。



通过 matlab 的计算可得 G_{ij} 的表达式

$$\begin{aligned} & 4610 s^3 + 3.4527e06 s^2 + 2.008e08 s + 4.433e09 \\ 1: & \text{-----} \\ & s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4533e08 \\ & 1439 s^3 + 1.04e05 s^2 + 1.878e06 s + 6.491e-08 \\ 2: & \text{-----} \\ & s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4533e08 \\ & -1439 s^3 - 1.04e05 s^2 - 1.878e06 s - 8.889e-07 \\ & \text{-----} \\ & s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4533e08 \end{aligned}$$

$$\frac{-3923 s^3 - 1.686e06 s^2 - 5.578e07 s + 4.306e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 4: -----
 From input 2 to output...

$$\frac{1439 s^3 + 1.04e05 s^2 + 1.878e06 s - 7.891e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 1: -----

$$\frac{4610 s^3 + 3.4.527e06 s^2 + 2.008e08 s + 4.433e09}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 2: -----

$$\frac{-3923 s^3 - 1.686e06 s^2 - 5.578e07 s - 3.955e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 3: -----

$$\frac{-1439 s^3 - 1.04e05 s^2 - 1.878e06 s - 6.179e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 4: -----
 From input 3 to output...

$$\frac{-1439 s^3 - 1.04e05 s^2 - 1.878e06 s + 3.772e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 1: -----

$$\frac{-3923 s^3 - 1.686e06 s^2 - 5.578e07 s + 1.523e-07}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 2: -----

$$\frac{4610 s^3 + 3.4.527e06 s^2 + 2.008e08 s + 4.433e09}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 3: -----

$$\frac{1439 s^3 + 1.04e05 s^2 + 1.878e06 s + 1.283e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 4: -----
 From input 4 to output...

$$\frac{-3923 s^3 - 1.686e06 s^2 - 5.578e07 s + 2.449e-06}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 1: -----

$$\frac{-1439 s^3 - 1.04e05 s^2 - 1.878e06 s + 1.093e-07}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$
 2: -----

$$s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08$$

$$3: \frac{1439 s^3 + 1.04e05 s^2 + 1.878e06 s - 7.516e-07}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$

$$4: \frac{4610 s^3 + 3.4.527e06 s^2 + 2.008e08 s + 4.433e09}{s^4 + 970.5 s^3 + 2.45e05 s^2 + 1.409e07 s + 3.4.533e08}$$

对上面的式子进行简化,

$$1: \frac{4.61 s^3 + 3.4.527 s^2 + 0.2008 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s}$$

$$2: \frac{1.439 s^3 + 0.104 s^2 + 0.001878 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s}$$

$$3: \frac{-1.439 s^3 - 0.104 s^2 - 0.001878 s + 6.773e-19}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$4: \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s}$$

From input 2 to output...

$$1: \frac{1.439 s^2 + 0.104 s + 0.001878}{s^3 + 0.9705 s^2 + 0.245 s + 0.01409}$$

$$2: \frac{4.61 s^3 + 3.4.527 s^2 + 0.2008 s + 0.004433}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$3: \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s + 2.215e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$4: \frac{-1.439 s^3 - 0.104 s^2 - 0.001878 s + 2.133e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

From input 3 to output...

$$1: \frac{-1.439 s^3 - 0.104 s^2 - 0.001878 s + 3.345e-19}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$2: \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s - 1.043e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$3: \frac{4.61 s^3 + 3.4.527 s^2 + 0.2008 s + 0.004433}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$4: \frac{1.439 s^3 + 0.104 s^2 + 0.001878 s + 9.2e-19}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

From input 4 to output...

$$1: \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s - 2.501e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$2: \frac{-1.439 s^3 - 0.104 s^2 - 0.001878 s - 5.323e-20}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$3: \frac{1.439 s^3 + 0.104 s^2 + 0.001878 s - 1.633e-19}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

$$4: \frac{4.61 s^3 + 3.4.527 s^2 + 0.2008 s + 0.004433}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}$$

对各式子中零次项系数相比其它的项较小，进行忽略。

$$1: \frac{4.61 s^2 + 3.4.527 s + 0.2008}{s^3 + 0.9705 s^2 + 0.245 s + 0.01}$$

$$2: \frac{1.439 s^2 + 0.104 s + 0.001878}{s^3 + 0.9705 s^2 + 0.245 s + 0.01}$$

$$3: \frac{-1.439 s^2 - 0.104 s - 0.001878}{s^3 + 0.9705 s^2 + 0.245 s + 0.01}$$

$$\begin{array}{l}
 \text{4: } \frac{-3.923 s^2 - 1.686 s - 0.05578}{s^3 + 0.9705 s^2 + 0.245 s + 0.01} \\
 \text{From input 2 to output...} \\
 \text{1: } \frac{1.439 s^3 + 0.104 s^2 + 0.001878 s + 1.089e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333} \\
 \text{2: } \frac{4.61 s^3 + 3.4.527 s^2 + 0.2008 s + 0.004433}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333} \\
 \text{3: } \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s + 2.215e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333} \\
 \text{4: } \frac{-1.439 s^3 - 0.104 s^2 - 0.001878 s + 2.133e-18}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s + 0.0002333}
 \end{array}$$

$$\begin{array}{l}
 \text{From input 3 to output...} \\
 \text{1: } \frac{-1.439 s^2 - 0.104 s - 0.001878}{s^3 + 0.9705 s^2 + 0.245 s + 0.01409} \\
 \text{2: } \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s} \\
 \text{3: } \frac{4.61 s^3 + 3.4.527 s^2 + 0.2008 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s} \\
 \text{4: } \frac{1.439 s^3 + 0.104 s^2 + 0.001878 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s}
 \end{array}$$

$$\begin{array}{l}
 \text{From input 4 to output...} \\
 \text{1: } \frac{-3.923 s^3 - 1.686 s^2 - 0.05578 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s} \\
 \text{2: } \frac{-1.439 s^3 - 0.104 s^2 - 0.001878 s}{s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s}
 \end{array}$$

$$s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s$$

$$1.439 s^3 + 0.104 s^2 + 0.001878 s$$

3: -----

$$s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s$$

$$4.61 s^3 + 3.4.527 s^2 + 0.2008 s$$

4: -----

$$s^4 + 0.9705 s^3 + 0.245 s^2 + 0.01409 s$$

通过计算得出传递函数的矩阵中对角元素的，耦合扰动的增益一般小于实际的有用的传递函数。

搭建 simulink 模型

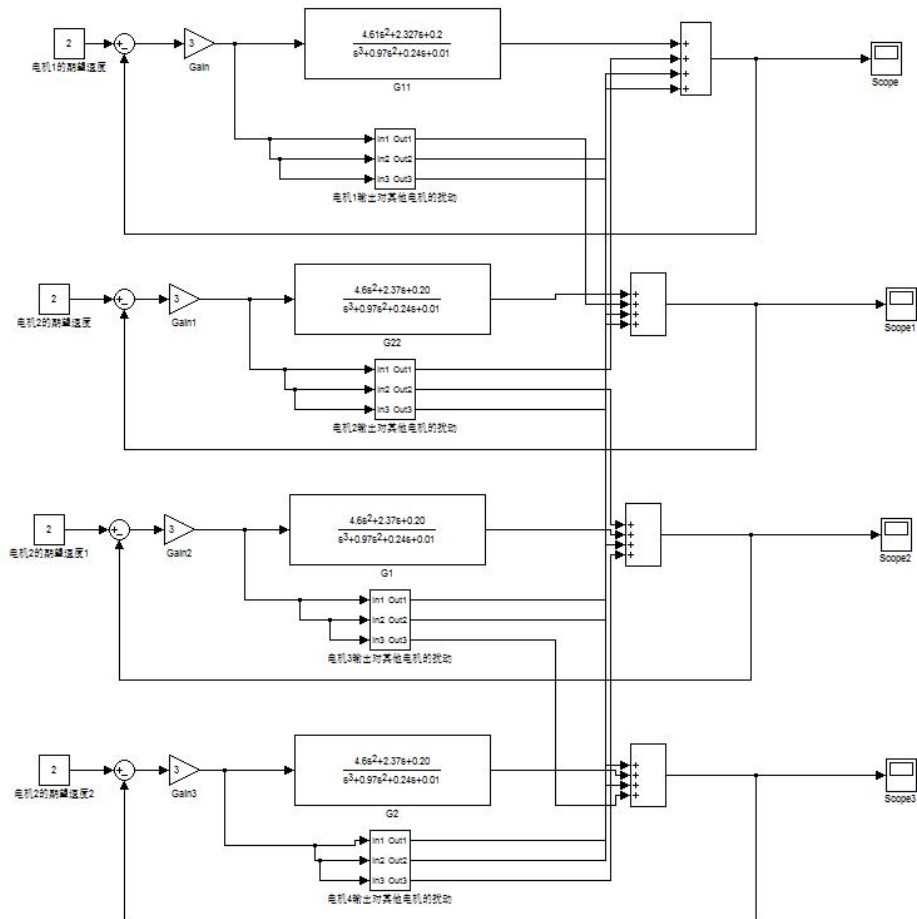


图 2.5 simulink 模型

其仿真结果为

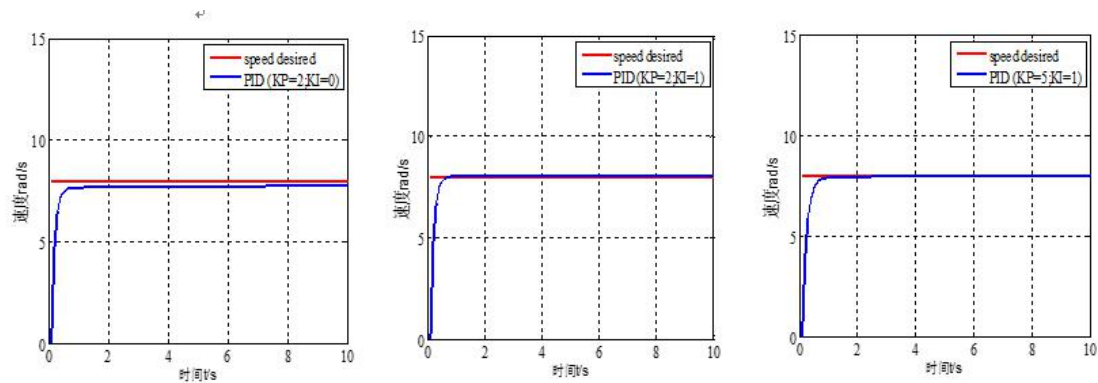


图 2.6(a) 无耦合扰动下 PID

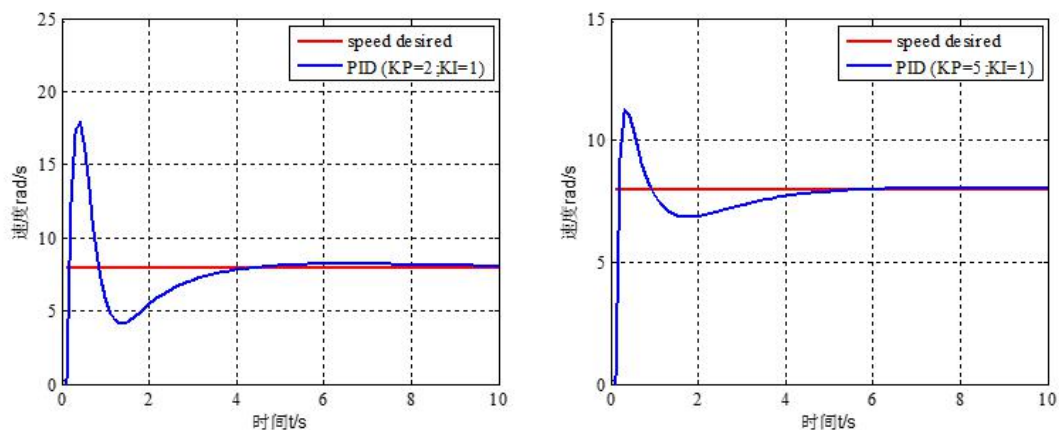


图 2.6(b) 有耦合扰动下 PID

2.3.2 云台控制方案

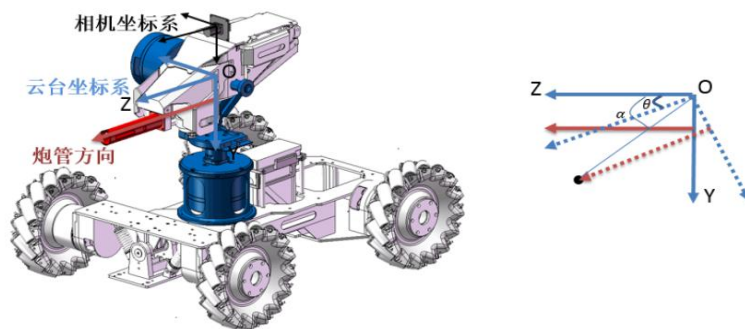


图 2.7 系统模型

系统主要分为两个子系统：视觉反馈子系统和控制子系统

战车的随动云台主要包括方位 **YAW** 轴和俯仰 **PITCH** 轴的两套随动系统，其主要功能是：接收视觉发送的场地上目标的当前位置坐标，即基于转台的方位角和俯仰角，由方位轴和俯仰轴的随动系统进行实时跟踪，使得固联在俯仰轴上的子

弹能够实时瞄准目标，保证子弹顺利发射并击毁目标。

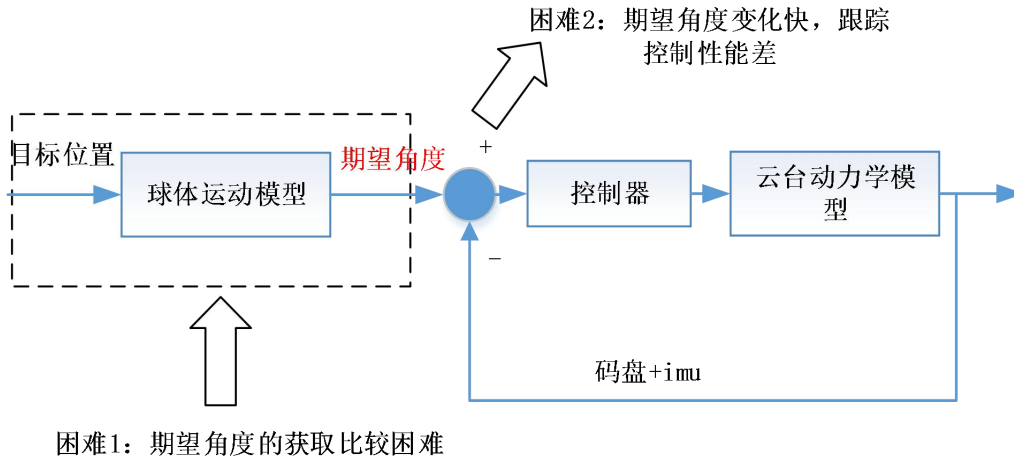


图 2.8 系统框图

其中，期望角度的获取困难的原因包括：

1.步兵战车在锁定目标进行打击的过程中，在近距离的情况下，目标的位置往往存在一个移动距离，当移动距离变化剧烈时，云台的跟踪性能有所降低，从而，战车无法对目标进行精确打击。

2.球体运动模型的目标位置检测在炮管快速移动的时候，容易出现炮管上的摄像头读取数据准确性降低。

2.3.2.1 期望角度解算

经过前期的射弹测试，我们发现子弹飞行轨迹满足重力模型，其出射角度为枪管 pit 角度，以相机 x、y 轴为坐标系， yaw 电机的期望角度增量应为 $\beta = \text{atan2f}(x,d)$ 。在 y 轴方向上，忽略重力作用，则 pit 电机的期望角度增量应为 $\alpha = \text{atan2f}(y,d)$ 。考虑重力模型，则需要将 pit 电机往上增加一个偏移角度以抵消重力影响，该角度与子弹速度、目标的距离等有关。近似情况下，考虑子弹到达目标的时间为 $t = d/v$ ，d 为云台到目标的距离，v 为子弹的初速度（忽略风阻），假设子弹速度为水平方向，则重力加速度带来的垂直向下的位移为 $(gt^2)/2$ ，因此在计算角度时将该距离考虑到 Y 轴的坐标上。真实情况下，由于子弹初速度方向与目标位置和云台当前 Pitch 轴角度有关。通过 Pitch 角度，可将目标在云台位置转换至水平坐标系中，利用下述程序即可算出云台的角度。在底盘运动时，其转动角度与平移量将对图像产生影响，我们可以将其作为前馈量进行目标点预测。其编程实现如下：

```
float x = -ReceData.visionDataX/1000.0f;
float y = -ReceData.visionDataY/1000.0f;
float d = ReceData.visionDataD/1000.0f;
x+=-mecanum.hvx*0.03f-tanf(mecanum.hvw*0.005f)*d;
d+=-mecanum.hvy*0.03f;
```

```

theta = gimbal.pit_angle_fdb*D2R;
alpha = atan2f(y,d);
beta = atan2f(x,d);
yita
atan2f((speed2-sqrtf(speed4-2.f*imu.OneG*(d*tanf(theta+alpha))*speed2-SQR(imu.
OneG)*SQR(d)))/imu.OneG,d) - alpha - theta;
gimbal.pit_angle_ref = gimbal.pit_angle_fdb+((alpha+yita)*R2D+pit_off);
gimbal.yaw_angle_ref = gimbal.yaw_angle_fdb+(beta*R2D+yaw_off);

```

2.3.2.2 模型辨识

云台的各种电机模型可以通过牛顿定律计算得到。大多是采用一个一阶惯性环节来表示电机，输入是单片机内的值，输出为速度云台控制的模型的最内环为电流环，但是电流环由比赛提供的电调完成，一般电流环的频带几百上千 HZ，比速度环的频带要小很多倍。所以可以认为电流环是所带来的零极点不重要。所以可以近似电流环是一个常数。

所以系统的结构和结构框图如下所示

$$\frac{K}{sJ(Ts+1)}$$

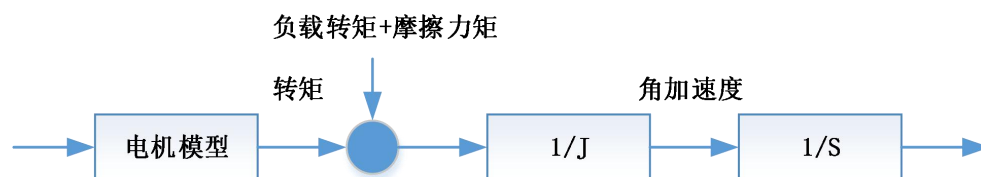


图 2.9(a) YAW 轴的模型

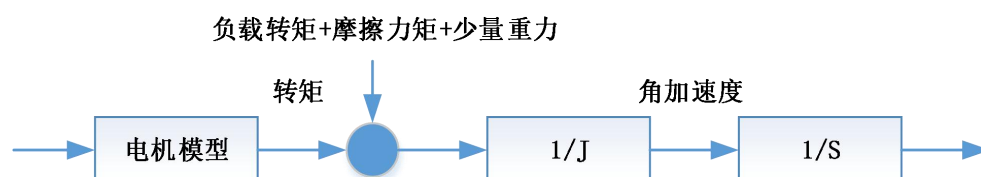


图 2.9(b) PITCH 轴的模型

电机的相关系数可以参考官方的测试资料可知

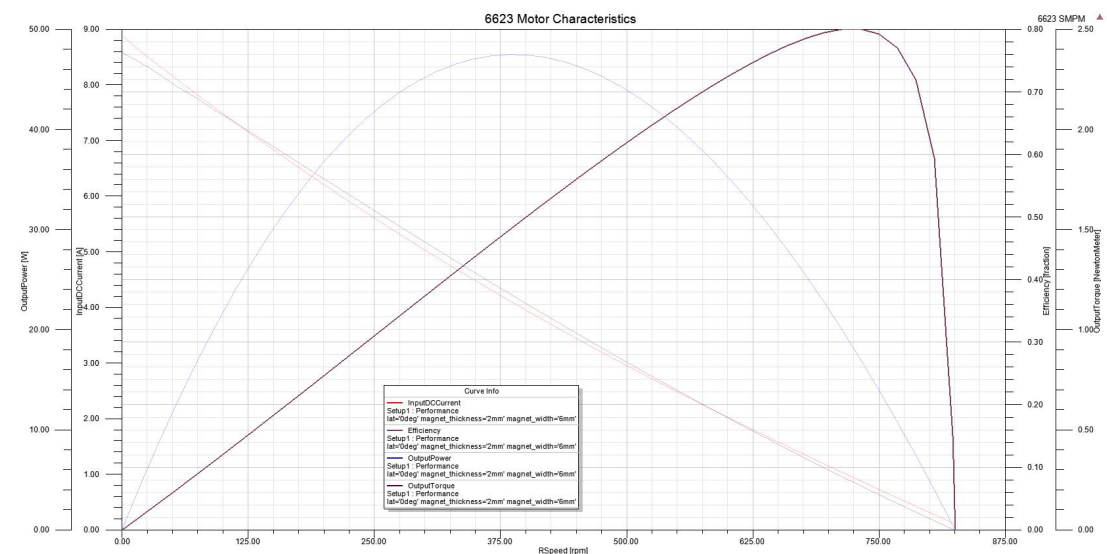


图 2.10 电机参数图表

可得 $K_e: 0.274 \text{ Vs/rad}$, $K_V: 35 \text{ RPM/V}$, $K_t: 0.268 \text{ Nm/A}$ 。

辨识技术可以分为两类，即时域辨识和频域辨识。频域法虽然只能用于线性系统，但在处理动力学系统和非定常线性动力学系统的参数辨识方面有独特优点，因此频域辨识广泛的应用于系统辨识。频域系统辨识是利用飞行器的输入输出数据的频率响应来估计给定参数模型的不确定参数。一般是利用快速傅立叶变换 (FFT) 计算出频率响应，然后通过辨识算法，获得模型参数。

2.3.2.3 扫频实验

根据以上的要求，按照如下的实验方案：采用扫频实验获得数据。即通过 STM32 单片机输入一个逐渐增加输入频率的正弦信号，实验在各个通道 (YAW 轴) 开环下进行。先将云台稳定在悬停状态，然后分别对横滚和俯仰通道输入扫频信号，控制输入和云台状态数据以 10ms 的采样速率被飞控计算机记录下来。通过 wifi 传输链路传送到地面站。如图所示即为一次典型的横向通道扫频控制信号输入图，刚开始时输入频率很低，确保低频带的激励信号质量，逐渐增大输入频率，最终最新回到平衡点。

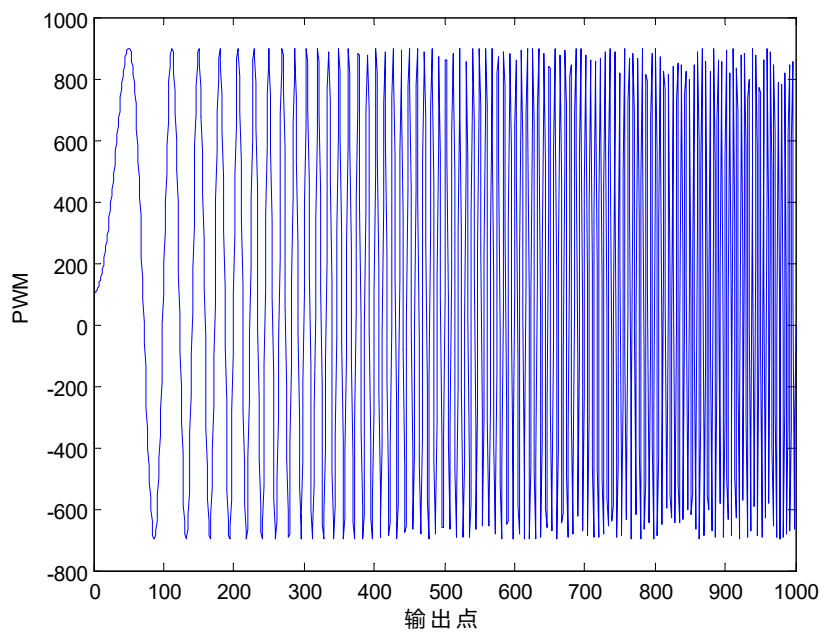


图 2.11 YAW 轴通道舵机输入

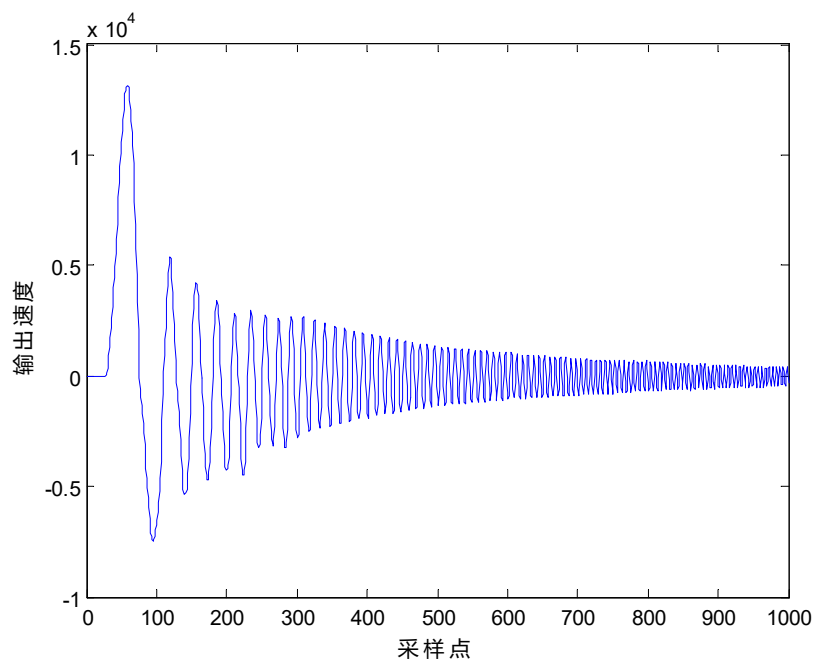


图 2.12 实际输出信号

2.3.2.4 实验结果与分析

对所输出的角速度信号进行 FFT 运算，和 BODE 图得绘制。

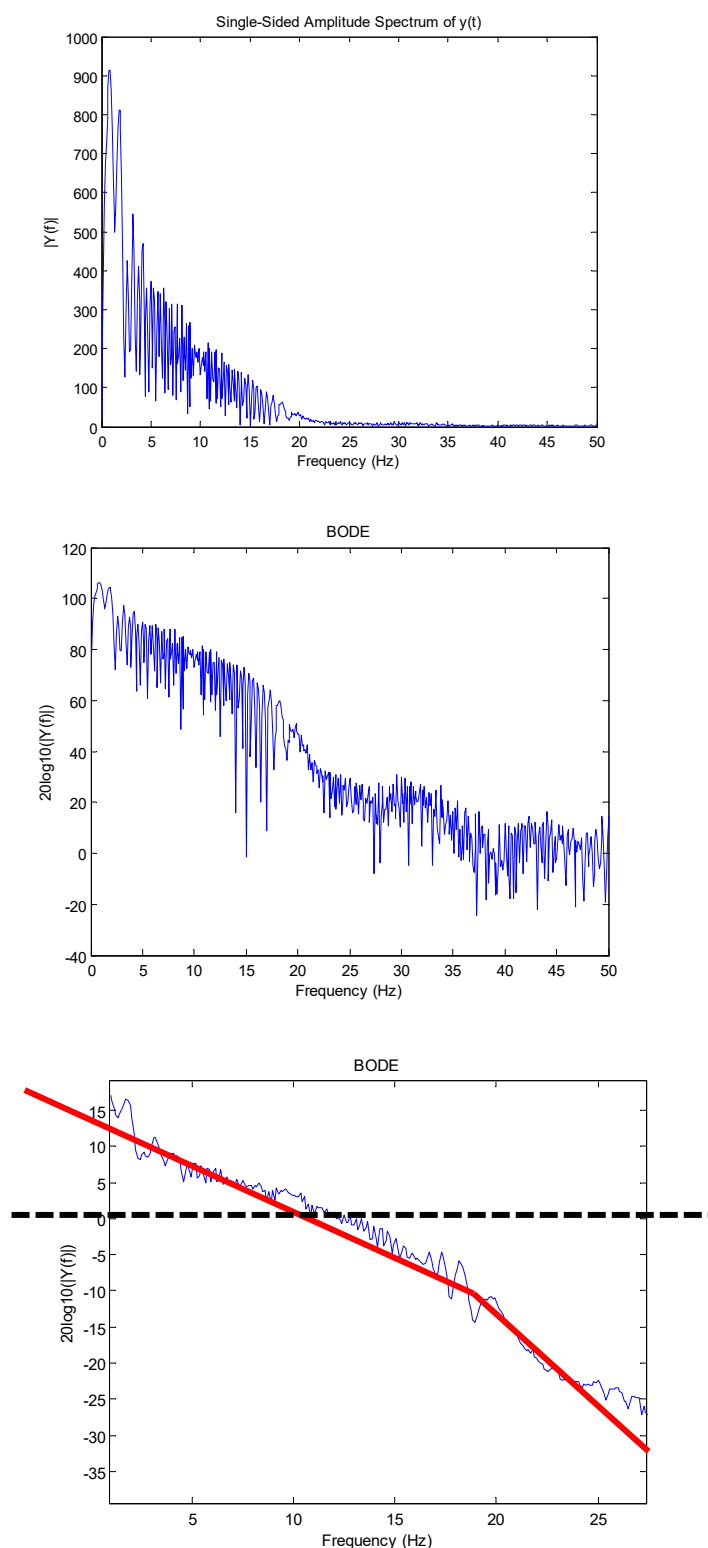


图 2.13 滤波结果与伯德图

通常扫频实验获得的传递函数在低频和高频区域准确性较差。高频激励容易超过云台的正常工作范围。但是我们可以利用频域辨识获得的传递函数来确定系统的关键参数。从而利用参数化模型确定适应整个频段的模型。由图可知 18HZ 左右是系统的一个转折频率，系统的截至频率在 12HZ 左右。所以 YAW 轴的开环

传递函数可以近似为

$$\frac{15/600}{s(\frac{s}{18} + 1)}$$

（备注：理论上需要对所输出得伯德图得数据进行拟合得出拟合曲线，并计算拟合曲线和实际数据得相关性分析，但是从图中可知其趋势比较明显，就省略这个步骤了）

2.3.2.5 串级 pid 控制

利用上一节得到的云台模型，采用多回路串级 PID 控制方法设计云台控制系统。如下图所示。

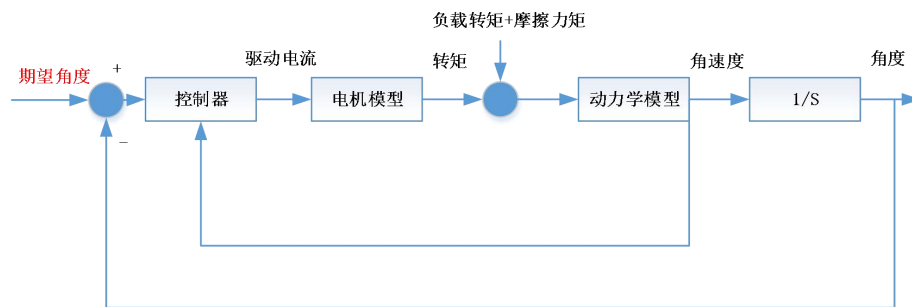


图 2.14 系统框图

采用码盘和 IMU 构成云台的角和角速度得反馈。

多闭环控制系统的设计通常采用由内而外设计的方法，即先设计内环，然后以内环为控制对象设计中环，再以中环为控制对象接着设计外环。云台控制系统的设计也是如此，其中内部的电流环官方封装，所以直接以电流环为控制对象设计速度环，再以双闭环的调速系统为控制对象来设计位置环。

根据上述辨识的模型为一个二阶系统，所以采用 PID 或 PD 算法可以做到对系统的零极点的任意配置，

```
pid_calc(&pid_yaw, gimbal_para.yaw_angle_fdb, gimbal_para.yaw_angle_ref);//  
速度环 PID
```

```
pid_calc(&pid_yaw_speed, -imu.gz / IMU_GYR_FCT / 10.0f, pid_yaw.pos_out /  
10.0f);//位置环得 PID
```

针对 pitch 轴存在一点重力的非线性，其消除的方案是在 pitch 轴上用一根橡皮筋对其进行平衡。使云台在静止时候处于平衡点的位置。

橡皮筋



图 2.15 平衡解决方案

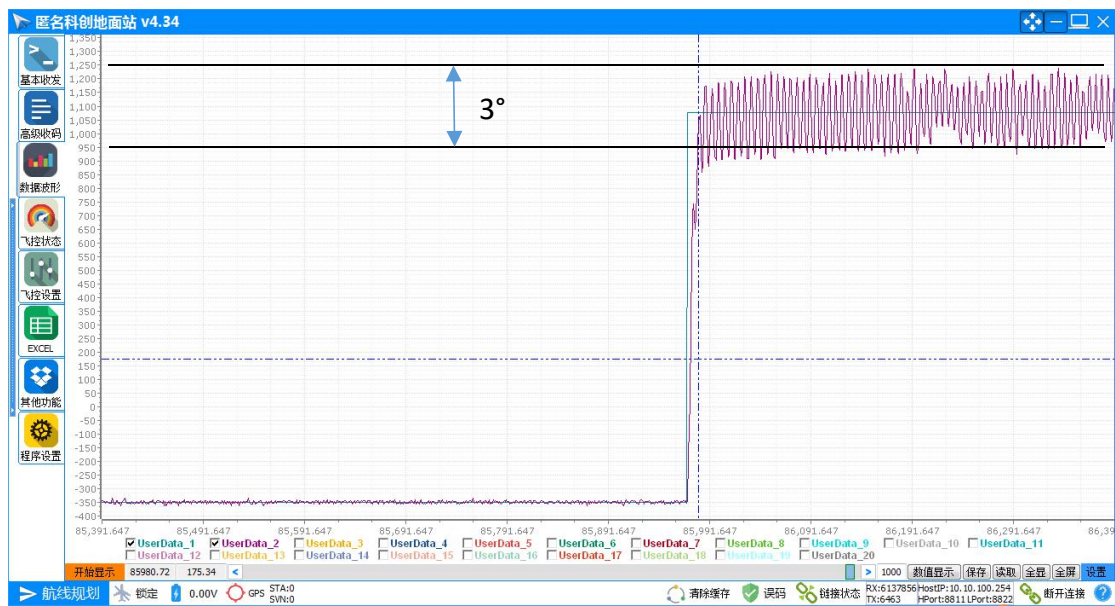


图 2.16(a) 未射弹过程中的响应曲线

内环: $K_P=40$, $K_I=0.3$, $K_D=10$

外环: $K_P=200$

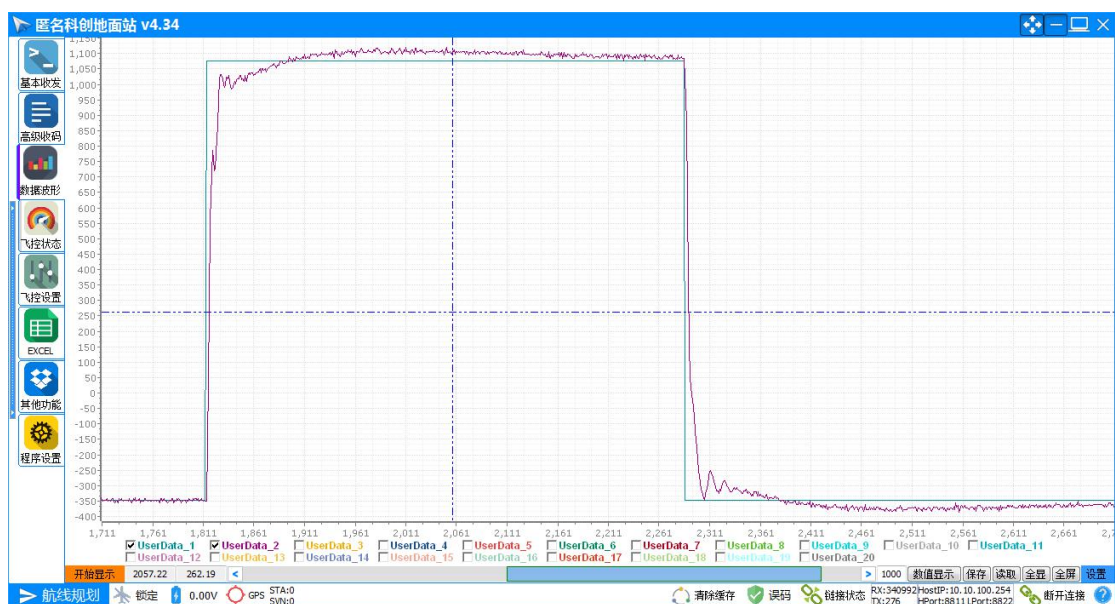


图 2.16(b) 未射弹过程中的响应曲线

内环: $K_P=20$, $K_I=0.3$, $K_D=10$

外环: $K_P=200$, $K_I=0.5$

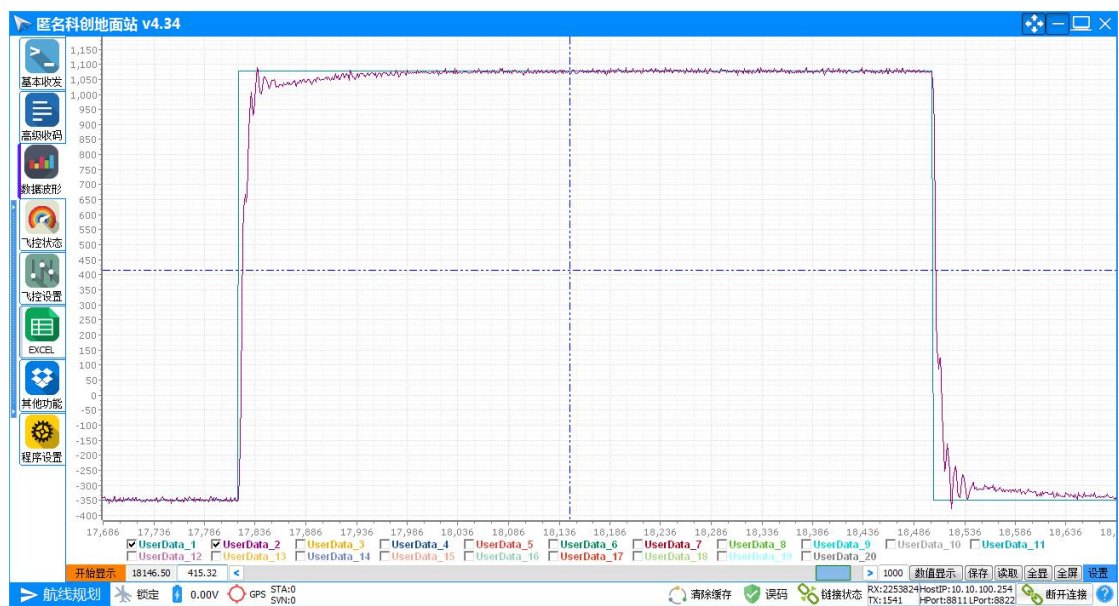


图 2.16(c) 未射弹过程中的响应曲线

内环:

$K_P=25$, $K_I=0.3$, $K_D=10$

外环:

$K_P=200$,

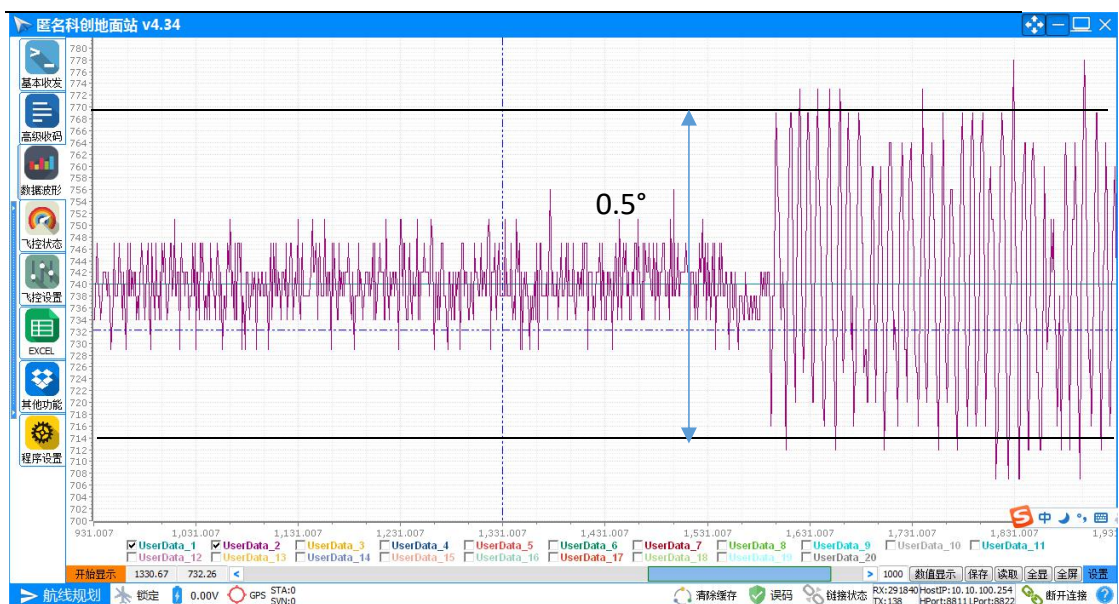


图 2.17(a) 射弹过程中电机响应曲线

相应的 PID 参数，内环 PID 外环 PI 未变

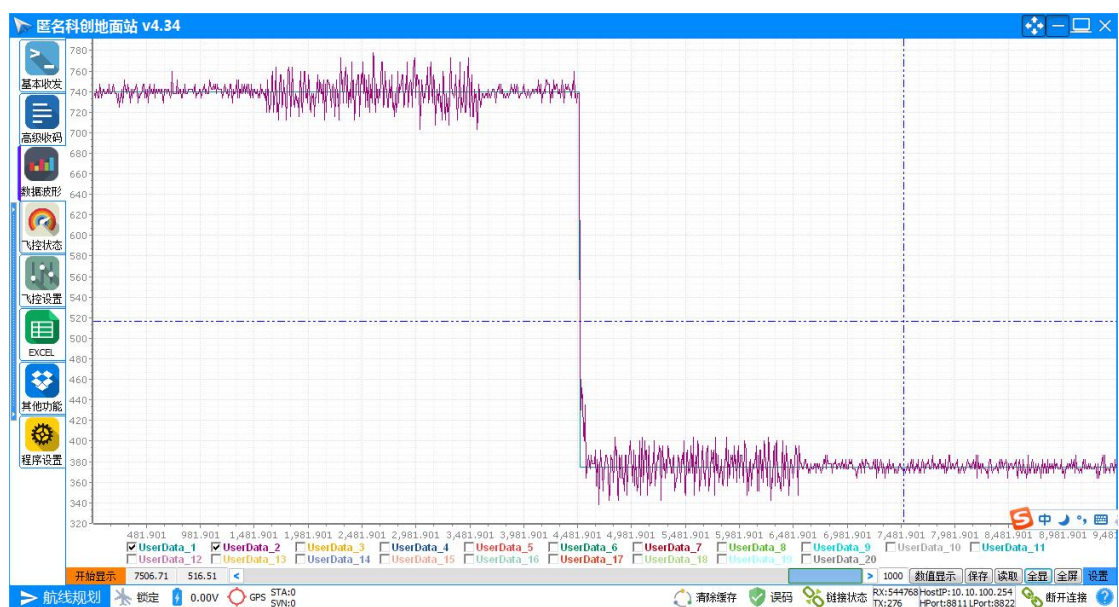


图 2.17(b) 射弹过程中电机响应曲线

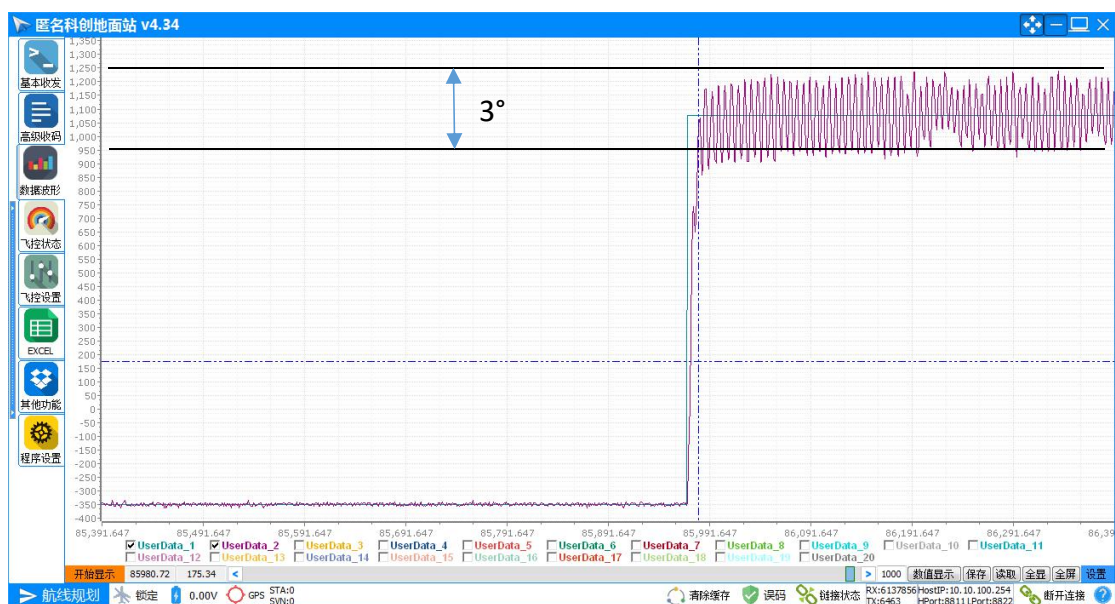


图 2.18(a) 未射弹过程中的响应曲线

内环: $K_P=40$, $K_I=0.3$, $K_D=10$

外环: $K_P=200$

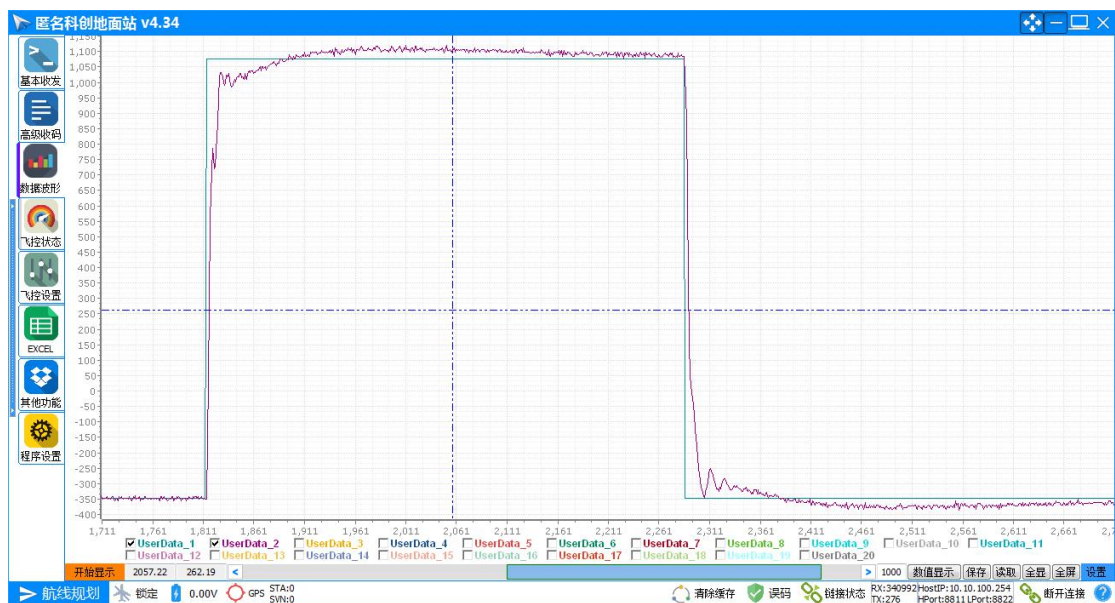


图 2.18(b) 未射弹过程中的响应曲线

内环: $K_P=20$, $K_I=0.3$, $K_D=10$

外环: $K_P=200$, $K_I=0.5$

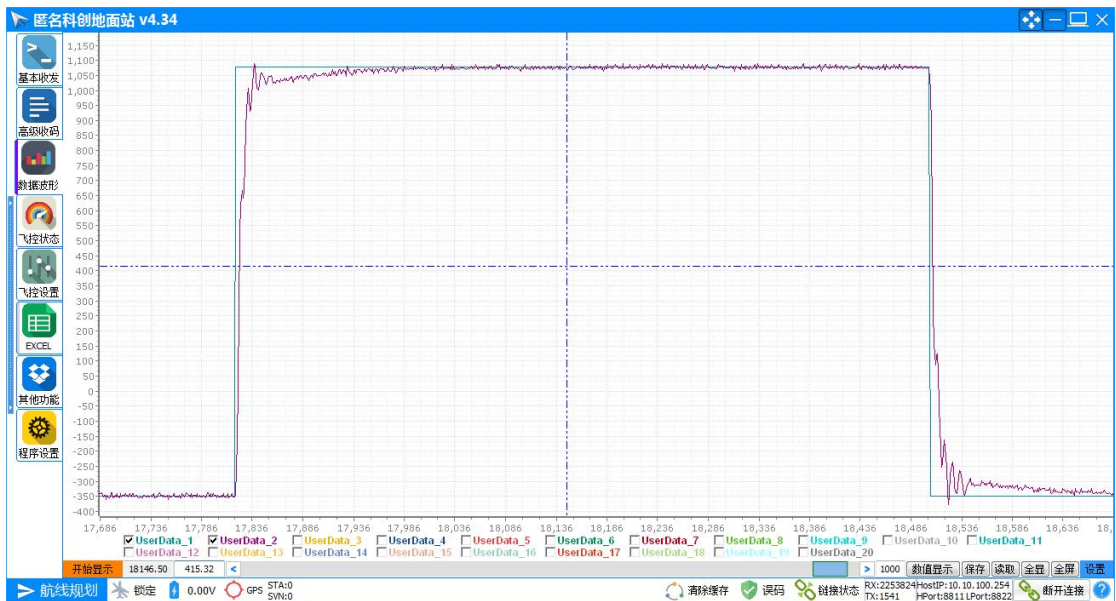


图 2.18(c) 未射弹过程中的响应曲线

内环：

$K_P=25$, $K_I=0.3$, $K_D=10$

外环：

$K_p=200$,

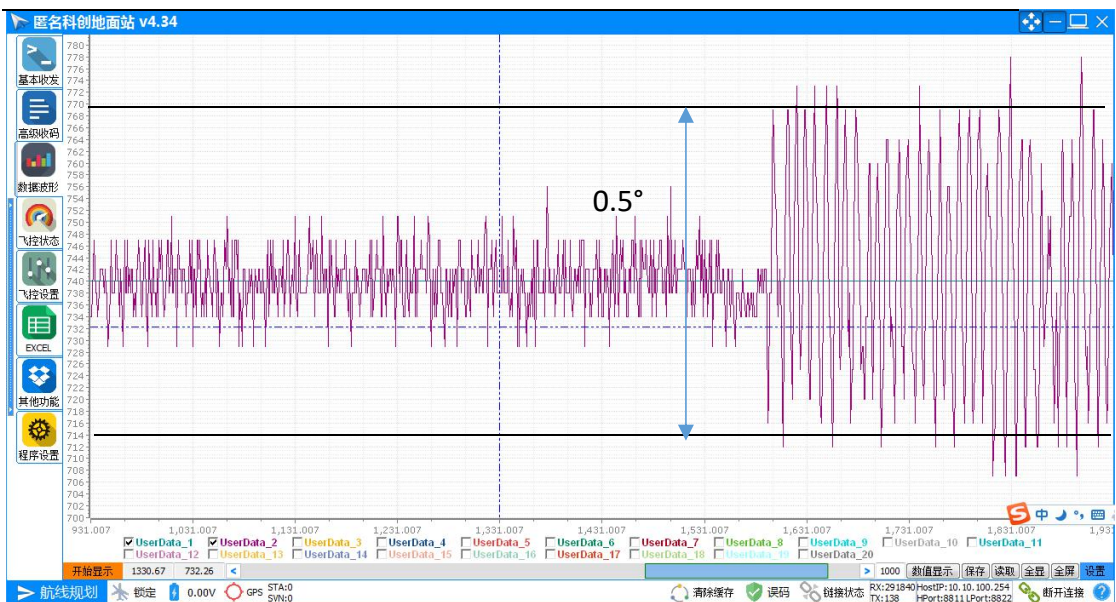


图 2.19(a) 射弹过程中电机响应曲线

相应的 PID 参数，内环 PID 外环 PI 未变

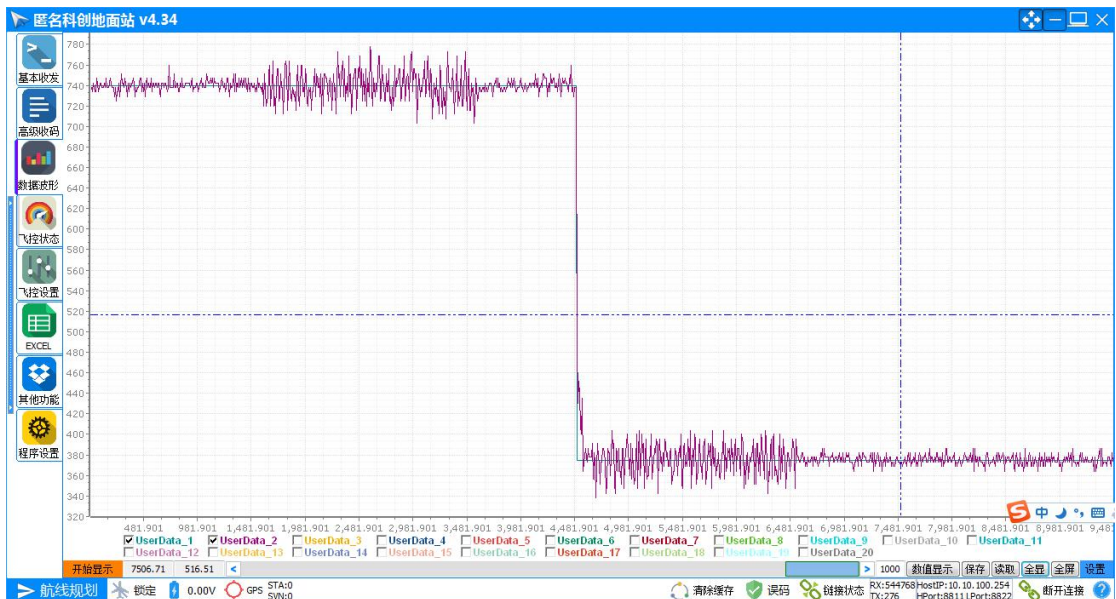


图 2.19(b) 射弹过程中电机响应曲线

2.4 串口高效读取

串口的准确读取与高效性非常重要，在系统中，我们利用串口 1 进行遥控器数据读取，串口 2 与 PC 上位机进行数据交互，串口 3 读取裁判系统数据，串口 6 与 TXone 进行数据交互。在不使用 DMA 的情况下，对 CPU 的负荷是非常重的。官方代码使用 HAL 库进行串口数据读取，采用了 HAL 库自带的串口中断函数，但 HAL 库的串口中断服务函数进行很多冗余处理，大大增大了系统的运行开销，我们重写 HAL 库中断函数以达到高效运行的目的。

在所有串口通讯中，我们均采用 DMA 接收或发送的模式，在接收数据时，我们启用串口空闲中断进行接收。这点与官方程序类似，但我们采取重写中断服务函数来进行高效传输。代码如下。

```
void USART6_IRQHandler(void)
{
    __HAL_UART_CLEAR_IDLEFLAG(&huart6);
    __HAL_DMA_DISABLE(huart6.hdmarx);
    __HAL_DMA_CLEAR_FLAG(huart6.hdmarx,
        __HAL_DMA_GET_TC_FLAG_INDEX(huart6.hdmarx));
    memcpy(&ReceData, uart6_rx, sizeof(tReceTXoneData));
    ros_update = true;
    __HAL_DMA_SET_COUNTER(huart6.hdmarx, UART6_MAX_LEN);
    __HAL_DMA_ENABLE(huart6.hdmarx);
}
```

在裁判系统数据的读取上，由于裁判系统发送比赛信息为循环式模式，而射击信息与打击信息则为注入式模式，利用空闲中断可能读出两帧不同类型的信息。我们在官方原有基础加入对帧头的重新识别，再复原。

```
void USART3_IRQHandler(void)
{
    if((__HAL_UART_GET_FLAG(&huart3, UART_FLAG_IDLE) != RESET))
    {
        uart3_time = HAL_GetTick() - uart3_pretime;
        uart3_pretime = HAL_GetTick();
        u8 len = UART3_MAX_LEN - __HAL_DMA_GET_COUNTER(huart3.hdmarx);
        for(u8 i = 0; i < 100 ; i++)
        {
            if(uart3_rx[i] == SOF_FIXED)
            {
                u8 *realdata = &uart3_rx[i];
                tFrameHeader* data = (tFrameHeader*)realdata;
                if((1 == Verify_CRC8_Check_Sum(&uart3_rx[i], 5)) && (1 ==
                Verify_CRC16_Check_Sum(&uart3_rx[i], 9 + data->dataLenth)))
                {
                    switch((realdata[5] | realdata[6] << 8))
                    {
                        case GameInfoId:
                            memcpy(&GameInfo, realdata + 7, data->dataLenth);
                            uwb_update = true;
                            break;
                        case RealBloodChangedDataId:
                            memcpy(&RealBloodChangedData, realdata + 7,
                            data->dataLenth);
                            blood_update = true;
                            break;
                        case RealShootDataId:
                            memcpy(&RealShootData, realdata + 7,
                            data->dataLenth);
                            shoot_update = true;
                            break;
                    }
                }
            }
        }
        __HAL_UART_CLEAR_IDLEFLAG(&huart3);
        __HAL_DMA_DISABLE(huart3.hdmarx);
        __HAL_DMA_CLEAR_FLAG(huart3.hdmarx,
        __HAL_DMA_GET_TC_FLAG_INDEX(huart3.hdmarx));
    }
}
```

```
    __HAL_DMA_SET_COUNTER(huart3.hdmarx, UART3_MAX_LEN);  
    __HAL_DMA_ENABLE(huart3.hdmarx);  
}  
}
```

2.5 难点与不足

在实验中，我们经常进行射击测试，卡弹是非常严重的问题，曾困扰了我们好久，因为每次的卡弹都要进行必要的机械拆卸，这是非常枯燥的过程。通过仔细观察思索，我们采用对拨弹电机进行角度控制，在判别到堵转情况下进行反转退弹，我们只要保证拨弹口上预留有反转空间即可，故使用薄碳板遮盖出这个空间来，从软件上暂时解决卡弹问题。在 pit 电机控制上，我们前期采用了重力进行前馈补偿，但云台负载在这些阶段不断有变化，后来尝试使用橡皮经进行重力初步补偿，也基本能达到消除重力扰动的效果。

不足之处，没有进行底盘的速度优化。

3. 算法部分

3.1 开发环境介绍

3.1.1 硬件环境

在这次的自动化战车射击比赛中，战车的定位导航用到的计算设备是 Jetson TX1，传感器是 RPLIDAR A2 激光雷达。

系统的坐标系有四个，分别是地图全局坐标系、里程计坐标系、车身坐标系和激光雷达坐标系。地图全局坐标系是用于定位战车在地图上的坐标和发送目标位置，里程计坐标系是用于定位战车的里程计坐标，激光雷达的安装位置基本与战车中心重合，因而车身坐标系和激光雷达坐标系基本重合。

硬件框架是激光雷达获取环境数据，发送到 Jetson TX1 进行计算分析，进而让战车感知周围环境，从而实现定位和导航功能。

3.1.2 算法环境

虽然这次官方提供搭建好环境的 TX1 镜像，但是为了加速项目开发，我们在台式机和笔记本上都搭建跟 TX1 一样的环境进行仿真和测试，具体搭建环境如下：

- 1、首先搭建 ROS 环境，可以参照 ROS 官网的安装教程。我们安装的版本是 Kinetic，平台是 Ubuntu。值得注意的是最好将镜像源改为国内的，这样下载安装速度会快很多，而且最好安装 Desktop-Full，免去后面需要安装很多的依赖包。即使这样，后面在项目开发的过程中也要安装挺多的依赖包，这里介绍一个小技巧，依赖包可以通过命令 `sudo apt-get install ros-kinetic-` 来进行安装，假设我们要安装 navigation 的其他一些依赖包，可以在 `sudo apt-get install ros-kinetic-` 后面输入 navigation，然后敲两下 Tab 键，这时 navigation 的其它可以安装的依赖就会显示出来，这时直接安装就好了。就算忘了 navigation 如何拼写，在输入 navi 后敲一下 Tab 键，navigation 就会被补全，因此我们要学会善用 Tab 键，这样不但减轻了记忆量方便输入命令，而且补全的命令会更准

确，对于新手来说是很好的。附上 ROS 官网的安装教程：
<http://wiki.ros.org/ROS/Installation>

3.2 整体技术方案概述

3.2.1 技术原理介绍

定位：amcl 是移动机器人二维环境下的概率定位系统。它实现了自适应（或 kld 采样）的蒙特卡罗定位方法，其中针对已有的地图使用粒子滤波器跟踪一个机器人的姿态。

导航：move_base 提供 ROS 导航的配置，运行，交互接口，主要包括两个部分：

- (1) 全局路径规划：根据给定的目标位置进行总体路径的规划
- (2) 局部路径规划：根据附近的障碍物信息进行躲避路线规划

3.3 算法整体框架设计

这次比赛中战车的定位导航具体要实现的功能是战车得知自己在地图上的位置，获取到达目标的位置和路线，感知导航过程中的障碍物并且懂得躲避。因此算法的整体设计架构是：

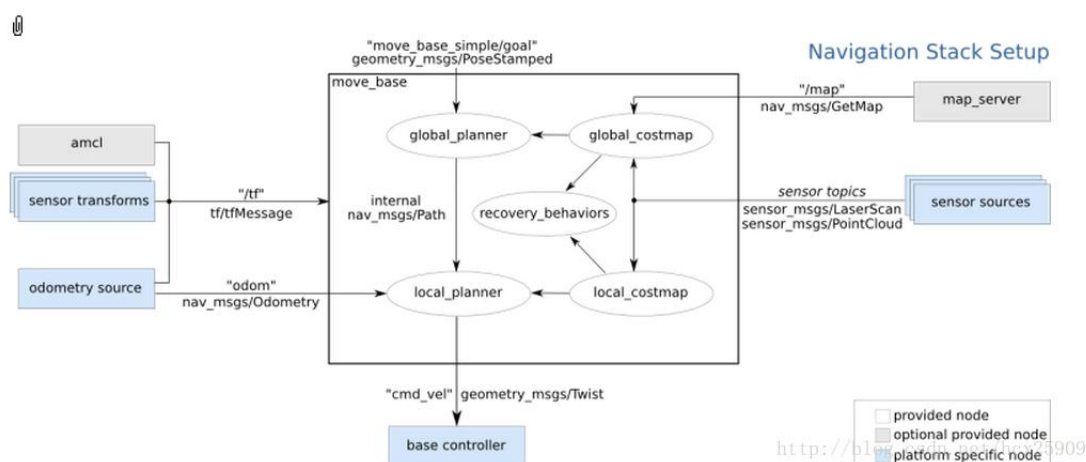


图 3.1 算法整体架构

在总体框架图中可以看到，`move_base` 提供了 ROS 导航的配置、运行、交互接口，它主要包括两个部分：

(1) 全局路径规划 (global planner): 根据给定的目标位置进行总体路径的规划;

(2) 本地实时规划 (local planner): 根据附近的障碍物进行躲避路线规划。

自主定位

全局定位: 通过测机器人的绝对未知来定位, 定位的精度较高, 并且可以用来修复局部定位的定位误差

局部定位: 通过测量相对于机器人初始位置的距离和方向来确定当前的位姿, 但随着时间的累计造成定位的误差较大, 无法精确定位

路径导航

1、在 ROS 的导航中, 首先会通过全局路径规划, 计算出机器人到目标位置的全局路线, 这一功能是 navfn 这个包实现的。

navfn 通过 Dijkstra 最优路径的算法, 计算 costmap 上的最小花费路径, 作为机器人的全局路线。

2、本地的实时规划是利用 base_local_planner 包实现的, 该包使用 Trajectory Rollout 和 Dynamic Window Approaches 算法计算每个周期内应该行驶的速度和角度(dx, dy, dtheta velocities)。

base_local_planner 这个包通过地图数据, 通过算法搜索到达目标的多条路径, 利用一些评价标准(是否会撞到障碍物, 所需要的时间等等), 选取最优的路径, 并且计算所需要的实时速度和角度。

Trajectory Rollout 和 Dynamic Window Approaches 算法的主要思路如下:

- (1) 采样机器人当前的状态(dx, dy, dtheta)
- (2) 针对每个采样速度, 计算机器人以该速度行驶一段时间后的状态, 得出一条行驶路线
- (3) 利用一些评价标准为多条路线打分
- (4) 根据打分, 选择最优路线
- (5) 重复上面的过程

3.4 算法功能模块说明

算法功能尽可能的模块化, 使得整个系统做到高内聚, 低耦合, 提高系统的稳定性以及可维护性。这一章尽可能的有详细的算法流程图说明。

3.4.1 定位算法

定位算法是 AMCL（自适应蒙特卡罗定位）。

主要存在的问题是战车需要从定位的始点走一段路程才能有比较好的定位，定位点才会收敛。

3.4.1 导航算法

在 ROS 的导航中，首先会通过全局路径规划，计算出机器人到目标位置的全局路线，这一功能是 navfn 这个包实现的。navfn 通过 Dijkstra 最优路径的算法，计算 costmap 上的最小花费路径，作为机器人的全局路线。

该算法的优点是对于地图上可以到达的目的点能够规划出一条比较短的路线，而缺点是由于周围有比较多障碍物的情况，通常不能规划出路线而死掉。

3.4.2 跟踪射击

通过摄像头捕捉敌方战车装甲板位置实现跟踪射击。具体是先定位装甲板两侧灯条，再通过装甲板中间的数字来进行捕捉装甲板的。当得到装甲板在图像中的位置后再通过 PNP 算法求解得到实际空间距离，将信息传递给底层控制云台转动实现跟踪射击。

在实际自动化对战过程中发现制约跟踪射击准确率的是图像识别的准确率，加上动态预测效果反而没有那么理想。因为战车自动化移动还是较为缓慢的，因此提高跟踪射击的准确率在于提高装甲板识别的准确率和稳定性。

3.4.3 单兵逻辑

单兵作战的逻辑是比赛开始的时候战车先在启动区等待飞机补弹，在这过程中摄像头一直进行扫描，一旦发现敌方就会进行攻击。在补弹时间过后战车就会出动巡逻，当发现敌方后就交由底层控制云台和底盘进行跟踪射击。值得一提的是处于射击模式时前后装甲板被射击后底盘会左右移动，左右装甲板被射击后会前后移动，而处于巡逻模式的时候左装甲板被射击底盘会左转，右装甲板被射击底盘会右转，后装甲板被射击底盘会往后转。这样做的目的是处于射击模式时

候能够躲避敌方子弹，处于巡逻模式的时候能让战车在被攻击的时候及时发现敌方进行回击。当子弹不足的时候就会撤退躲避，或是回己方启动区进行补弹。

3.4.4 多兵作战

开始时打算通过 ROS 的多机通信机制进行通信，后期发现实际场地无线通信质量不是很好，有时会断线，而且延时较大，有几百毫秒延时。考虑实际作战的稳定可靠性，也就没让战车之间通信而是让操作手获取战车的信息再做决策判断。最后多兵作战是在单兵作战的基础上，在比赛开始时每辆战车从飞机获取一次补弹，然后每辆战车走不同的路线从前、侧、后攻击敌方战车，操作员再根据战车血量和剩余弹量决定继续攻击还是撤退。

3.4.5 控制

路线规划其实是通过发送目的点的位置然后再通过 ROS 的导航包生成路线，再发送对应的前后左右速度和旋转角速度到底层进行控制。

3.5 测试结果

由于实际场地会有很多干扰，因此前期我会在 stage 下进行仿真测试，当数据可行后我再去实际场地进行测试，而仿真出来的结果基本上是没问题的，只需要实地测试再确认一下。

一开始我会先让战车把需要到达的点给跑对，然后再去考虑跑动过程中战车的朝向问题，尽可能利用地形的优势让枪口一直处于有效扫描射击的状态。因此实际的测试主要是目的点的标定，包括方向。

3.6 可优化方案

现有系统对于加血点没有很好解决，其实较为简便高效的方法是让底层判断有没有加到血，一旦加血成功就马上停下来进行加血；

攻击敌方的时候让多辆战车同时射击同一辆战车，这时可以通过 ROS 的多机通信机制让战车之间进行通信和协作。

4 夏令营感想、总结

40 天的夏令营快要接近尾声了，回想走过的这一段路，的确有不少的收获。首先认识了那么多有激情和能力的小伙伴是很高兴的，然后有一件“小事”是我最为难忘的。

我问硕哥在去到加血点的时候能不能通按键让战车走到加血卡上进行加血，因为定位导航是很难精确到达某一个点的。这时硕哥很不高兴地跟我说这样实在太 low 了，做下去也没多大意思了。那时我正在场地上进行调试，距离联盟赛也还只剩一天了，想的是怎么通过一些小窍门完成目标，而没有考虑怎么实实在在解决问题。现在回想起来这段时间自己基本没做什么开拓性的工作，只是在前人工作基础上做些修修补补，实实在在学到的东西很少很少。说真的在这方面我是挺后悔的，太注重比赛结果而忽视了技术本身，我想硕哥想说的是能力的真正提升才是最大的收获，最后的名次和奖品只存在于瞬间，在我们人生道路上并没有留下多大的痕迹。

可能就这样结束了吧，对大疆有了更深的认识。忘不了两点多骑车回宿舍，也忘不了队友们共同度过的这一段难忘时光。



RoboMaster 大赛组委会

邮箱：robomaster@dji.com

官方论坛：<http://bbs.robomaster.com>

官方网站：<http://www.robomasters.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

ROBOMASTER™ 是大疆创新的商标。

Copyright © 2017 大疆创新 版权所有