RoboMaster 冬令营技术报告（对抗赛）

1、项目规划

1.1 项目规划

目标是搭建一台能够实现基本功能及抓取障碍快的步兵机器人，同时在基本框架如代码和结构上进行优化以其达到更加稳定的操作。

工作安排为三位负责机械，两位负责嵌入式，两位负责算法。

时间计划为 2.5 号完成底板设计，2.6 号完成云台设计并开始测试底盘，2.7 号完成弹仓设计和代码编译，2.8 号调试，2.9 号比赛。

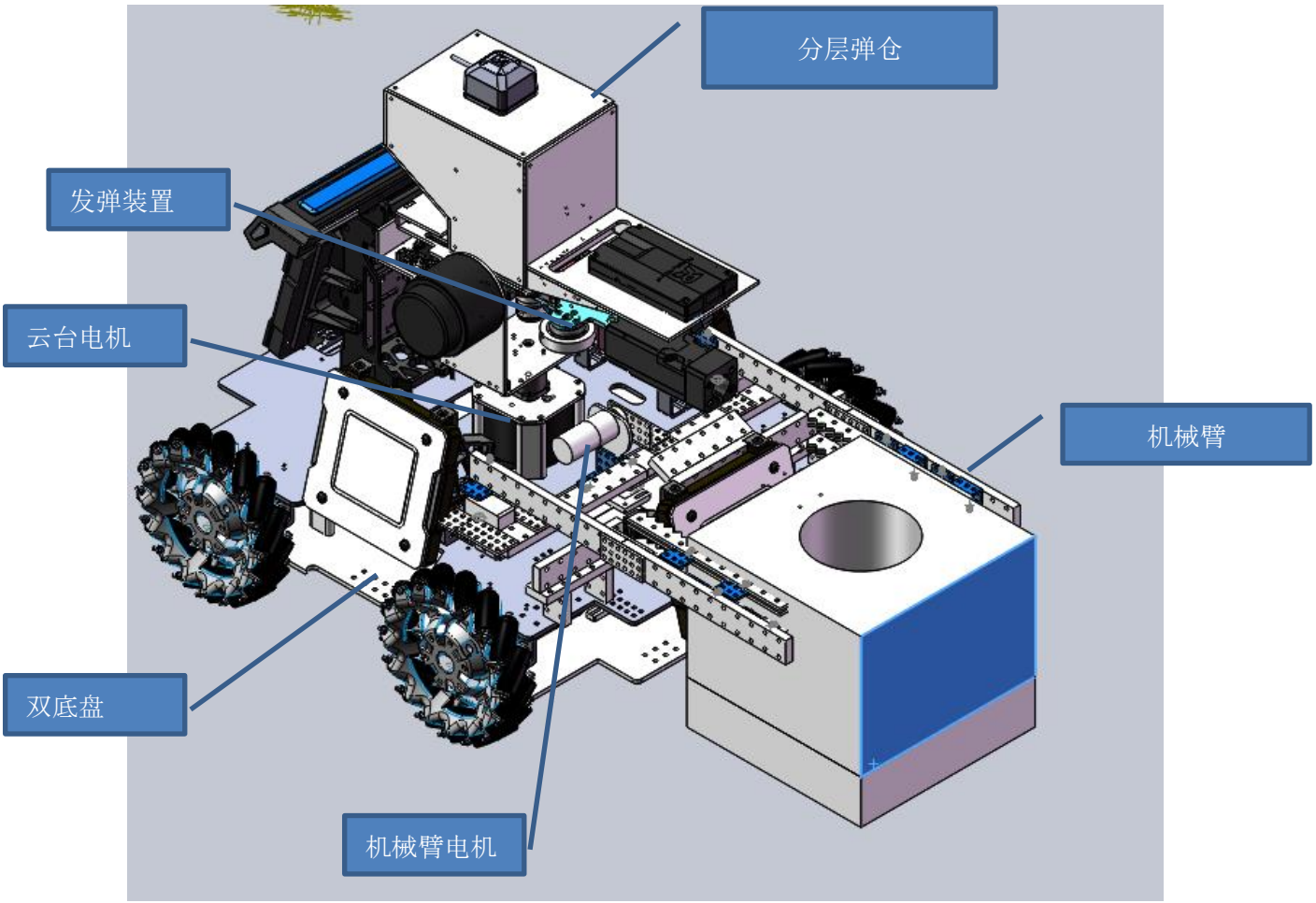
风险点在于任务完成时间很紧，会缺少校对核查的时间；因此本组明确设计思路为先使用建模软件安装确保无误后再使用材料搭建。在底板与车身切割的同时用 makeblock 的零件搭出底盘测试车，加快嵌入式调试的进程。

1.2 主要设计思路

底盘方面初始设计思路为带悬挂的分解底盘，分为底盘和前后轮两部分，但是考虑带悬挂的底盘会需要更长设计和检验时间，并且工作室缺乏悬挂固定器，本组决定采用整体式双层底盘。

2、机械设计

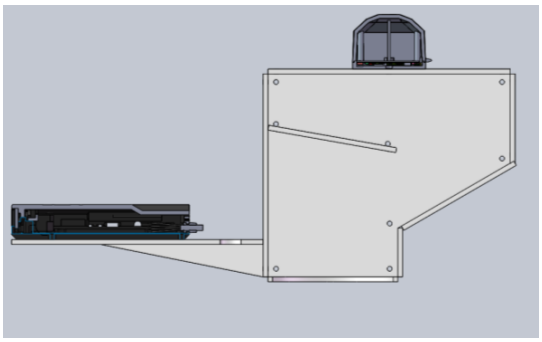
2.1 项目模型



2.2 各模块设计思路

底盘 :底盘是本次比赛的基础, 为了得到稳定迅速的机器人本组采用官方配置的 M3508 减速电机为动力驱动麦克纳姆轮, 考虑到机械臂结构和云台需要支撑, 在上底盘根据目标孔预留一定量的孔位 ; 同时使用建模软件将底盘和以底盘为支撑的结构安装好确认模型无误后用碳板把底盘加工成型。

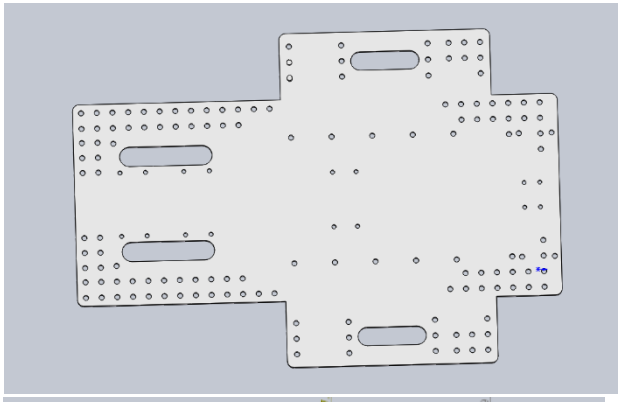
云台：最终使用官方提供的装置，但因为一开始要做下供弹，所以自己设计了云台，但后来得知不能用下供弹的系统，怕时间不够，供弹:首先，为了能装更多的弹，在比赛中有更大的优势，我们设计了下供弹的供弹方式，使用滚珠轴承，将供弹管套分别套在轴承内部和外部，是供弹管能够随着云台的转动而转动，但是因为材料(供弹转角件数量不足)和时间限制的原因，我们将供弹方式改为上供弹，同样为了增加容弹量，以及防止因为弹丸太多，压力太大而卡弹，我们决定将弹仓内部做成分层的结构。

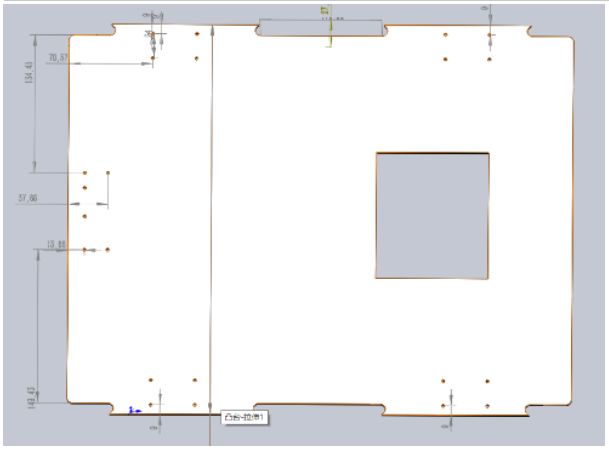


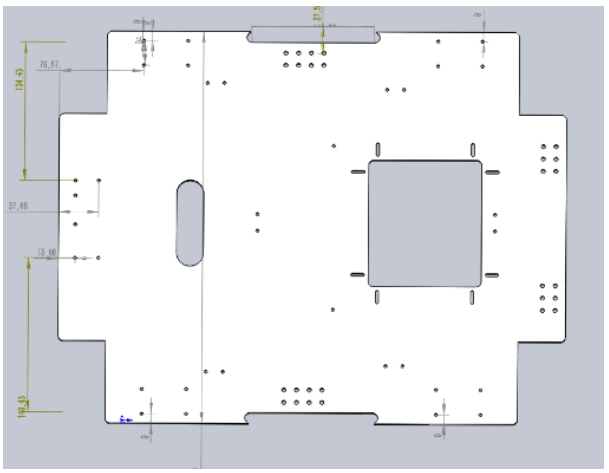
嵌入式：主控板和代码库方面我们决定使用 RM 官方提供的主控板和官方给的封装代码。首先是考虑到 RM 主控板适合步兵这种以无刷电机为主要动力的机器人，主控板上丰富的 CAN 总线接口和 PWM 输出极大的方便了同时驱动许多电机（虽然后来发现有分线板，但是这一点也很重要）。其次 8 路 UART 接口也方便与其它模块通信。至于 RM 的官方代码的封装已经很成熟，上手也非常快，针对步兵机器人设计，省去了自己封装代码的步骤，可以加快进度。

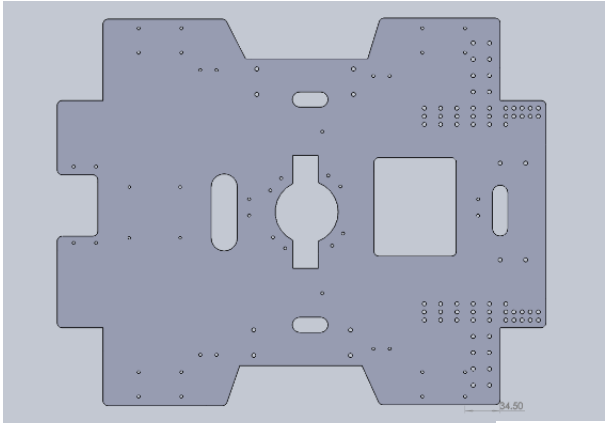
电调方面我们选择使用 CAN 总线驱动，放弃了 PWM，一方面是官方代码中就使用 CAN 总线驱动，其次是如果使用 PWM 就难以获取电调回传的电机的转速等数据。

2.3 优化和改动









最初的设计，这是主体部分的底盘，结构较为紧凑并且留给云台的地方不太够，由于悬挂方案难度较大而且云台和供弹设计没有协商好无法添加固定孔位而放弃

下底盘

1.1 设计好电机座电池架和 RFID 位置

1.2 添加支撑柱和预留孔位

1.3 修正边角

上底盘

1.1 将下底盘支撑点的孔位设计好

1.2 将上下底盘对齐矫正误差

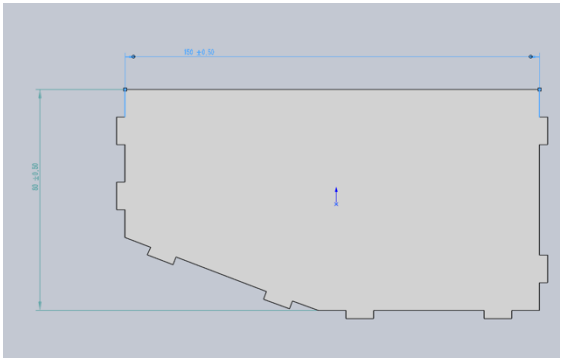
1.3 修正边角并添加云台和机械臂预留孔

弹仓：

第一代弹仓

优点：重心在中间，转向时不与灯柱碰撞

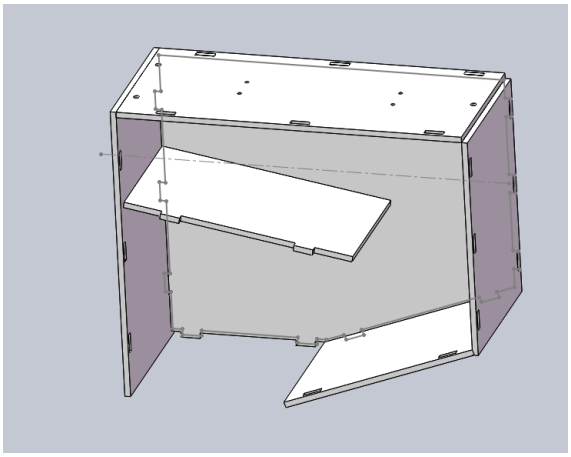
缺点：容量小



第二代弹仓

优点：重心在中间，容量大，对拨弹轮到压力小

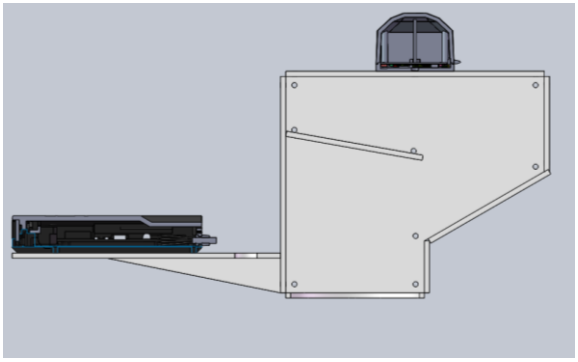
缺点：转向时与灯柱碰撞



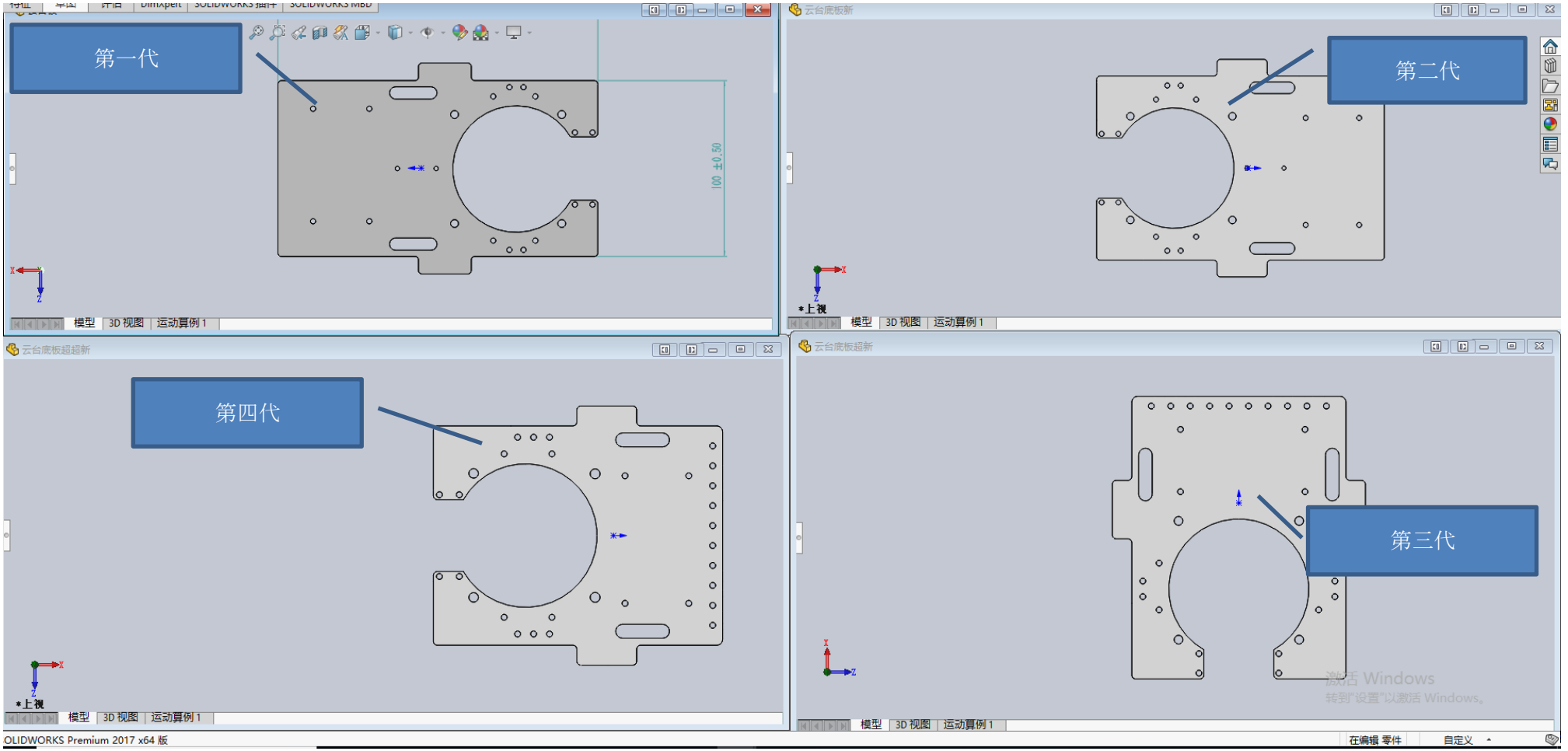
第三代弹仓

优点：重心在中间，容量大，对拨弹轮到压力小，转向时不与灯柱碰撞

缺点：图传支架不稳定



云台底板



第一代云台底板

长度刚刚好与灯柱相摩擦，所以改到了第二代

第二代云台底板

机械没有问题，但嵌入式告诉我们 RM 主控板没有办法倒着装，所以改良了底板

第三代云台底板

因为暂时没有碳纤维板了, 只能亚克力板, 因为我没有计算静力分析, 导致晚上试车的时候，抬头太猛把亚克力板弄断了。

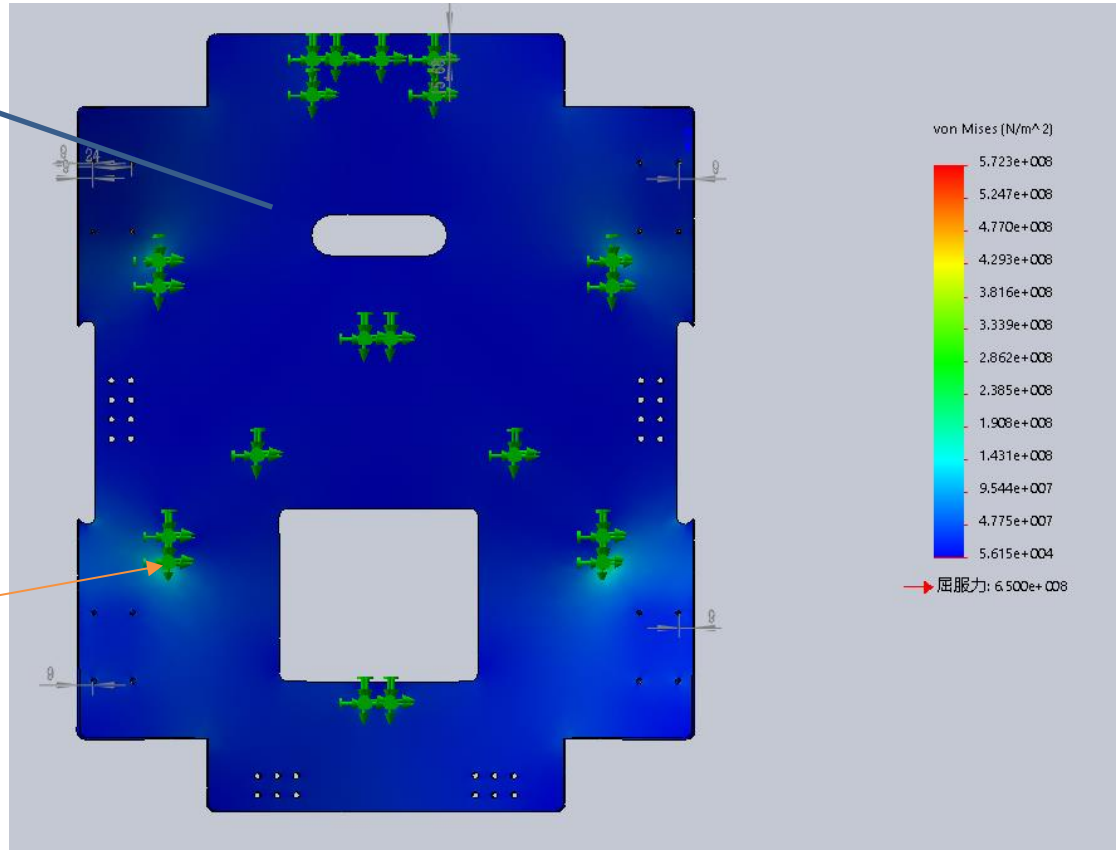
第四代云台底板

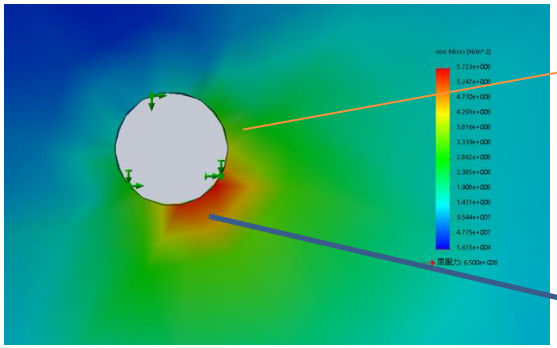
改良了第三代，并用玻璃纤维板切割

受力分析

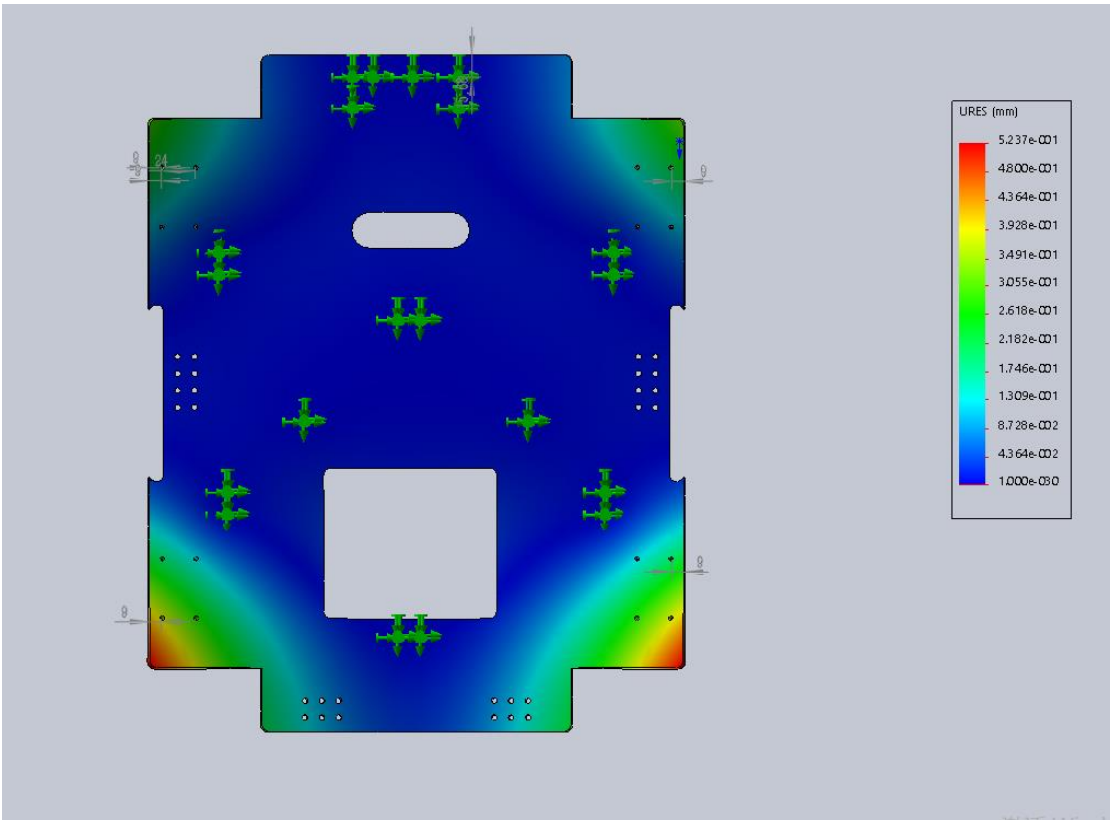
在云台底板设计的时候我忘了受力分析，使用导致这样问题，所以我把每个主要的板子都做了受力分析，算出哪里最受力。

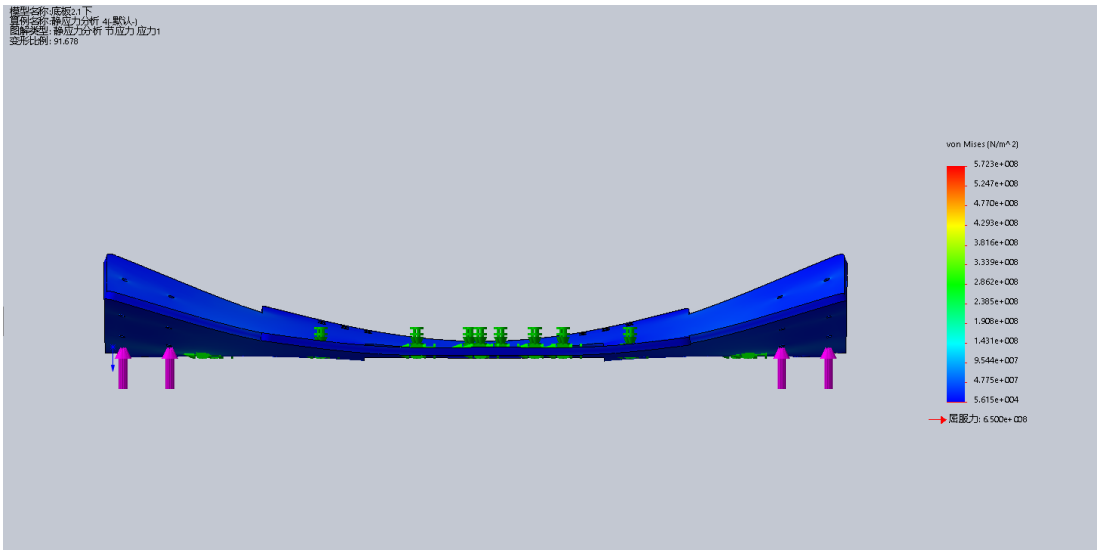
下底板受力分析



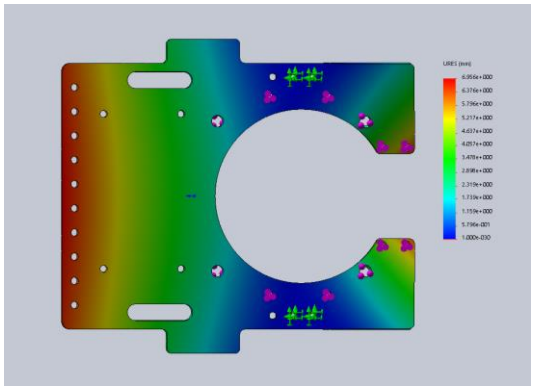


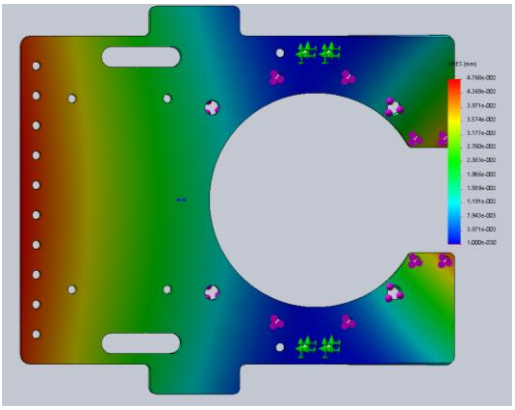
这个底板上的螺丝是受力最大的，所以当我们每打完一场比赛就检查一遍这个螺丝

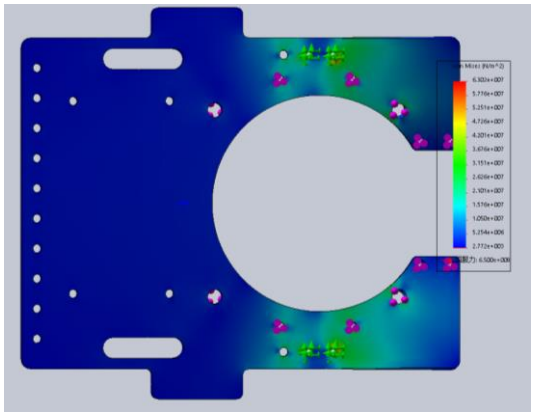


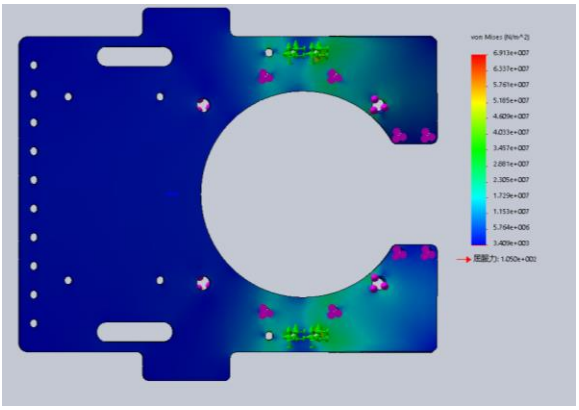


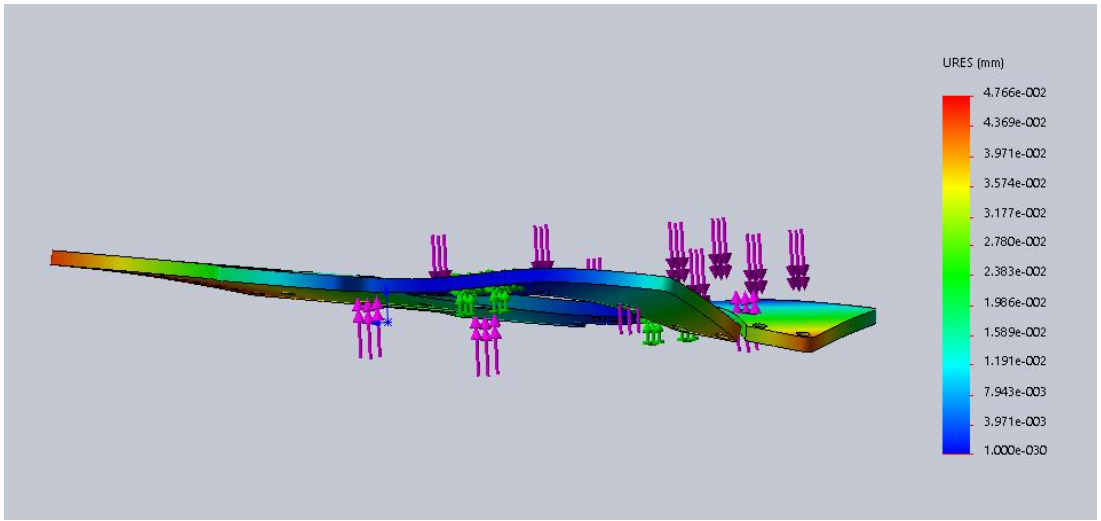
底板变形（图片是软件方便人看，而增加的变形尺度，实际变形不到 0.05mm）

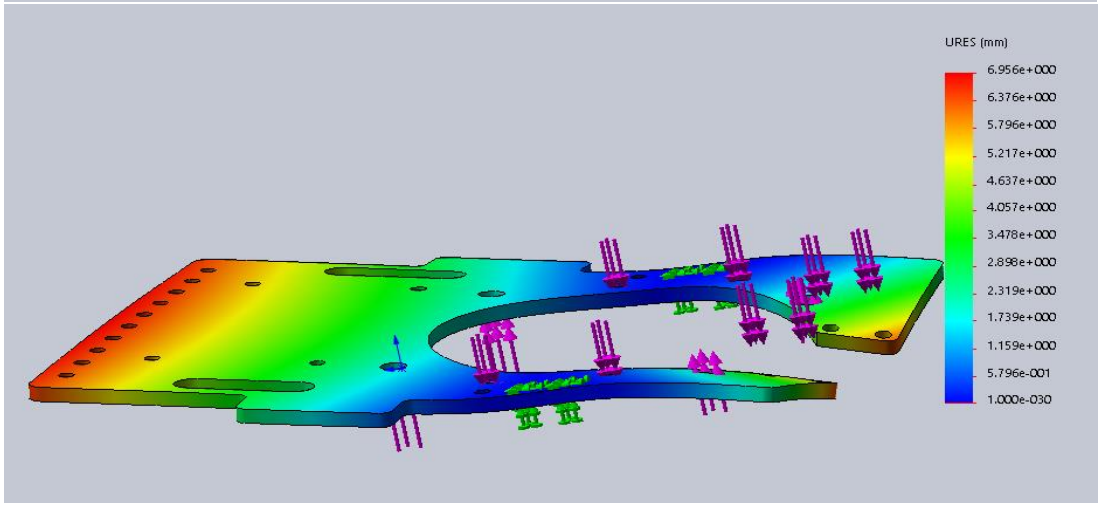









URES (mm)
4.766e-002
4.369e-002
3.971e-002
3.574e-002
3.177e-002
2.780e-002
2.383e-002
1.986e-002
1.589e-002
1.191e-002
7.943e-003
3.971e-003
1.000e-030


URES (mm)
6.956e+000
6.376e+000
5.796e+000
5.217e+000
4.637e+000
4.057e+000
3.478e+000
2.898e+000
2.319e+000
1.739e+000
1.159e+000
5.796e-001
1.000e-030

云台底板（亚克力板与碳纤维板对比）

嵌入式：我们发现了代码中关于步兵通过扭腰避弹的彩蛋，并且将这个功能加入到了键盘控制的代码中

```
if (rc.kb.bit.X)//如果x被按下，开始扭腰
  {
     chassis_control_information_get();
     //底盘扭腰旋转速度处理，覆盖前面计算出来的值
     chassis_twist_handle();
  }
```

林昕泽还编写了使用舵机控制机械臂夹取障碍块的代码，并也将其加入了键盘的控制中，

```
if (rc.kb.bit.C)//如果c被按下，且夹子没有合住
{
  int i;
  uncaughted=~uncaughted;
  if(state_jiazi!=uncaughted){
    state_jiazi=uncaughted;
  if(uncaughted==1)
    {
      for(i=1600;i<2100;i+=20)
      {
      set_pwm_param(PWM_IO4, i);
      set_pwm_param(PWM_IO7, 3700-i);
      osDelay(20);
      }
   }
  else if(uncaughted==0){
    for(i=2100;i>1600;i-=20){
      set_pwm_param(PWM_IO4, i);
      set_pwm_param(PWM_IO7, 3700-i);
      osDelay(20);}
}
}
```

2.4 可行性测试

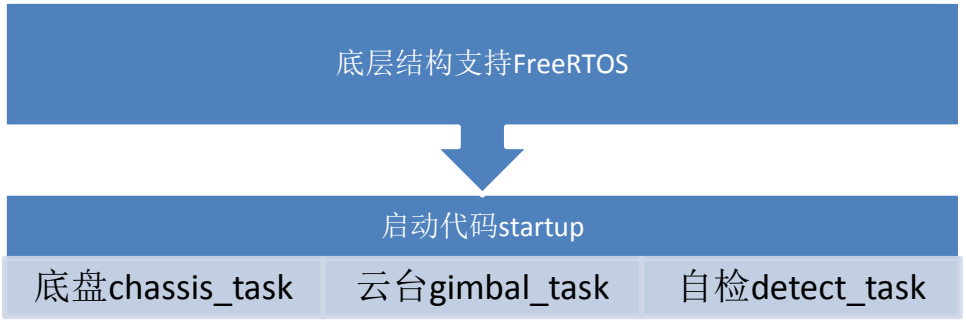
林昕泽和王豪还用 Open mv 尝试进行了有关机器视觉的代码编写，计划用 UART 将 open mv 模块中计算的数据回传到主控板，

帮助瞄准。但是由于后来模块质量问题，摄像头感光板损坏，更换的话时间又很紧迫变不得不放弃。于睿华本计划使用红外传感器放在两机械臂上帮助机械臂夹取障碍块（当红外被遮挡时，电平变化，使用中断或轮询控制机械臂夹取），但由于机械臂的制作进度比较慢，最后不得不放弃。

3、控制部分

3.1 项目软件架构

3.1.1 软件框图



底层代码：

底层代码由 FreeRTOS 封装，具有 io 操作，PWM 输出，UART 通信，CAN 总线通信，蜂鸣器控制及 RTOS 基本功能（进程控制，延时，定时器等）。

启动代码：

启动代码主要由 startup.c 组成，用途是初始化任务中用到的 IO 端口，初始化 UART 和 CAN 总线通信，注册接收遥控器用 DBUS 和电机用 CAN 总线的数据的回调函数和中断函数。

3.1.2 操作说明

右侧拨杆：

上：正常模式，云台和底盘都正常工作

中：单独底盘模式，此时云台断电

下：机器人失能，所有机构都切断输出

左侧拨杆：

上：在和中间切换时，控制摩擦轮开启和关闭

中：进入键盘控制模式，此时遥控器也可以控制

下：在和中间切换时，控制单发发弹，停留 2s 后连续发射

在右侧拨杆为上时，左侧拨杆为下可启动摆腰

键盘：

Q：开启摩擦轮

Q+shift：关闭摩擦轮

W/S/A/D：前后左右控制

shift：快速模式

ctrl：慢速模式

C：启动/关闭摆腰（在 keyboard.c 添加接口）

G：启动/关闭机械臂（在 keyboard.c 添加接口）

鼠标：

左键：

单击：单发子弹

长按 1s：连续发射

3.2 云台模块

云台任务进程即 gimbal_task（）函数，该函数先初始化云台的 pid 控制，然后从主控板 flash 读取 glb_cali_data 结构体，通过检测该结构体中的 gimbal_cali_data.calied_flag 标志位是否为 CALIED_FLAG 来判断云台是否经过校准，如果未校准则进入死循环等待校准，如果校准了则等待遥控启动云台。并按照遥控数据用 pid 调控云台电机。

遇到的问题在于云台电机采用 can 总线发送电流数据，但是 can 总线上的设备需要一个唯一的 ID，然而起初我们并不知道云台电机还需要设置 ID，但是当所有电机都接好时，主控板蜂鸣器仍叫三声表示云台 pitch 轴电机离线，所以于睿华决定去找助教寻求帮助，但是当时许多队还没有做出云台，助教也不太清楚什么问题，后来杜助教过来说要拆开电机后盖调 ID，我们才看到电机芯片上的拨码开关，之后于睿华和助教及他组同学一起根据电机手册调整了电机的 ID 后才能正常驱动云台。

3.3 底盘模块

底盘任务进程即 chassis_task()函数，该进程先初始化底盘的 pid 参数，然后挂起底盘任务进程，等待云台任务进程唤醒。唤醒后通过

get_chassis_mode(void)函数从接收到的遥控数据获取底盘模式。底盘一共有四种模式：底盘跟随云台模式（CHASSIS_FOLLOW_GIMBAL），底盘开环模式（CHASSIS_OPEN_LOOP），底盘扭腰模式（CHASSIS_TWIST），底盘锁死模式。然后根据模式控制再用 pid 计算底盘速度。底盘遇到的问题是要让电调的 ID 与代码中的 ID 相符，并且符合麦克纳姆轮的原理，所以于睿华先确定了电调的位置，再根据代码中的图示调整了电调的 ID，最终使底盘可以正常移动。

```
* @map    2 %+++++++% 1
              ++++
              ++++
           3 %+++++++% 4
*/
```

还有就是底盘装好时云台还没有就绪，所以为了测试底盘，我们注释了 osThreadSuspend(NULL);语句，阻止挂起底盘，因而得以在不连接云台的情况下测试底盘的移动效果。

3.4 发射模块

发射模块主要有 shoot_customs.c 组成，包含对摩擦轮电机，拨弹电机，和激光瞄准的控制。其中摩擦轮电机使用 PWM 方波控制，拨弹电机采用 CAN 总线控制（但其电流值随云台电机的电流值一起发送）。为了在步兵整体完成之前测试摩擦轮电机和拨弹电机，于睿华编写了以下代码：

```
while(1)
{

  set_pwm_param(PWM_IO1, 1000);
  set_pwm_param(PWM_IO2, 1000);
  osDelay(5000);
  write_led_io(LED_IO1, LED_ON);//led点亮时电机启动
  set_pwm_param(PWM_IO1, 1300);
  set_pwm_param(PWM_IO2, 1300);
  osDelay(10000);
  write_led_io(LED_IO1, LED_OFF);
}
//让拨弹电机加速旋转
//  trigger_moto_current=0;
//  for(i=0;i<500;i=i+5)
//  {
//  trigger_moto_current=i;
//  send_gimbal_moto_current(0,0);
//  write_led_io(LED_IO1, LED_ON);
//  osDelay(10);
//  write_led_io(LED_IO1, LED_OFF);
//  osDelay(200);
```

4、总结

4.1 项目收获

我在本次冬令营对抗赛中主要负责底盘以及机械臂的设计，结果是底盘工作正常而机械臂表现不佳。底盘一开始的问题是要不要使用悬挂，经过多方讨论我们认为放弃悬挂采用简单可靠的双层设计能如期完成进度并留出充足时间调试，实践证明调试的时间越多发挥则越好。底盘在建模软件中安装的时候孔位总是无法对齐，之后发现是测量误差，如果是先加工再检查不经建模则会拖累整组进度。机械臂主要的问题集中在时间上，机械臂只有一天的测试时间，但是本组的舵机由于固定不稳，而且程序运行没有有效调试。如果能够使用自主设计的舵机固定架并且让嵌入式有足够时间调试，我觉得机械臂是可以

有效运行的。总结这次冬令营，最重要的是有先设计后安装的思想和系统思维，不能没有设计和检查就直接动手做，只能拖慢进度；而且不能将眼界锁在单一的领域，比如机械和嵌入式需要良好的沟通而不是推卸责任才能解决矛盾。

所有的事情都不是想的这么简单，能想到的不都是能实现的，在实现之前，要考虑各种影响因素，时间的把握，在冬令营期间，对 solidworks 有了更加深入的了解，把原来先搭建再建模的习惯改为了先建模再搭建，这样在搭建之前可以更全面的分析可行性。

我在这个队伍完成了云台，供弹，弹仓，受力分析，整车装配的工作，在调试云台的时候我感觉与嵌入式的队员沟通有问题，互相都不了解对方到底付出了多少，而导致不必要的冲突，我想在以后的队伍里可以加一个项目管理的人，这样可以加快我们的整体的进度。

这次对抗赛中我第一次接触到了 stm32 芯片，也是第一次使用 CAN 总线驱动设备，还是第一次使用遥控操控设备，甚至是我第一次接触 pid 控制。可以说这次比赛让我收益匪浅，极大地丰富了我的项目经历。

完成了摆腰和机械臂的功能实现，结构与执行动作不稳定，考虑是否有必要实现这个功能，不能以牺牲基本功能为代价实现扩展功能。

4.2 马后炮

我觉得需要能在机械和嵌入式中游刃有余的人才，而且算法也要

更加主动地帮助机械和嵌入式，整个队伍需要更多开诚布公的讨论和交流。我认为搭建一套电机驱动而不是舵机的机械臂会更可靠，而且悬挂在这个比赛场地是必不可少的。

嵌入式：我希望机械会给云台更大的空间，并且把 pitch 轴的转动范围减小，不要给云台太大的重量，还有就是把机械臂的结构做的更合理。我也会优化 pid 参数，杜绝云台抖动的问题。对拨弹速度等细节参数做更多的优化。

实现了一个要功能赶紧调试，稳定了再做下一个。

用事实验证自己的想法是否合理，绝不能仅凭感觉判断