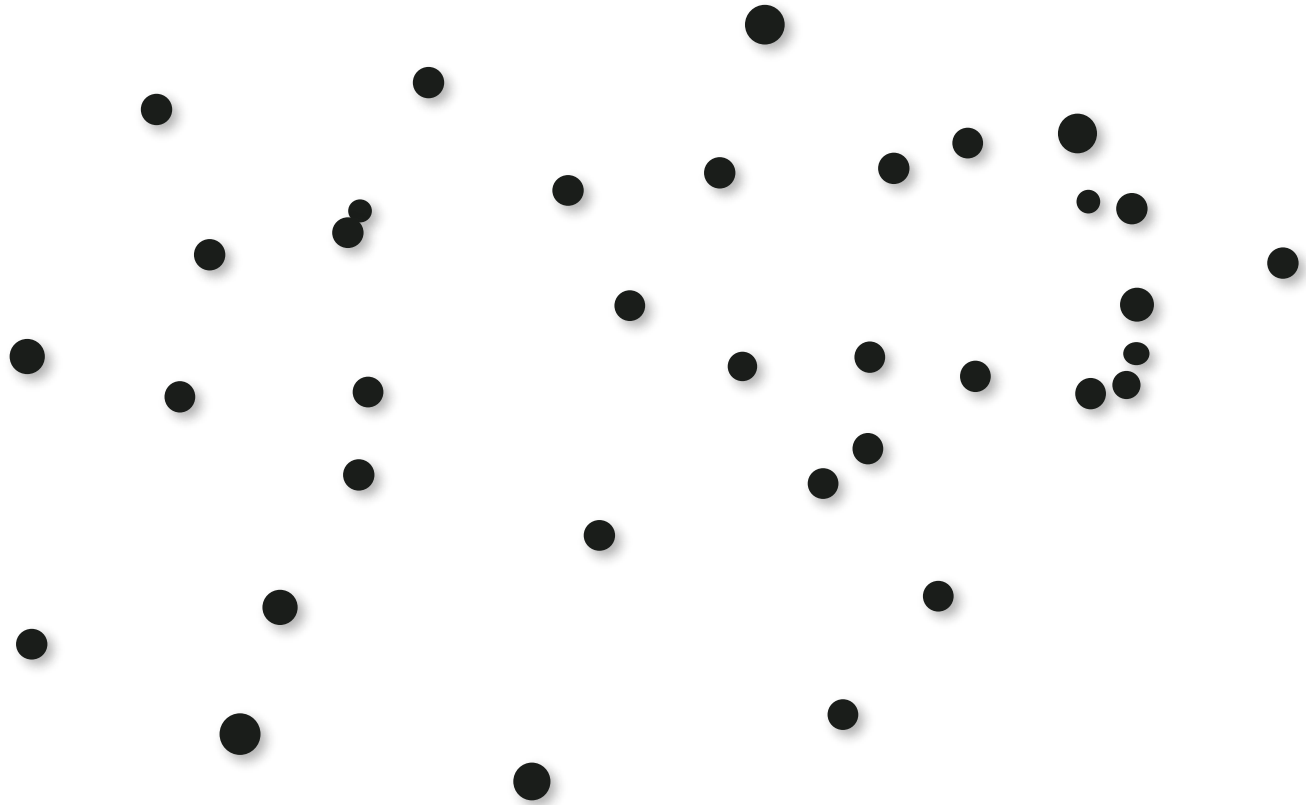


RoboMaster

2018

第11组答辩

汇报人：第11组

1 人员安排

2 大致情况

3 详细方案

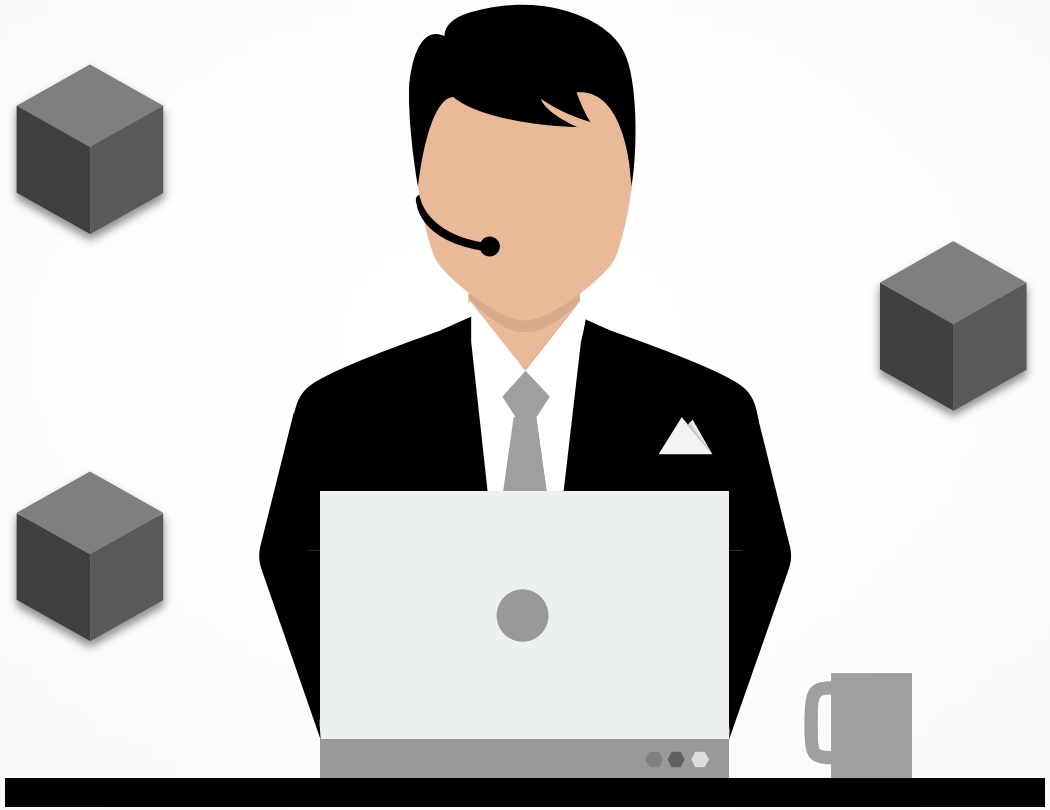
组长：张文翰

机械组

张文翰、文宇飞

嵌入式组

侯彦颀、陈泽祺、袁梓豪



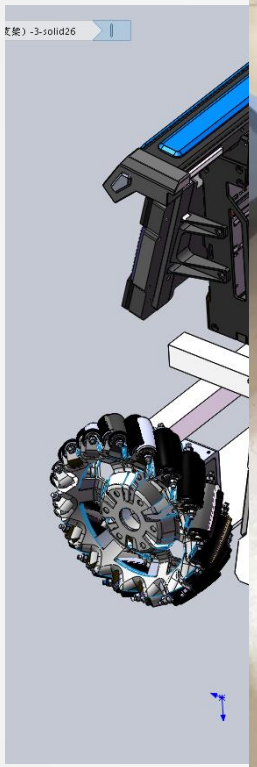
算法组

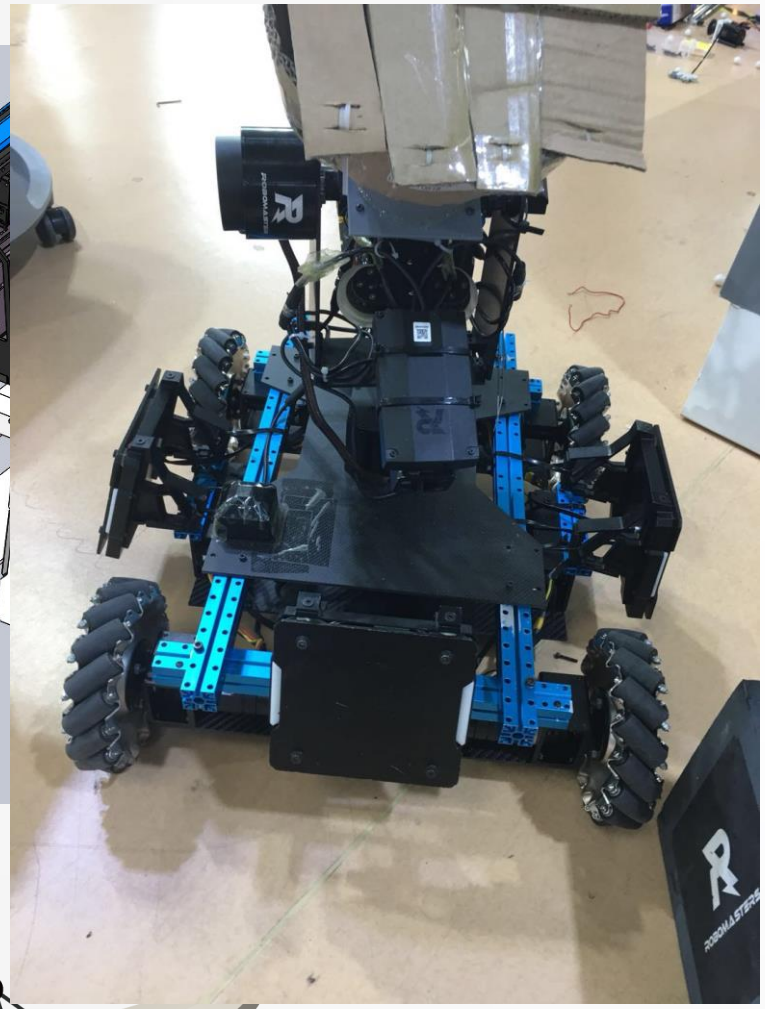
薛浩楠、屈子铭

- 第一阶段机械设计

  - 底盘
  - 云台
  - 发射装置
  - 夹取结构

- 底盘——底盘结构

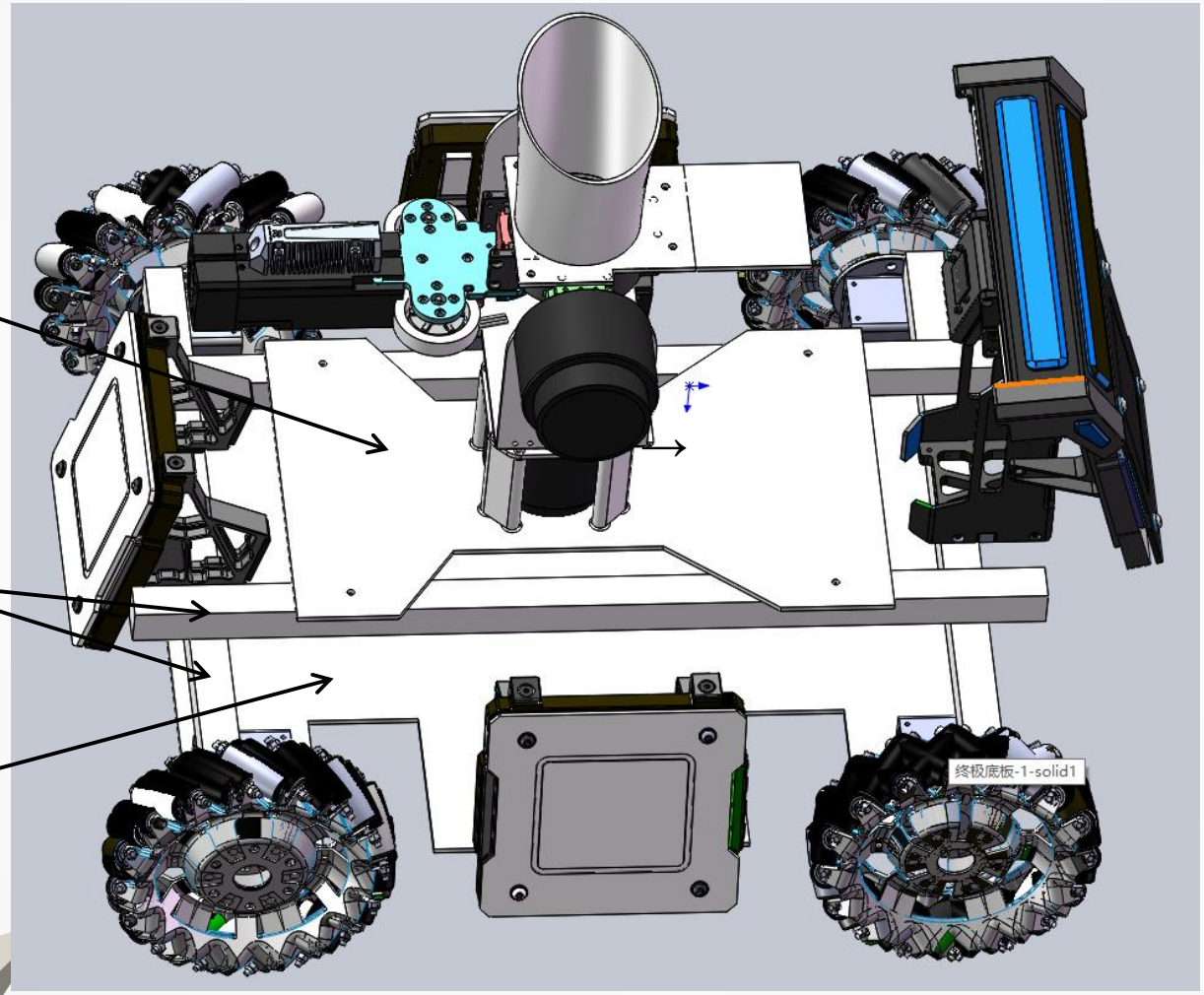


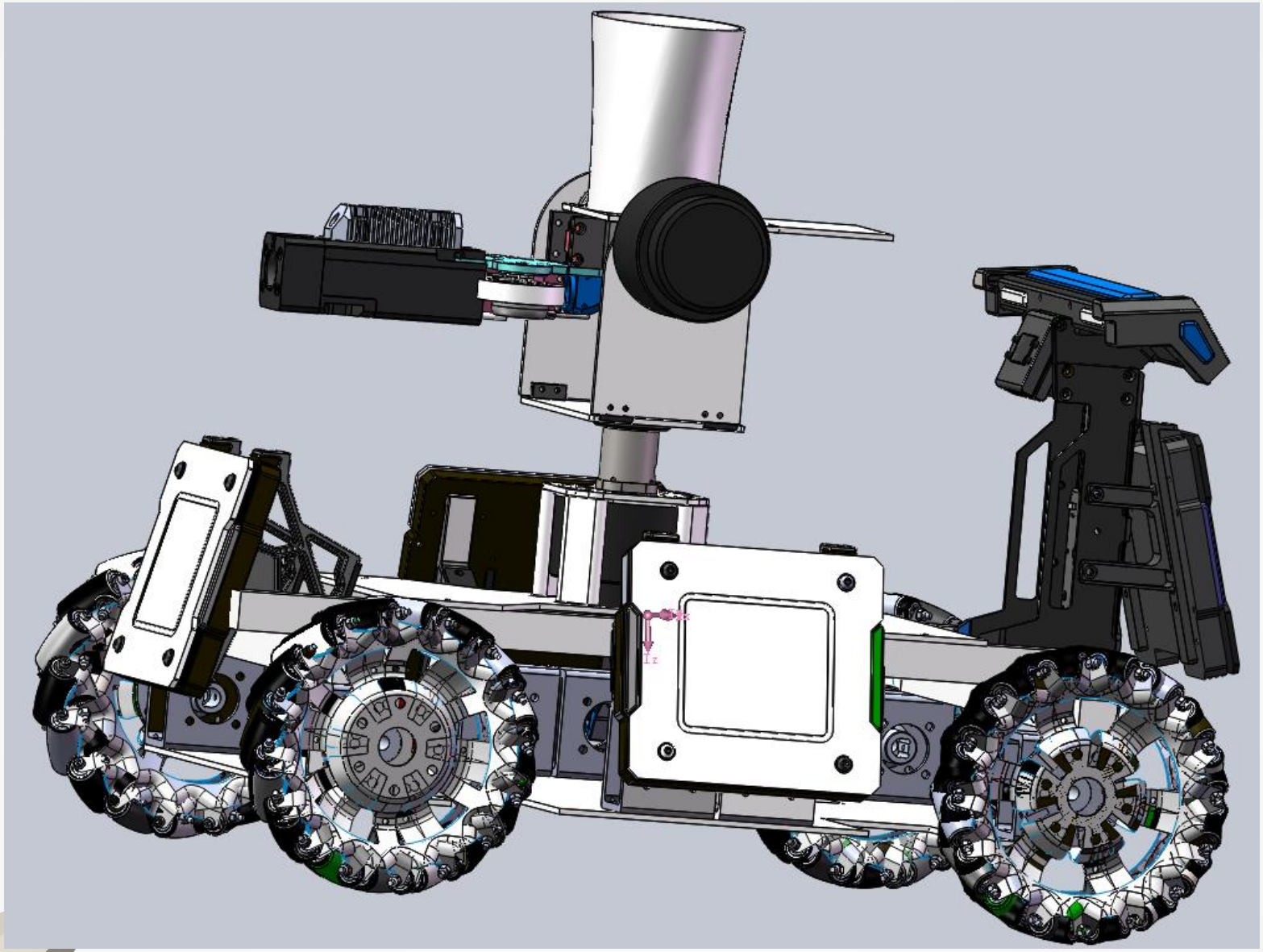


上底盘

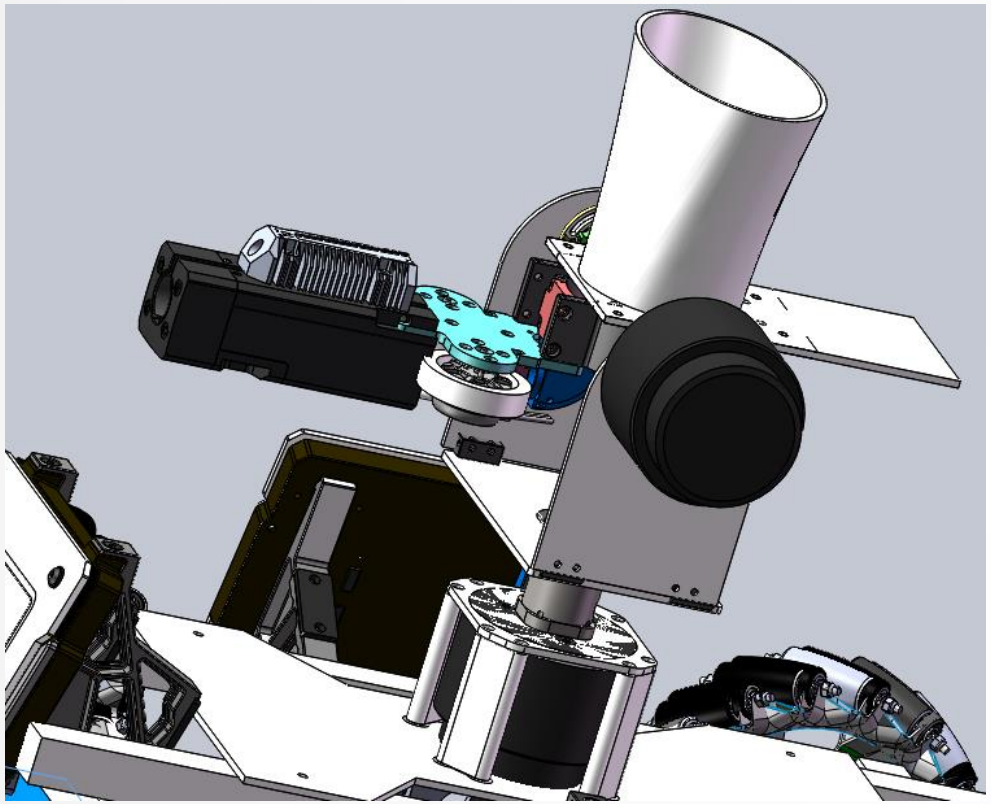
铝架结构

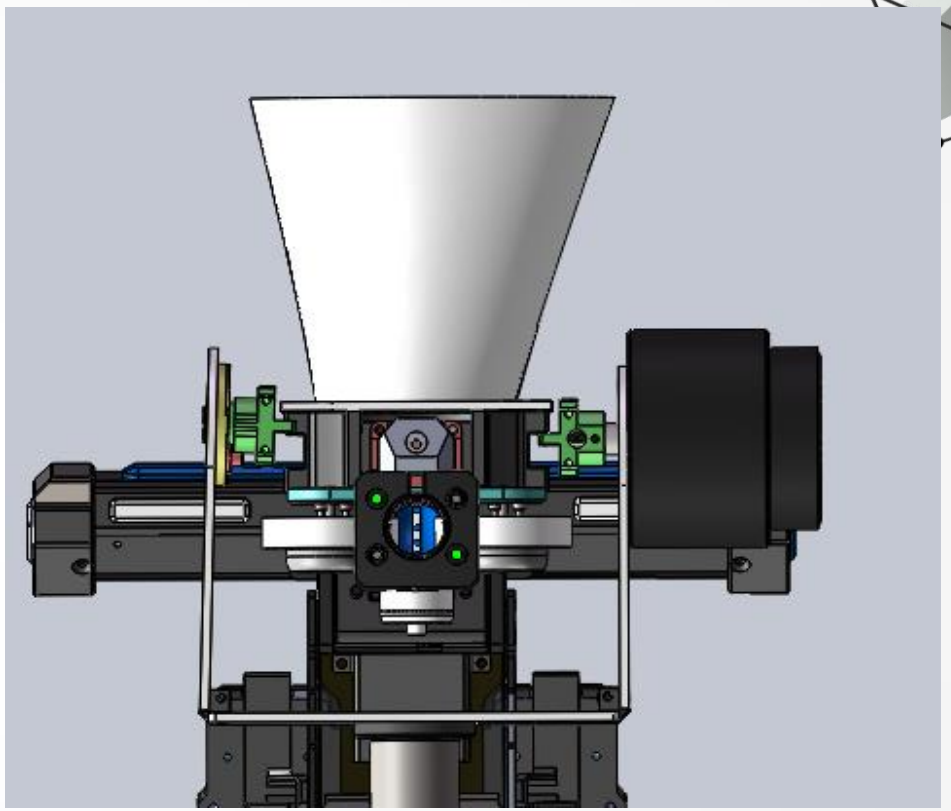
下底盘



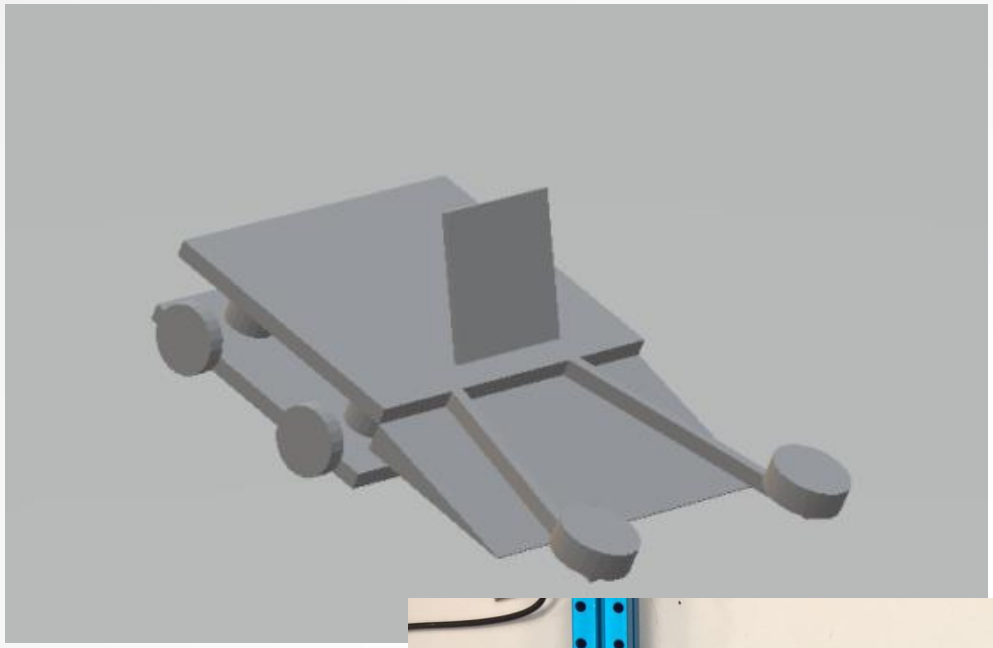


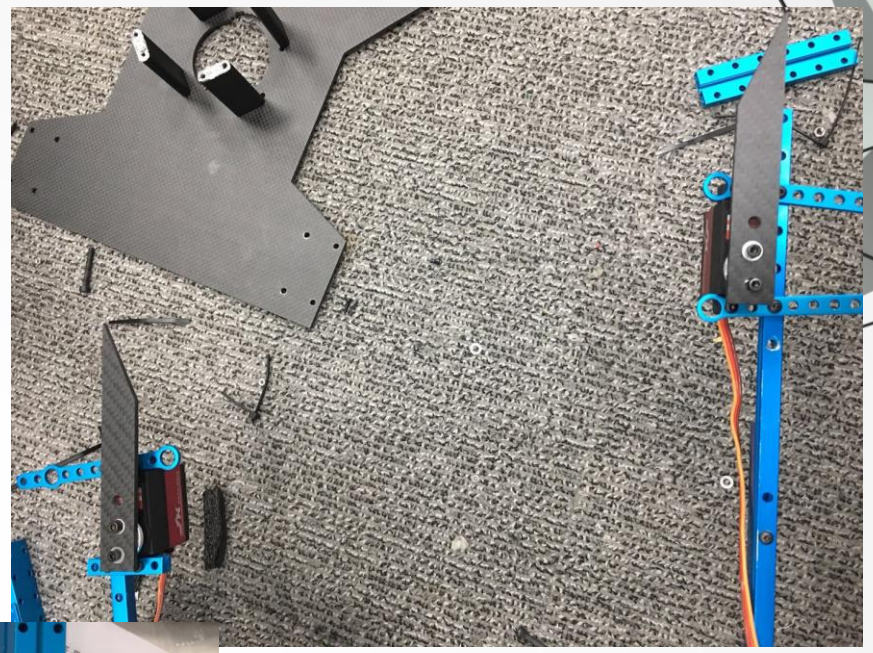
- 云台与发射结构

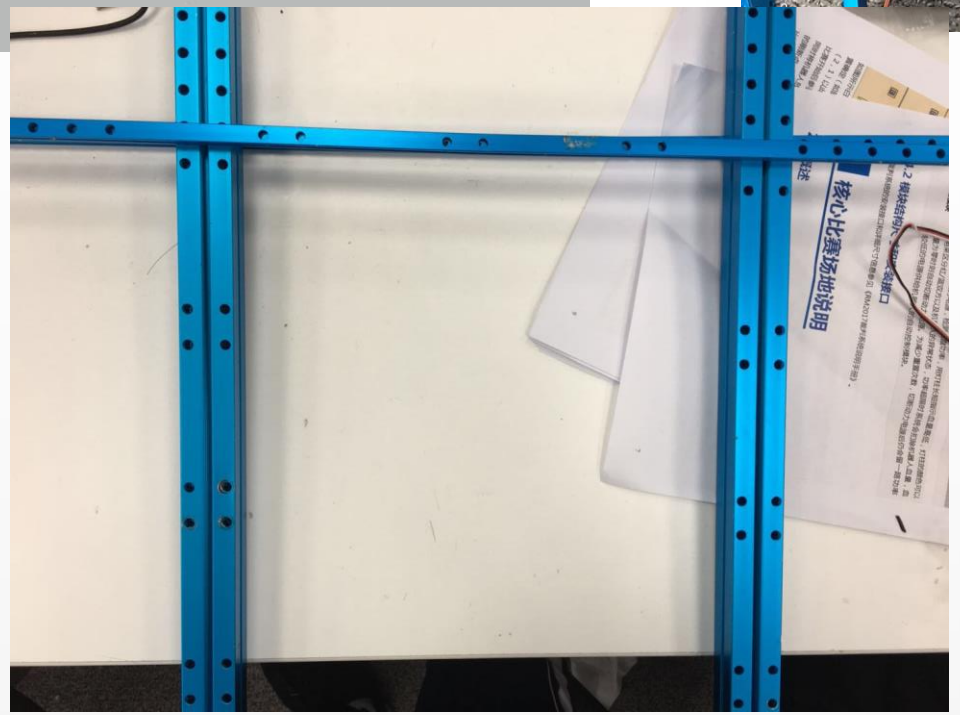




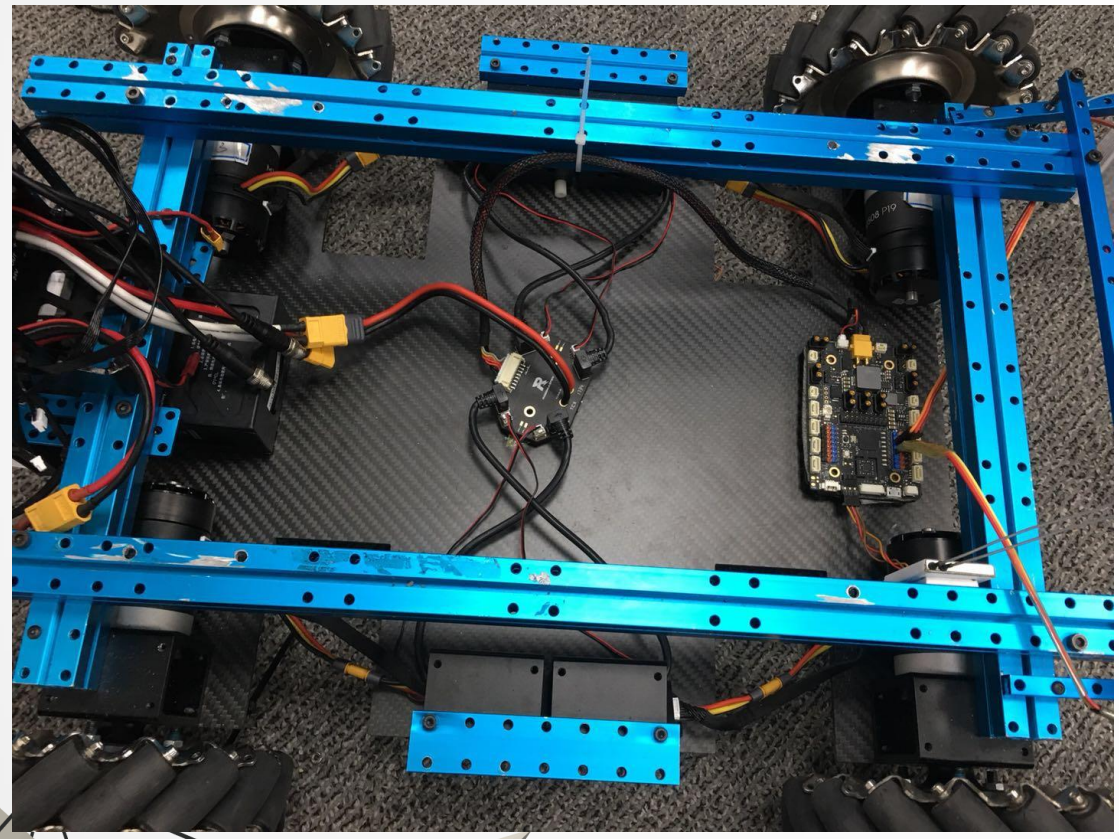
夹取结构

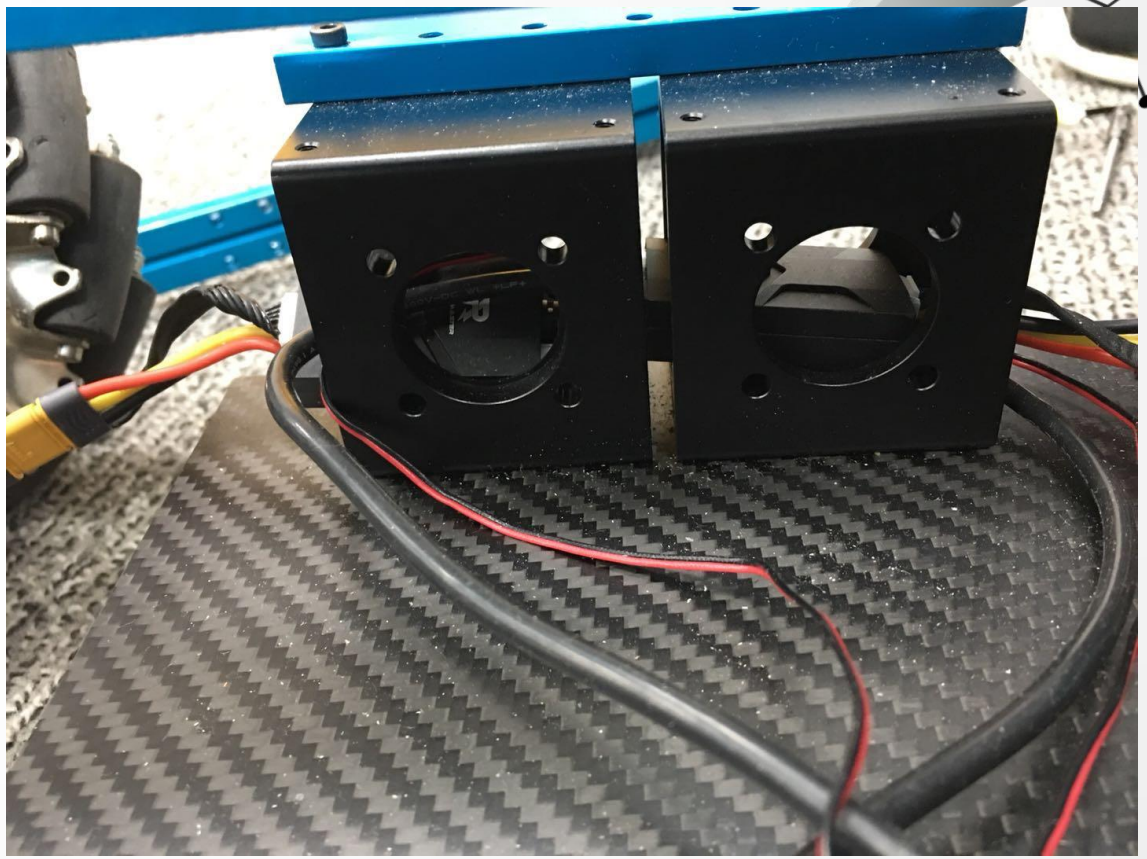




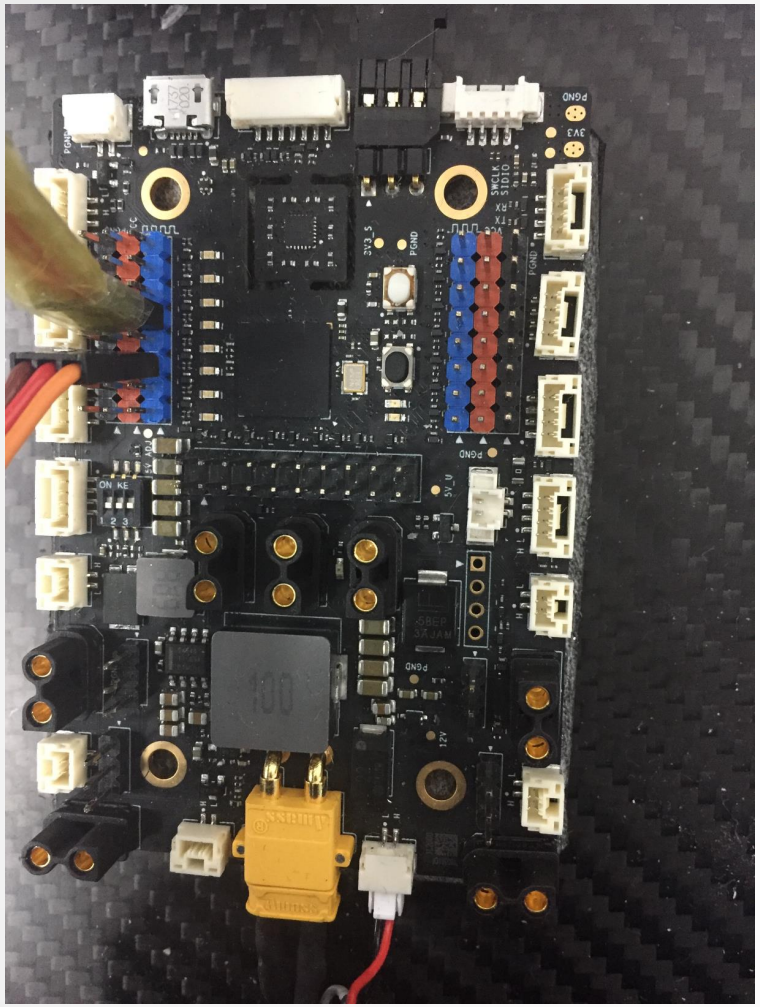


底盘+引线



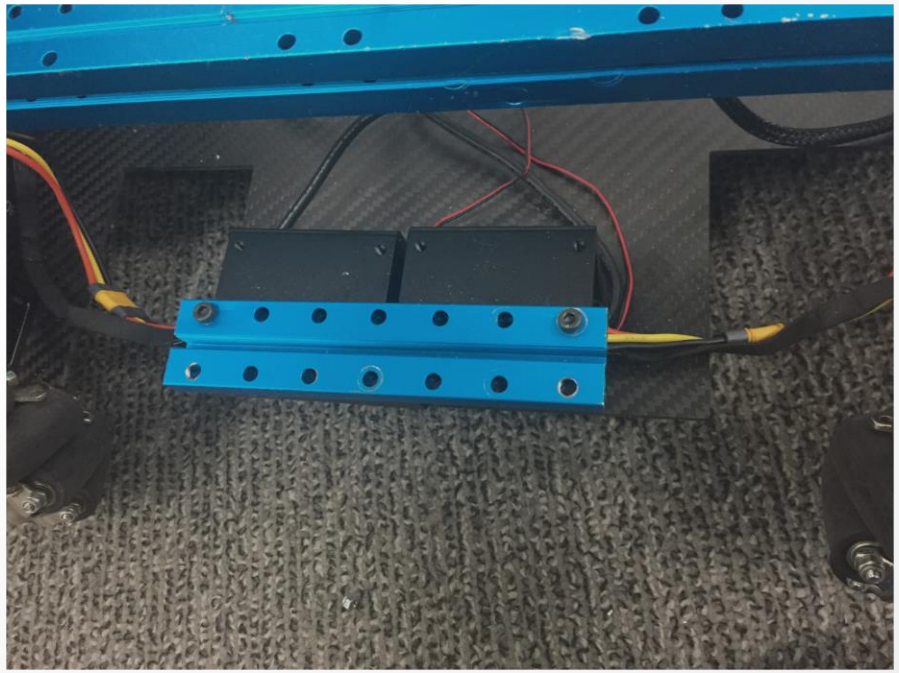


- 嵌入式设计

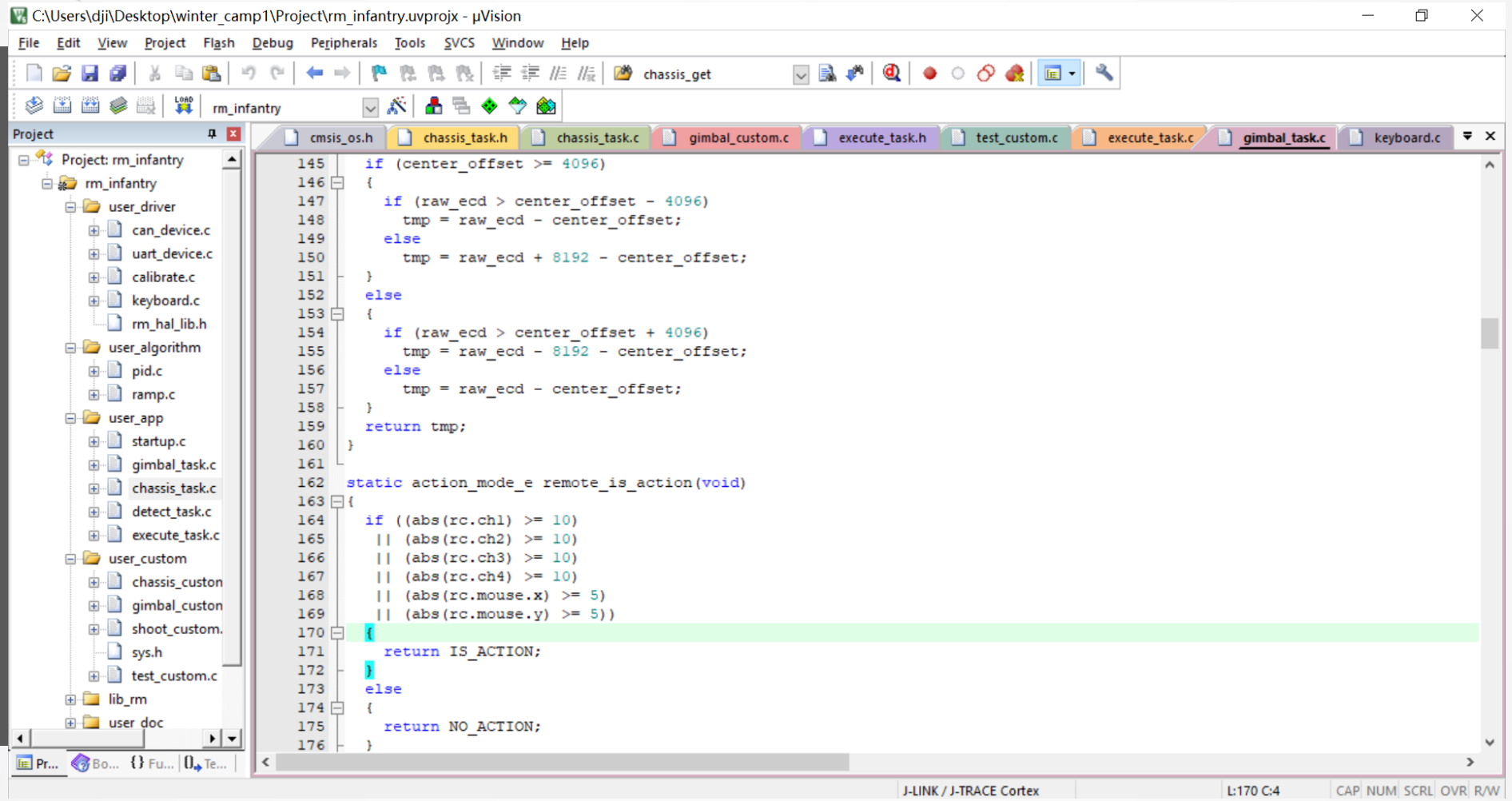


学习这块东西

1. 了解各个借口的用处，并将线正确接入
2. 拨码，识别正确的信号。
3. 进行走线，不影响电机及云台的工作。



读代码


C:\Users\dji\Desktop\winter_camp1\Project\rm_infantry.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
chassis_get
rm_infantry
Project
Project: rm_infantry
rm_infantry
user_driver
can_device.c
uart_device.c
calibrate.c
keyboard.c
rm_hal_lib.h
user_algorithm
pid.c
ramp.c
user_app
startup.c
gimbal_task.c
chassis_task.c
detect_task.c
execute_task.c
user_custom
chassis_custon
gimbal_custon
shoot_custom.
sys.h
test_custom.c
lib_rm
user_doc
cmsis_os.h
chassis_task.h
chassis_task.c
gimbal_custom.c
execute_task.h
test_custom.c
execute_task.c
gimbal_task.c
keyboard.c
145 if (center_offset >= 4096)
146 {
147 if (raw_ecd > center_offset - 4096)
148 tmp = raw_ecd - center_offset;
149 else
150 tmp = raw_ecd + 8192 - center_offset;
151 }
152 else
153 {
154 if (raw_ecd > center_offset + 4096)
155 tmp = raw_ecd - 8192 - center_offset;
156 else
157 tmp = raw_ecd - center_offset;
158 }
159 return tmp;
160 }
161
162 static action_mode_e remote_is_action(void)
163 {
164 if ((abs(rc.ch1) >= 10)
165 || (abs(rc.ch2) >= 10)
166 || (abs(rc.ch3) >= 10)
167 || (abs(rc.ch4) >= 10)
168 || (abs(rc.mouse.x) >= 5)
169 || (abs(rc.mouse.y) >= 5))
170 {
171 return IS_ACTION;
172 }
173 else
174 {
175 return NO_ACTION;
176 }
J-LINK / J-TRACE Cortex
L:170 C:4
CAP NUM SCRL OVR R/W

- 找到有用信息

```
/*****************************发射速度设置*************************************/
#define SHOT_FRIC_WHEEL_SPEED    1300 //最大为2500

/*****************************发射频率设置*************************************/
#define TRIGGER_MOTOR_SPEED      -5000 //
```

- 射击调试

我们拍摄了拨弹轮的旋转视频，然后慢动作处理，计算出通过一个口所需要的时间，从而得出子弹速度。完美贴近每秒5发。

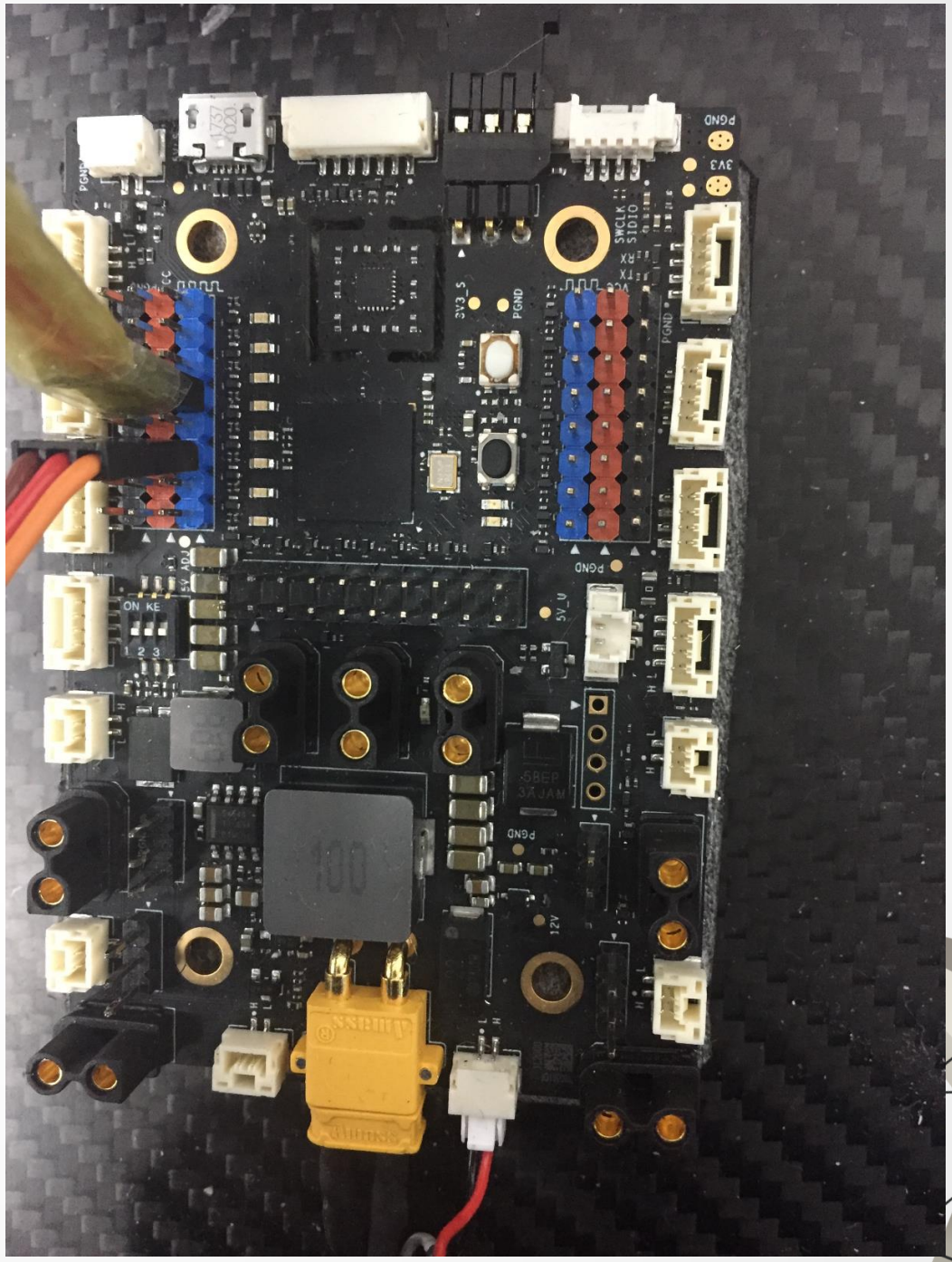
• 控制开发

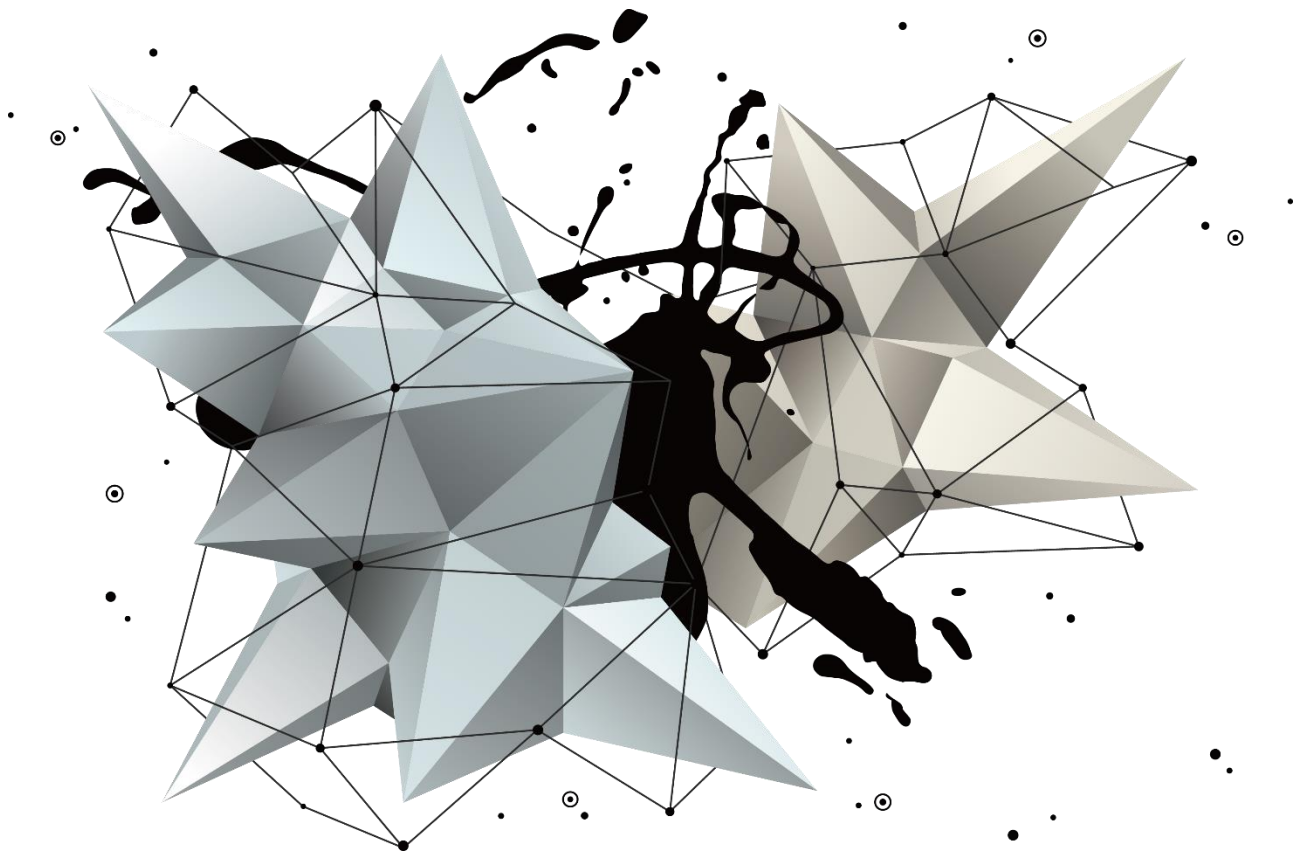
```
//add ramp
if (rc.kb.bit.W)
  km.vy += delta_spd;
else if (rc.kb.bit.S)
  km.vy -= delta_spd;
else
  km.vy = 0;

if (rc.kb.bit.A)
  km.vx += -delta_spd;
else if (rc.kb.bit.D)
  km.vx += delta_spd;
else
  km.vx = 0;
```

```
/* PC 键盘按键数据 */
union
{
  uint16_t key_code;
  struct
  {
    uint16_t W:1;
    uint16_t S:1;
    uint16_t A:1;
    uint16_t D:1;
    uint16_t SHIFT:1;
    uint16_t CTRL:1;
    uint16_t Q:1;
    uint16_t E:1;
    uint16_t R:1;
    uint16_t F:1;
    uint16_t G:1;
    uint16_t Z:1;
    uint16_t X:1;
    uint16_t C:1;
    uint16_t V:1;
    uint16_t B:1;
  }bit;
}kb;
```

代码中定义了多个按键，但有许多未被使用，打算使用另外的按键来实现步兵的左右扭腰，而云台保持不动

BUT

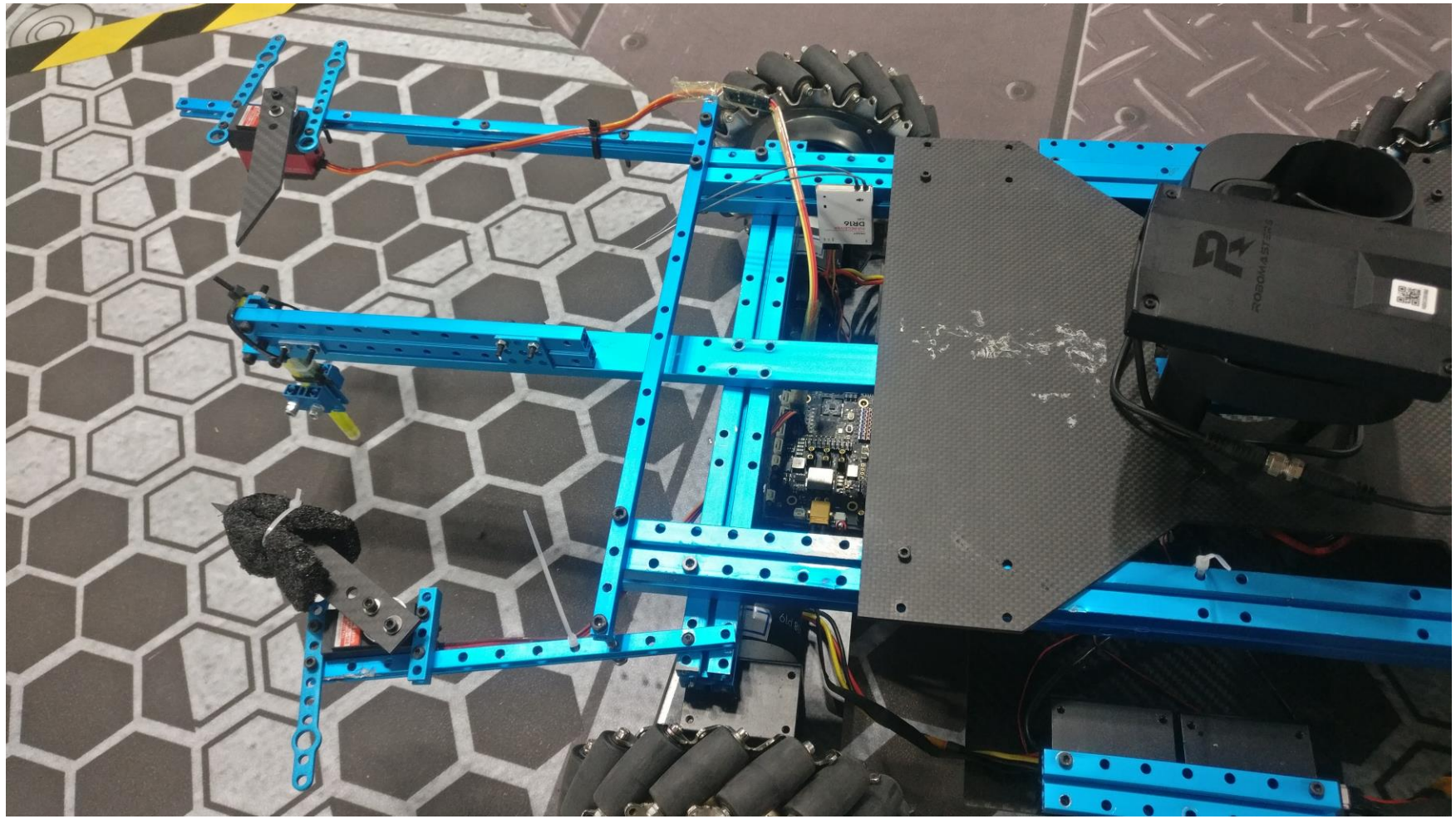

PGND
3V3
SWCLK
SIDIO
RX
TX
3V3_S
PGND
PGND
ON KE
1 2 3
5V_U
PGND
12V



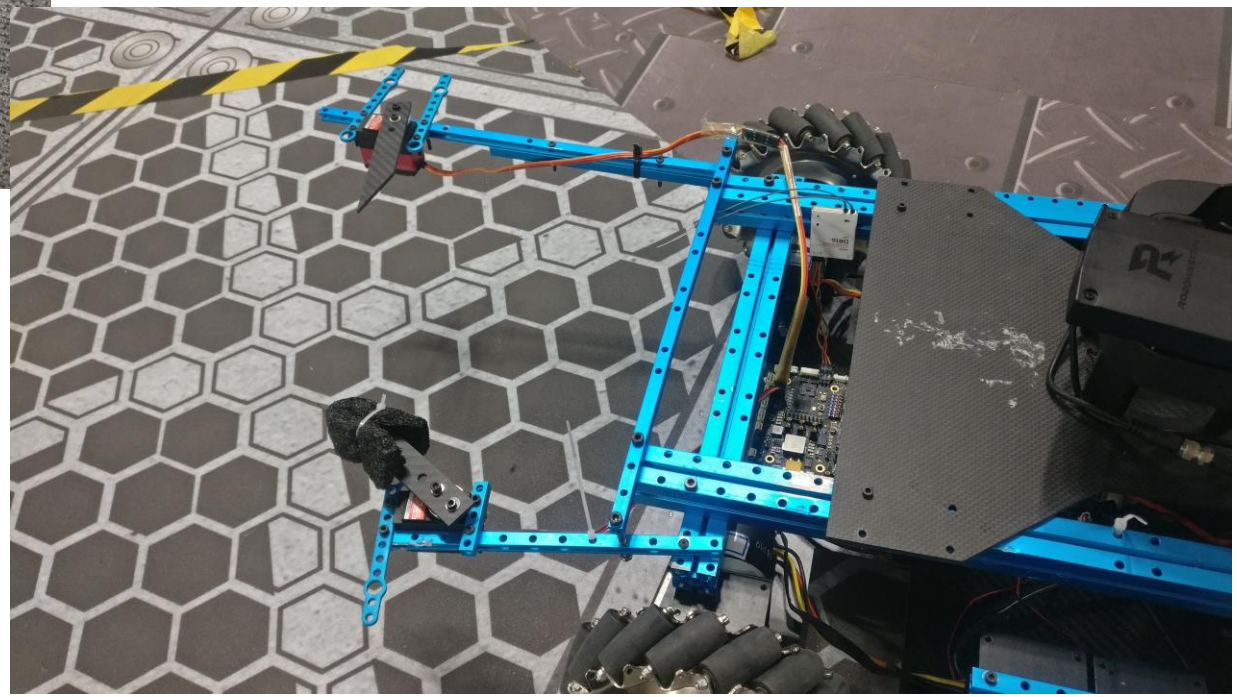
挑战赛

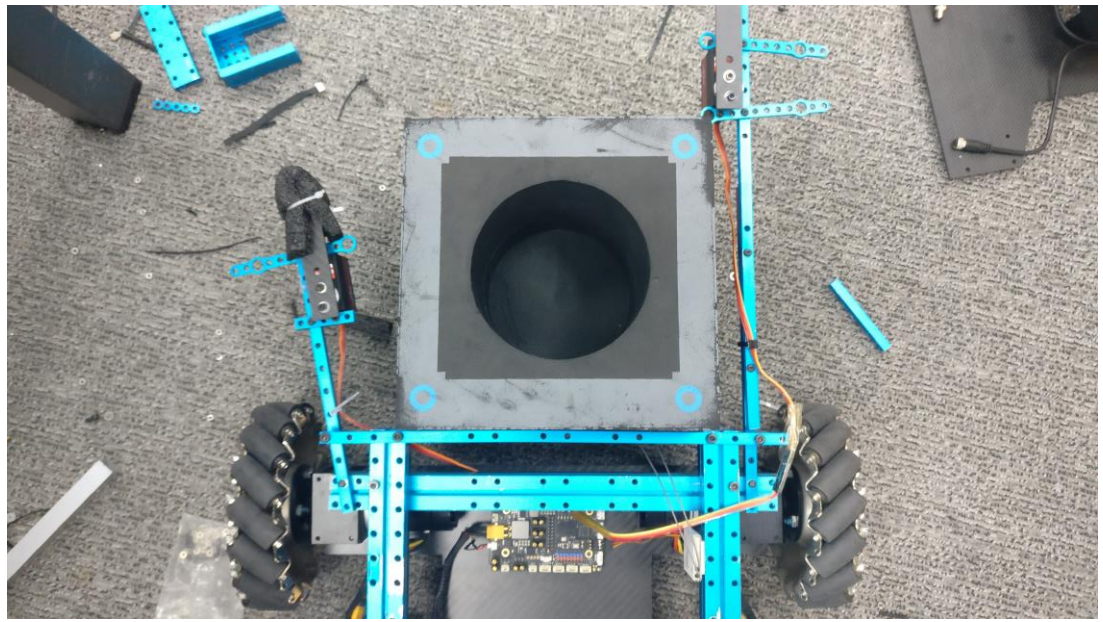
华容道机械臂

- 第二阶段机械设计



连连看抓取机构





具体实现情况

嵌入式设计

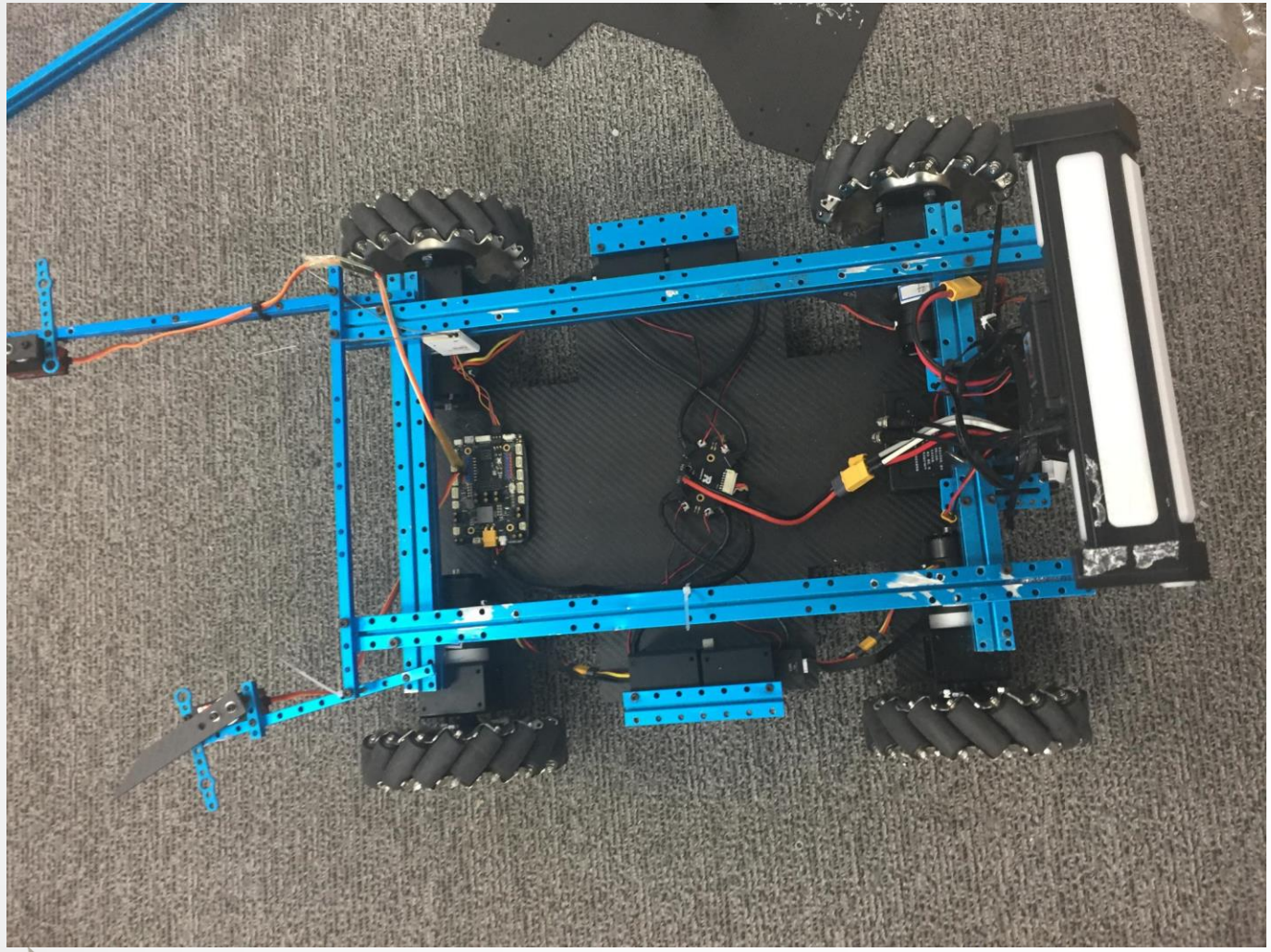
- 找到有用信息

```
/******** 底盘最大速度设置 *******/
/* 底盘移动最大速度，单位是毫米每秒 */
#define MAX_CHASSIS_VX_SPEED 3300
#define MAX_CHASSIS_VY_SPEED 3300
/* 底盘旋转最大速度，单位是度每秒 */
#define MAX_CHASSIS_VR_SPEED 300
```

- 底盘速率调试（华容道）

```
#define PWM_IO15          15
#define PWM_IO16          16
//PWM IO 相关函数
 * @brief      开启 PWM 输出
 * @param      pwm_id: PWM IO 的 ID
 */
void start_pwm_output(uint8_t pwm_id);
/**
 * @brief      设置 PWM 组对应参数
 * @param      pwm_group: PWM 组别，目前有 4 组PWM
 * @param      period:  每组 PWM 对应周期时间，单位是微秒(us)
 */
void set_pwm_group_param(uint8_t pwm_group, uint32_t period);
/**
 * @brief      设置 PWM IO 参数
 * @param      pwm_id: PWM IO 的 ID
 * @param      pulse: 配置 PWM 的高电平时间，单位是微秒(us)
 */
void set_pwm_param(uint8_t pwm_id, uint32_t pulse);
```

- 舵机运行（连连看）



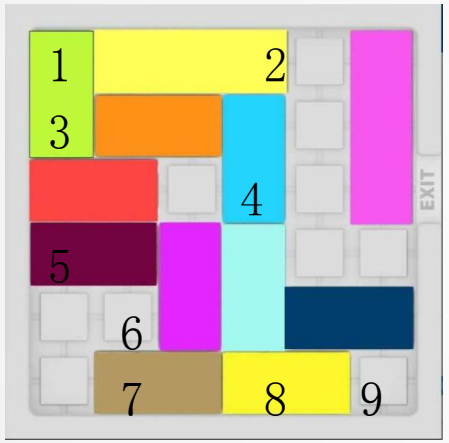
既然我们已经有可以实现二维移动的车了，那我们只需把它的速度调低，调低，在调低。令他的精度上升，即可完成华容道的任务

华容道

算法组

1. 物理模型转换为数学模型

物理模型



10

11

12

数据输入

			横竖	行
	块数	12		
列	长度	1	1	1
	木块1	2		
	木块2	0	1	4
	木块3	3		
	木块4	1	1	6
	木块5	3		
	木块6	0	2	3
	木块7	2		
	木块8	1	2	4
	木块9	2		
	木块10	0	3	2
	木块11	2		
	木块12	0	4	2
	推出的目标编号	2		
		1	4	3
		2		
		1	4	4

数学模型

1 2 2 2 0
3
1 4 4 5 0
3
6 6 0 5 0
3
7 7 8 9 0
0
0 0 8 9 10
10
0 11 11 12 12 0

2.状态压缩

map[6][6]

1 2 2 2 0 3

1 4 4 5 0 3

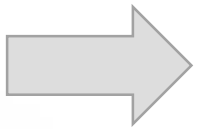
6 6 0 5 0 3

7 7 8 9 0 0

0 0 8 9 10 10

0 11 11 12 12 0

（16）



q[6]

1188355

1328387

6685955

7833856

3524

769216

（10）

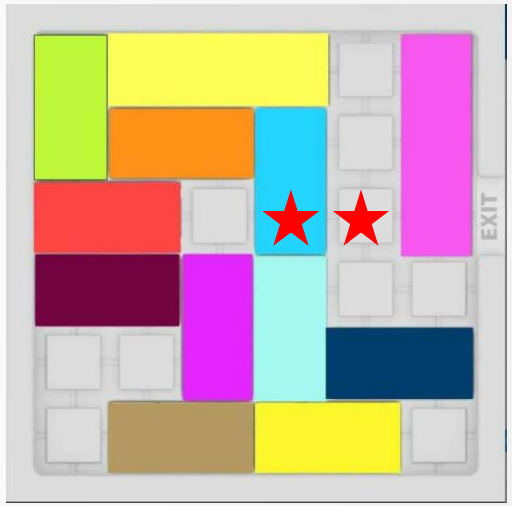
3. 搜索过程数据存储

- 压缩后地图（用于判重）
- 每个木块的坐标（用于改变每个木块位置生成子地图）
- 该地图的父亲地图队列编号（用于得出到此地图的所有操作）
- 该步移动的木块编号和移动方向（记录操作）
- 到此地图所需总步数（用于输出答案）

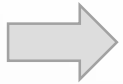
4. 移动过程输出

- 搜索：需滑出的木块为右图位置且右边格为空时输出

1. 不断将父辈地图的操作木块和移动方向添加到新数组
2. 逆序输出该数组，如连续移动相同木块，则合并输出



例如：
木块1向右移动1格
木块1向右移动1格

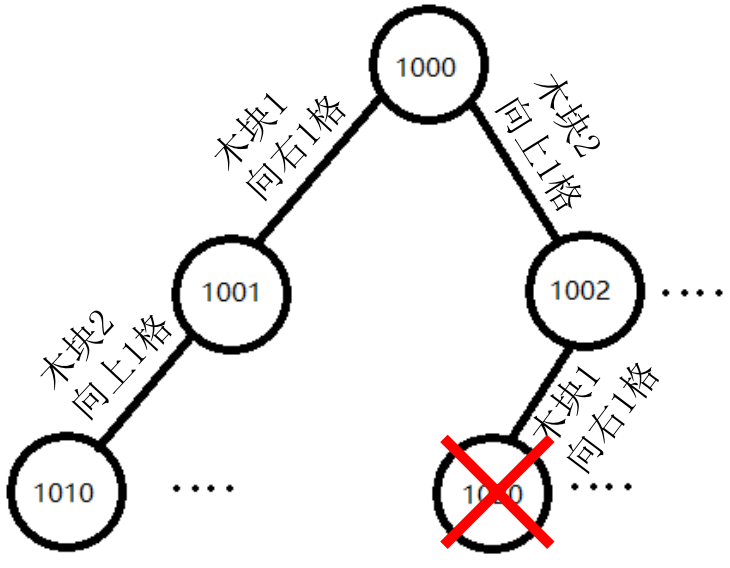


木块1向右移动2格

迭代过程

//t为地图队列队尾编号，h为父地图
第一次失败案例：

- 失败原因：生成map[t]后，为节约时间，与map[h]及以前地图比较判重，答案几十分钟跑不出来。
- 初步判断：生成大量重复地图。
- 调试程序：每次生成新地图后输出队列编号及当前所需步数，发现总步数为16左右已经生成了60万幅地图了。
- 检查判重是否有效：导出华容道程序生成的前10000幅地图，另外写一个软件判断这些地图有无重复，发现大量重复。
- 解决方法：生成map[t]后，与map[t-1]及以前地图比较判重，答案秒出。

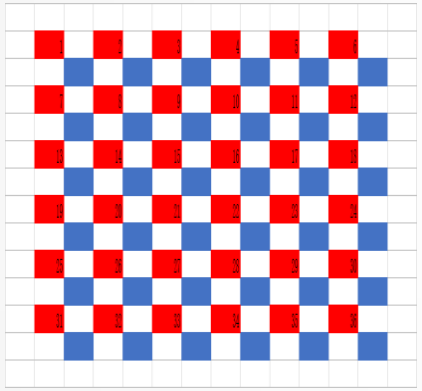


连连看

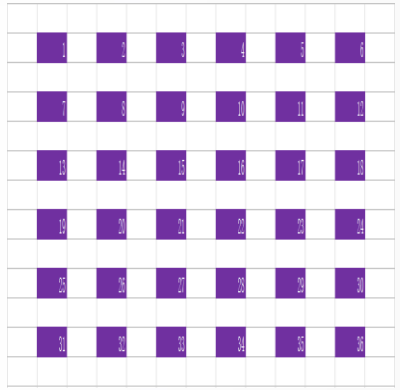
算法组

1. 物理模型转换为数学模型

物理模型



数学模型



从具体的起点终点转化为存有起点终点信息的图

算法组

2. 搜索过程数据存储

对起点按顺序进行编号

- 终点坐标
- 到终点的距离length[]
- 以该点为起点的贪心最优解f[]
- 地图信息

3.移动过程输出

搜索：

每次在当前指针节点周围寻找当前状态转移到下一个状态的最小距离，将指针指向当前状态转移到下一个状态的最小距离的节点，并重复进行如上搜索。方程如下：

f[beginning] = f[beginning] + min{ distance(now,i) + length[i] }

（其中，distance(now,i)表示点now与点i之间的距离，这里我们使用哈夫曼距离）

输出：

在搜索完成后，比较各个起点的贪心最优解的最小距离，找出最小距离，并输出以该节点为起点的经过各点路径。

算法&嵌入式

无人机算法实现

向右飞

为什么我程序写了向右飞，但飞机升空后不动呢？

我在起飞
无法执行命令

收到命令

算法&嵌入式

无人机算法实现

等待3秒

向右飞

成功了

收到命令

收到命令

问题总结

合作效率不高

人员安排不够合理，出现资源闲置

东西摆放较为散乱，找工具花费大量时间

总结

感谢聆听！