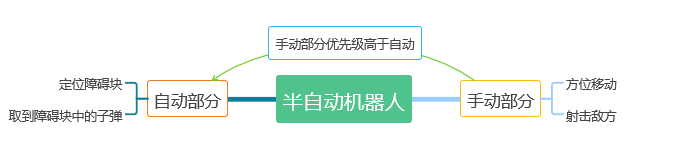
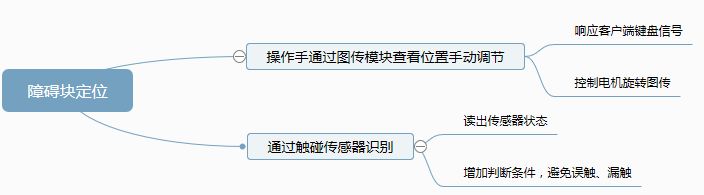
## Robomaster 2017 高中生夏令营 学习和技术报告

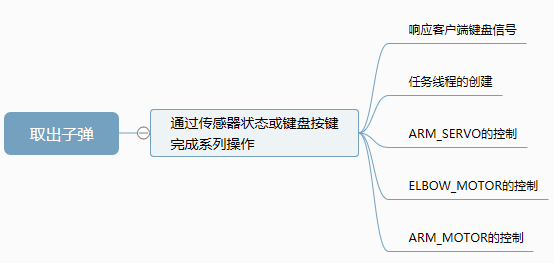
**高中1组 丁致远**

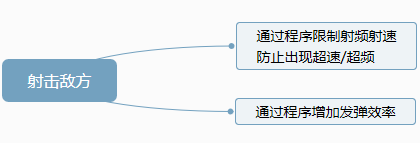
在本次夏令营的比赛中，我所在的高中1组获得了高中组亚军的成绩。本次夏令营七分钟比赛时间内，需要完成的主要任务有两个：一是从放置在“资源岛”的障碍块中取到弹药，二是在取到弹药后射击敌方。为了更高效地实现此次比赛的两个任务，本组进行了如下图所示的任务规划。



每个任务可以分为如下几个部分。







在这些任务的要求下，我担任了本组的嵌入式系统工作，负责控制各电机和伺服电机的运转(功率和角度)、键盘的逻辑操作、传感器数据的读入和相应、自动取子弹部分以及射击部分优化的编写。

利用此前有限的STM32系统板的开发基础和C++的应用软件编程经验，我与嵌入组的刘李正合作，完成了电机和伺服电机通过GPIO电平和PWM输出的方式，实现了控制电机旋转的方向和速度以及控制伺服电机的旋转角度，完成了机器人机械臂的单步操作，使用*camera\_turn*， *arm\_motor*，*arm\_lift*和 *elbow\_lift*变量的值使电机和舵机运动。然而将单步操作融合到原有线程的循环中，出现了不响应变量的情况。因此我们设置了临时变量s1~s9，分别临时存储带操作的数值。部分代码如下：

*if (camera\_turn)*

*s3 = 650;*

*else*

*s3 = 2050;*

*if (arm\_motor == 0)*

*{*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*TIM8->CCR2 = 700;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_RESET);*

*TIM8->CCR3 = 700;*

*osDelay(1600);*

*arm\_motor = 2;*

*}*

*if (arm\_motor == 1)*

*{*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_RESET);*

*TIM8->CCR2 = 700;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*TIM8->CCR3 = 700;*

*osDelay(1600);*

*arm\_motor = 2;*

*}*

*if (arm\_motor == 2)*

*{*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*s6 = 00;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*s7 = 00;*

*}*

*if (arm\_lift)*

*s1 = 1850;*

*else*

*s1 = 970;*

*if (elbow\_lift)*

*s2 = 550;*

*else*

*s2 = 1500;*

*TIM2->CCR1 = s1;*

*TIM2->CCR2 = s2;*

*TIM2->CCR3 = s3;*

*TIM2->CCR4 = s4;*

*TIM8->CCR1 = s5;*

*TIM8->CCR2 = s6;*

*TIM8->CCR3 = s7;*

*TIM8->CCR4 = s9;*

为了使操作手在操作时有更好的操作体验，我将取子弹的操作合为一个，当*test*变量为真时，机器人便会自动开始收取子弹的流程。

*if (test)*

*{*

*TIM2->CCR2 = 550; //elbow\_lift = 1*

*osDelay(1000);*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_RESET);*

*TIM8->CCR2 = 700;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*TIM8->CCR3 = 700;*

*osDelay(1650);*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*s6 = 00;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*s7 = 00;*

*osDelay(1200);*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_SET); //arm\_motor*

*TIM8->CCR2 = 700;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_RESET);*

*TIM8->CCR3 = 700;*

*osDelay(300);*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_SET); //arm\_motor*

*TIM8->CCR2 = 00;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_SET);*

*TIM8->CCR2 = 00;*

*osDelay(200);*

*TIM2->CCR2 = 1500; //elbow\_lift = 0*

*osDelay(500);*

*HAL\_GPIO\_WritePin(GPIOC, GPIO\_PIN\_5, GPIO\_PIN\_SET); //arm\_motor*

*TIM8->CCR2 = 700;*

*HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_RESET);*

*TIM8->CCR3 = 700;*

*osDelay(1250);*

*TIM2->CCR1 = 1850;*

*arm\_lift = 0; //restore to wait the next operation*

*elbow\_lift = 0;*

*camera\_turn = 0;*

*test = 0;*

*}*

同时，我在设置键盘操作时尽量简化操作手的记忆和操作难度，将按键数量由6（X，Q，C，V，B，CTRL）个缩减为3个（X，Q，C），按下SHIFT可以取消相应操作。

*if (rc.kb.bit.SHIFT && !rc.kb.bit.CTRL)*

*{*

*if (rc.kb.bit.X)*

*arm\_lift = 0;*

*if (rc.kb.bit.C)*

*{*

*arm\_lift = 0; //restore all*

*elbow\_lift = 0;*

*camera\_turn = 0;*

*test = 0;*

*}*

*if (rc.kb.bit.Q)*

*camera\_turn = 0;*

*}*

*else if (!rc.kb.bit.SHIFT && !rc.kb.bit.CTRL)*

*{*

*if (rc.kb.bit.X)*

*arm\_lift = 1;*

*if (rc.kb.bit.C)*

*test = 1;*

*if (rc.kb.bit.Q)*

*camera\_turn = 1;*

*}*

在实际测试中，我们发现机器人的子弹射击经常超频和超速，从而受到裁判系统的处罚。其中有在机械发弹结构的原因，也有在操作手操作时鼠标点击频率过高的原因。为了防止后者造成的违规，我在原有的发弹程序中加入了时间限制，单发子弹间隔不得超过200ms。同时，我加入了右键连发功能，当操作手按下右键时，可以连续发出4颗子弹，每次连发间隔不得低于1000ms。如果操作手操作过于频繁，程序会等待200ms再响应刚才的操作，达到了防止超频的目的。

*if (fric\_wheel\_run)*

*{*

*if (rc.sw1 == RC\_DN || left\_key)*

*{*

*if (HAL\_GetTick() - shoot\_time\_begin\_l > 200*

*&& HAL\_GetTick() - shoot\_time\_begin\_r > 800)*

*{*

*trigger\_pos\_ref = moto\_trigger.total\_ecd;*

*trigger\_pos\_ref += 130922 \* trigger\_dir;*

*shoot\_time\_begin\_l = HAL\_GetTick();*

*}*

*else*

*{*

*osDelay(200);*

*trigger\_pos\_ref = moto\_trigger.total\_ecd;*

*trigger\_pos\_ref += 130922 \* trigger\_dir;*

*shoot\_time\_begin\_l = HAL\_GetTick();*

*}*

*}*

*if (right\_key)*

*{*

*if (HAL\_GetTick() - shoot\_time\_begin\_l > 200*

*&& HAL\_GetTick() - shoot\_time\_begin\_r > 800)*

*{*

*trigger\_pos\_ref = moto\_trigger.total\_ecd;*

*trigger\_pos\_ref += 130922 \* 4 \* trigger\_dir;*

*shoot\_time\_begin\_r = HAL\_GetTick();*

*}*

*else*

*{*

*osDelay(200);*

*trigger\_pos\_ref = moto\_trigger.total\_ecd;*

*trigger\_pos\_ref += 130922 \* 4 \* trigger\_dir;*

*shoot\_time\_begin\_r = HAL\_GetTick();*

*}*

*}*

后期，算法组需要从灰度传感器读入几个模拟量，而此时我们正在做触碰传感器的功能。组委会为我们提供了一块通过CAN通信的一块基于STM32F1的开关板。因为时间紧迫，我们也没有成员有STM32直接读入模拟数据或者CAN通信的经验，所以我决定使用一块Arduino Mega 2560兼容板来读入各个传感器的数据，然后通过UART发送给STM32，从而获得数据并将数据存储在*arduino\_buff*。串口部分代码如下：

*#define AUTOP\_SIZE 7*

*#define SEND\_SIZE 28*

*tSendTXoneData SendData;*

*tReceArduinoData ReceData;*

*uint8\_t arduino\_buff[AUTOP\_SIZE+1];*

*uint8\_t auto\_send[SEND\_SIZE+1];*

*void arduino\_uart\_init()*

*{*

*\_\_HAL\_UART\_CLEAR\_IDLEFLAG(&ARDUINO\_HUART);*

*\_\_HAL\_UART\_ENABLE\_IT(&ARDUINO\_HUART, UART\_IT\_IDLE);*

*UART\_Receive\_DMA\_No\_IT(&ARDUINO\_HUART, arduino\_buff, AUTOP\_SIZE);*

*}*

*void MyUartFrameIRQHandler(UART\_HandleTypeDef \*huart)*

*{*

*if (huart == &ARDUINO\_HUART)*

*{*

*memcpy(&ReceData, arduino\_buff, sizeof(tReceArduinoData));*

*}*

*uart\_reset\_idle\_rx\_callback(huart);*

*}*

*void uart\_send\_data(UART\_HandleTypeDef \*huart)*

*{*

*memcpy(auto\_send, &SendData, sizeof(tSendTXoneData));*

*HAL\_UART\_Transmit(huart, auto\_send, sizeof(tSendTXoneData), 30);*

*SendData.sof = 123;*

*}*

Arduino部分代码如下：

*void setup() {*

*Serial3.begin(115200);*

*}*

*int mydigitalRead(int a);*

*void loop() {*

*unsigned int data[4];*

*digitalWrite(LED\_BUILTIN, HIGH);*

*data[1] = analogRead(A6);*

*data[2] = analogRead(A9);*

*data[3] = mydigitalRead(A15);*

*data[4] = mydigitalRead(A5);*

*if (data[1] < 100){*

*Serial3.print(0);*

*}*

*Serial3.print(data[1]);*

*if (data[2] < 100){*

*Serial3.print(0);*

*}*

*Serial3.print(data[2]);*

*Serial3.print(data[3]);*

*Serial3.println(data[4]);*

*delay(50);*

*digitalWrite(LED\_BUILTIN, LOW);*

*delay(50);*

*}*

*int mydigitalRead(int a){*

*if (analogRead(a) < 950) return(0);*

*else return(1);*

*}*

在本次夏令营中，我运用了此前的STM32系列单片机的简单基础、C++程序的开发经验、Arduino系列开发板的应用以及基于Ubuntu 和Windows IoT的嵌入式系统应用经验，通过此段时间的学习，应用了新学习的STM32单片机的GPIO电瓶和PWM输出、定时器TIM的配置和UART通信，完成了一次较大型的C/C++嵌入式系统程序开发，基本完成了此次夏令营中我所负责的工作，实现了比赛中需要达到的目的。