

HAL-Template

1 简介

- 开发工具: Keil V5.38a, VsCode
- 软件环境: Window11
- 硬件环境: 大疆RoboMaster开发板C型 (STM32F407IGHX)
- 编译工具: Arm Compiler V5.06u7, C/C++编译

2 目录结构

```
HAL-Template
├── Application/API
├── Application/Tasks
├── BSP
├── Components/Algorithm
├── Components/Controller
├── Components/Device
├── Core
├── Docs
├── Drivers
├── MDK-ARM
├── Middlewares
├── Third_Party
└── USB_DEVICE
```

3 模块功能说明

3.1 IMU 惯性测量单元

- 模块参考[哈尔滨工程大学创梦之翼战队惯导姿态解算项目](#)。
- 详情见[Quaternion](#)。

适配常见问题:

1. STM32CubeMX添加DSP库

- (a) 点击[Software Packs]/[Select Components], 在弹出的[Software Packs Component Selector]窗口中, 勾选[STMicroelectronics.X-CUBE-ALGOBUILD]/[DSP Library Library]/[DSP Library 1.3.0];
- (b) 关闭[Software Packs Component Selector]窗口, 在[Middle and Software Packs]/[X-CUBE-ALGOBUILD]栏勾选[DSP Library Library];
- (c) 此时在工程中默认添加的LIB文件为 `arm_cortexM4l_math.lib` (Little endian on Cortex-M4), 而实际需求为 `arm_cortexM4lf_math.lib` (Little endian and Floating Point Unit on Cortex-M4), 后者支持浮点单元。

2. malloc函数内存申请失败

在startup_stm32f407xx.s中分配的堆空间只有0x0200个字节，而在初始化扩展卡尔曼时所申请的空间超过了0x0200，需要在STM32CubeMX的[Project Manager]/[Project]/[Linker Settings]栏修改Minimum Heap Size的值以达到使用需求，修改后可在startup_stm32f407xx.s文件中的Heap_Size体现。

3.2 MiniPC通信

- 使用MicroUSB连接STM32和上位机
- 在./Device/Src/minipc.c中封装了适配rm_serial_driver的数据交互函数

– void MiniPC_RecvFrameInfo(uint8_t* Buf, uint32_t *Len)

函数在Application/User/USB_DEVICE/App/usbd_cdc_if.c的

static int8_t CDC_Receive_FS(uint8_t* Buf, uint32_t *Len)

函数中调用,实现了上位机数据的接收。

– void MiniPC_SendFrameInfo(MiniPC_SendPacket_Typedef *SendPacket)

函数应在RTOS任务中以500Hz的频率实现下位机数据的发送。

3.3 弹道解算

- 模块参考弹道解算。
- 在./Application/Tasks/src/Vision_Task.c中存在待测参数:
 - Camera_Yaw_Vertical: 相机相对yaw轴的垂直距离, 单位/m;
 - Camera_Yaw_Horizontal: 相机相对yaw轴的前推距离, 单位/m;
 - FireSystem_BiasTime: 系统造成的总延时: 通信延时和击发延时等, 单位/s;
- 解算部分在./Application/Tasks/Src/Vision_Task.c中以500Hz的频率进行,

```
void SolveTrajectory_Update(SolveTrajectory_Typedef *SolveTrajectory,
                           float picth,
                           float yaw,
                           float target_yaw,
                           float v_yaw,
                           float r1,
                           float r2,
                           float dz,
                           float bullet_speed,
                           float armor_num)
```

函数更新弹道解算参数时, 请根据云台RoboMaster开发板C型的安装位置调整输入的位姿数据, 即

在./Application/API/Inc/config.h中修改IMU reslove constants栏相关宏定义, 此处与云台控制相对应, 若后者正常则无需修改。

```
void SolveTrajectory_Transform(MiniPC_SendPacket_Typedef *MiniPCTxData,
                               MiniPC_ReceivePacket_Typedef *MiniPCRxData,
                               SolveTrajectory_Typedef *SolveTrajectory)
```

函数解算得出云台期望姿态。

3.4 裁判系统

- 使用STM32串口DMA实现裁判系统数据交互

- 数据接收使用双缓存循环模式
- 数据发送使用单缓存普通模式

- 在 `./Components/Device/Src/Referee_info.c` 中封装了适配[RoboMaster 裁判系统串口协议附录 V1.4](#)及[RoboMaster 裁判系统串口协议附录 V1.4 增补修订说明](#)的通信解码函数，其在 `./BSP/bsp_uart.c` 中的

```
void HAL_UARTEx_RxEventCallback(UART_HandleTypeDef *huart, uint16_t Size)
```

串口传输完成回调函数中调用，以实现裁判系统的数据更新。

- 在 `./Components/Device/Src/ui.c` 中封装了适配[RoboMaster 裁判系统串口协议附录 V1.4](#)的客户端绘制UI所需通信编码函数，用户可调用此类函数实现客户端绘制自定义UI的通信数据帧编码，并利用

```
HAL_StatusTypeDef HAL_UART_Transmit_DMA(UART_HandleTypeDef *huart, const uint8_t *pData, uint16_t Size)
```

函数以串口DMA的形式，向裁判系统链路发送相关通信数据帧。

4 贡献

- 完善项目过程中，请尽量遵循以下设计原则和规范：

- [API](#) 应用接口层对应应用接口，是一类功能的抽象，请不要在该层相关文件中定义实体变量；该层调用各组件层以实现功能；
- [Bsp](#) 板级支持包面向底层组件，是唯一允许直接出现STM32HAL库函数的代码层；
- 请不要跨层调用；
- 请注意[代码规范](#)，建议参考[Google C++风格指南](#)。

- 欢迎提交Issues和Pull Requests帮助我们改进。