通讯主控板及SDK说明文档

本文档的2和3在出厂前已经配置好，如果需要进行二次开发的人员可以根据2和3的说明进行配置

1.环境依赖

1.1 使能嵌入式板子(香橙派)的SPI, 在Linux系统的终端里面执行以下命令：

```
sudo vim /boot/orangepiEnv.txt

//进入文件后，在最后添加
overlays=spi4-m0-cs1-spidev

//保存后重启
sudo reboot
```

1.2 安装cmake和pip，执行以下命令

```
sudo apt update
sudo apt install cmake
sudo apt install python3-pip
```

1.3 搭建python SDK环境，执行以下命令

```
//打开".\LivelyMotorControl\SDK\Python"
git init
git submodule add https://gitee.com/mysticalwing/pybind11.git
third_party/pybind11-2.5.0
cd third_party/pybind11-2.5.0/
git checkout tags/v2.5.0
```

2.配置通讯主控板和SDK(出厂已执行)

打开".\LivelyMotorControl\Core\Inc\Config.h" 和 ".\LivelyMotorControl\SDK\include\Config.h"两个文件，可以看到以下的配置信息，我们需要确保两个文件的配置信息相同(一般在出厂时会将其设置为对应的机器人的配置)

```
//用户配置区
#define CAN1_NUM            6     //CAN1的电机数量
#define CAN2_NUM            6     //CAN2的电机数量
#define ENABLE_IMU          1     //是否使能IMU
#define ENABLE_FOOTSENSOR   1     //是否使能足底传感器(只有双足机器人有这个传感器)
```

```
#define ENABLE_STOP         1     //是否使能通讯断开让所有电机停在当前位置(即当上层嵌入式板子和通讯主控板通讯停止时，是否让当前所有电机维持速度为0的转态)
```

3. 编译通讯主控板并烧录(出厂已执行)

3.1 先确保本地电脑已经安装了MDK5以及STM32H730的硬件库.
3.2 接着，使用MDK5打开".\LivelyMotorControl\Core\Inc\Config.hMDK-ARM\LivelyMotorControl.uvprojx"
3.3 使用MDK5进行编译然后进行烧录

4. 编译C++版本的SDK

4.1 首先将"\LivelyMotorControl\SDK"文件夹移到嵌入式板子里(香橙派)
4.2 创建build文件夹，进入build文件夹里面，使用以下指令：

```
1. cmake ..
2. make
```

中途没有报错则编译通过！

5. 测试C++版本的SDK

5.1 先确保电机在CAN1线id为1，并且能够自由转动。
5.2 进入"SDK\build"文件夹下，使用下面的指令执行example程序

```
sudo ./example
```

5.3 执行完毕后，你可以看到类似如下的输出，

```
motor_pos: 97520    //电机反馈的位置
accX:-5, accY:12, accZ:2021   //imu加速度值
angVelX:0, angVelY:0, angVelZ:0 //imu角加速度值
angle_roll:62, angle_pitch:23, angle_yaw:-26919 //imu的横滚角，俯仰角，偏航角的值
magX:-2247, magY:-3607, magZ:-17580 //imu的磁偏角
```

数值正常，电机在转动则测试正常!
5.4 查看测试源码，打开"SDK\example\example.cpp"文件，可以看到下面代码

```
//创建Liveltbot_Driver对象,传入参数
Livelybot_Driver my_Driver("/dev/spidev4.1");

//设置can1线id为1的电机的位置
my_Driver.set_motor_position(0x11, i);

//发送指令
```

```
my_Driver.spi_send();

//获得can1线id为1的电机的状态
motor_state = my_Driver.get_motor_state(0x11);
printf("motor_pos: %d\n", motor_state.motor.position);

//获得imu的数值
imu_data = my_Driver.get_imu_data();
printf("accX:%d, accY:%d, accZ:%d\nangVelX:%d, angVelY:%d,
angVelZ:%d\nangle_roll:%d, angle_pitch:%d, angle_yaw:%d\nmagX:%d, magY:%d,
magZ:%d\n",
imu_data.accX, imu_data.accY, imu_data.accZ, imu_data.angVelX, imu_data.angVelY,
imu_data.angVelZ,
imu_data.angle_roll, imu_data.angle_pitch, imu_data.angle_yaw, imu_data.magX,
imu_data.magY, imu_data.magZ);
```

更详细的用法请参考".\LivelyMotorControl\Doc\sdk_api.md"的说明。

6. 编译Python版本的SDK

6.1 打开".\LivelyMotorControl\SDK\Python"文件夹，执行以下命令：

```
mkdir build
cd build
cmake ..
make
```

6.2 编译成功后，将build文件夹下生成的.so文件移到"SDK/example-python"文件夹中,直接执行example.py文件, 执行完毕后，可以看到类似如下的输出：

```
motor_pos: 97520    //电机反馈的位置
accX:-5, accY:12, accZ:2021   //imu加速度值
angVelX:0, angVelY:0, angVelZ:0 //imu角加速度值
angle_roll:62, angle_pitch:23, angle_yaw:-26919 //imu的横滚角，俯仰角，偏航角的值
magX:-2247, magY:-3607, magZ:-17580 //imu的磁偏角
```

可通过".\LivelyMotorControl\Doc\sdk_api.md"去查看API的使用