目录

修订记录

修订版本	修订时间	作者	备注
V1.0	2020.8.7	谭	初始版本
V2.0	2021.4.14	谭	

1. 软件安装

参考根目录 README.md

2. 启动软件

2.1 打开 can 通信

终端输入: rosrun canopen_communication can_prepare.sh

输入系统密码，即可通信。若没有连接 can 卡，将出现如下错误：

Cannot find device "can0"

Cannot find device "can0"

SIOCGIFINDEX: No such device

通讯测试：

另外打开终端，输入 cansend can0 123#112233，查看上述终端是否显示该 can 帧，显示表示 usb-to-can 驱动成功，通讯正常。

2.2 打开软件界面

打开新的终端输入：roslaunch ui ui.launch

至此，若无报错，软件成功打开。

3. 软件功能使用

3.1 机器人选择界面

点击机器人按钮，进入相应的机器人控制界面。

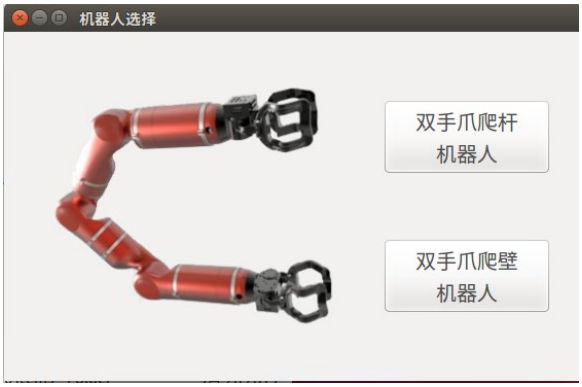

图 2-1 机器人选择界面

3.2 机器人控制主界面

机器人控制主界面包含关节位置控制、关节速度控制、笛卡尔空间位置控制和离线轨迹四种模式。如图 2-2 所示，机器人初始状态处于“未使能状态”，点击启动按钮，机器人状态将转换为“机器人使能中”，等待五秒左右，机器人状

态将转换为“机器人使能成功”。另外，可通过暂停、停止和急停按钮控制机器人运动。

3.2.1 关节空间位置控制

如图 2-2 所示，机器人默认关节速度为 5 度每秒，速度调节范围限制在 1~30 度每秒。通过输入各关节对应的关节位置目标，点击相应的运行按钮，机器人关节及产生运动。

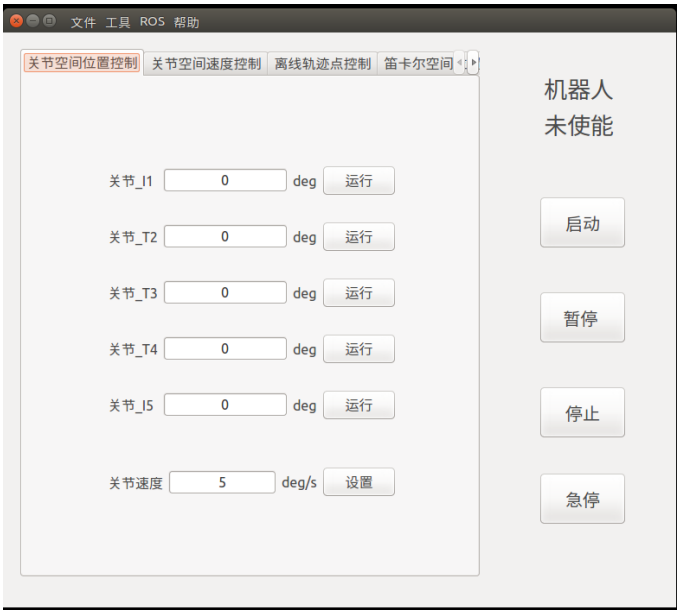

图 2-2 机器人关节位置控制界面

3.2.2 关节空间速度控制

如图 2-3 所示，机器人默认关节速度为 5 度每秒，速度调节范围限制在-30~30 度每秒。点击相对关节运行按钮，机器人将以设定速度运行。再次点击运行按钮，机器人立刻停止。

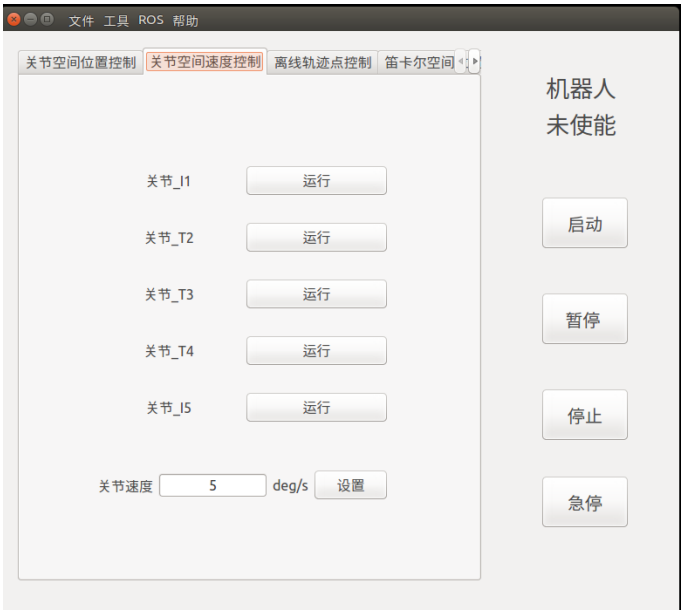

图 2-3 机器人关节速度控制界面

3.2.3 离线轨迹控制

如图 2-4 所示，通过复制相应路径点集合（如下述 example）到文本框，设置路径最大速度。点击载入文件按钮，界面将弹窗提示路径点格式是否正确。若格式正确，点击运行按钮，机器人将以指定路径运行。

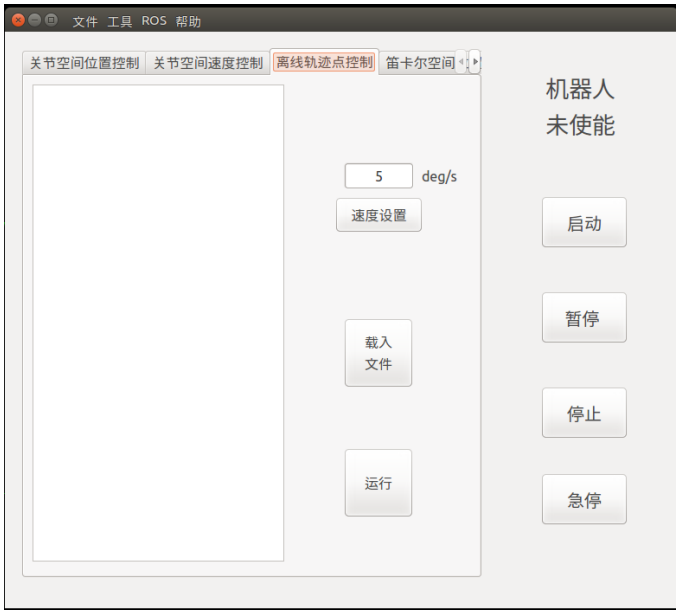

图 2-4 机器人离线轨迹控制界面

路径代码 example：

P=0,0,0,0,0,;

P=0.0,13.654,-0.0,15.605,0.0,;

P=0.0,2.774,-0.0,2.78,0.0,;

P=0.0,-14.26,-0.0,2.78,0.0,;

P=0.0,-14.26,-0.0,-26.912,0.0,;

P=0.0,0.0,-0.0,-0.0,0.0,;

3.2.4 笛卡尔空间位置控制

如图 2-5 所示，笛卡尔空间位置速度命令默认为机器人工作初始位姿。在机器人工作空间范围内，选择合适的笛卡尔位置速度命令；点击获取逆解按钮；窗口将显示关节位置、速度命令，确认安全无误，点击运行按钮。

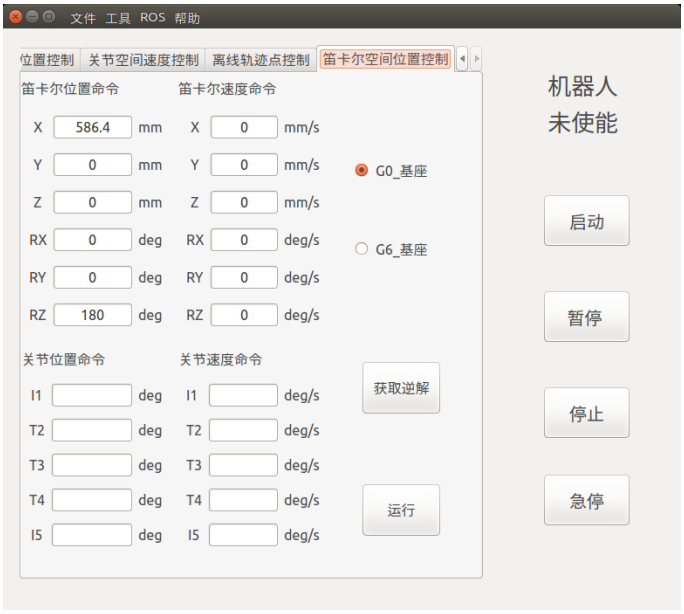

图 2-5 机器人笛卡尔空间控制界面

3.2.5 笛卡尔空间增量控制

如图 2-6 所示，为机器人笛卡尔增量控制界面，相对于基坐标系。通过文本框设置相应的位置、姿态位移增量以及速度，点击各轴增量加减按钮，机器人即可产生相对应的笛卡尔轴向运动。

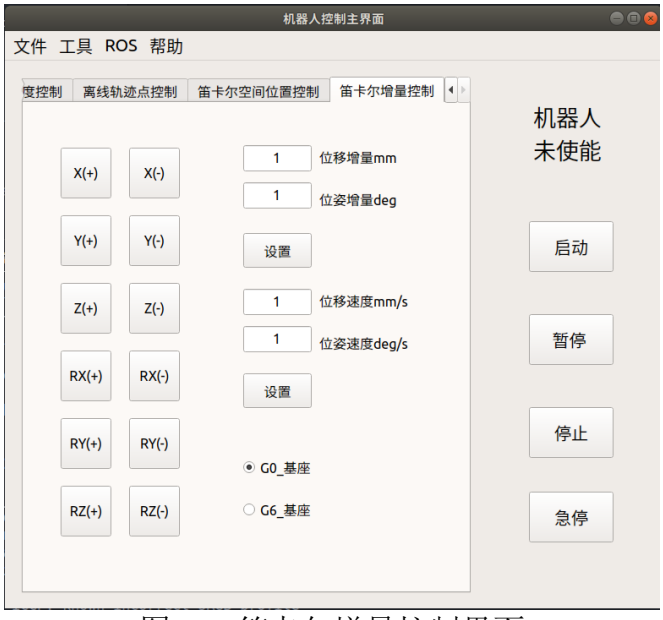

图 2-6 笛卡尔增量控制界面

3.3 机器人工具栏

3.3.1 夹持器控制

点击工具 → 夹持器。如图 2-7 所示，夹持器电流默认 200mA，右侧文本动态显示夹持器控制电流。点击开启按钮，将使能电流控制打开相应夹持器，再次点击开启按钮，夹持器停止使能电流，即夹持器停止运动。

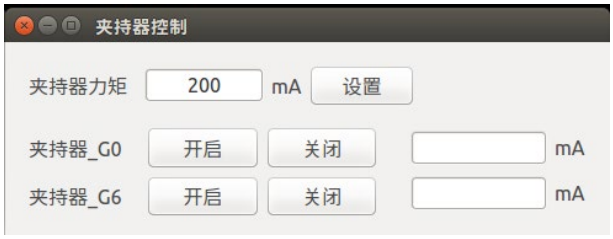

图 2-7 夹持器控制界面

3.3.2 机器人示教记录

点击 工具→ 示教记录。如图 2-8 所示，机器人运动到待记录位置，点击插入点按钮，左侧文本将显示插入路径点。添加路径点过程中，可通过删除点按钮删除错误采样点。采样路径完毕，通过左下侧文本框输入保存路径文件名，点击保存按钮，文件将保存在 third_modular_robot/ui/path_point_file/目录下。

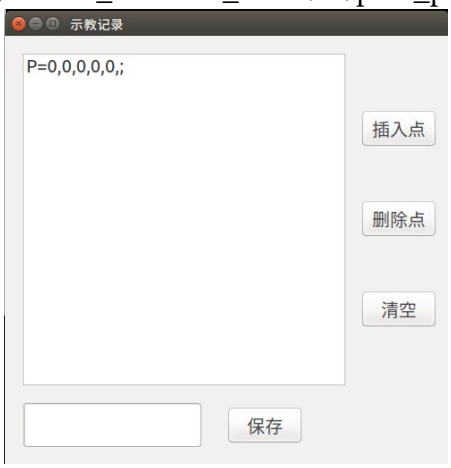

图 2-8 机器人示教记录界面

3.3.3 机器人状态显示

如图 2-9 所示，机器人启动成功后，点击 工具→机器人状态反馈，该界面动态显示笛卡尔位姿、关节位置、关节速度和关节电流状态。

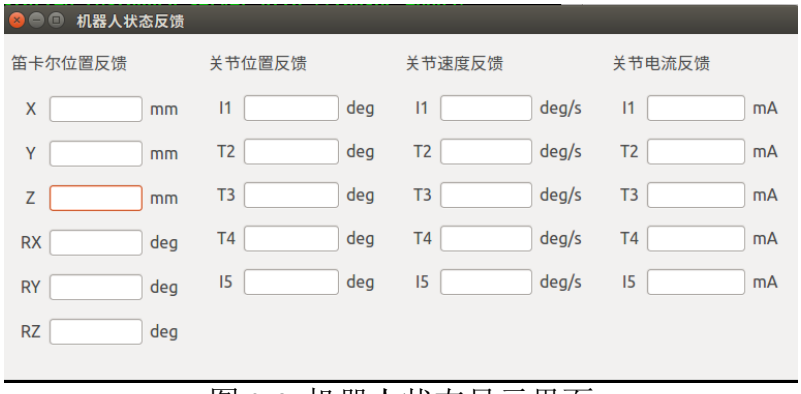

图 2-9 机器人状态显示界面

3.3.4 零点设置

点击 工具→ 零点设置，如图 2-10 所示。点击获取当前位置按钮，关节文本框将更新关节位置。关节运动方向默认为正方向。若机器人关节运动方向与预

期相反，可通过正向选择框更改运动方向。确认机器人零点位置后，点击设置按钮设置关节零点。

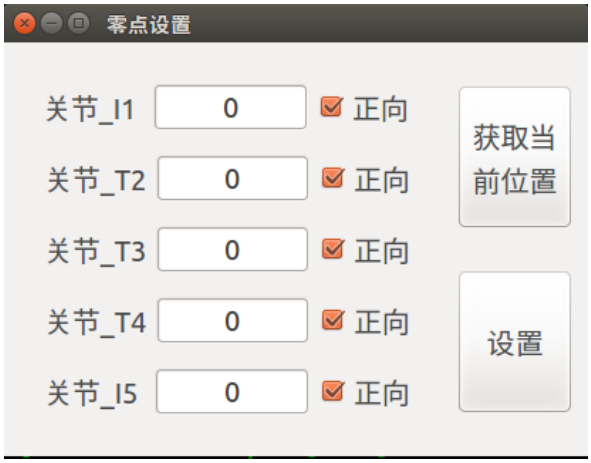

图 2-10 机器人零点设置界面

3.3.5 自主抓夹

点击 工具→ 自主抓夹，如图 2-11 所示。

界面功能有效前提：杆件位姿捕捉程序需要通过 Ros topic（/detection/pole model）发布，话题消息类型为 Float64MultiArray，数据格式为[p1_x,p1_y,p1_z, p2_x,p2_y,p2_z] → 杆件两点位姿， 单位 mm。

订阅话题程序：ui/script/auto_gripper_get_pole_point_thread.py

点击获取夹持点按钮，后台将计算抓夹点 XYZ，RxRyRz，显示在对应的文本框中。根据抓夹点，可先点击调整姿态按钮，后台将根据夹持器姿态计算机器人各个关节位移，确认关节位移无误后，点击发送控制命令，机器人将产生运动。点击调整位置按钮，机器人后台将计算对应的关节位移，确认关节位移无误后，点击发送控制命令按钮。

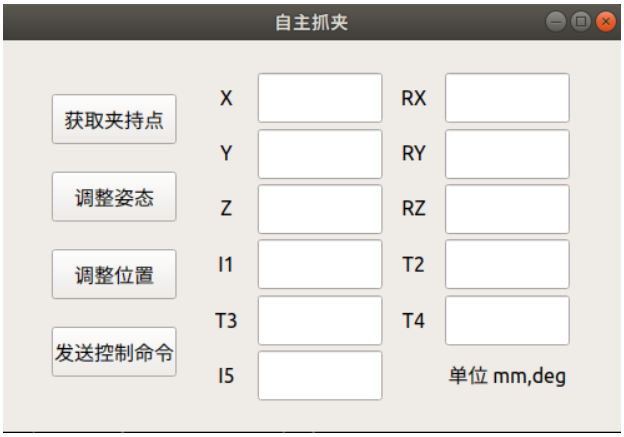

图 2-11 机器人自主抓夹界面

手眼标定结果设置文件：birl_modular_robot/file/eye_hand_calibration_consquence.json

自主抓夹后台测试文件：birl_modular_robot/src/test.cpp，

自主抓夹后台测试命令：rosrun birl_modular_robot find_grasp_point_test

4. ROS 控制

4.1 Ros 控制命令

1. 机器人关节控制命令消息类型 third_modular_robot/ui/msg/robot_command.msg：

```
# 时间戳
Header timeHeader
# 关节位置命令 I1 T2 T3 T4 I5
float32[] CommandPosData
# 关节速度命令
float32[] CommandVelData
```

2. 机器人接收话题：/low_level/joints_point_command

采用上述消息类型、话题；控制程序可编写相对应的 ros 话题发布器进行机器人控制。点击工具栏 ROS → 控制命令。如图 2-12 所示，ros 发布器发布出机器人控制命令，将显示在文本框中。通过删除按钮，可删除相关错误控制命令。清空按钮可清空接收的所有的控制命令。确认控制命令安全无误后，点击发送控制命令按钮控制机器人运动。

测试脚本：/ui/test/test_pub_joints_point_command.py,

测试命令：rosrun ui test_pub_joints_point_command.py

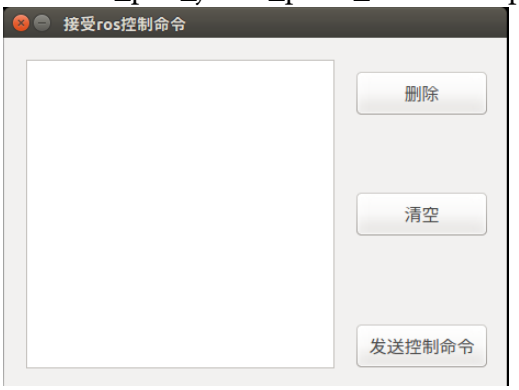

图 2-12 机器人接受 ros 命令界面

4.2 Ros 状态反馈

反馈消息类型，third_modular_robot/ui/msg/robot_feedback.msg：

```
# 时间戳
Header timeHeader
# 反馈关节位置数据 I1 T2 T3 T4 I5
```

float32[] feedbackPosData

反馈关节速度数据

float32[] feedbackVelData

反馈关节电流数据

float32[] feedbackCurrData

ros 状态反馈系统默认关闭状态，通过工具栏 ROS → 勾选状态反馈即可打开。

机器人状态消息订阅话题：/low_level/joints_point_feedback

5. Qt 界面

5.1 Qt 添加功能界面

1. 使用 qtcreator 新建界面文件，添加或删除控件。

2. 将 ui 文件转为 python 文件。

命令: sudo pyuic5 -o 输出 python 文件 ui 源文件。

示例： sudo pyuic5 -o modular_robot_control.py modular_robot_control.ui

生成的 *.py 放在 third_modular_robot/ui/script, *.ui 放在 third_modular_robot/ui/ui_file/.

5.2 Qt 机器人信号控制

信号位置：third_modular_robot/ui/script/modular_robot_control_func.py

1. 机器人关节位置模式控制信号：sin_joint_position = pyqtSignal(list)

信号内容格式:

Python 列表 : [temp_pos_command, temp_velocity_command]

temp_pos_command ：关节控制位置列表（I1, T2, T3, T4, I5）,单位角度。

temp_velocity_command ：关节控制速度列表（I1, T2, T3, T4, I5）,单位度每秒。

2 机器人关节速度模式控制信号：sin__joint_velocity = pyqtSignal(list)

信号内容格式：

Python 列表： 关节控制速度列表（I1, T2, T3, T4, I5）,单位度每秒。

6. 文件目录结构

```
├── install.sh（程序依赖安装脚本）

├── birl_modular_robot（运动学、自主抓夹）
│   ├── CMakeLists.txt
│   ├── file(手眼标定文件)
│   │   ├── CamInToolPose_G0_needrotation.dat
│   │   ├── CamInToolPose_G6_need_rotation.dat
│   │   ├── eye_hand_calibration_consquence.json（手眼标定配置文件）
│   │   └── README.md
│   ├── msg
│   │   └── pole_position.msg
│   ├── package.xml
│   ├── script
│   │   ├── find_grasp_point_client.py（自主抓夹 ros 客户端）
│   │   ├── inverse_solution_client.py（机器人运动学逆解客户端）
│   │   ├── positive_solution_client.py（机器人运动学正解客户端）
│   │   ├── robot_increTransTool_client.py
│   │   └── test_pub_pole_position_msg.py
│   ├── src
│   │   ├── find_grasp_point_server.cpp（自主抓夹 ros 服务器）
│   │   ├── grasp_intelligent.cpp（自主抓夹实现代码）
│   │   ├── grasp_intelligent.h（自主抓夹实现代码）
│   │   ├── inverse_solution_server.cpp（机器人运动学逆解 ros 服务端）
│   │   ├── positive_solution_server.cpp（机器人运动学正解 ros 服务端）
│   │   ├── robot_increTransTool_server.cpp
│   │   └── test.cpp
│   └── srv
│       ├── grasp_point.srv
│       ├── inverse_solution.srv（逆解服务）
│       ├── positive_solution.srv（正解服务）
│       └── robot_increTransTool.srv

├── canopen_communication（canopen 底层通讯）
```

```
│   ├── CMakeLists.txt
│   ├── file
│   │   └── Copley.eds（copley 字典文件）
│   ├── modular
│   │   ├── modular_G100.py（夹持器通讯子类）
│   │   ├── modular_I100.py（I100 通讯子类）
│   │   ├── modular_I85.py（I85 通讯子类
│   │   ├── modular_joint.py（电机通讯基类）
│   │   ├── modular_T100.py（T100 通讯子类）
│   │   ├── modular_T85.py（T85 通讯子类）
│   ├── package.xml
│   └── scripts
│       ├── can_prepare.sh（can 卡启动文件）
│       └── test.py
├── CMakeLists.txt -> /opt/ros/melodic/share/catkin/cmake/toplevel.cmak
e

├── kinematics（运动学实现代码）
│   ├── Kine.cpp
│   ├── Kine.h
│   ├── Matrix.cpp
│   ├── Matrix.h
│   ├── README.md
│   ├── Robot.h
│   ├── Setup.cpp
│   └── Setup.h

├── manual（手册）
│   ├── manual.doc
│   └── manual.pdf
├── README.md

├── ui（交互界面）
│   ├── CMakeLists.txt
```

```
│   ├── file
│   │   ├── README.md
│   │   └── robot_state.json（机器人程序启动默认配置数据）
│   ├── launch
│   │   └── ui.launch（机器人交互界面启动文件）
│   ├── msg
│   │   ├── robot_command.msg（机器人控制命令消息类型）
│   │   └── robot_feedback.msg（机器人反馈状态消息类型）
│   ├── package.xml
│   ├── path_point_file
│   │   └── test_path.txt（轨迹记录文件）
│   ├── pic（图标文件）
│   │   ├── biped5d_.png
│   │   ├── climbot5d_.png
│   │   └── robot.ico
│   ├── script（界面实现文件）
│   │   ├── auto_gripper_func.py（自主抓夹界面实现文件）
│   │   ├── auto_gripper_get_grasp_point_thread.py（获取抓夹点线程）
│   │   ├── auto_gripper_get_pole_point_thread.py（获取杆件位姿线程）
│   │   ├── auto_gripper.py（自主抓夹界面文件）
│   │   ├── biped_receive_control_command_func.py（爬壁机器人接受控制命令界面实现文件）
│   │   ├── biped_receive_control_command.py（爬壁机器人接受控制命令界面文件）
│   │   ├── get_fifo_relative_position_thread.py
│   │   ├── get_inverse_solution_thread.py（获取逆解线程）
│   │   ├── get_positive_solution_thread.py（获取正解线程）
│   │   ├── gripper_control_func.py（夹持器控制界面实现文件）
│   │   ├── gripper_control.py（夹持器控制界面文件）
│   │   ├── main.py（程序入口）
│   │   ├── modular_robot_control_func.py（控制软件主界面实现文件）
│   │   ├── modular_robot_control.py（控制软件主界面文件）
│   │   ├── path_point_recorder_func.py（示教记录界面实现文件）
│   │   ├── path_point_recorder.py（示教记录界面文件）
```

```
│   │   ├── path_process.py（离线轨迹处理）
│   │   ├── README.md
│   │   ├── receive_ros_command_func.py（接受 ros 轨迹命令界面实现文件）
│   │   ├── receive_ros_command.py（接受 ros 轨迹命令界面文件）
│   │   ├── receive_ros_command_thread.py（获取 ros 轨迹命令线程）
│   │   ├── robot_choice_func.py（机器人选择界面实现文件）
│   │   ├── robot_choice.py（机器人选择界面文件）
│   │   ├── robot_feedback_fun.py（机器人反馈状态界面实现文件）
│   │   ├── robot_feedback.py（机器人反馈状态界面文件）
│   │   ├── robot_lowlevel_control_muti_thread.py（机器人底层控制线程）
│   │   ├── zero_point_set_func.py（零点设置界面实现文件）
│   │   ├── zero_point_set.py（零点设置界面文件）
│   ├── test
│   │   └── test_pub_joints_point_command.py（机器人 ros 控制命令发布测试脚本）
│   └── ui_file（qt ui 设计文件）
│       ├── auto_gripper.ui（自主抓夹界面）
│       ├── biped_receive_control_command.ui（爬壁机器人接受控制命令界面）
│       ├── gripper_control.ui（夹持器控制界面）
│       ├── modular_robot_control.ui（主界面）
│       ├── path_point_recorder.ui（示教记录界面）
│       ├── README.md
│       ├── receive_ros_command.ui（获取 ros 控制命令界面）
│       ├── robot_choice.ui（机器人选择界面）
│       ├── robot_feedback.ui（机器人状态反馈界面）
│       └── zero_point_set.ui（零点设置界面）

└── usb-to-can_socketcan（usb-to-can 驱动文件）
    ├── ixx_usb_core.c
    ├── ixx_usb_core.h
    ├── ixx_usb_fd.c
```

```
├── ixx_usb_v2.c
└── Makefile
```