1. 安装和调试

参考资料包里的视频。

注：上电和关电，运行代码和中断代码运行前一定要先手轻轻扶住手柄，否则黑色柱体会掉落损坏。

1. SDK安装

这里是在虚拟机VMware上的ubuntu22.04系统下安装。

（1）将sdk-3.14.0-linux-x86\_64-gcc解压到ubuntu系统上，在sdk-3.14.0下有很多文件夹。

-****bin**** 包含编译生成的可执行文件以及一些二进制文件  
-****doc**** 说明文档以及许可  
-****example**** c++源代码  
-****external**** 软件开发工具包需要的一些外部库，例如Eigen  
-****include**** 头文件，主要是  
-****lib**** 静态库（.a）动态库（.so）

1. 测试手柄

·首先插上手柄电源线和usb线，并对手柄进行调试，使调试指示灯常亮

·进入bin文件夹，右键gravity -> 属性 -> 权限 -> 允许执行文件

·打开终端，进入bin文件夹，执行：sudo ./gravity

！！！执行前和终止程序前用手稍微扶住手柄

gravity例程是一个有重力补偿的显示末端执行器位置的程序（把手柄运动到某处松手，手柄保持不动，但在比较高的地方也会稍稍掉落）。该例程可以测试手柄是否运行正常。

1. 配置自己的功能包
2. 首先更改usb权限

这么做是为了可以正常运行ros节点，否则每次都要sudo su。





1. 创建一个新的功能包

·创建功能包

执行

ros2 pkg create omega7\_sensor build-type ament\_cmake --dependencies rclcpp std\_msgs

这样创建了一个新的功能包，当然，工作空间可以自己先创建一个。

·配置库函数

- 在功能包下建立lib文件夹，将SDK中lib文件夹下的四个文件复制过来，将两个带3.14.0的文件重命名为libdhd.so.3和libdrd.so.3。

- 将SDK中include文件夹下的两个.h文件复制到自己功能包下的include中。

- 将SDK中externals文件夹下的Eigen文件夹复制到自己功能包下的include中。（我是另下了一个，区别在于代码中引用库时不同）

·CmakeLists.txt修改

增加如下代码：

#添加为可执行文件，并使用install指令将其安装到install目录

add\_executable(omega7\_sensor src/omega7\_sensor.cpp)

ament\_target\_dependencies(omega7\_sensor rclcpp std\_msgs)

install(TARGETS

omega7\_sensor

DESTINATION lib/${PROJECT\_NAME}

)

include\_directories(

include ${catkin\_INCLUDE\_DIRS} # 添加头文件

)

link\_directories(

${catkin\_LIB\_DIRS} lib # 加入链接库的目录

)

find\_library(DHDLIB libdhd.so.3 lib) # 设置动态链接库环境变量

find\_library(DRDLIB libdrd.so.3 lib)

find\_library(USB1LIB libusb-1.0.so)

target\_link\_libraries(omega7\_sensor # 链接节点依赖的库

${DHDLIB}

${DRDLIB}

${USB1LIB}

${catkin\_LIBRARIES}

)

·引用库函数

#include "rclcpp/rclcpp.hpp"

#include "std\_msgs/msg/string.hpp"

#include "std\_msgs/msg/float64\_multi\_array.hpp"

#include "/home/yx/omega7\_sensor/src/omega7\_sensor/include/omega7\_sensor/dhdc.h"

#include "/home/yx/omega7\_sensor/src/omega7\_sensor/include/omega7\_sensor/drdc.h"

#include <stdio.h>

#include <stdlib.h>

#define \_USE\_MATH\_DEFINES

#include <math.h>

#include "/home/yx/omega7\_sensor/src/omega7\_sensor/include/eigen/Eigen/Eigen"

using namespace Eigen;

Note：单独#include dhdc.h我这里提示找不到文件，就用绝对路径了，需要修改，其他绝对路径同理。

1. 代码简单介绍
2. 主函数初始化并启动节点，检测ctrl c随时退出程序。
3. 运行节点，首先是一系列初始化过程，如果相应函数执行不成功，会打印错误信息，并退出程序；如果运行成功，会显示（可能是如下的信息，反正是你的设备被检测到了）：

Omega.7 right-hand haptic device detected

（3）

使手柄运动至中心位置（7个自由度都设为0）

drdMoveTo (nullPose);

将所有力置零，否则你运动手柄松手后手柄会产生力动起来。

drdSetForceAndTorqueAndGripperForce (0.0, 0.0, 0.0, // force

0.0, 0.0, 0.0, // torque

0.0); // gripper force

这里base和grip的值都是false，让基座和抓手可以自由移动，omega7好像不支持wrist相关操作

drdRegulatePos (base);

//drdRegulateRot (wrist);

drdRegulateGrip (grip);

（4）在终端打印相关信息

（5）建立发布者和定时器，每隔1ms进入回调函数timecaller

Note： 这里进入timecaller的周期应该小一些，如果太大，手柄无法实时回位，具体原因在后面解释。

1. timecaller中首先进行了一系列读取设备状态，计算回位的力和力矩，设置力和力矩的操作，以实现回位功能。

回位功能是当手柄运行至中心位置后，向任何位置移动手柄，松手后手柄会回到原位。

所以如果进入timecaller的周期太长，就无法实时计算，从而无法回位。

1. timecaller第二部分是打印信息至终端，可以更新终端的信息状态。主要是可以通过按下键盘的b锁住手柄的位置，同时终端更新手柄的状态为hold。
2. timecaller最后是获取手柄的位置，姿态，力，力矩信息，并通过话题发布出去。