

## 目录

1、 安装 CMake.....	2
1.1 Windows.....	2
1.2 Ubuntu.....	4
2、 安装 Visual Studio 2022.....	5
3、 编译工程（Windows） .....	5
3.1 方法一：使用命令行编译.....	5
3.2 方法二：打开 Visual Studio 编译.....	6
4、 编译工程（Ubuntu x64） .....	8
4.1 安装 cmake.....	8
4.2 安装 gcc.....	8
4.3 生成工程并编译.....	8
5、 编译工程（Ubuntu ARM x64） .....	9
6、 打开 MotionGlove.....	9
7、 打开例程测试.....	10
7.1 motionGloveSDK_example1.....	10
7.2 motionGloveSDK_example2.....	11
7.3 motionGloveSDK_example3.....	12
7.4 motionGloveSDK_example4.....	13

# 1、安装 CMake

## 1.1 Windows

在windows上安装cmake并  
添加到环境变量

打开网址

- <https://cmake.org/>

下载最新版安装

点击右上角DOWNLOAD

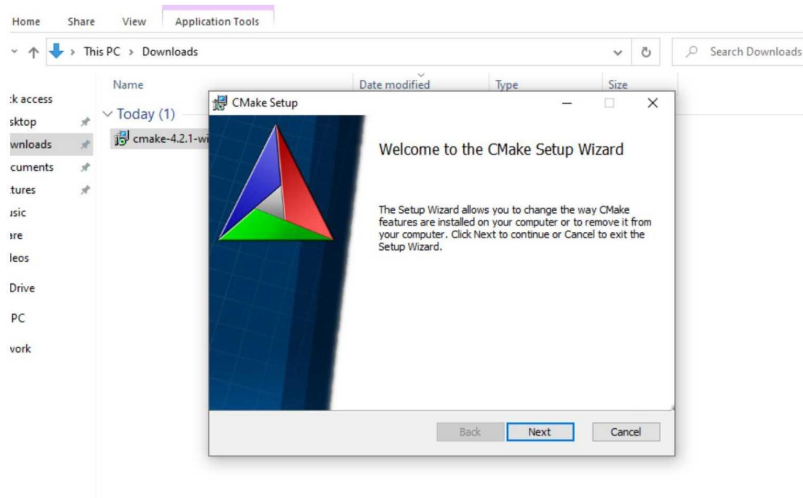


下载这个

Binary distributions:

Platform	Files
Windows x64 Installer:	<a href="#">cmake-4.2.1-windows-x86_64.msi</a>
Windows x64 ZIP	<a href="#">cmake-4.2.1-windows-x86_64.zip</a>
Windows i386 Installer:	<a href="#">cmake-4.2.1-windows-i386.msi</a>
Windows i386 ZIP	<a href="#">cmake-4.2.1-windows-i386.zip</a>
Windows ARM64 Installer:	<a href="#">cmake-4.2.1-windows-arm64.msi</a>
Windows ARM64 ZIP	<a href="#">cmake-4.2.1-windows-arm64.zip</a>

# 一直下一步直到安装完成



## 安装时，安装文件已经将cmake添加到了环境变量里

- 打开控制台，输入
- `cmake --version`

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\foheart>cmake --version
cmake version 4.2.1

CMake suite maintained and supported by Kitware (kitware.com/cmake).
```

说明安装成功

## 1.2 Ubuntu

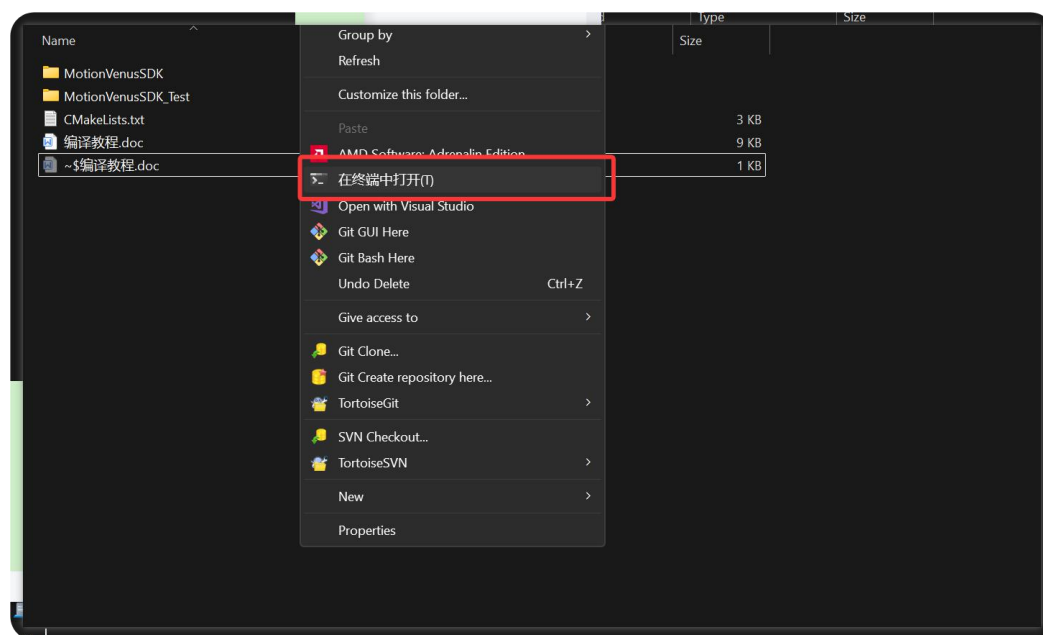
## 2、安装 Visual Studio 2022

<https://visualstudio.microsoft.com/zh-hans/vs/>

## 3、编译工程（Windows）

以 Visual Studio 2022 为例，编译程序。

在工程目录鼠标右键，点击“在终端中打开”。如果没有这个选项，在 cmd 命令行中手动导航到工厂目录。



在终端输入：

```
cmake -S . -B build -G "Visual Studio 17 2022" -DCMAKE_CONFIGURATION_TYPES="Debug;Release"
```

```
Command Prompt
Microsoft Windows [Version 10.0.26200.7171]
(c) Microsoft Corporation. All rights reserved.

D:\DEV\HMAX_SDK_V1.0.0_cmake>cmake -S . -B build -G "Visual Studio 17 2022" -DCMAKE_CONFIGURATION_TYPES="Debug;Release"
-- Selecting Windows SDK version 10.0.26100.0 to target Windows 10.0.26200.
-- The CXX compiler identification is MSVC 19.44.35221.0
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:/Program Files/Microsoft Visual Studio/2022/Community/VC/Tools/MSVC/14.44.35207/bin/Hostx64/x64/cl.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- MotionVenusSDK库根目录: D:/DEV/HMAX_SDK_V1.0.0_cmake/MotionVenusSDK
-- Configuring done (3.0s)
-- Generating done (0.0s)
-- Build files have been written to: D:/DEV/HMAX_SDK_V1.0.0_cmake/build
```

### 3.1 方法一：使用命令行编译

编译 Debug 文件输出：

```
cmake --build build --config Debug
```

```
D:\DEV\MotionGloveSDK>cmake --build build --config Debug
MSBuild version 17.14.23+b0019275e for .NET Framework
```

```
MotionGloveSDK.vcxproj -> D:\DEV\MotionGloveSDK\build\MotionGloveSDK\Debug\MotionGloveSDKd.dll
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example1.vcxproj -> D:\DEV\MotionGloveSDK\build\Debug\motionGloveSDK_example1d.exe
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example2.vcxproj -> D:\DEV\MotionGloveSDK\build\Debug\motionGloveSDK_example2d.exe
motionGloveSDK_example3.vcxproj -> D:\DEV\MotionGloveSDK\build\Debug\motionGloveSDK_example3d.exe
motionGloveSDK_example4.vcxproj -> D:\DEV\MotionGloveSDK\build\Debug\motionGloveSDK_example4d.exe
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
```

编译 Release 文件输出:

cmake --build build --config Release

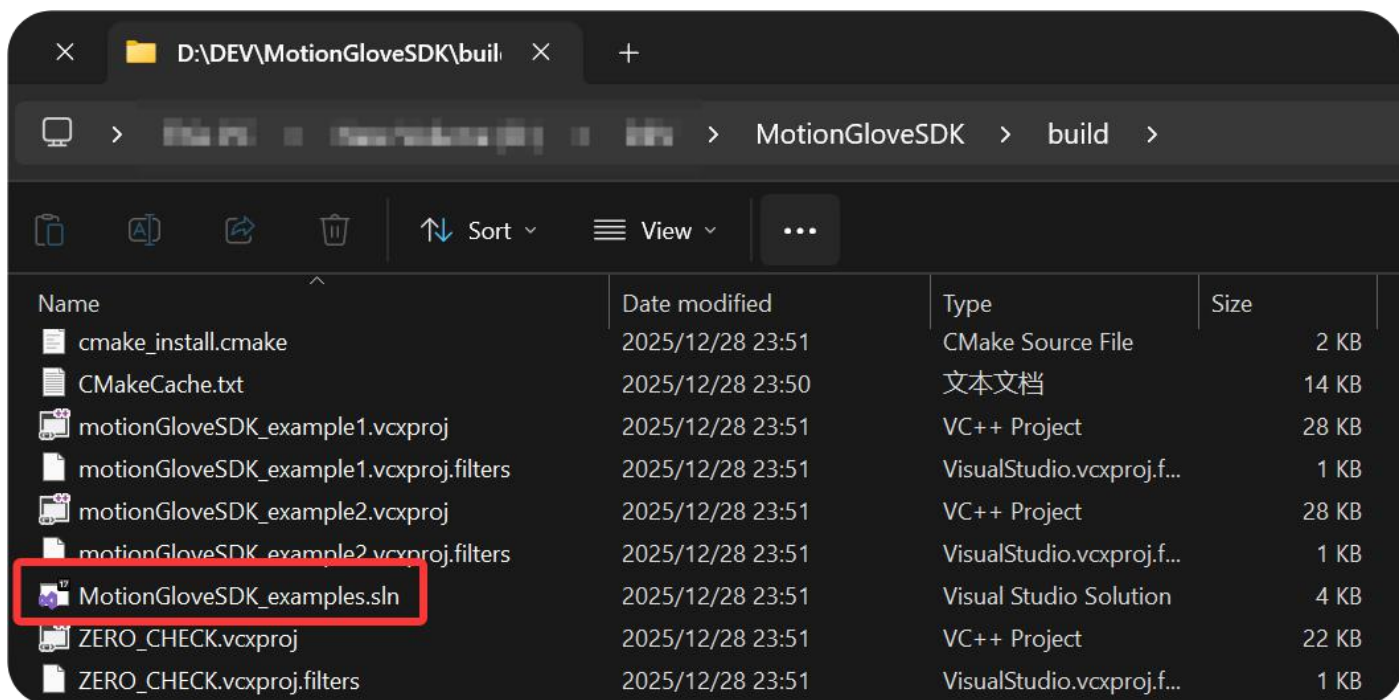
```
D:\DEV\MotionGloveSDK>cmake --build build --config Release
MSBuild version 17.14.23+b0019275e for .NET Framework
```

```
1>Checking Build System
decodeAsGloveCSV.cpp
eulerToQuat.cpp
motionGloveSDK.cpp
motionGloveSdkHelper.cpp
version.cpp
Generating Code...
Creating library D:/DEV/MotionGloveSDK/build/MotionGloveSDK/Release/MotionGloveSDK.lib and object D:/DEV/MotionGloveSDK/build/MotionGloveSDK/Release/MotionGloveSDK.exp
MotionGloveSDK.vcxproj -> D:\DEV\MotionGloveSDK\build\MotionGloveSDK\Release\MotionGloveSDK.dll
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example1.cpp
motionGloveSDK_example1.vcxproj -> D:\DEV\MotionGloveSDK\build\Release\motionGloveSDK_example1.exe
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example2.cpp
motionGloveSDK_example2.vcxproj -> D:\DEV\MotionGloveSDK\build\Release\motionGloveSDK_example2.exe
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example3.cpp
motionGloveSDK_example3.vcxproj -> D:\DEV\MotionGloveSDK\build\Release\motionGloveSDK_example3.exe
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example4.cpp
motionGloveSDK_example4.vcxproj -> D:\DEV\MotionGloveSDK\build\Release\motionGloveSDK_example4.exe
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
```

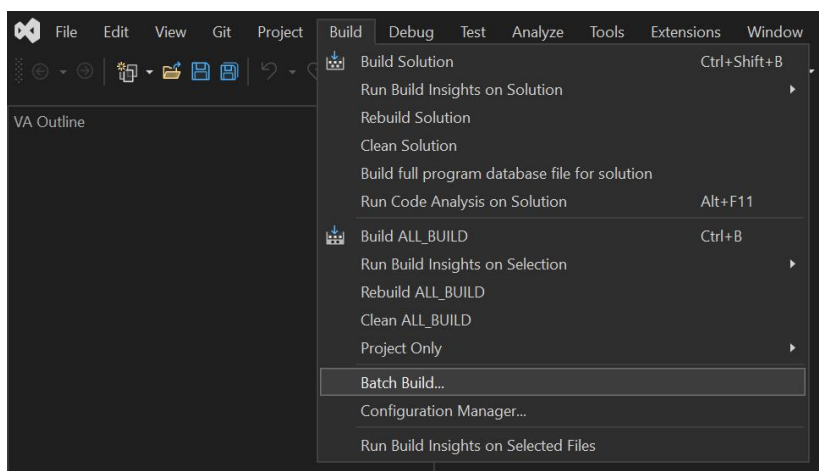
## 3.2 方法二：打开 Visual Studio 编译

打开 build 目录下的:

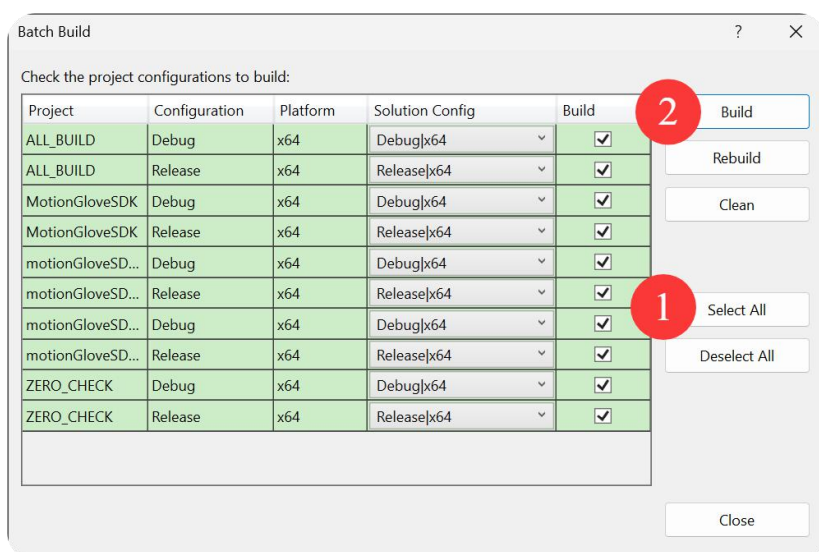
MotionGloveSDK\_examples.sln



打开批处理:



全选并点击 Build 编译:



输出中显示全部成功，无失败:



```
Output
Show output from: Build
motionGloveSDK_example2.cpp
motionGloveSDK_example2.vcxproj -> D:\DEV\MotionGloveSDK\build\Debug\motionGloveSDK_example2d.exe
----- Build started: Project: motionGloveSDK_example1, Configuration: Debug x64 -----
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
motionGloveSDK_example1.cpp
motionGloveSDK_example1.vcxproj -> D:\DEV\MotionGloveSDK\build\Debug\motionGloveSDK_example1d.exe
----- Build started: Project: ALL_BUILD, Configuration: Debug x64 -----
Building Custom Rule D:/DEV/MotionGloveSDK/CMakeLists.txt
----- Build: 10 succeeded, 0 failed, 0 up-to-date, 0 skipped -----
----- Build completed at 23:51 and took 11.815 seconds -----
```

## 4、编译工程（Ubuntu x64）

### 4.1 安装 cmake

参照 1.2 节

### 4.2 安装 gcc

参照 1.2 节

### 4.3 生成工程并编译

在源文件目录打开终端，输入：

```
cmake -S . -B build -DCMAKE_CONFIGURATION_TYPES="Debug;Release"
```

```
doro@doro-virtual-machine:/mnt/hgfs/share/HMAX_SDK_V1.0.0_cmake$ cmake -S . -B build -DCMAKE_CONFIGURATION_TYPES="Debug;Release"
-- The CXX compiler identification is GNU 11.4.0
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- MotionGloveSDK库根目录: /mnt/hgfs/share/HMAX_SDK_V1.0.0_cmake/MotionGloveSDK
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/hgfs/share/HMAX_SDK_V1.0.0_cmake/build
```

编译 Debug 文件输出：

```
cmake --build build --config Debug
```

```
doro@doro-virtual-machine:/mnt/hgfs/share/HMAX_SDK_V1.0.0_cmake$ cmake --build build --config Debug
[ 12%] Building CXX object MotionGloveSDK/CMakeFiles/MotionGloveSDK.dir/src/decodeAsGloveCSV.cpp.o
[ 25%] Building CXX object MotionGloveSDK/CMakeFiles/MotionGloveSDK.dir/src/eulerToQuat.cpp.o
[ 37%] Building CXX object MotionGloveSDK/CMakeFiles/MotionGloveSDK.dir/src/motionGloveSDK.cpp.o
[ 50%] Building CXX object MotionGloveSDK/CMakeFiles/MotionGloveSDK.dir/src/motionGloveSdkHelper.cpp.o
[ 62%] Building CXX object MotionGloveSDK/CMakeFiles/MotionGloveSDK.dir/src/version.cpp.o
[ 75%] Linking CXX shared library libMotionGloveSDK.so
[ 75%] Built target MotionGloveSDK
[ 87%] Building CXX object CMakeFiles/MotionGloveSDK_Test.dir/MotionGloveSDK_Test/src/MotionGloveSDK_test.cpp.o
[100%] Linking CXX executable MotionGloveSDK_Test
[100%] Built target MotionGloveSDK_Test
```

编译 Release 文件输出：

```
cmake --build build --config Release
```



```
doro@doro-virtual-machine:/mnt/hgfs/share/HMAX_SDK_V1.0.0_cmake$ cmake --build build --config Release
Consolidate compiler generated dependencies of target MotionGloveSDK
gmake[2]: Warning: File 'MotionGloveSDK/CMakeFiles/MotionGloveSDK.dir/compiler_depend.make' has modification time 0.5 s in the future
gmake[2]: warning: Clock skew detected. Your build may be incomplete.
[ 75%] Built target MotionGloveSDK
Consolidate compiler generated dependencies of target MotionGloveSDK_Test
gmake[2]: Warning: File 'CMakeFiles/MotionGloveSDK_Test.dir/compiler_depend.make' has modification time 0.22 s in the future
gmake[2]: warning: Clock skew detected. Your build may be incomplete.
[100%] Built target MotionGloveSDK_Test
```

## 5、编译工程（Ubuntu ARM x64）

适用于树莓派或 RDK X3/X5、RK3568、RK3588 等运行 Ubuntu 的开发板

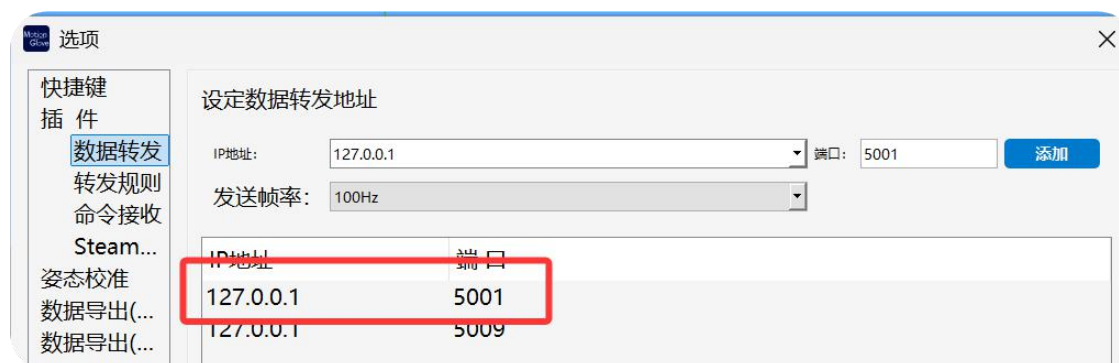
## 6、打开 MotionGlove

数据手套开机。

首先打开 MotionGlove 数据手套客户端，确保客户端已经连接数据手套并正常运行：



如果 Windows 系统在本机下测试，在软件菜单栏->设置->插件->数据转发中添加了 127.0.0.1 端口 5000（软件默认已添加）



如果在局域网内测试，填入运行 MotionGloveSDK 的 PC 网络 IP 地址。

如果在 Ubuntu 下测试，填入运行 MotionGloveSDK 的 PC 网络 IP 地址，确保与 Ubuntu 系统的网络畅

通。  
转发规则如下：



## 7、打开例程测试

以 Debug 版本为例讲解功能。

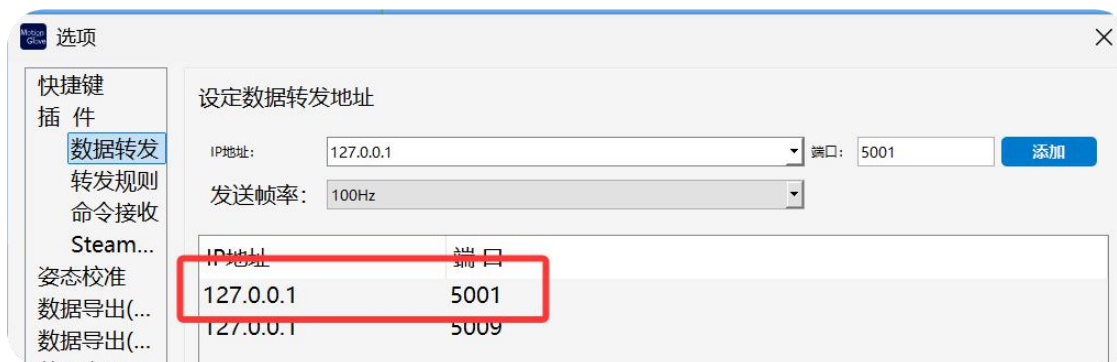
### 7.1 motionGloveSDK\_example1

测试目的：

用来测试是否接收到了 MotionGloveSDK 的数据转发，默认使用 UDP 监听本地 5001 端口。

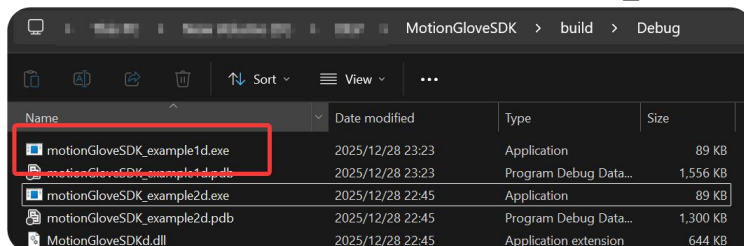
使用流程：

- 1、打开 MotionGlove 软件，并且手套连接正常
- 2、确保在软件菜单栏->设置->插件->数据转发中添加了 127.0.0.1 端口 5000（软件默认已添加）



"如果运行正常，会打印出收到数据帧的帧序号，每秒 60 帧

双击运行编译输出目录中的 motionGloveSDK\_example1d.exe



打印 New frame received 代表收到了转发的数据。

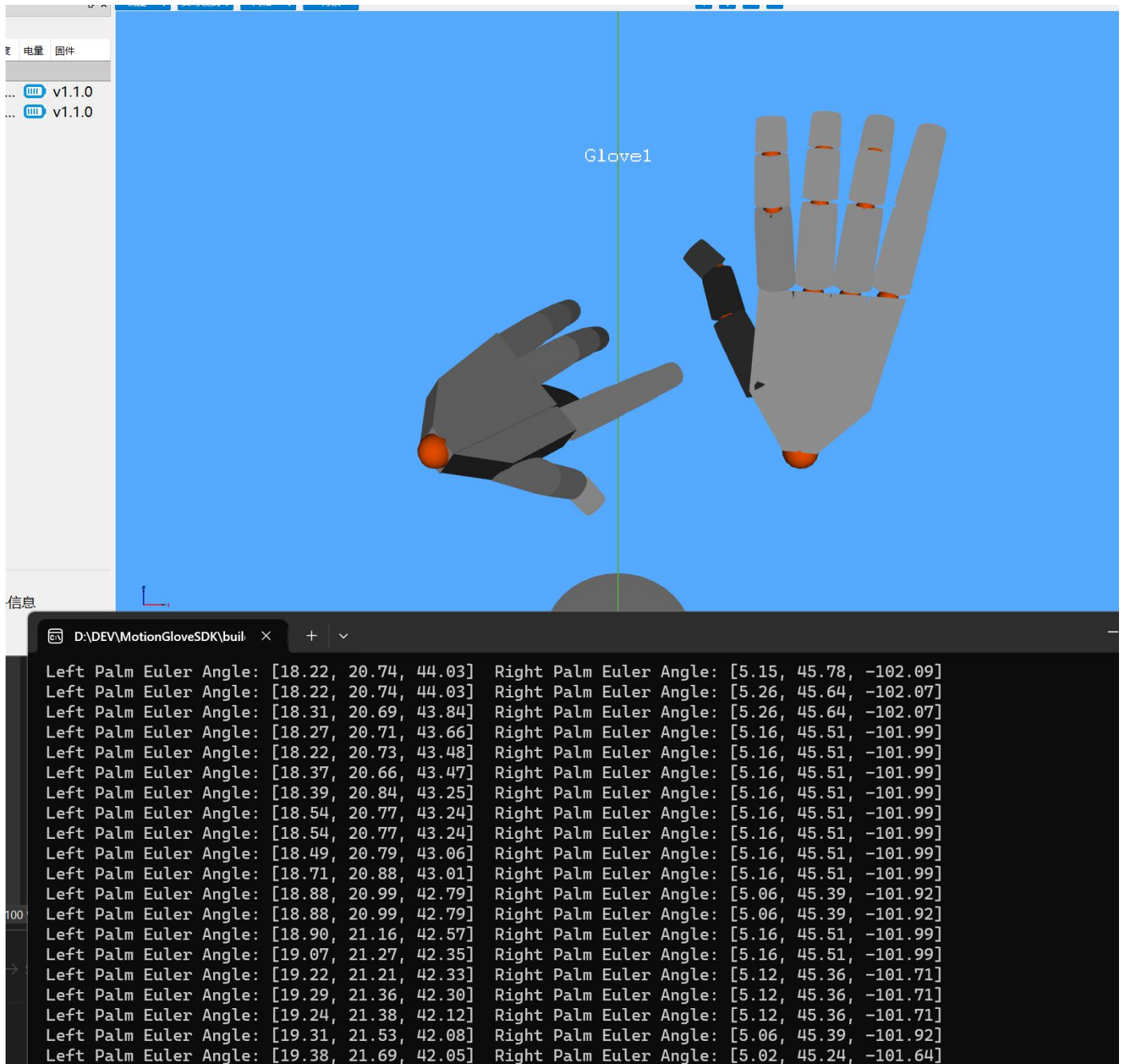


## 7.2 motionGloveSDK\_example2

测试目的：

用来测试左右手手背的角度。

这里的手背角度所在的坐标系为 MotionGlove 软件 3D 窗口左下角标识的坐标系。



## 7.3 motionGloveSDK\_example3

测试目的:

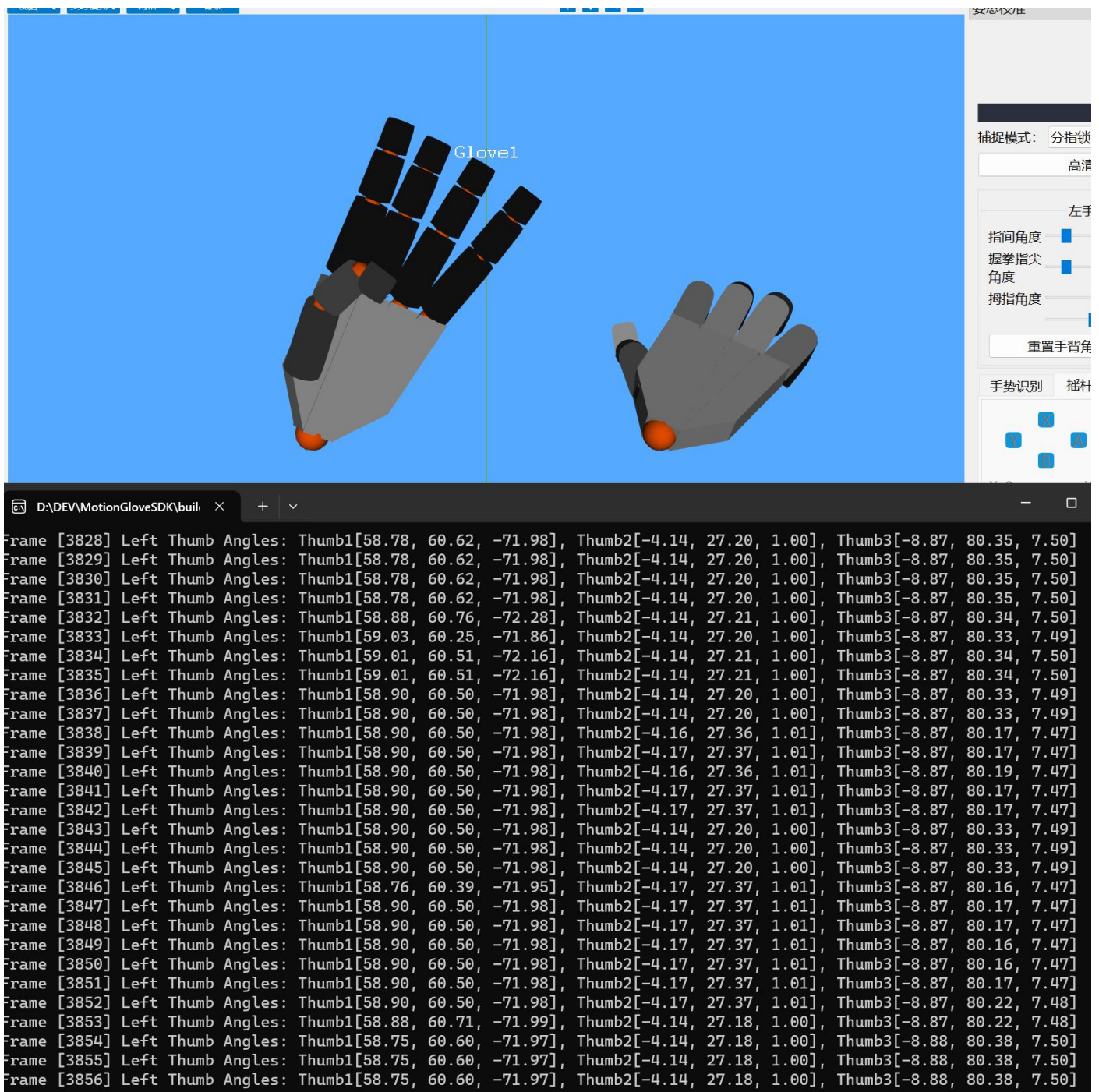
用来测试左手拇指三段骨骼的角度

拇指第 1 段的角度为相对于手背的角度

拇指第 2 段的角度为相对于拇指第 1 段的角

拇指第 3 段的角度为相对于拇指第 2 段的角





取最后一行分析:

Thumb1[58.75,60.60,-71.97], Thumb2[-4.14,27.18,1.00], Thumb3[-8.88, 80.38,7.50]

第 1 段的骨骼角度是相对于手背的角度,也是所有手指里角度最复杂的一段。

第 2 段的骨骼角度是相对于的第 1 段的角度,由于指关节自由度限制,可以忽略 XZ 轴,只关心 Y 轴的 27.18 度。

第 3 段的骨骼角度是相对于的第 2 段的角度,由于指关节自由度限制,可以忽略 XZ 轴,只关心 Y 轴的 80.38 度。

如果想获得手指的整体弯曲角度,可以将 2、3 段骨骼的 Y 轴角度相加,  $27.18+80.38=107.56$  度。

## 7.4 motionGloveSDK\_example4

测试目的:

用来测试左手中指三段骨骼的角度

中指第 1 段的角度为相对于手背的角度

中指第 2 段的角度为相对于中指第 1 段的角度

中指第 3 段的角度为相对于中指第 2 段的角



取最后一行分析：

Middle1[0.00, 57.37, 0.00] Middle2[-0.00, 62.69, -0.00] Middle3[-0.00, 62.69, -0.00]

第 1 段的骨骼角度是相对于手背的角度，由于指关节自由度限制，可以忽略 XZ 轴，只关心 Y 轴的 57.37 度。。

第 2 段的骨骼角度是相对于的第 1 段的角度，由于指关节自由度限制，可以忽略 XZ 轴，只关心 Y 轴的 62.69 度。

第 3 段的骨骼角度是相对于的第 2 段的角度，由于指关节自由度限制，可以忽略 XZ 轴，只关心 Y 轴的 62.69 度。

如果想获得手指的整体弯曲角度，可以将 1、2、3 段骨骼的 Y 轴角度相加， $57.37+62.69+62.69=182.75$  度。

食指、中指、无名指、小指都可以按照上面方法计算相对角度。