PicoZense TOF Camera

SDK 开发者指南

Windows

2019.11

Pico Technology Co.， Ltd.

关于本指南

本指南文档主要介绍如何使用 PicoZense TOF Camera 及 PicoZense SDK 进行开发。

文档结构

章节	标题	内容
1	概述	介绍 SDK 的概况
2	支持设备	介绍 PicoZense 产品
3	安装	介绍 PicoZense 产品及 SDK 的安装
4	SDK 使用说明	介绍如何使用 PicoZense SDK
5	SDK 接口介绍	介绍 PicoZense SDK 的接口
6	固件升级说明	介绍 PicoZense 产品固件升级

版本发布记录

日期	版本	发布说明
2019.12.24	V3.0.0.7	优化 SDK 框架及代码， 请阅读解压后 ReleaseNotes.txt

目 录

1 概述

PicoZense TOF Camera 是 Pico 公司采用飞行时间测距技术（TOF: Time of Flight）研发的一系列 3D 相机模组，适应不同场景需求，具有精度高、环境适应性强、尺寸小等优点。

PicoZense SDK 是基于 PicoZense TOF Camera 开发的软件开发工具包，该开发包目前适用于 Windows、Linux、Android，为应用开发者提供一系列友好的 API 和简单的应用示例程序。

用户基于该开发包，可获取高精度的深度数据信息、灰度图像信息和彩色图像信息，方便用户开发刷脸支付、手势识别、投影触控、人脸识别、疲劳检测、三维重建、导航避障等应用。

2 支持设备

2.1 DCAM710

图 2.1 PicoZense TOF Camera：DCAM710

DCAM710 是 Pico Zense 针对包括智慧物流、工业自动化、智能零售等专门化应用场景开发的一款通用的基于 ToF 技术的 3D 相机硬件模块，可提供高精度的深度距离图像、IR 图像和 RGB 图像数据，可满足即插即用的面部识别、手势识别、骨骼点获取、深度图像实时渲染、SLAM 等等功能。

2.2 DCAM800

图 2.2 PicoZense TOF Camera：DCAM800

DCAM800 是 Pico Zense 专门针对工业应用场景开发的一款基于 ToF 技术的 3D 相机，具有易安装、高可靠性，IP56 等级防护等特点。可以从容应对不同的工业场景需求，并且具有更远距离的侦测能力。

3 安装

3.1 推荐系统配置

配置项	推荐配置
操作系统	Win7 32/64 位 Win8/8.1 32/64 位 Win10 32/64 位
内存	4G 及以上

3.2 安装说明

3.2.1 硬件连接

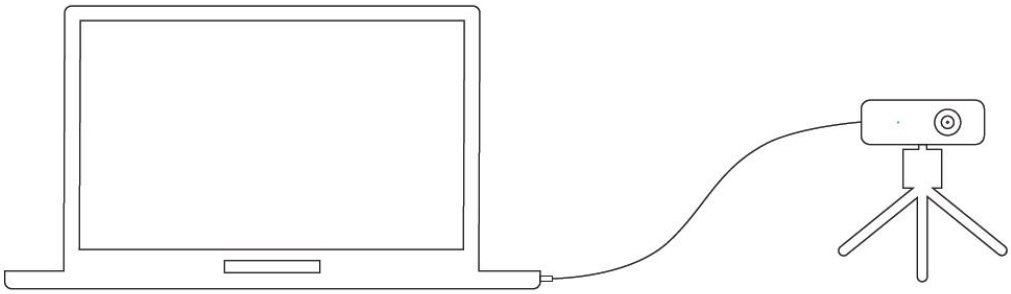


图 3.1 硬件模组安装示意图

3.2.1.1USB 方式

USB 连接线一端连接模组，另一端连接台式机或笔记本的 USB 接口。

在 Windows 系统下,连接成功后,系统桌面会弹出正在安装设备驱动程序软件的提示，安装完成后，设备管理器中会出现 PicoZense RGBD Camera 设备，如图

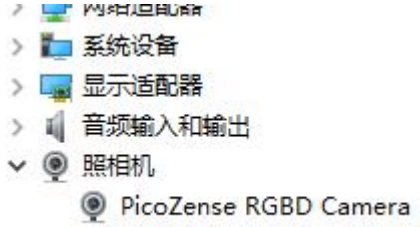


图 3.2 PicoZense RGBD Camera

3.2.1.2网口方式

网线连接分为固定地址直连与 DHCP 连接两种方式。

一、固定地址

固定地址连接可以相机与电脑直连，也可以配置在同一网段的交换机中使用。

直连：一端连接相机，另一端连接 PC 主机的网线接口。相机默认 IP 为 192.168.1.101，在 PC 端将“本地连接”的，子网掩码设为 255.255.255.0，IP 地址设为同一网段（如 192.168.1.100）。

图 3.3 直连方式

二、DHCP

DHCP 连接方式，需要将相机连接在开启 DHCP 功能的路由器上，使用在相同局域网中的 PC 进行连接，推荐将 PC 的“本地连接”设置为自动获取 IP 地址。

图 3.4 dhcp 方式

注意：

1、PC 端使用的网卡、路由器、交换机都要满足千兆要求。

2、在首次运行 SDK 时，要为 SDK 设置通过系统防火墙的权限，如下图所示。

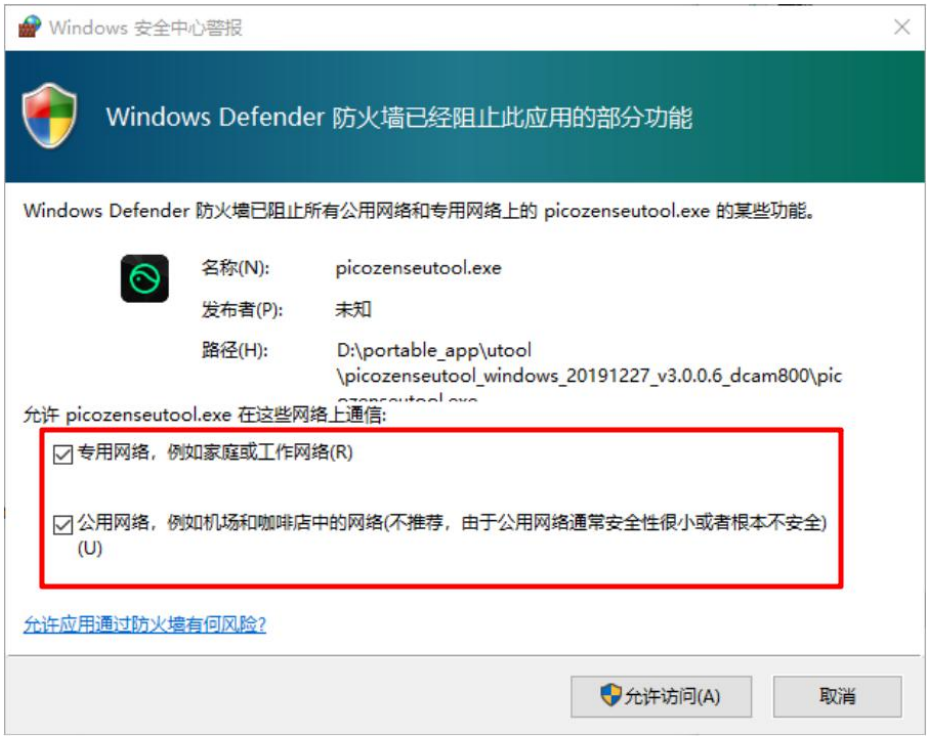


图 3.5 防火墙配置

3.2.2 软件环境配置

Windows 系统不需要额外的环境配置。

4 SDK 使用说明

4.1 SDK 目录结构

PicoZense SDK 包含 Bin，Document，Include，Lib，Samples，Tools 等目录。

Windows 目录结构如下图所示：

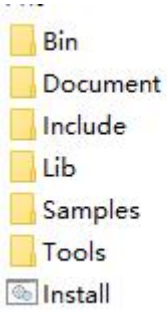
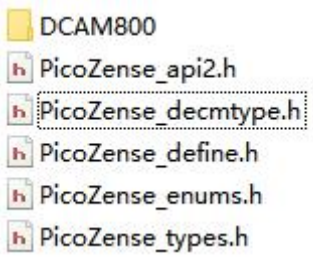


图 4.1 Windows SDK 目录结构

Bin 目录主要包含 PicoZense SDK 的动态链接库，如 PicoZense_api.dll，包括 x64 和 x86 的版本，运行基于该 SDK 开发的应用之前，需要先将相应平台的 dll 文件拷贝到可执行程序所在的目录。

Document 包含 SDK 的说明文档。

Include 主要包含 SDK 的通用头文件（PicoZense_api2.h 、PicoZense_define.h 、PicoZense_enums.h 、PicoZense_types.h、PicoZense_decmtype.h）和包含不同型号产品所需特定头文件的文件夹，如 DCAM800。如下图



Lib 主要包含 SDK 的 lib 文件，如 PicoZense_api.lib。

Samples 主要包含使用 PicoZense SDK 开发的例程。

Tools 中的 FrameViewer 可查看 PicoZense 深度摄像头的深度图像和 IR 灰度图像，针对不同设备，可自行编译 Samples 目录中的 FrameViewer 进行相应展示。

4.2 Tool 使用说明

将 PicoZense 深度摄像头连接到 PC，运行 Tools 目录里的 FrameViewer.exe，会启动两个窗口分别显示深度图像和 IR 灰度图像，如下图所示，图像正常显示且不卡顿，即表明 PicoZense 深度摄像头硬件运行正常。

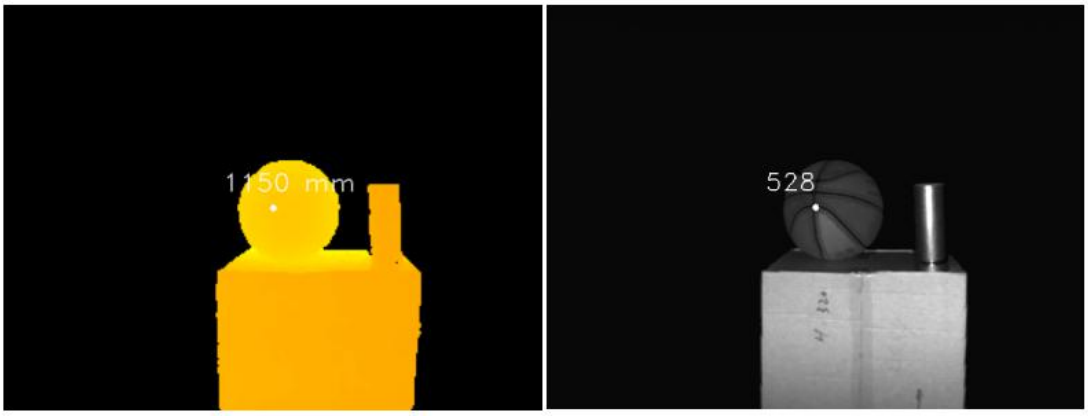


图 4.2 FrameViewer 运行效果

4.3 开发流程

4.3.1 项目配置

Windows 下使用 Visual Studio 2013 开发。新建应用项目工程，设置工程属性，将 Include 目录添加到包含目录中，将 Lib 目录添加到库目录中。另外，需要将 PicoZense_api.lib 添加到附加依赖项中。可参考 Samples 中的项目配置。

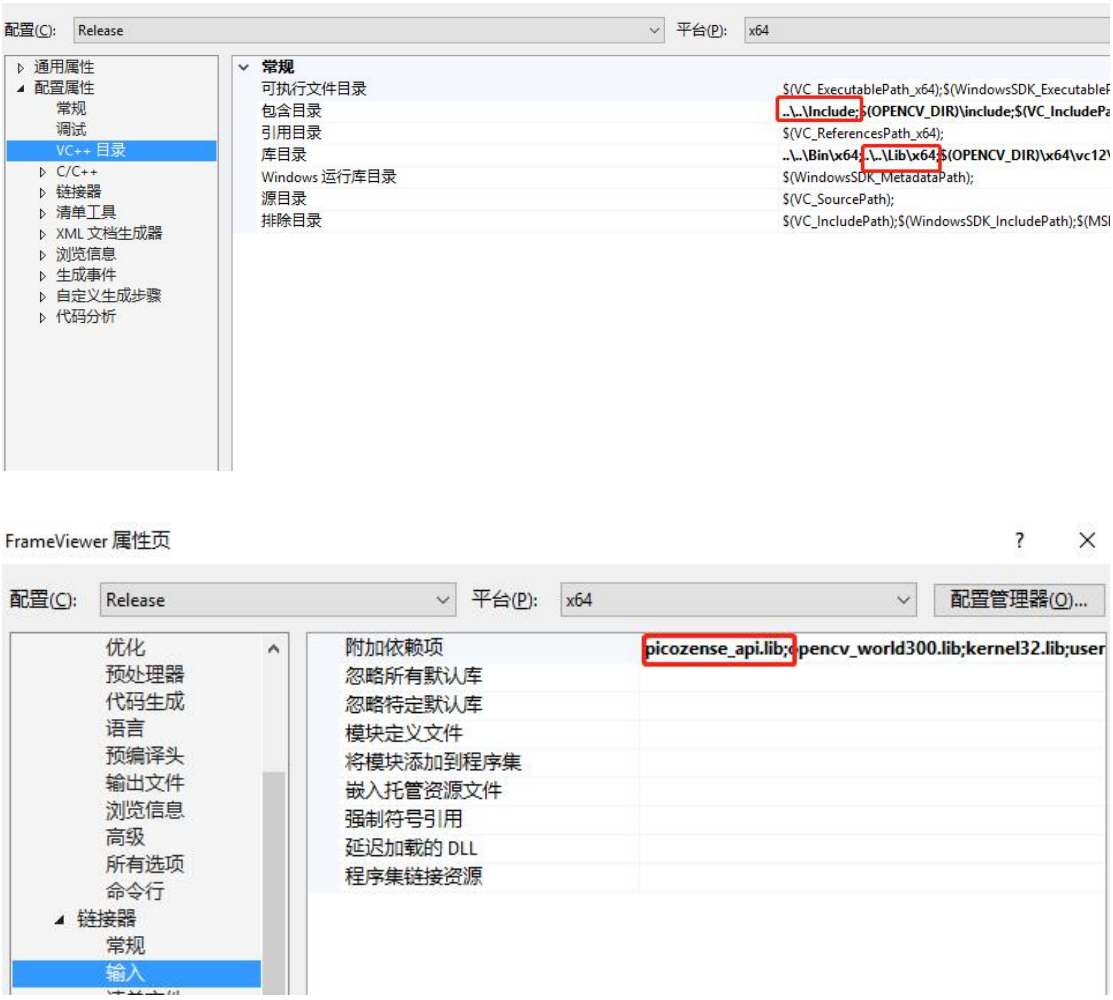


图 4.3 SDK 开发项目配置

4.3.2 接口调用流程

PicoZense SDK 的 API 接口调用流程图如下：

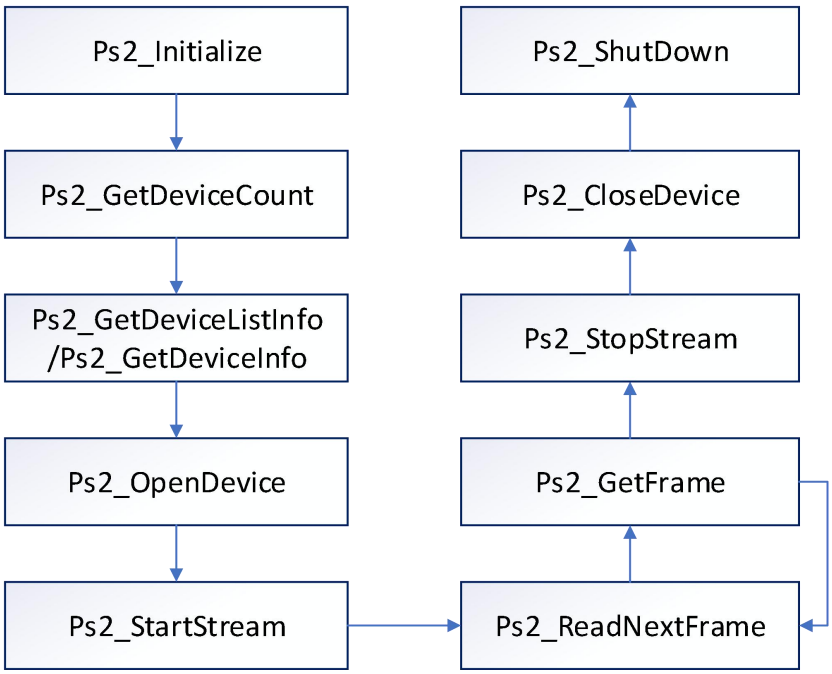


图 4.4 SDK 接口调用流程

1. Ps2_Initialize 和 Ps2_Shutdown

调用 Ps2_Initialize 接口，初始化 SDK ；调用 Ps2_Shutdown 接口，注销 SDK，释放 SDK 创建的所有资源。

2. Ps2_GetDeviceCount、Ps2_GetDeviceListInfo/Ps2_GetDeviceInfo

调用 Ps2_GetDeviceCount 接口，获取当前连接的设备数，请确保此接口返回的设备数量大于 0，在进行后续接口的调用；

调用 Ps2_GetDeviceListInfo Ps2_GetDeviceInfo 接口，获取当前连接的设备信息；

3. Ps2_OpenDevice 和 Ps2_CloseDevice

调用 Ps2_OpenDevice 接口，打开指定的深度摄像头设备；调用 Ps2_CloseDevice 接口，关闭指定设备。

4. Ps2_StartStream 和 Ps2_StopStream

调用 Ps2_StatrStream 接口，打开深度摄像头设备视频流；调用 Ps2_StopStream 接口，关闭指定设备视频流。

5. Ps2_ReadNextFrame 和 Ps2_GetFrame

在图像处理的主循环里，每次先调用 Ps2_ReadNextFrame 采集一帧图像，然后再调用 Ps2_GetFrame 获取指定图像类型的一帧图像数据，并用于相应的图像处理。

6. Set 和 Get

SDK 提供了丰富的 Set 和 Get 类型的接口，以便设置与获取相机属性、参数和数据等各类功能，详见 5.3 节。如果取图之前需要修改相机初始设置，请在 Ps2_StatrStream 接口调用之后，Ps2_ReadNextFrame 接口调用之前进行设置。请参考以下流程：

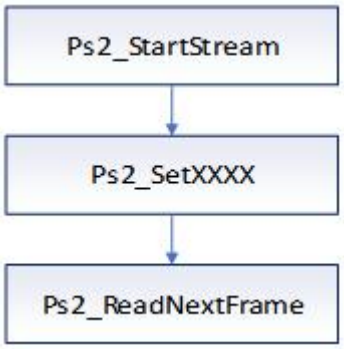


图 4.5 取图前，修改默认设置 SDK 接口调用流程

4.4 SDK Sample 使用流程

PicoZense SDK 开发包提供相应的 Sample 用于演示 SDK 的 API 接口使用，位于 SDK 安装路径的 Samples 目录下。

1. 到 OpenCV 官网 http://opencv.org/releases.html，下载并安装 OpenCV 3.0.0。

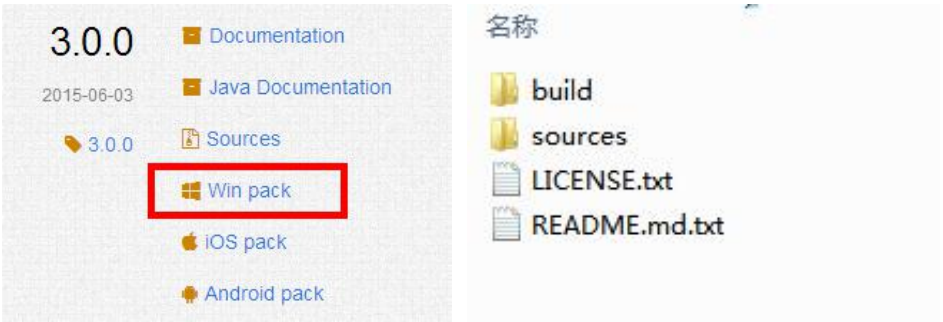


图 4.6 OpenCV3.0.0 下载安装

2. 设置环境变量 OPENCV_DIR， 其值为安装的 OpenCV 的 build 目录的绝对路径，例如 D:\Program Files\opencv-3.0.0\build。

图 4.7 设置 OPENCV_DIR 环境变量

3. 使用 Visual Studio 2013 打开（Visual Studio 2015、2017 均支持）PicoZenseSDK_Windows_xxx\Samples\FrameViewer 目录下的 FrameViewer.sln，直接编译。

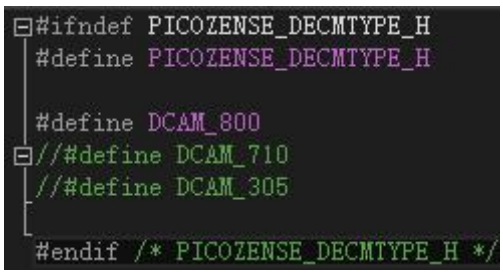
图 4.8 FrameViewer 工程

4. 编译选项配置了 opencv_world300.dll，PicoZense_api.dll 自动拷贝功能，会自动拷贝到编译运行的 Target 目录。开发时也可手动将以上文件拷贝到运行目录。

5. 编译并运行程序，其效果与 Tools 里的 FrameViewer 工具运行效果一致。

4.5 重点说明

1．Include 使用时，需根据不同型号设备，修改 PicoZense_decmtype.h 中的宏定义状态，如使用 DCAM800，仅保留 **DCAM_800** 宏定义，注释掉其他定义。



2．Samples 目录中的 FrameViewer 工程编译时，FrameViewer.cpp 已根据 PicoZense_decmtype.h 中的宏定义，进行不同模块的定义，在实际开发过程，请以 Include 中具体型号文件夹下的头文件为准。

5 SDK 接口介绍

V3.0 版本的 SDK 在接口定义上为区分原 V2.0 版本接口，采用 Ps2_的命名方式，枚举类型及结构体类型延用 V2.0 的命名方式。 5.3 节仅介绍 V3.0 接口不在赘述 V2.0 接口，另 V3.0 版本 SDK 支持 V2.0 的接口使用，具体详见 5.4 节。

5.1 Enum 数据类型

5.1.1 PsDepthRange

功能:

Depth Range 模式。

枚举值:

PsNearRange 表示设置为 Near Range 模式，Range0

PsMidRange 表示设置为 Middle Range 模式，Range1

PsFarRange 表示设置为 Far Range 模式，Range2

PsXNearRange 表示设置为 XNear Range 模式，Range3

PsXMidRange 表示设置为 XMiddle Range 模式，Range4

PsXFarRange 表示设置为 XFar Range 模式，Range5

PsXXNearRange 表示设置为 XXNear Range 模式，Range6

PsXXMidRange 表示设置为 XXMiddle Range 模式，Range7

PsXXFarRange 表示设置为 XXFar Range 模式，Range8

注意：每个相机可能支持这九种模式中的几种，不一定全部支持。

5.1.2 PsDataMode

功能：

数据类型设置，设置输出图像帧的类型和帧数。

PS:不同型号产品对应的枚举值定义可能不同，请以 Include 中具体型号文件夹下定义为准。

枚举值：

PsDepthAndRGB_30 表示以 30fPs 同时输出 Depth 和 RGB 两路图像，Depth 图像分辨率为 640*480，RGB 图像分辨率可通过 Ps2_SetRGBResolution 接口设置，支持 1920*1080/1280*720/640*480/640*360 四种分辨率。

PsDepth_30 表示以 30fPs 仅输出 Depth 图像，Depth 图像分辨率为 640*480。

PsIRAndRGB_30 表示以 30fPs 同时输出 IR 和 RGB 两路图像，IR 图像分辨率为 640*480，RGB 图像分辨率可通过 Ps2_SetRGBResolution 接口设置，支持 1920*1080/1280*720/640*480/640*360 四种分辨率。

PsIR_30 表示以 30fPs 仅输出 IR 图像，IR 图像分辨率为 640*480。

PsDepthAndIR_30 表示以 30fPs 同时输出 Depth 和 IR 两路图像，分辨率均为 640*480。

PsDepthAndIR_15_RGB_30 表示以 15fPs 输出 Depth 和 IR 图像，Depth 和 IR

图像分辨率为 640*480，RGB 图像分辨率可通过 Ps2_SetRGBResolution 接口设置，支持 1920*1080/1280*720/640*480/640*360 四种分辨率。

PsDepthAndIR_15 表示以 15fPs 输出 Depth 和 IR 图像，Depth 和 IR 图像分辨率为 640*480。

PsWDR_Depth 表示以 30fPs 输出 Depth 图像。Depth 图像分辨率为 640*480，支持 2-3 种距离模式交替输出。可以通过 Ps2_SetWDROutputMode 设置所需的距离模式。

5.1.3 PsPropertyType

功能：

获取或设置的属性

PS:不同型号产品对应的枚举值个数可能不同，请以 Include 中具体型号文件夹下定义为准。

枚举值：

PsPropertySN_Str 表示设备 SN 序列号，大小不超过 64 个字节

PsPropertyFWVer_Str 表示设备固件版本号，大小不超过 64 个字节

PsPropertyHWVer_Str 表示设备硬件版本号，大小不超过 64 个字节

PsPropertyDataMode_UInt8 表示数据模式，参考 PsDataMode 与接口 Ps2_SetDataMode 一致

PsPropertyDataModeList 表示设备支持的数据模式列表，参考 PsDataMode

PsPropertyDepthRangeList 表示设备支持的 range 列表，参考 PsDepthRange

PsPropertyDeviceUpgradeFlag 表示设备升级标志位

PsPropertyDeviceSN 表示设备 SN

PsPropertyDeviceMACAddr 表示设备 MAC 地址

PsPropertyDeviceSoftVer 表示设备软件版本

PsPropertyDeviceIPAddr 表示设备 IP 地址

PsPropertyDeviceSubnetMask 表示设备子网掩码

5.1.4 PsFrameType

功能：

图像数据流类型。

PS:不同型号产品对应的枚举值个数可能不同，请以 Include 中具体型号文件夹下定义为准。

枚举值：

PsDepthFrame 表示深度图像流

PsIRFrame 表示 IR 灰度图像流

PsGrayFrame 表示灰度图像流，

PsRGBFrame 表示彩色图像流

PsMappedRGBFrame 表示映射到深度图空间的 RGB 图像流

PsMappedDepthFrame 表示映射到 RGB 空间的深度图像流

PsMappedIRFrame 表示映射到 RGB 空间的 IR 图像流

PsRawDepthFrame 表示原始深度图像流（没有滤波和反畸变处理）

PsConfidenceFrame 表示置信度数据流

PsWDRDepthFrame 表示 WDR 深度图像流

5.1.5 PsSensorType

功能：

摄像头类型。

PS:不同型号产品对应的枚举值个数可能不同，请以 Include 中具体型号文件夹下定义为准。

枚举值：

PsDepthSensor 表示深度摄像头

PsRgbSensor 表示 RGB 摄像头

5.1.6 PsPixelFormat

功能：

图像的像素类型。

PS:不同型号产品对应的枚举值个数可能不同，请以 Include 中具体型号文件夹下定义为准。

枚举值：

PsPixelFormatDepthMM16 表示每像素数据为 16 位的深度值，以毫米为单位

PsPixelFormatGray16 表示每像素数据为 16 位的灰度值

PsPixelFormatGray8 表示每像素数据为 8 位的灰度值

PsPixelFormatRGB888 表示每像素数据为 24 位的 RGB 值

PsPixelFormatBGR888 表示每像素数据为 24 位的 BGR 值

5.1.7 PsReturnStatus

功能：

函数调用的返回状态值

枚举值：

PsRetOK 表示函数调用正常返回

PsRetNoDeviceConnected 表示当前无设备连接

PsRetInvalidDeviceIndex 表示传入的设备序号不是有效值

PsRetDevicePointerIsNull 表示设备结构指针为空

PsRetInvalidFrameType 表示传入的 frame type 不是有效值

PsRetFramePointerIsNull 表示获取到的图像数据帧的指针为空

PsRetNoPropertyValueGet 表示无法获取当前属性值

PsRetNoPropertyValueSet 表示无法设置当前属性值

PsRetPropertyPointerIsNull 表示传入的存储属性值的 buffer 的指针为空

PsRetPropertySizeNotEnough 表示传入的 buffer 的 size 太小

PsRetInvalidDepthRange 表示传入的 Depth Range 值不是有效值

PsRetReadNextFrameError 表示采集下一帧图像数据时出错

PsRetCameraNotOpened 表示未打开相机

PsRetInvalidCameraType 表示无效相机类型

PsRetInvalidParams 表示无效参数

PsRetOthers 表示其他错误

5.1.8 PsWDRTotalRange

功能：

WDR 模式下可选择的输出距离模式的数量。

枚举值：

PsWDRTotalRange_Two 表示输出两种距离模式，如 Near/Far/Near/Far…

PsWDRTotalRange_Three 表示输出三种距离模式，如 Near/Mid/Far/Near/Mid/Far…

5.1.9 PsWDRStyle

功能：

WDR 融合类型，用于 Ps2_SetWDRStyle，用于决定输出的 WDR 图像是否做融合。

枚举值:

PsWDR_FUSION 表示多距离融合模式

PsWDR_ALTERNATION 表示距离切换模式

5.1.10 PsResolution

功能:

RGB 图像分辨率类型。

PS:不同型号产品可能不支持 RGB，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

枚举值:

PsRGB_Resolution_1920_1080 表示 1080P

PsRGB_Resolution_1280_720 表示 720P

PsRGB_Resolution_640_480 表示 480P

PsRGB_Resolution_640_360 表示 360P

5.2 Struct 数据类型

5.2.1 PsRGB888Pixel

功能:

RGB 格式彩色图像像素类型。

PS:不同型号产品可能不支持 RGB，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

成员：

r：red

g：green

b：blue

5.2.2 PsBGR888Pixel

功能：

BGR 格式彩色图像像素类型。

PS:不同型号产品可能不支持 RGB，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

成员：

b：blue

g：green

r：red

5.2.3 PsVector3f

功能：

向量容器

成员：

float x, y, z

5.2.4 PsDepthVector3

功能：

深度图像向量

成员：

depthX ： 像素横坐标 x

depthY ： 像素纵坐标 y

depthZ ： 深度值 z，单位为毫米

5.2.5 PsCameraParameters

功能：

相机内参和畸变系数

成员：

fx Focal length x (pixel)

fy Focal length y (pixel)

cx Principal point x (pixel)

cy Principal point y (pixel)

k1 Radial distortion coefficient, 1st-order

k2 Radial distortion coefficient, 2nd-order

p1 Tangential distortion coefficient

p2 Tangential distortion coefficient

k3 Radial distortion coefficient, 3rd-order

k4 Radial distortion coefficient, 4st-order

k5 Radial distortion coefficient, 5nd-order

k6 Radial distortion coefficient, 6rd-order

5.2.6 PsCameraExtrinsicParameters

功能：

相机外参

成员：

rotation[9] ： 3×3 的旋转矩阵

translation[3] ： 三维平移矢量

5.2.7 PsFrame

功能：

图像信息

成员：

frameIndex 表示帧索引

frameType 表示图像数据流类型

pixelFormat 表示像素类型

imuFrameNo 表示对应的 IMU 数据的帧号，用于与 IMU 同步

pFrameData 图像数据缓存的指针

dataLen 表示数据长度，以字节为单位

exposureTime 表示曝光时间，单位为 ms

depthRange 表示当前帧的深度范围，仅对深度图像帧

width 表示当前帧的宽度

height 表示当前帧的高度

5.2.8 PsWDROutputMode

功能：

WDR 输出模式设置

成员：

totalRange 表示设置的 range 总数，目前支持 2 或 3 种距离模式

range1 表示第 1 个 range 值

range1Count 表示 wdr 模式下 range1 输出帧的数量

range2 表示第 2 个 range 值

range2Count 表示 wdr 模式下 range2 输出帧的数量

range3 表示第 3 个 range 值，仅当 totalRange 设置为 3 时生效

range3Count 表示 wdr 模式下 range3 输出帧的数量

5.2.9 PsMeasuringRange

功能：

相机标定范围，可用于选取深度图像的颜色映射阈值

成员：

depthMode 表示 depth 的模式，0/1/2 分别对应（near/mid/far）/（xnear/xmid/xfar）/（xxnear/xxmid/xxfar） 这三种模式

depthMaxNear 表示 depthMode 模式下的 Near 理论最大值

depthMaxMid 表示 depthMode 模式下的 Mid 理论最大值

depthMaxFa 表示 depthMode 模式下的 Far 理论最大值

effectDepthMaxNear 表示 depthMode 模式下的 Near 有效标定大值

effectDepthMaxMid 表示 depthMode 模式下的 Mid 有效标定大值

effectDepthMaxFar 表示 depthMode 模式下的 Far 有效标定大值

effectDepthMinNear 表示 depthMode 模式下的 Near 有效标定小值

effectDepthMinMid 表示 depthMode 模式下的 Mid 有效标定小值

effectDepthMinFar 表示 depthMode 模式下的 Far 有效标定小值

5.2.10 PsDeviceInfo

功能：

设备信息

成员：

sessionCount 表示有几个 Tof Sensor

devicetype 表示设备类型

uri 表示设备的标识

fw 表示设备固件版本号

status 表示连接状态

5.2.11 PsDataModeList

功能：

相机支持可切换数据模式列表

成员：

index 固定 0x00

count 表示 DataMode 总个数

datamodelist 表示 DataMode 列表

5.2.12 PsDepthRangeList

功能：

相机可切换的 range 列表

成员：

index 固定 0x01

count 表示 DepthRange 总个数

depthrangelist 表示 DepthRange 列表

5.2.13 PsFrameReady

功能：

图像是否可直接获取，1 可直接取图，0 则不可。

PS:不同型号产品可获取的图像类型不同，请以 Include 中具体型号文件夹下定义为准。

成员：

depth 表示 depth 图的可获取状态

ir 表示 ir 图的可获取状态

rgb 表示 rgb 图的可获取状态

mappedRGB 表示对齐后 rgb 图的可获取状态

mappedDepth 表示 depth 图的可获取状态

mappedIR 表示对齐后 ir 图的可获取状态

confidence 表示自信度图的可获取状态

wdrDepth 表示 wdr 模式下 depth 图的可获取状态

reserved 表示预留状态，暂未使用

5.3 API 接口

5.3.1 Ps2_Initialize

函数原型：

PsReturnStatus Ps2_Initialize()

函数功能：

SDK 初始化，需要在调用任何 SDK 其它接口之前先调用 Ps2_Initialize 接口

函数参数：

无

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.2 Ps2_Shutdown

函数原型：

PsReturnStatus Ps2_Shutdown()

函数功能：

SDK 注销，释放 SDK 创建的所有资源，该接口调用之后，则不能再调用 SDK 其他接口

函数参数：

无

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.3 Ps2_GetDeviceCount

函数原型：

PsReturnStatus Ps2_GetDeviceCount(int32_t* pDeviceCount)

函数功能：

获取已连接的设备数

函数参数：

pDeviceCount **[out]**：存储返回的设备数的变量指针，需要先创建一个 int 类型变量并将其指针传给该函数

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.4 Ps2_GetDeviceListInfo

函数原型：

PsReturnStatus Ps2_GetDeviceListInfo(PsDeviceInfo* pDevicesList，uint32_t deviceCount)

函数功能：

获取 deviceCount 个设备的信息列表

函数参数：

deviceCount**[in]**：deviceCount 个设备

pDevicesList**[out]**：返回 deviceCount 个设备的信息

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.5 Ps2_GetDeviceInfo

函数原型：

PsReturnStatus Ps2_GetDeviceInfo(PsDeviceInfo* pDevices， uint32_t deviceIndex)

函数功能：

获取第 deviceIndex 个设备的设备信息

函数参数：

deviceIndex**[in]**：第 deviceIndex 个设备

pDevicesList**[out]**：返回第 deviceIndex 个设备的信息

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.6 Ps2_OpenDevice

函数原型：

PsReturnStatus Ps2_OpenDevice(const char* uri， PsDeviceHandle *pDevice)

函数功能：

打开

函数参数：

uri**[in]**： 设备的标识，可从设备信息中得到

pDevice**[out]**：返回设备句柄

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.7 Ps2_CloseDevice

函数原型：

PsReturnStatus Ps2_CloseDevice(PsDeviceHandle device)

函数功能：

关闭句柄为 device 的设备

函数参数：

device**[in]**：设备句柄

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.8 Ps2_StartStream

函数原型：

PsReturnStatus Ps2_StartStream(PsDeviceHandle device， uint32_t sessionIndex);

函数功能：

打开设备句柄为 handle 的第 sessionIndex 个会话流

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，会话是对 Tof 传感器和 Rgb 传感器操作的一个抽象概念，会话可能有 N 个 Tof 最多有 N 个 Rgb，如设备中有 2 个 tof，1 个 rgb，sessionIndex 为 0 时，Tof 与 RGB 同时开启流，如果 sessionIndex 为 1 时，只打开 Tof 流

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.9 Ps2_StopStream

函数原型：

PsReturnStatus Ps2_StopStream(PsDeviceHandle device， uint32_t sessionIndex)

函数功能：

关闭设备句柄为 handle 的第 sessionIndex 个会话流

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，会话是对 Tof 传感器和 Rgb 传感器操作的一个抽象概念，会话可能有 N 个 Tof 最多有 N 个 Rgb，如设备中有 2 个 tof，1 个 rgb，sessionIndex 为 0 时，Tof 与 RGB 同时关闭流，如果 sessionIndex 为 1 时，只关闭 Tof 流

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.10 Ps2_ReadNextFrame

函数原型：

PsReturnStatus Ps2_ReadNextFrame(PsDeviceHandle device， uint32_t

sessionIndex， PsFrameReady* pFrameReady)

函数功能：

采集指定设备的下一帧图像，在使用 Ps2_GetFrame 获取图像数据之前，需要先调用该函数

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pFrameReady**[out]**：输出图像是否获取标识，可参考 Ps2_FrameReady 说明

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.11 Ps2_GetFrame

函数原型：

PsReturnStatus Ps2_GetFrame(PsDeviceHandle device， uint32_t sessionIndex， PsFrameType frameType， PsFrame* pPsFrame)

函数功能：

获取指定帧类型的当前帧的图像数据，调用该函数之前需要先调用

Ps2_ReadNextFrame 采集一帧图像。

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

frameType **[in]**：图像数据帧类型，参考 PsFrameType

pPsFrame **[out]**：图像信息，参考 PsFrame

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.12 Ps2_SetDataMode

函数原型：

PsReturnStatus Ps2_SetDataMode(PsDeviceHandle device， uint32_t sessionIndex， PsDataMode dataMode)

函数功能：

设置输出的数据类型

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

dataMode **[in]**：数据类型，参考 PsDataMode

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.13 Ps2_GetDataMode

函数原型：

PsReturnStatus Ps2_GetDataMode(PsDeviceHandle device， uint32_t sessionIndex， PsDataMode dataMode)

函数功能：

获取当前使用的数据类型

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

dataMode **[out]**：存储输出数据类型，参考 PsDataMode

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.14 Ps2_GetDepthRange

函数原型：

PsReturnStatus Ps2_GetDepthRange(PsDeviceHandle device， uint32_t sessionIndex， PsDepthRange* pDepthRange)

函数功能：

获取指定设备的 Depth Range 模式

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pDepthRange **[out]**：存储 Depth Range 模式的变量指针，参考 PsDepthRange

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.15 Ps2_SetDepthRange

函数原型：

PsReturnStatus Ps2_SetDepthRange(PsDeviceHandle device， uint32_t

sessionIndex， PsDepthRange depthRange)

函数功能：

设置指定设备的 Depth Range 模式

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

depthRange **[out]**：Depth Range 模式，参考 PsDepthRange

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.16 Ps2_GetThreshold

函数原型：

PsReturnStatus Ps2_GetThreshold(PsDeviceHandle device， uint32_t

sessionIndex， uint16_t* pThreshold)

函数功能：

获取背景滤波阈值大小

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pThreshold**[out]**：阈值大小

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.17 Ps2_SetThreshold

函数原型:

PsReturnStatus PsSetThreshold(PsDeviceHandle device， uint32_t sessionIndex， uint16_t threshold)

函数功能:

设置背景滤波阈值大小

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pThreshold**[in]**：阈值大小，0 表示关闭背景滤波功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.18 Ps2_GetPulseCount

函数原型：

PsReturnStatus Ps2_GetPulseCount(PsDeviceHandle device， uint32_t sessionIndex， uint16_t* pPulseCount)

函数功能：

获取脉冲计数个数

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pulseCount **[out]**：脉冲计数个数

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.19 Ps2_SetPulseCount

函数原型：

PsReturnStatus Ps2_SetPulseCount(PsDeviceHandle device， uint32_t sessionIndex， uint16_t pulseCount)

函数功能：

设置脉冲计数个数

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pulseCount **[in]**：脉冲计数个数

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.20 Ps2_GetGMMGain

函数原型：

PsReturnStatus Ps2_GetGMMGain(PsDeviceHandle device， uint32_t sessionIndex， uint16_t* gmmgain)

函数功能：

获取设备 Gamma 增益值

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

gmmgain **[out]**：存储返回的 Gamma 值变量指针，需要先创建一个 unsigned short 类型变量并将其指针传给该函数

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.21 Ps2_SetGMMGain

函数原型:

PsReturnStatus Ps2_SetGMMGain(PsDeviceHandle device， uint32_t sessionIndex， uint16_t gmmgain)

函数功能:

设置设备 Gamma 增益值

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

gmmgain **[in]**：需要设置的 Gamma 增益值

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.22 Ps2_GetProperty

函数原型：

PsReturnStatus Ps2_GetProperty(PsDeviceHandle device， uint32_t sessionIndex， int32_t propertyType， void* pData， int32_t* pDataSize)

函数功能：

获取指定设备的属性值

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

propertyType **[in]**：属性类型，参考 PsPropertyType

pData **[out]**：存储返回的属性值的内存指针

pDataSize **[in/out]**：传入 pData 指向的内存空间大小，同时传回用于存储返回的属性值占用的实际内存空间大小，以字节为单位

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.23 Ps2_SetProperty

函数原型：

PsReturnStatus Ps2_SetProperty(PsDeviceHandle device， uint32_t sessionIndex，int32_t propertyType，const void* pData，int32_t dataSize)

函数功能：

设置指定设备的相应属性

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

propertyType **[in]**：属性类型，参考 PsPropertyType

pData **[in]**：存储需要设置的属性值的内存指针

dataSize **[in]**：需要设置的属性值占用的内存空间大小，以字节为单位

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.24 Ps2_GetCameraParameters

函数原型：

PsReturnStatus Ps2_GetCameraParameters(PsDeviceHandle device，

uint32_t sessionIndex， PsSensorType sensorType， PsCameraParameters*

pCameraParameters)

函数功能：

获取相机内参

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

sensorType **[in]**：传感器类型，参考 PsSensorType

pCameraParameters**[out]**：输出的相机内参，参考 PsCameraParameters

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.25 Ps2_GetCameraExtrinsicParameters

函数原型：

PsReturnStatus Ps2_GetCameraExtrinsicParameters(PsDeviceHandle device， uint32_t sessionIndex， PsCameraExtrinsicParameters* pCameraExtrinsicParameters)

函数功能：

获取相机外参

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pCameraExtrinsicParameters[out]: 指向用于存储返回的摄像机参数的结构变量的指针，参考 PsCameraExtrinsicParameters

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.26 Ps2_SetColorPixelFormat

函数原型：

PsReturnStatus Ps2_SetColorPixelFormat(PsDeviceHandle device, uint32_t sessionIndex, PsPixelFormat pixelFormat)

函数功能：

设置像素类型。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pixelFormat **[in]**: 像素类型，参考PsPixelFormat

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.27 Ps2_SetRGBResolution

函数原型：

PsReturnStatus Ps2_SetRGBResolution(PsDeviceHandle device, uint32_t sessionIndex, PsResolution resolution)

函数功能：

设置 RGB 分辨率。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

resolution**[in]**: 分辨率类型，参考PsResolution

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.28 Ps2_GetRGBResolution

函数原型:

PsReturnStatus Ps2_GetRGBResolution(PsDeviceHandle device, uint32_t sessionIndex, uint16_t* resolution)

函数功能:

获取 RGB 分辨率。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

resolution**[out]**：分辨率类型，参考PsResolution

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.29 Ps2_SetWDROutputMode

函数原型：

PsReturnStatus Ps2_SetWDROutputMode(PsDeviceHandle device，

uint32_t sessionIndex， PsWDROutputMode * pWDRMode)

函数功能：

设置 WDR 输出模式

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pWDRMode **[out]**：WDR 输出模式，参考 PsWDROutputMode

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.30 Ps2_GetWDROutputMode

函数原型：

PsReturnStatus Ps2_GetWDROutputMode(PsDeviceHandle device，

uint32_t sessionIndex， PsWDROutputMode * pWDRMode)

函数功能：

获取 WDR 输出模式

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pWDRMode **[out]**：WDR 输出模式，参考 PsWDROutputMode

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.31 Ps2_SetWDRStyle

函数原型：

PsReturnStatus Ps2_SetWDRStyle(PsDeviceHandle device， uint32_t

sessionIndex， PsWDRStyle wdrStyle)

函数功能：

设置 WDR 模式的输出类别，融合或不融合

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

wdrStyle **[in]**：WDR 融合类型，参考 PsWDRStyle

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.32 Ps2_GetMeasuringRange

函数原型：

PsReturnStatus Ps2_GetMeasuringRange(PsDeviceHandle device，uint32_t sessionIndex， PsDepthRange depthRange， PsMeasuringRange* pMeasuringRange)

函数功能：

获取 range 标定范围

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pMeasuringRange**[out]**：标定范围，参考 PsMeasuringRange

返回值：

PsRetOK：调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.33 Ps2_ConvertWorldToDepth

函数原型：

PsReturnStatus Ps2_ConvertWorldToDepth(PsDeviceHandle device，

uint32_t sessionIndex， PsVector3f* pWorldVector， PsDepthVector3*

pDepthVector， int32_t pointCount)

函数功能：

将输入的点坐标从世界坐标系转换为深度坐标系

函数参数：

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pWorldVector **[in]**：存储需要转换的点的世界坐标系三维坐标值的内存指针，

mm 为单位，参考 PsVector3f

pDepthVector **[out]**：存储需要转换输出的点的深度坐标系的坐标值的内存指针

(x， y)以像素为单位，(0，0)点位于图像的左上角

z以mm为单位,为转换点的深度值,参考PsDepthVector3

pointCount **[in]**：需要转换的点的数目

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.34 Ps2_ConvertDepthToWorld

函数原型:

PsReturnStatus Ps2_ConvertDepthToWorld(PsDeviceHandle device，

uint32_t sessionIndex， PsDepthVector3* pDepthVector， PsVector3f*

pWorldVector， int32_t pointCount)

函数功能:

将输入的点坐标从深度坐标系转换为世界坐标系

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pDepthVector **[in]**：存储需要转换输出的点的深度坐标系的坐标值的内存指针

(x，y)以像素为单位，(0，0)点位于图像的左上角

z以mm为单位,为转换点的深度值,参考PsDepthVector3

pWorldVector **[out]**：存储需要转换的点的世界坐标系三维坐标值的内存指针，

以 mm 为单位，参考 PsVector3f

pointCount **[in]**：需要转换的点的数目

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.35 Ps2_ConvertDepthFrameToWorldVector

函数原型:

PsReturnStatus Ps2_ConvertDepthFrameToWorldVector(PsDeviceHandle device， uint32_t sessionIndex， const PsFrame& depthFrame，PsVector3f* pWorldVector)

函数功能:

将输入的深度图像全部点坐标从深度坐标系转换为世界坐标系

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

pixelFormat **[in]**: 像素类型，参考PsPixelFormat

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.36 Ps2_SetSynchronizeEnabled

函数原型:

PsReturnStatus Ps2_SetSynchronizeEnabled (PsDeviceHandle device，

uint32_t sessionIndex， bool bEnabled)

函数功能:

设置输出的 RGB、Depth、IR 等图像是否进行时间同步

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]** : true 表示设为同步， false 表示设置为不同步

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.37 Ps2_GetSynchronizeEnabled

函数原型:

PsReturnStatus Ps2_GetSynchronizeEnabled(PsDeviceHandle device，

uint32_t sessionIndex， bool *bEnabled)

函数功能:

获取输出的 RGB、Depth、IR 等图像进行时间同步的状态

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]** : true 表示设为同步， false 表示设置为不同步

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.38 Ps2_SetDepthDistortionCorrectionEnabled

函数原型：

PsReturnStatus Ps2_SetDepthDistortionCorrectionEnabled

(PsDeviceHandle device， uint32_t sessionIndex， bool bEnabled)

函数功能：

设置启用或禁用深度畸变矫正

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]**: true 表示启用该功能， false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.39 Ps2_GetDepthDistortionCorrectionEnabled

函数原型：

PsReturnStatus Ps2_GetDepthDistortionCorrectionEnabled

(PsDeviceHandle device， uint32_t sessionIndex， bool *bEnabled);

函数功能：

获取深度畸变矫正状态启用或禁用

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.40 Ps2_SetIrDistortionCorrectionEnabled

函数原型：

PsReturnStatus Ps2_SetIrDistortionCorrectionEnabled(PsDeviceHandle device， uint32_t sessionIndex， bool bEnabled)

函数功能：

设置启用或禁用 IR 畸变矫正

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.41 Ps2_GetIrDistortionCorrectionEnabled

函数原型:

PsReturnStatus Ps2_GetIrDistortionCorrectionEnabled(PsDeviceHandle device， uint32_t sessionIndex， bool *bEnabled)

函数功能:

获取 IR 畸变矫正状态启用或禁用

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.42 Ps2_SetRGBDistortionCorrectionEnabled

函数原型：

PsReturnStatus Ps2_SetRGBDistortionCorrectionEnabled (PsDeviceHandle device, uint32_t sessionIndex, bool bEnabled)

函数功能：

设置启用或禁用 RGB 畸变矫正。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]**: true 表示启用该功能, false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.43 Ps2_GetRGBDistortionCorrectionEnabled

函数原型：

PsReturnStatus Ps2_GetRGBDistortionCorrectionEnabled(PsDeviceHandle device, uint32_t sessionIndex, bool *bEnabled)

函数功能：

获取 RGB 畸变矫正状态启用或禁用。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能, false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.44 Ps2_SetComputeRealDepthCorrectionEnabled

函数原型：

PsReturnStatus Ps2_SetComputeRealDepthCorrectionEnabled

(PsDeviceHandle device， uint32_t sessionIndex， bool bEnabled)

函数功能：

设置启用或禁用深度图像拉直操作

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.45 Ps2_GetComputeRealDepthCorrectionEnabled

函数原型：

PsReturnStatus Ps2_GetComputeRealDepthCorrectionEnabled

(PsDeviceHandle device， uint32_t sessionIndex， bool *bEnabled);

函数功能：

获取深度图像拉直操作状态开启或禁用

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.46 Ps2_SetSpatialFilterEnabled

函数原型:

PsReturnStatus Ps2_SetSpatialFilterEnabled(PsDeviceHandle device，

uint32_t sessionIndex， bool bEnabled);

函数功能:

设置启用或禁用深度图像空间滤波操作

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.47 Ps2_GetSpatialFilterEnabled

函数原型：

PsReturnStatus Ps2_GetSpatialFilterEnabled(PsDeviceHandle device，

uint32_t sessionIndex， bool *bEnabled)

函数功能：

获取深度图像空间滤波操作状态启用或禁用

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能， false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.48 Ps2_SetTimeFilterEnabled

函数原型：

PsReturnStatus Ps2_SetTimeFilterEnabled(PsDeviceHandle device，

uint32_t sessionIndex， bool bEnabled)

函数功能：

设置启用或禁用深度图像时间滤波操作

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]**: true 表示启用该功能， false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.49 Ps2_GetTimeFilterEnabled

函数原型：

PsReturnStatus Ps2_GetTimeFilterEnabled(PsDeviceHandle device，

uint32_t sessionIndex， bool *bEnabled)

函数功能：

获取深度图像时间滤波操作状态启用或禁用

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能， false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.50 Ps2_SetMapperEnabledDepthToRGB

函数原型:

PsReturnStatus Ps2_SetMappedEnabledDepthToRGB(PsDeviceHandle device, uint32_t sessionIndex, bool bEnabled)

函数功能:

设置开启或关闭 RGB 图像映射到 Depth 空间。如果开启，可以通过 Ps2_GetFrame 并传入 PsMappedRGBFrame 类型来获取映射到 Depth 空间的 RGB 图像流，其分辨率与 RGB 分辨率一致。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]** : true 表示启用该功能, false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.51 Ps2_GetMapperEnabledDepthToRGB

函数原型:

PsReturnStatus Ps2_GetMapperEnabledDepthToRGB(PsDeviceHandle device, uint32_t sessionIndex, bool *bEnabled)

函数功能:

获取 RGB 图像到 Depth 空间的映射状态开启或关闭。如果开启，可以通过 Ps2_GetFrame 并传入 PsMappedRGBFrame 类型来获取映射到 Depth 空间的 RGB 图像流，其分辨率与 RGB 分辨率一致。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]**: true 表示启用该功能, false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.52 Ps2_SetMapperEnabledRGBToDepth

函数原型:

PsReturnStatus Ps2_SetMappedEnabledRGBToDepth(PsDeviceHandle device, uint32_t sessionIndex, bool bEnabled)

函数功能:

设置开启或关闭 Depth 图像映射到 RGB 空间。如果开启，可以通过 Ps2_GetFrame 并传入 PsMappedDepthFrame 类型来获取映射到 Depth 空间的 RGB 图像流，其分辨率与 Depth 分辨率一致。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]** : true 表示启用该功能, false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.53 Ps2_GetMapperEnabledRGBToDepth

函数原型：

PsReturnStatus Ps2_GetMapperEnabledRGBToDepth(PsDeviceHandle device, uint32_t sessionIndex, bool *bEnabled)

函数功能：

获取 Depth 图像到 RGB 空间的映射状态，开启还是关闭。如果开启，可以通过 Ps2_GetFrame 并传入 PsMappedDepthFrame 类型来获取映射到 Depth 空间的 RGB 图像流，其分辨率与 Depth 分辨率一致。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]** : true 表示启用该功能, false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.54 Ps2_SetMapperEnabledRGBToIR

函数原型：

PsReturnStatus Ps2_SetMappedEnabledRGBToIR(PsDeviceHandle device, uint32_t sessionIndex, bool bEnabled)

函数功能：

设置开启或关闭 IR 图像映射到 RGB 空间。如果开启，可以通过 Ps2_GetFrame 并传入 PsMappedIRFrame 类型来获取映射到 IR 空间的 RGB 图像流，其分辨率与 IR 分辨率一致。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[in]** : true 表示启用该功能, false 表示禁用该功能

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.3.55 Ps2_GetMapperEnabledRGBToIR

函数原型：

PsReturnStatus Ps2_GetMapperEnabledRGBToIR(PsDeviceHandle device, uint32_t sessionIndex, bool *bEnabled)

函数功能：

获取 IR 图像到 RGB 空间的映射状态，开启或关闭。如果开启，可以通过 Ps2_GetFrame 并传入 PsMappedIRFrame 类型来获取映射到 IR 空间的 RGB 图像流，其分辨率与 IR 分辨率一致。

PS:不同型号产品可能不支持此接口，如 DCAM800，请以 Include 中具体型号文件夹下定义为准。

函数参数:

device**[in]**：设备句柄

sessionIndex**[in]**：会话索引，可参考 Ps2_StartStream Ps2_StopStream 说明

bEnabled **[out]** : true 表示启用该功能, false 表示禁用该功能

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 5.1.7 节说明

5.4 V2.0 接口支持

本节主要介绍对 DCAM710 V2.0 版本接口的支持。

5.4.1 已开发应用

基于 V2.0 开发的应用，根据开发平台，将 V3.0 版本 SDK 目录结构 Bin 文件夹下

x64/x84 文件夹中的全部文件， 尤其是 picozense_api.dll、ImgPreProcess.dll，拷贝到应用的执行目录下，即可正常使用。另，如非因特殊需要必须更新，建议沿用 V2.0。

5.4.2 开发支持

V2.0 版本 SDK 开发的应用可以使用 V3.0 SDK 进行基于 V2.0 原有接口的开发，但无法同时使用 Ps2_和 Ps_接口。具体步骤：

1．仍使用 V2.0 版本 SDK 目录结构中的 Include 文件夹中的头文件，切不替换成 V3.0 的 Include 文件夹中头文件；

2．用 V3.0 版本 SDK 目录结构中的 Lib 和 Bin 替换 V2.0 所使用的 Lib 和 Bin 文件夹；

3．应用发布时拷贝 SDK 目录结构 Bin 文件夹下 x64/x84 文件夹中全部文件，尤其注意一定要拷贝 ImgPreProcess.dll；

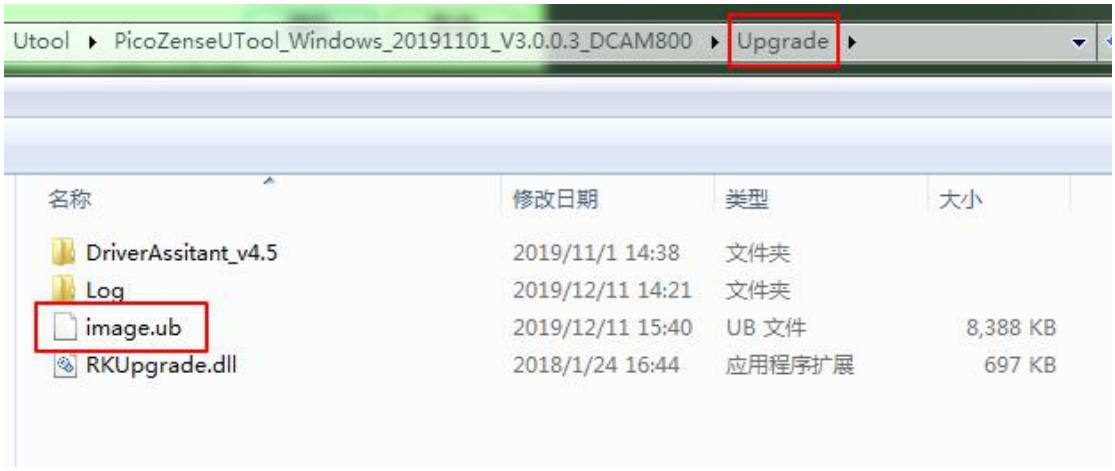
注：因 V3.0 接口更加完善，建议使用新接口（Ps2_)进行开发。

6 固件升级说明

6.1 DCAM800 升级

固件升级需要用到升级工具 UTool，可从我司官网下载，也可联系我司同事获取。具体使用步骤如下：

1．将升级文件（image.ub）放到 Utool 的 Upgrade 目录下：



2．将设备与 PC 相连（直连、DHCP 方式均可）阶段

3．打开 Utool，待“Start”按钮显示为绿色后点击“Start”，将 DeviceInfo 选到“Upgrade Flag”，点击“Write”按钮。之后设备自动进入升级状态，在此状态下设备的红色指示灯常亮。

4．升级完成后设备的红色指示灯会一直闪烁，此时断电重启设备就可以正常使用了。

7 FAQ

7.1 USB 方式

7.2 网口方式

7.2.1 SDK 无法打开相机的排查步骤

1、首先确认相机是否被修改过静态 IP，若不确定请使用 DHCP 的方式打开相机，参照 3.2.1.2，否则进行下一步。

2、将相机与 PC 直连，打开控制面板的“网络连接”，确定以太网是否连接成功，如下图所示。若以太网网络连接失败，请排查物理通路是否正常，若连接成功则进行下一步。

图 7.1 网络连接

3、在 PC 端 ping 相机的静态 IP（默认为 192.168.1.101），若无法 ping 通则排查 PC 的 IP 地址配置，参见 3.2.1.2。若成功 ping 通则进行下一步。

图 7.2 ping

4、在控制面板的“Windows Defender 防火墙”中点击“允许应用通过防火墙” ，为 SDK 添加通过防火墙的权限。

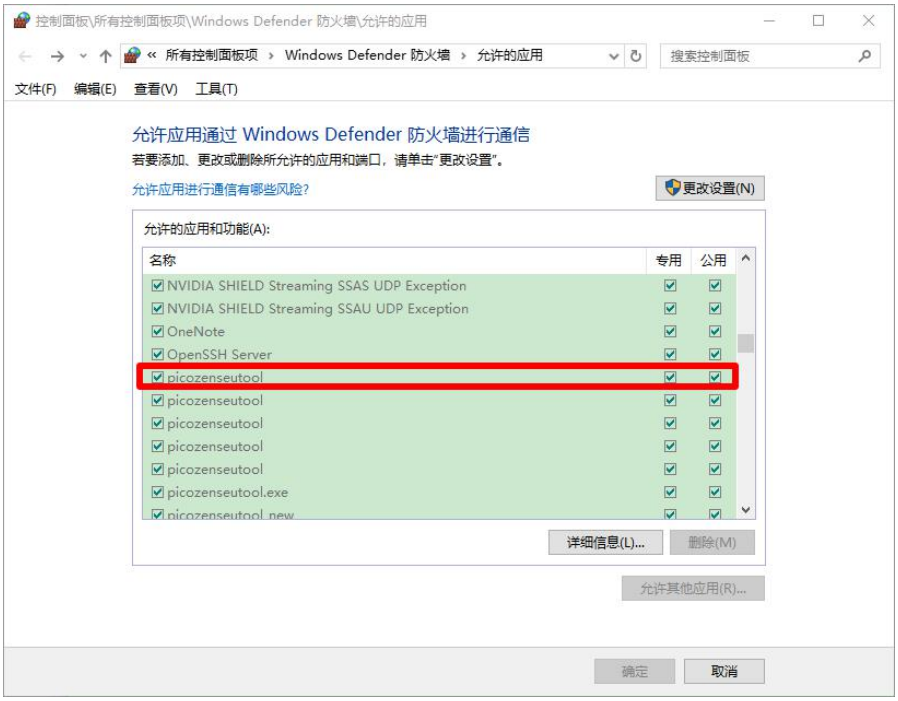


图 7.3 防火墙设置

或者直接将 windows 防火墙关闭。

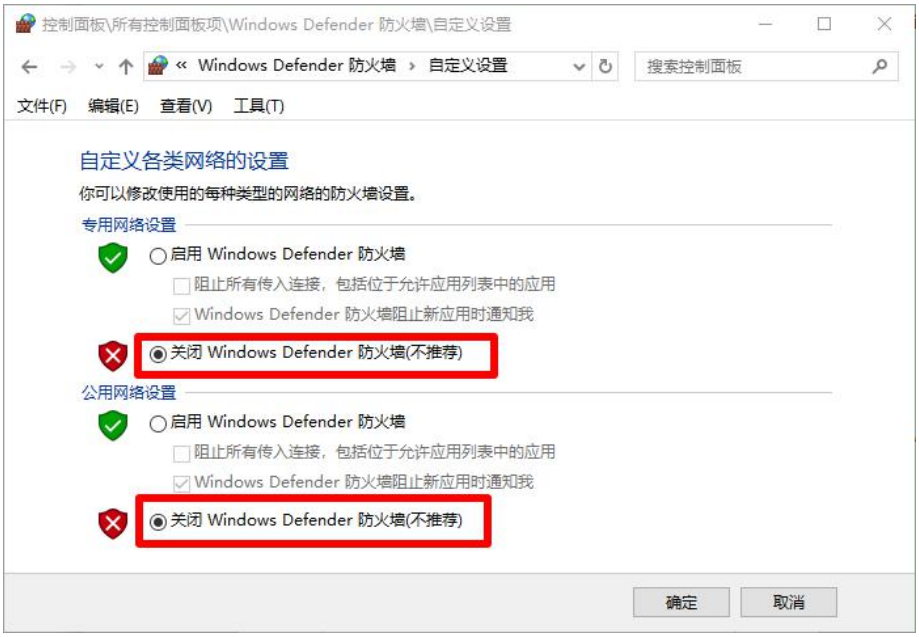


图 7.4 关闭防火墙