

Vzense TOF Camera

SDK 开发者指南

Linux

2020.06

Vzense Technology ,Inc.

关于本指南

本指南文档主要介绍如何使用 Vzense TOF Camera 及 Vzense 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 Vzense 产品
3	安装	介绍 Vzense 产品及 SDK 的安装
4	SDK 使用说明	介绍如何使用 Vzense SDK
5	SDK 接口介绍	介绍 Vzense SDK 的接口
6	固件升级说明	介绍 Vzense 产品固件升级
7	API 调用示例	介绍常用功能的初始化流程
8	FAQ	

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
2020.12.04	V3.2.2	支持 DCAM550
2021.01.12	V3.4.8	支持 DCAM560

目 录

1 概述	11
2 支持设备	12
2.1 DCAM710	12
2.2 DCAM550U	12
2.3 DCAM550P	13
2.4 DCAM550E	13
2.5 DCAM560CPRO	14
2.6 DCAM560CLITE	14
3 安装	15
3.1 推荐系统配置	15
3.2 安装说明	15
3.2.1 硬件连接	15
3.2.2 软件环境配置	19
4 SDK 使用说明	23
4.1 SDK 目录结构	23
4.2 Tool 使用说明	24

4.3 开发流程	24
4.3.1 项目配置	24
4.3.2 接口调用流程	25
4.4 SDK Sample 使用流程	27
5 SDK 接口介绍	29
5.1 Enum 数据类型	29
5.1.1 PsDepthRange	29
5.1.2 PsDataMode	30
5.1.3 PsPropertyType	31
5.1.4 PsFrameType	31
5.1.5 PsSensorType	32
5.1.6 PsPixelFormat	32
5.1.7 PsReturnStatus	33
5.1.8 PsWDRTotalRange	35
5.1.9 PsWDRStyle	35
5.1.10 PsResolution	36
5.2 Struct 数据类型	36

5.2.1 PsRGB888Pixel	36
5.2.2 PsBGR888Pixel	37
5.2.3 PsVector3f	37
5.2.4 PsDepthVector3	37
5.2.5 PsCameraParameters	38
5.2.6 PsCameraExtrinsicParameters	39
5.2.7 PsFrame	39
5.2.8 PsWDROutputMode	40
5.2.9 PsMeasuringRange	41
5.2.10 PsDeviceInfo	42
5.2.11 PsDataModeList	42
5.2.12 PsDepthRangeList	43
5.2.13 PsFrameReady	43
5.3 API 接口	44
5.3.1 Ps2_Initialize	44
5.3.2 Ps2_Shutdown	44
5.3.3 Ps2_GetDeviceCount	45

5.3.4 Ps2_GetDeviceListInfo	46
5.3.5 Ps2_GetDeviceInfo	46
5.3.6 Ps2_OpenDevice	47
5.3.7 Ps2_CloseDevice	48
5.3.8 Ps2_StartStream	48
5.3.9 Ps2_StopStream	49
5.3.10 Ps2_ReadNextFrame	50
5.3.11 Ps2_GetFrame	51
5.3.12 Ps2_SetDataMode	52
5.3.13 Ps2_GetDataMode	53
5.3.14 Ps2_GetDepthRange	53
5.3.15 Ps2_SetDepthRange	54
5.3.16 Ps2_GetThreshold	55
5.3.17 Ps2_SetThreshold	56
5.3.18 Ps2_GetPulseCount	56
5.3.19 Ps2_SetPulseCount	57
5.3.20 Ps2_GetGMMGain	58

5.3.21 Ps2_SetGMMGain	59
5.3.22 Ps2_GetProperty	60
5.3.23 Ps2_SetProperty	61
5.3.24 Ps2_GetCameraParameters	61
5.3.25 Ps2_GetCameraExtrinsicParameters	62
5.3.26 Ps2_SetColorPixelFormat	63
5.3.27 Ps2_SetRGBResolution	64
5.3.28 Ps2_GetRGBResolution	65
5.3.29 Ps2_SetWDROutputMode	66
5.3.30 Ps2_GetWDROutputMode	66
5.3.31 Ps2_SetWDRStyle	67
5.3.32 Ps2_GetMeasuringRange	68
5.3.33 Ps2_ConvertWorldToDepth	69
5.3.34 Ps2_ConvertDepthToWorld	70
5.3.35 Ps2_ConvertDepthFrameToWorldVector	71
5.3.36 Ps2_SetSynchronizeEnabled	72
5.3.37 Ps2_GetSynchronizeEnabled	72

5.3.38 Ps2_SetDepthDistortionCorrectionEnabled	73
5.3.39 Ps2_GetDepthDistortionCorrectionEnabled	74
5.3.40 Ps2_SetIrDistortionCorrectionEnabled	75
5.3.41 Ps2_GetIrDistortionCorrectionEnabled	75
5.3.42 Ps2_SetRGBDistortionCorrectionEnabled	76
5.3.43 Ps2_GetRGBDistortionCorrectionEnabled	77
5.3.44 Ps2_SetComputeRealDepthCorrectionEnabled	78
5.3.45 Ps2_GetComputeRealDepthCorrectionEnabled	79
5.3.46 Ps2_SetSpatialFilterEnabled	79
5.3.47 Ps2_GetSpatialFilterEnabled	80
5.3.48 Ps2_SetTimeFilterEnabled	81
5.3.49 Ps2_GetTimeFilterEnabled	82
5.3.50 Ps2_SetDepthFrameEnabled	82
5.3.51 Ps2_SetIrFrameEnabled	83
5.3.52 Ps2_SetRgbFrameEnabled	84
5.3.53 Ps2_SetImageMirror	85
5.3.54 Ps2_SetImageRotation	85

5.3.55 Ps2_SetMapperEnabledDepthToRGB	86
5.3.56 Ps2_GetMapperEnabledDepthToRGB	87
5.3.57 Ps2_SetMapperEnabledRGBToDepth	88
5.3.58 Ps2_GetMapperEnabledRGBToDepth	89
5.3.59 Ps2_SetHotPlugStatusCallback	90
5.3.60 Ps2_SetHotPlugStatusCallback_	90
5.3.61 Ps2_GetWDRPulseCount	91
5.3.62 Ps2_SetWDRPulseCount	92
5.3.63 Ps2_GetSerialNumber	93
5.3.64 Ps2_GetFirmwareVersionNumber	93
5.3.65 Ps2_SetDSPEnabled	94
5.3.66 Ps2_GetDSPEnabled	95
5.3.67 Ps2_SetSlaveModeEnabled	96
5.3.68 Ps2_SetTofFrameRate	97
5.3.69 Ps2_GetTofFrameRate	97
5.3.70 Ps2_SetStandByEnabled	98
5.3.71 Ps2_OpenDeviceByAlias	99

5.3.72 Ps2_SetWaitTimeOfReadNextFrame	100
5.3.73 Ps2_GetSDKVersion	100
5.3.74 Ps2_GetMappedPointDepthToRGB	101
5.3.75 Ps2_SetSlaveTrigger	102
5.3.76 Ps2_GetDeviceIP	103
5.3.77 Ps2_GetDeviceMAC	103
5.3.78 Ps2_SetRGBBrightness	104
5.3.79 Ps2_GetRGBBrightness	105
5.3.80 Ps2_SetRGBExposure	106
5.3.81 Ps2_GetRGBExposure	107
5.3.82 Ps2_RebootCamera	107
6 固件升级说明	108
7 API 调用示例	108
8 FAQ	110
8.1 USB 方式	110
8.1.1 无法识别 USB 设备	110

1 概述

Vzense TOF Camera 是 Vzense 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。

Vzense SDK 是基于 Vzense 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。

SDK 下载链接:

国内:

https://gitee.com/Vzense/Vzense_SDK_windows

海外:

https://github.com/Vzense/Vzense_SDK_Windows

2 支持设备

2.1 DCAM710



图 2.1 Vzense TOF RGBD Camera: DCAM710

DCAM710 是 Vzense 一款通用的开发套件，基于 TOF 技术的 3D+RGB 相机，可提供高精度的深度距离图像、IR 图像和 RGB 图像数据，USB2.0 接口即插即用。适用于不同 3D 应用场景，如面部识别、手势识别、骨骼点获取、深度图像实时渲染、SLAM 等等。

2.2 DCAM550U



图 2.2 Vzense TOF Camera: DCAM550U

DCAM550U 是 Vzense 针对工业应用场景开发的一款基于 TOF 技术的 3D 相机，带 Vzense Technology, Inc

锁的 Type A USB2.0 接口，支持外部触发与多相机同步，满足不同使用方式。

2.3 DCAM550P



图 2.3 Vzense TOF Camera: DCAM550P

DCAM550P 是 Vzense 针对工业应用场景开发的一款基于 TOF 技术的 3D 相机，支持 100M Ethernet 可以满足更远的传输距离，支持外部触发与多相机同步。

2.4 DCAM550E



图 2.4 Vzense TOF Camera: DCAM550E

DCAM550E 是 Vzense 针对工业应用场景开发的一款基于 TOF 技术的 3D 相机，支
Vzense Technology, Inc

持 100M Ethernet, 同时支持外部触发与多相机同步, IP67 防护等级, 可以从容应对不同的工业场景需求。

2.5 DCAM560CPRO



图 2.5 Vzense TOF RGBD Camera: DCAM560CPRO

DCAM560CPRO 是 Vzense 针对工业应用场景开发的一款基于 TOF 技术的 3D+RGB 相机, 支持 1000M Ethernet, 同时支持外部触发与多相机同步, IP67 防护等级。

2.6 DCAM560CLITE



图 2.6 Vzense TOF RGBD Camera: DCAM560CLITE

DCAM560CLITE 是 Vzense 针对工业应用场景开发的一款基于 TOF 技术的 3D+RGB 相机, 支持 1000M Ethernet, 同时支持外部触发与多相机同步。

3 安装

3.1 推荐系统配置

配置项	推荐配置
操作系统	Ubuntu 20.04 64 位
	Ubuntu 18.04 64 位
	Ubuntu 16.04 64 位
	ARM Linux(AArch64)
内存	4G 及以上

3.2 安装说明

3.2.1 硬件连接

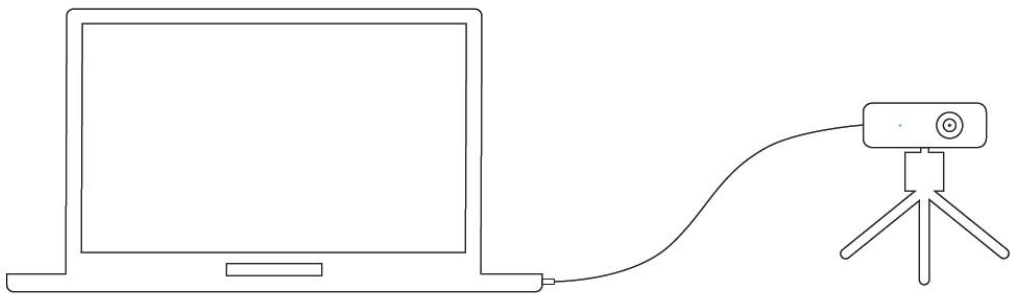
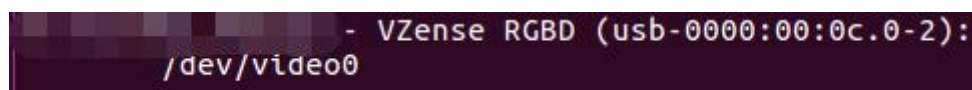


图 3.1 硬件模组安装示意图

3.2.1.1 USB 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 Ubuntu 系统下，在终端中运行 `sudo apt-get install v4l-utils`，安装 v4l 相关工具。安装成功后，运行 `v4l2-ctl --list-devices`，如果提示如下图信息，表示系统已识别连接的设备。



```
- VZense RGBD (usb-0000:00:0c.0-2):  
/dev/video0
```

图 3.2 Vzense RGBD Camera

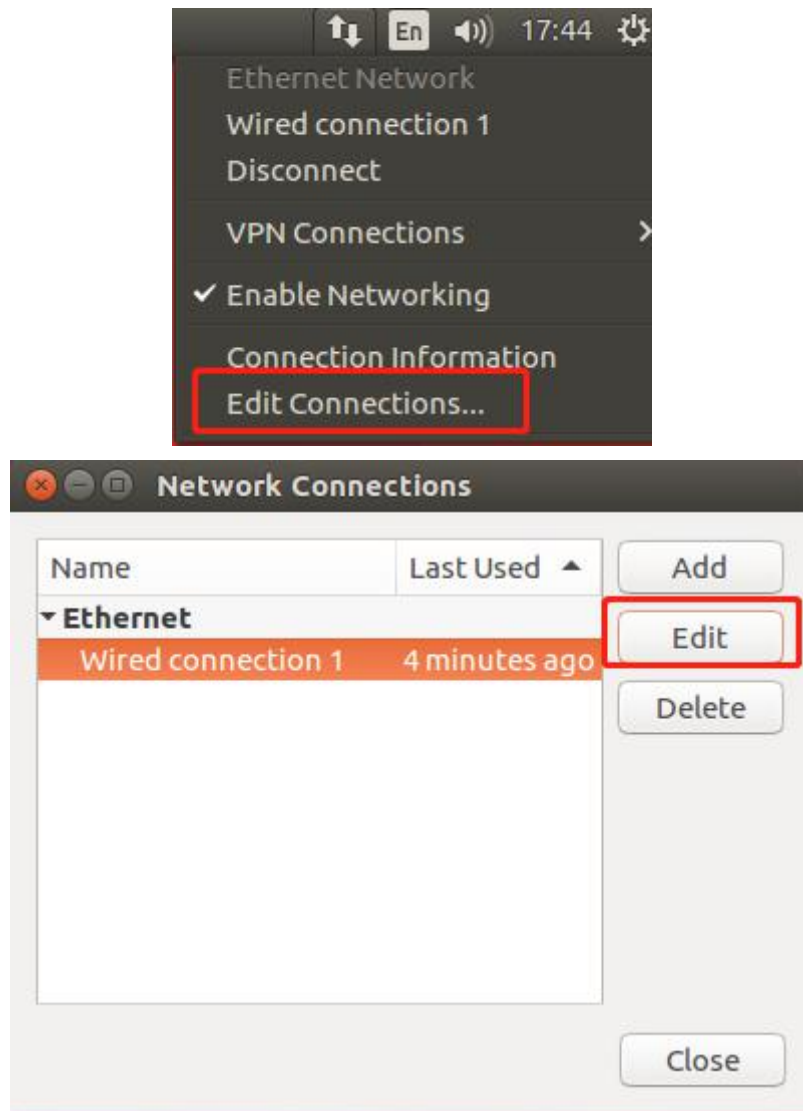
3.2.1.2 网口方式

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）。



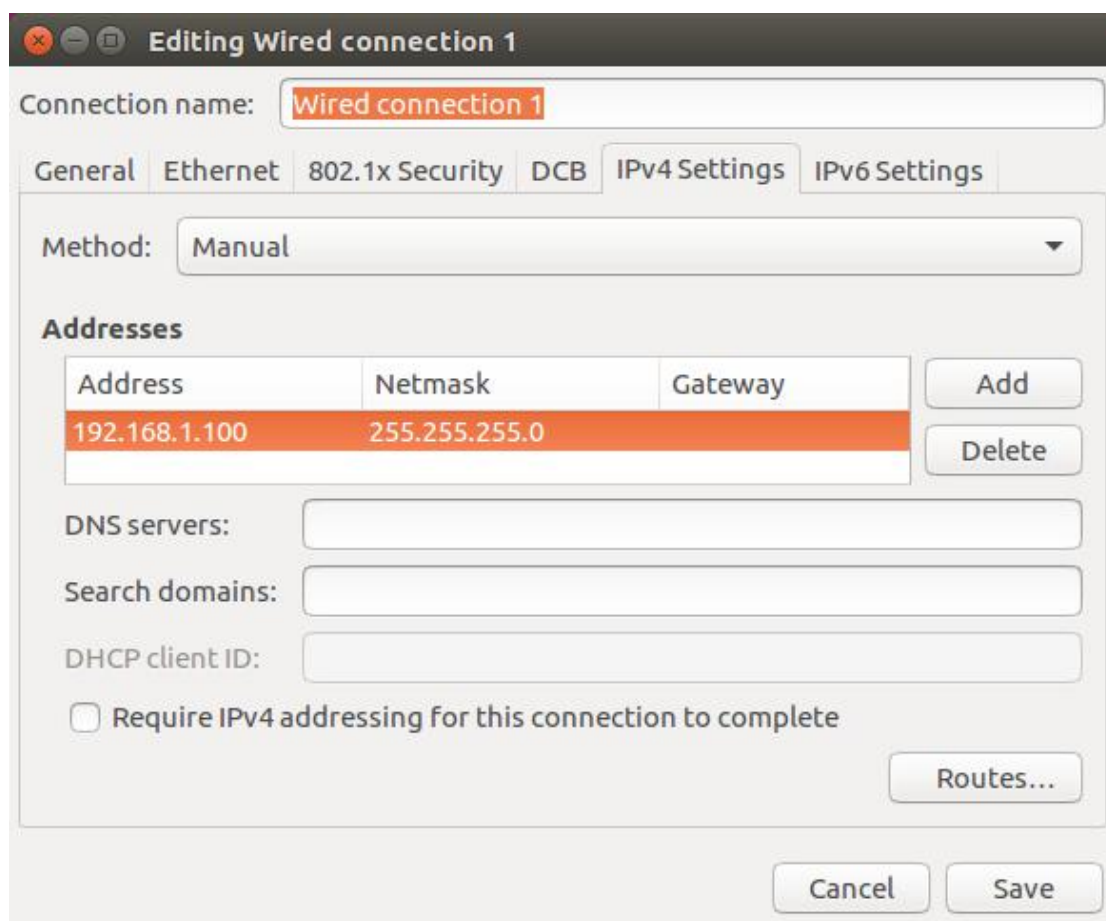


图 3.3 直连方式

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。

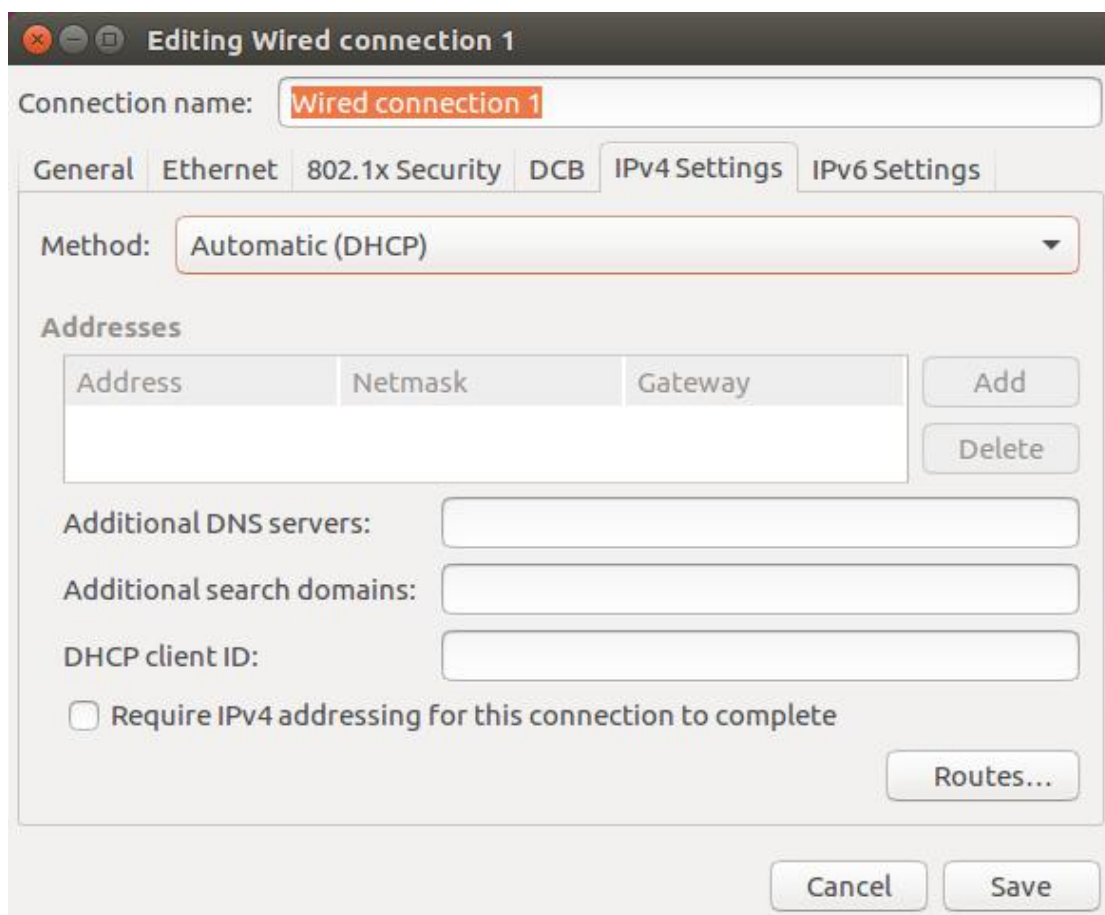


图 3.4 DHCP 方式

注意：PC 端使用的网卡、路由器、交换机都要满足千兆要求。

3.2.2 软件环境配置

Linux 系统需要对显卡驱动做的配置如下：

NVIDIA 显卡

对于 NVIDIA 显卡的电脑，请检查驱动配置，如下图。

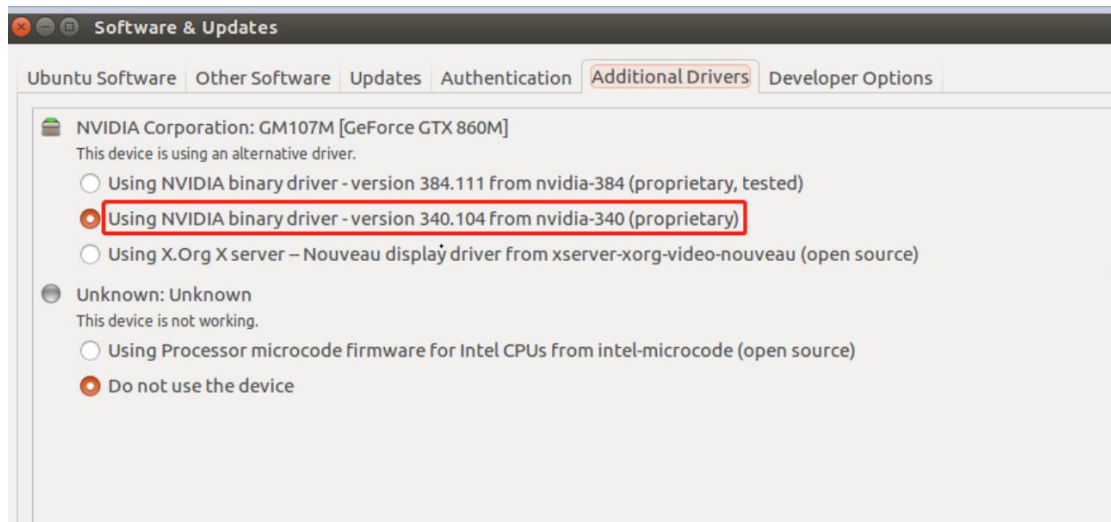


图 3.5 显卡驱动设置

安装工具

安装 VDPAU: libvdpau-dev, vdpauinfo。

```
sudo apt-get install libvdpau-dev
```

```
sudo apt-get install vdpauinfo
```

安装完成后运行 vdpauinfo，如下图：

```

teemo@teemo-ASM100:~/Downloads/opencv-2.4.9/build$ vdpauinfo
display: :0 screen: 0
API version: 1
Information string: NVIDIA VDPAU Driver Shared Library 340.104 Thu Sep 14 16:45:03 PDT 2017

Video surface:

name      width height types
-----
420       4096  4096  NV12 YV12
422       4096  4096  UYVY YUYV

Decoder capabilities:

name                      level macbs width height
-----
MPEG1                     0 65536 4080 4080
MPEG2_SIMPLE              3 65536 4080 4080
MPEG2_MAIN                3 65536 4080 4080
H264_BASELINE             --- not supported ---
H264_MAIN                 41 65536 4096 4096
H264_HIGH                  41 65536 4096 4096
VC1_SIMPLE                 1 8190 2048 2048
VC1_MAIN                   2 8190 2048 2048
VC1_ADVANCED               4 8190 2048 2048
MPEG4_PART2_SP             3 8192 2048 2048
MPEG4_PART2_ASP            5 8192 2048 2048
DIVX4_QMOBILE              0 8192 2048 2048
DIVX4_MOBILE              0 8192 2048 2048
DIVX4_HOME_THEATER         0 8192 2048 2048
DIVX4_HD_1080P             0 8192 2048 2048
DIVX5_QMOBILE              0 8192 2048 2048
DIVX5_MOBILE              0 8192 2048 2048
DIVX5_HOME_THEATER         0 8192 2048 2048
DIVX5_HD_1080P             0 8192 2048 2048
H264_CONSTRAINED_BASELINE --- not supported ---
H264_EXTENDED             --- not supported ---
H264_PROGRESSIVE_HIGH     --- not supported ---
H264_CONSTRAINED_HIGH     --- not supported ---
H264_HIGH_444_PREDICTIVE  --- not supported ---
HEVC_MAIN                  --- not supported ---
HEVC_MAIN_10               --- not supported ---
HEVC_MAIN_STILL            --- not supported ---
HEVC_MAIN_12               --- not supported ---
HEVC_MAIN_444              --- not supported ---

```

图 3.6 安装 VDPAU

Intel 显卡

对于 Intel 显卡的电脑，需要安装以下工具：

```
sudo add-apt-repository ppa:nilarimogard/webupd8
```

```
sudo apt-get update
```

```
sudo apt-get install libvdpau-va-gl1
```

```
sudo apt-get install i965-va-driver
```

```
sudo apt-get install vdpauinfo
```

安装完成后运行 vdpauinfo，成功如图 3.6，如果出现如下错误：

display: :0 screen: 0

Failed to open VDPAU backend libvdpau_i965.so: cannot open shared object

file: No such file or directory

Error creating VDPAU device: 1

则执行如下命令：

```
cd /usr/lib/x86_64-linux-gnu/vdpau/
```

```
sudo ln -s libvdpau_va_gl.so libvdpau_i965.so
```

```
sudo ln -s libvdpau_va_gl.so.1 libvdpau_i965.so.1
```

4 SDK 使用说明

4.1 SDK 目录结构

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

Linux 目录结构如下图所示：

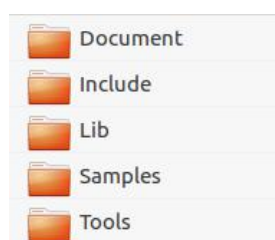
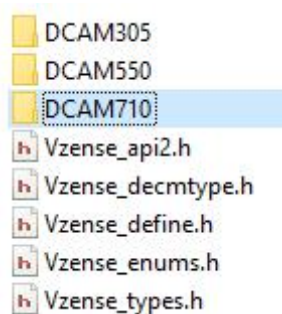


图 4.1 Linux SDK 目录结构

Document 包含 SDK 的说明文档。

Include 主要包含 SDK 的通用头文件（Vzense_api2.h 、 Vzense_define.h 、 Vzense_enums.h 、 Vzense_types.h、 Vzense_decmttype.h）和包含不同型号产品所需特定头文件的文件夹，如 DCAM710。如下图



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

Samples 主要包含使用 Vzense SDK 开发的例程。

Tools 中的 FrameViewer 可查看 Vzense 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。

install.sh 运行 FrameViewer 之前，需先运行 install.sh 以配置所需环境。在 SDK 路径下执行 `sudo ./install.sh`。

4.2 Tool 使用说明

将 Vzense 深度摄像头连接到 PC，运行 Tools 目录里匹配相机的 FrameViewer_DCAMXXX，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 Vzense 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

Linux 下使用 Vzense Linux SDK 开发新的项目，需要在 Makefile 中将 SDK 中的 Include 目录加入到包含路径，并且需要将 Lib/x64 目录加入到链接搜索路径中，并链接

libvzense_api.so, 如-l.././Include, -L.././Lib/x64, -lvzense_api。可参考 Samples 里 Makefile 的配置。

4.3.2 接口调用流程

Vzense SDK 的 API 接口调用流程图如下：

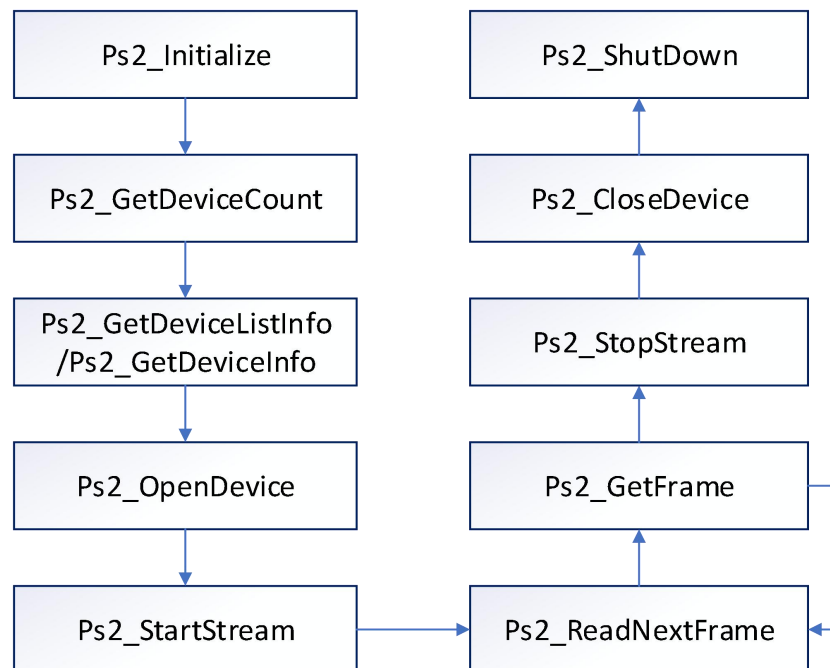


图 4.4 SDK 接口调用流程

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_StartStream](#) 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_StartStream](#) 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：

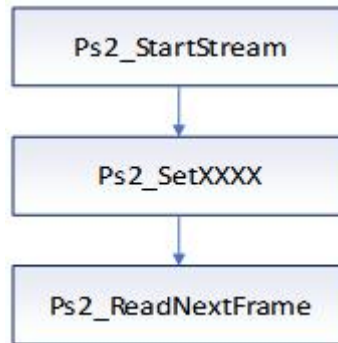


图 4.5 取图前，修改默认设置 SDK 接口调用流程

4.4 SDK Sample 使用流程

Vzense 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. Include 使用时, 需根据不同型号设备, 修改 Vzense_decmtpe.h 中的宏定义状态, 如使用 DCAM550, 仅保留 **DCAM_550** 宏定义, 注释掉其他定义。另, DCAM550 的接口同样适用于 DCAM800LITE, DCAM800, DCAM500, 不再单独设定编译类型及文件夹。

```

#ifndef VZENSE_DECMTYPE_H
#define VZENSE_DECMTYPE_H

// #define DCAM_550
// #define DCAM_305
#define DCAM_710

#endif /* VZENSE_DECMTYPE_H */
  
```

2. Samples 目录中的 FrameViewer 工程编译时, FrameViewer.cpp 已根据 Vzense_decmtpe.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。

3. 在 4.1 中运行 install.sh 会自动拷贝使用 opencv3.4.1, 如需使用其他版本请自行

替换，并修改 Makefile 文件。SDK 中使用 OpenCV 仅仅是显示彩色深度图像，如果不需要图像显示，可以根据自己需求进行修改。

5 SDK 接口介绍

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:

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](#)

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

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

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

PsRetCurrentVersionNotSupport 表示当前版本不支持

PsRetUpgradeImgError 表示标识升级镜像错误

PsRetUpgradeImgPathTooLong 表示镜像路径太长

PsRetUpgradeCallbackNotSet 表示未设置升级回调函数

PsRetNoAdapterConnected 表示没接电源

PsRetReInitialized 表示重复初始化

PsRetNoInitialized 表示没做初始化

PsRetCameraOpened 表示相机没打开

PsRetCmdError 表示命令下发失败

PsRetCmdSyncTimeOut 表示命令发送成功，同步匹配失败

PsRetIPNotMatch 表示设备 IP 与主机 IP 不在同一网段

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 550）或支持的 RGB 分辨率不同（如 560CPRO），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 DCAM550，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 DCAM550，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，表示世界坐标系的点坐标值，单位为 mm。

成员:

float x, y, z

5.2.4 PsDepthVector3

功能:

Vzense Technology, Inc

Copyright 2018

深度图像向量，表示相机图像坐标系的坐标值。

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 R 与 T，用于 depth 与 rgb 图像的对齐，参考公式如下：

$$\begin{bmatrix} x_{rgb} \\ y_{rgb} \\ z_{rgb} \end{bmatrix} = \begin{bmatrix} r0 & r1 & r2 \\ r3 & r4 & r5 \\ r6 & r7 & r8 \end{bmatrix} \times \begin{bmatrix} x_{depth} \\ y_{depth} \\ z_{depth} \end{bmatrix} + \begin{bmatrix} t0 \\ t1 \\ t2 \end{bmatrix}$$

成员：

rotation[9] : 3×3 的旋转矩阵

translation[3] : 3×1 平移矩阵

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

depthMaxFar 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 表示对齐后 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：

Vzense Technology, Inc

Copyright 2018

```
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:

Vzense Technology, Inc

Copyright 2018

```
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:

打开设备

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:

Vzense Technology, Inc

Copyright 2018


```
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 [in]: 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:

获取深度图像的背景滤波阈值大小

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：

设置深度图像的背景滤波阈值大小

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:

获取深度图像的脉冲计数个数, PulseCount 用于“曝光时间”调节

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:

设置深度图像的脉冲计数个数, PulseCount 用于“曝光时间”调节

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:

获取灰度图像的 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:

设置灰度图像的 Gamma 增益值, Gamma 值仅用于调节 IR 图的数字增益

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，不推荐用户使用

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

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:

Vzense Technology, Inc

Copyright 2018

```
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 DCAM550, 请以 Include 中具体型号文件夹下定义为准。

函数参数:

Vzense Technology, Inc

Copyright 2018

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 DCAM550, 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 DCAM550, 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：

Vzense Technology, Inc

Copyright 2018

```
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 DCAM550, 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 DCAM550, 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_SetDepthFrameEnabled

函数原型:

```
PsReturnStatus Ps2_SetTDepthFrameEnabled(PsDeviceHandle device,  
uint32_t sessionIndex, bool bEnabled)
```

函数功能:

设置启用或禁用深度图像

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

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_SetIrFrameEnabled

函数原型:

```
PsReturnStatus Ps2_SetIrFrameEnabled(PsDeviceHandle device, uint32_t  
sessionIndex, bool bEnabled)
```

函数功能:

设置启用或禁用 IR 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

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

5.3.52 Ps2_SetRgbFrameEnabled

函数原型:

```
PsReturnStatus Ps2_SetTRgbFrameEnabled(PsDeviceHandle device,  
uint32_t sessionIndex, bool bEnabled)
```

函数功能:

设置启用或禁用 RGB 图像

PS:不同型号产品可能不支持此接口, 如 DCAM550, 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_SetImageMirror

函数原型：

```
PsReturnStatus Ps2_SetImageMirror(PsDeviceHandle device, uint32_t  
sessionIndex, int32_t type)
```

函数功能：

设置图像镜像操作

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

type[in]：1 左右镜像，2 上下镜像，3 左右上下都做镜像

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_SetImageRotation

函数原型：

```
PsReturnStatus Ps2_SetImageRotation(PsDeviceHandle device, uint32_t
```

sessionIndex, int32_t type)

函数功能:

设置图像旋转操作

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

type[in]: 0 逆时针 90°, 1 逆时针 180°, 2 逆时针 270°

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_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 DCAM550, 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

5.3.56 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 DCAM550, 请以 Include 中具体型号文件夹下定义为准。

函数参数:

Vzense Technology, Inc

Copyright 2018

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

5.3.57 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 DCAM550, 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

5.3.58 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 DCAM550, 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

5.3.59 Ps2_SetHotPlugStatusCallback

函数原型:

```
PsReturnStatus Ps2_SetHotPlugStatusCallback(PtrHotPlugStatusCallback  
pCallback);
```

函数功能:

设置热插拔的回调函数

函数参数:

pCallback[in]: 回调函数

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.60 Ps2_SetHotPlugStatusCallback_

函数原型:

```
PsReturnStatus Ps2_SetHotPlugStatusCallback(PtrHotPlugStatusCallback_  
pCallback, void* contex);
```

函数功能:

设置热插拔的回调函数，C++的使用方式

函数参数:

pCallback[in]: 回调函数

context[in]: 指向一个 C++ 类对象

返回值:

PsRetOK: 调用成功

其他值: 调用失败，可参考 [5.1.7](#) 节说明

5.3.61 Ps2_GetWDRPulseCount

函数原型:

```
PsReturnStatus Ps2_GetWDRPulseCount(PsDeviceHandle device, uint32_t  
sessionIndex, PsWDRPulseCount* pwdrPulseCount)
```

函数功能:

获取 WDR 模式下的 pulsecount 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

pswdrPulseCount[**out**] : pulsecount 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

5.3.62 Ps2_SetWDRPulseCount

函数原型:

```
PsReturnStatus Ps2_SetWDRPulseCount(PsDeviceHandle device, uint32_t  
sessionIndex, PsWDRPulseCount wdrPulseCount)
```

函数功能:

设置 WDR 模式下的 pulsecount 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

wdrPulseCount[**in**] : pulsecount 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

5.3.63 Ps2_GetSerialNumber

函数原型:

```
PsReturnStatus Ps2_GetSerialNumber(PsDeviceHandle device, uint32_t  
sessionIndex, char* sn, int length)
```

函数功能:

获取设备 SN 序列号

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

sn[out]: sn 值

length[in]: 最大长度 63

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.64 Ps2_GetFirmwareVersionNumber

函数原型:

```
PsReturnStatus Ps2_GetFirmwareVersionNumber(PsDeviceHandle device,
```

```
uint32_t sessionIndex, char* fw, int length)
```

函数功能:

获取固件版本号

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

fw[out]: fw 值

length[in]: 最大长度 63

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.65 Ps2_SetDSPEnabled

函数原型:

```
PsReturnStatus Ps2_SetDSPEnabled(PsDeviceHandle device, uint32_t  
sessionIndex, bool bEnabled)
```

函数功能:

设置启用或禁用 DSP 模式

PS:不同型号产品可能不支持此接口, 如 DCAM550, 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

5.3.66 Ps2_GetDSPEnabled

函数原型:

```
PsReturnStatus Ps2_GetDSPEnabled(PsDeviceHandle device, uint32_t  
sessionIndex, bool* bEnabled)
```

函数功能:

获取是否启用 DSP 模式

PS:不同型号产品可能不支持此接口, 如 DCAM550, 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

5.3.67 Ps2_SetSlaveModeEnabled

函数原型:

```
PsReturnStatus Ps2_SetSlaveModeEnabled(PsDeviceHandle device,  
uint32_t sessionIndex, bool bEnabled)
```

函数功能:

设置启用或禁用触发模式

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

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

5.3.68 Ps2_SetTofFrameRate

函数原型:

```
PsReturnStatus Ps2_SetTofFrameRate(PsDeviceHandle device, uint32_t  
sessionIndex, uint8_t value)
```

函数功能:

设置 Tof 帧率

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

value[in]: 帧率值 3,5,6,10,15,30。以实际支持为准

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.69 Ps2_GetTofFrameRate

函数原型:

```
PsReturnStatus Ps2_GetTofFrameRate(PsDeviceHandle device, uint32_t  
sessionIndex, uint8_t *value)
```

函数功能:

获取 Tof 帧率

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

value[in]: 帧率值 3,5,6,10,15,30。以实际支持为准

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.70 Ps2_SetStandByEnabled

函数原型:

```
PsReturnStatus Ps2_SetStandByEnabled(PsDeviceHandle device, uint32_t  
sessionIndex, bool bEnabled)
```

函数功能:

设置启用或禁用低功耗模式

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

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

5.3.71 Ps2_OpenDeviceByAlias

函数原型:

```
PsReturnStatus Ps2_OpenDeviceByAlias(const char* alias, PsDeviceHandle  
*pDevice)
```

函数功能:

以别名的方式打开设备

函数参数:

alias[in]: 设备的别名, 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

5.3.72 Ps2_SetWaitTimeOfReadNextFrame

函数原型:

```
PsReturnStatus Ps2_SetWaitTimeOfReadNextFrame(PsDeviceHandle  
device, uint32_t sessionIndex, uint16_t time)
```

函数功能:

设置读取图像超时时间

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

time[in]: 超时时间

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.73 Ps2_GetSDKVersion

函数原型:

```
PsReturnStatus Ps2_GetSDKVersion(char* version, int length);
```

函数功能:

获取 SDK 的版本号

函数参数:

version[**out**]: 版本号

length[in]: 最大长度 63

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.74 Ps2_GetMappedPointDepthToRGB

函数原型:

```
PsReturnStatus Ps2_GetMappedPointDepthToRGB(const PsDeviceHandle  
device, const uint32_t sessionIndex, const PsDepthVector3 depthPoint,  
const PsVector2u16 rgbSize, PsVector2u16* pPosInRGB)
```

函数功能:

获取 Depth 图像到 RGB 空间的点对点映射

PS:不同型号产品可能不支持此接口, 如 DCAM550, 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

depthPoint[**in**] : depth 图上的点

rgbSize[**in**] : RGB 图的尺寸

pPosInRGB[**out**] : RGB 图上的点坐标

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.75 Ps2_SetSlaveTrigger

函数原型:

```
PsReturnStatus Ps2_SetSlaveTrigger(PsDeviceHandle device, uint32_t  
sessionIndex)
```

函数功能:

触发模式下, 设置单次触发

PS:不同型号产品可能不支持此接口, 如 DCAM710, 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

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.76 Ps2_GetDeviceIP

函数原型:

```
PsReturnStatus Ps2_GetDeviceIP(const char* uri, char* ip)
```

函数功能:

获取设备 IP, 仅支持网线连接方式

PS:不同型号产品可能不支持此接口, 如 DCAM710, 请以 Include 中具体型号文件夹下定义为准。

函数参数:

uri[in]: 设备的标识, 可从设备信息中得到

ip[out]: 返回设备的 ip, 设置数组大小为 17 个字节, 最后一位设置为 '\0'

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.77 Ps2_GetDeviceMAC

函数原型:

```
PsReturnStatus Ps2_GetDeviceMAC(PsDeviceHandle device, uint32_t
```

sessionIndex, char* mac)

函数功能:

获取设备 MAC 地址, 仅支持网线连接方式

PS:不同型号产品可能不支持此接口, 如 DCAM710, 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

mac[out]: 返回设备的 mac, 设置数组大小为 18 个字节, 最后一位设置为 '\0'

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.78 Ps2_SetRGBBrightness

函数原型:

```
PsReturnStatus Ps2_SetRGBBrightness(PsDeviceHandle device, uint32_t  
sessionIndex, char value)
```

函数功能:

设置 RGB 亮度, 仅支持 560 系列设备

PS:不同型号产品可能不支持此接口, 如 DCAM710, 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

value[in]: 亮度值, 范围[-64,64]

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.79 Ps2_GetRGBBrightness

函数原型:

```
PsReturnStatus Ps2_GetRGBBrightness(PsDeviceHandle device, uint32_t  
sessionIndex, char* value)
```

函数功能:

返回 RGB 亮度, 仅支持 560 系列设备

PS:不同型号产品可能不支持此接口, 如 DCAM710, 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

value[**out**]: 亮度值, 范围[-64,64]

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

5.3.80 Ps2_SetRGBExposure

函数原型:

```
PsReturnStatus Ps2_SetRGBExposure(PsDeviceHandle device, uint32_t  
sessionIndex, char value)
```

函数功能:

设置 RGB 曝光时间, 仅支持 560 系列设备

PS:不同型号产品可能不支持此接口, 如 DCAM710, 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

value[**in**]: 曝光时间, 范围[1,30]

返回值:

PsRetOK: 调用成功

其他值：调用失败，可参考 [5.1.7](#) 节说明

5.3.81 Ps2_GetRGBExposure

函数原型：

```
PsReturnStatus Ps2_GetRGBExposure(PsDeviceHandle device, uint32_t  
sessionIndex, char* value)
```

函数功能：

获取 RGB 曝光时间，仅支持 560 系列设备

PS:不同型号产品可能不支持此接口，如 DCAM710，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

value[out]：曝光时间，范围[1,30]

返回值：

PsRetOK: 调用成功

其他值：调用失败，可参考 [5.1.7](#) 节说明

5.3.82 Ps2_RebootCamera

函数原型：

Vzense Technology, Inc

Copyright 2018

```
PsReturnStatus Ps2_RebootCamera(PsDeviceHandle device, uint32_t  
sessionIndex)
```

函数功能:

重启相机

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

返回值:

PsRetOK: 调用成功

其他值: 调用失败, 可参考 [5.1.7](#) 节说明

6 固件升级说明

固件升级需要用到升级工具 VzenseUpgradeTool, 详情请参考升级工具文档, 下载地址如下:

<https://gitee.com/Vzense/VzenseUpgradeTool>

<https://github.com/Vzense/VzenseUpgradeTool>

7 API 调用示例

7.1 初始化为通用模式

Vzense Technology, Inc

Copyright 2018

```
Ps2_Initialize  
Ps2_GetDeviceCount  
Ps2_GetDeviceListInfo  
Ps2_OpenDevice  
Ps2_StartStream  
While(1)  
{  
    Ps2_ReadNextFrame  
    Ps2_GetFrame  
}  
Ps2_StopStream  
Ps2_CloseDevice  
Ps2_Shutdown
```

7.2 初始化为 slave 触发模式

```
Ps2_Initialize  
Ps2_GetDeviceCount  
Ps2_GetDeviceListInfo  
Ps2_OpenDevice  
Ps2_StartStream  
Ps2_SetSlaveModeEnabled  
While(1)  
{  
    Ps2_ReadNextFrame  
    Ps2_GetFrame  
}  
Ps2_StopStream  
Ps2_CloseDevice
```

```
Ps2_Shutdown
```

7.3 初始化为 WDR 模式

```
Ps2_Initialize
```

```
Ps2_GetDeviceCount
```

```
Ps2_GetDeviceListInfo
```

```
Ps2_OpenDevice
```

```
Ps2_StartStream
```

```
Ps2_SetWDROutputMode
```

```
Ps2_SetDataMode
```

```
Ps2_SetWDRStyle
```

```
While(1)
```

```
{
```

```
    Ps2_ReadNextFrame
```

```
    Ps2_GetFrame
```

```
}
```

```
Ps2_StopStream
```

```
Ps2_CloseDevice
```

```
Ps2_Shutdown
```

8 FAQ

8.1 USB 方式

8.1.1 无法识别 USB 设备

- 1、首先确认 USB 线材是否支持 USB 通信。
- 2、确保主机端的 USB 口的供电能力足够，尽量使用 USB3.0 接口。

