Pico Zense DCAM305
Linux SDK 开发者指南

型号: DCAM305

版本: V3.0.0.4

编制:

审核:

批准:

日期:

关于本指南

本指南文档主要介绍如何使用 Pico Zense DCAM305 及 PicoZense Linux SDK 进行开发。

文档结构

章节	标题	内容
1	概述	介绍 PicoZense 产品及 Linux SDK 的概况
2	安装	介绍 PicoZense 产品及 Linux SDK 的安装
3	SDK 使用说明	介绍如何使用 PicoZense Linux SDK
4	SDK 接口介绍	介绍 PicoZense Linux SDK 的接口

版本发布记录

日期	版本	发布说明
2019/11/20	V3.0.0.4	Linux 305 SDK 开发指南

目录

1. 概述

PicoZense TOF RGBD Camera（型号：DCAM305）是 Pico 公司采用飞行时间（Time of Flight，TOF）技术研发的一款 3D 成像模组，具有精度高、环境适应性强、尺寸小等优点。其输出的深度信息可适用于下一代基于手势识别的人机交互、TV 游戏体感交互、人脸识别与活体检测、机器人避障、先进汽车视觉系统、工业控制等前沿创新技术领域。

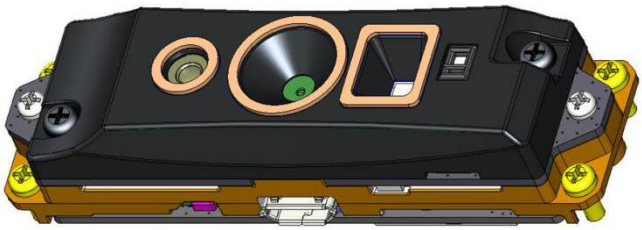

图 1 PicoZense TOF RGBD Camera: DCAM305

PicoZense Linux SDK 是基于 Pico Zense DCAM305 TOF RGBD Camera 开发的软件开发工具包，该开发包目前适用于 Linux 系统的 PC 或者 Arm 开发板，为应用开发者提供了一系列友好的 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，方便用户开发人脸识别与活体检测、手势识别、投影触控、疲劳检测、三维重建、导航避障等应用。

2. 安装

2.1. 推荐系统配置

配置项	推荐配置
开发环境	Ubuntu16.04 64 bit GCC 5.4.0
运行环境	Linux/Arm-Linux 系统 GCC 5.4.0 USB 2.0

2.2. 安装说明

USB 连接线一端连接模组，另一端连接 PC 或者开发板的 USB 接口，如图 2.1 所示。

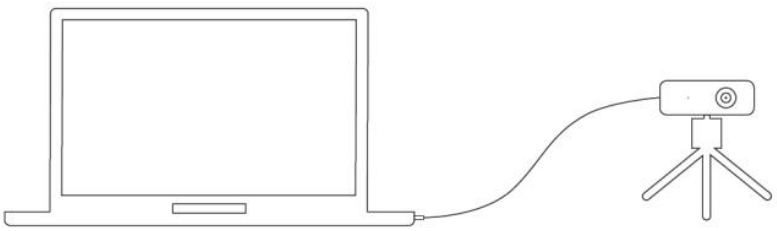

图 2.1 硬件模组安装示意图

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。

```
DCAM305: PicoZense RGBD Camera (usb-0000:00:14.0-2):
        /dev/video2
```

图 2.2 PicoZense RGBD Camera

1.1.1 软件环境配置

Arm-Linux 不需要额外的环境配置。

PC 下需要做的配置如下：

1. 对于 NVIDIA 显卡的电脑，请更新显卡驱动为 NVIDIA 驱动， 如下图。

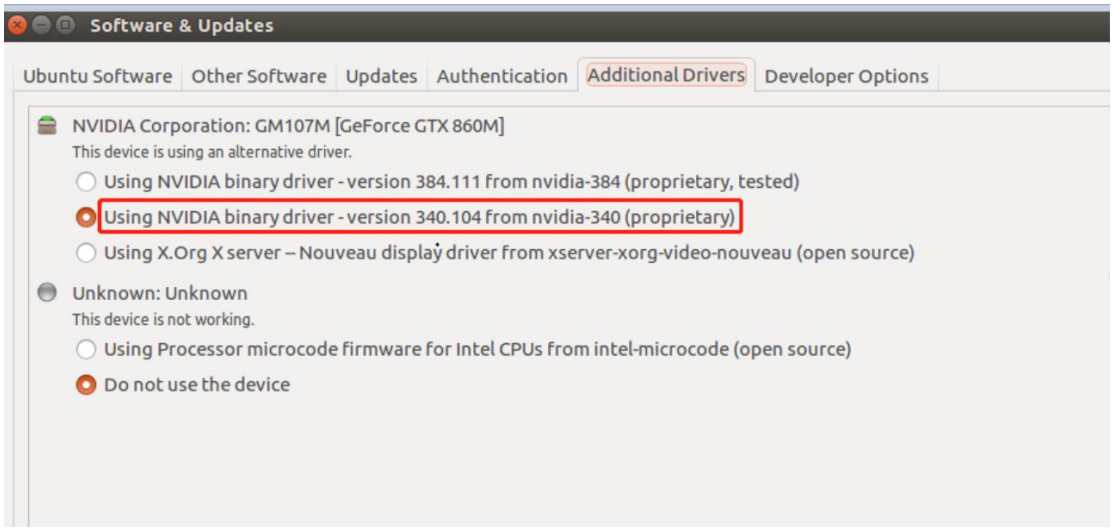

图 2.3 显卡驱动设置

2. 安装 VDPAU： 安装 libvdpau-dev 及 vdpauinfo。运行 vdpauinfo 查看，电脑是否支持 vdpau。支持时的输出参看下图：

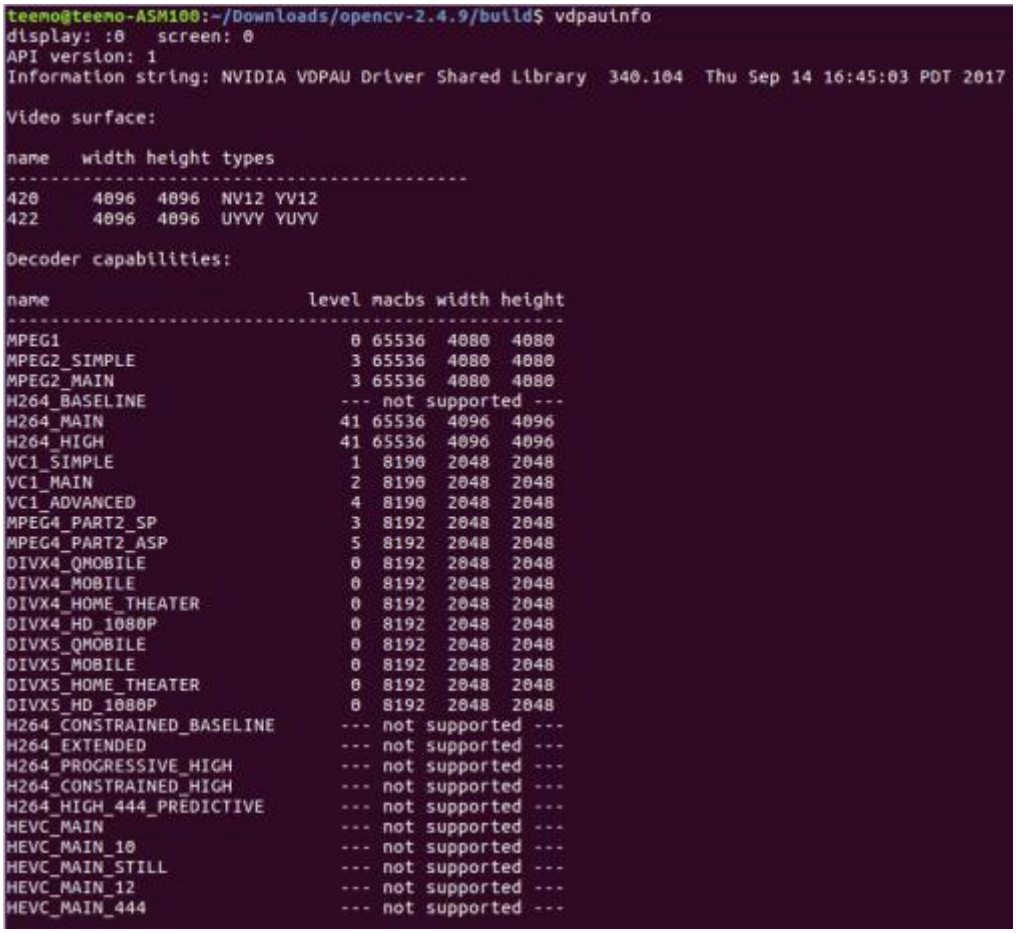

图 2.4 安装 VDPAU

3. 对于 Intel 显卡的电脑，需要进行以下操作：

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

运行：

vdpauinfo

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

3. SDK 使用说明

3.1. SDK 目录结构

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

Linux 目录结构如下图所示：

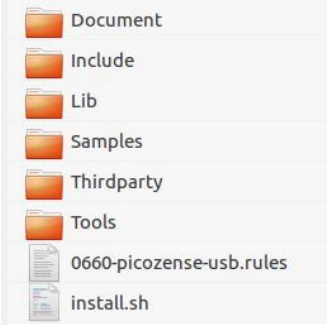

图 3.1 Linux SDK 目录结构

使用 SDK 前请先运行目录下的 install.sh 脚本。

Document 包含 SDK 的说明文档。

Include 主要包含 SDK 的头文件，如 PsCamera_api.h、PsCamera_define.h。

Lib 主要包含 SDK 的共享库 so 文件。

Samples 主要包含使用 PicoZense SDK 开发的例程。

Tools 包含可查看 PicoZense 深度摄像头的深度图像，IR 灰度图像和彩色图像的工具 FrameViewer，与使用 Samples 目录中的 FrameViewer 编译出来的一致。

3.2. 应用程序安装以及运行效果

将 PicoZense 深度摄像头连接到设备的 USB 接口，运行 Tools 目录里的 FrameViewer，会启动三个窗口分别显示深度图像，IR 灰度图像和彩色图像，如下图所示，彩色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。

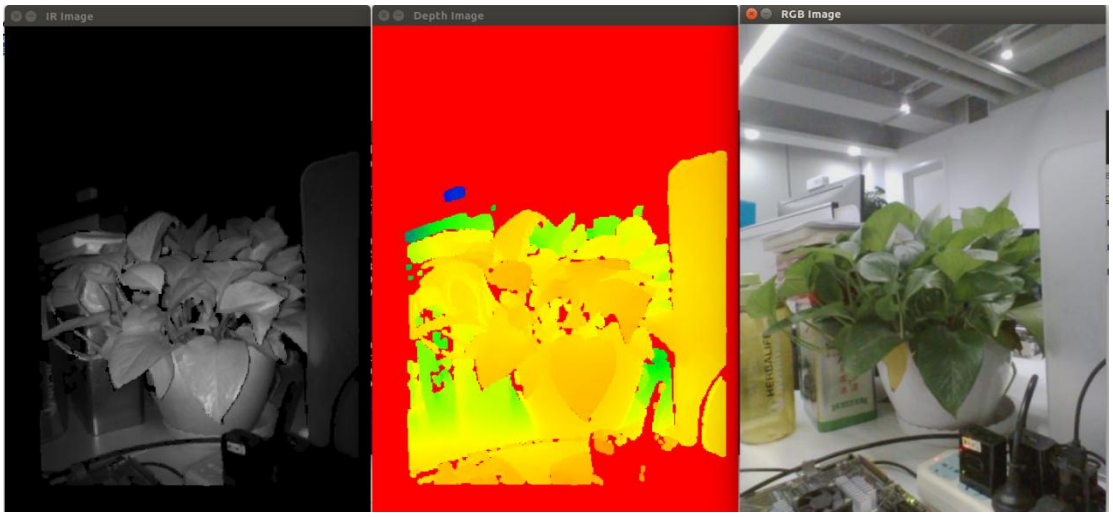

图 3.2 FrameViewer 运行效果

3.3. 开发流程

3.3.1. 项目配置

Linux 下使用 PicoZense 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 目录加入到链接搜索路径中，并链接 libpicozenseCamera_api.so，如-I../../Include，-L../../Lib/x64，-lpicozenseCamera_api。可参考 Samples 里 Makefile 的配置。

3.3.2. 接口调用流程

SDK 的 API 接口调用流程图如下：

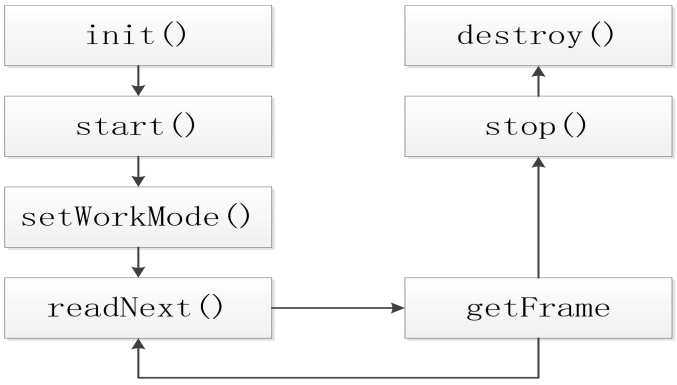

图 3.3 接口调用流程

1. init 和 destroy

调用 **init** 接口，初始化 SDK ；调用 **destroy** 接口，注销 SDK，释放 SDK 创建的所有资源。

2. start 和 stop

调用 **start** 接口，开启数据数据；调用 **stop** 接口，关闭数据获取。

3. setWorkMode

调用调用 **setWorkMode** 接口，设置相机模式。

4. readNext 和 getFrame

在图像处理的主循环里，每次先调用 **readNext** 采集一帧图像，然后再调用 **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。

3.3.3. 工作模式切换流程描述

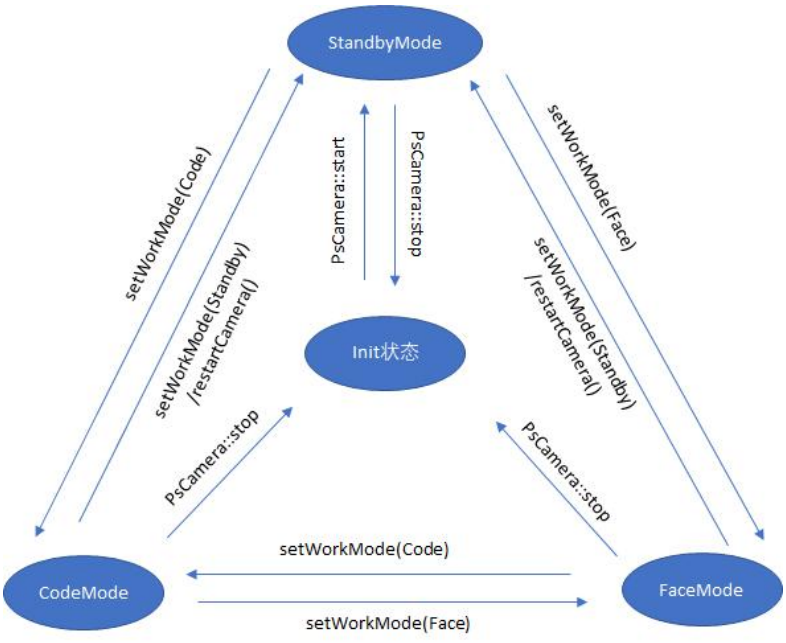

图 3.4 工作模式切换流程图

- 启动应用，通过 setWorkMode(Face)设置默认进入 FaceMode，此模式下默认以 Depth/IR/RGB 和单 RGB 交替输出图像，Depth，IR，RGB 图像的分辨率都为 480x640，Depth 和 IR 的帧率为 15Hz，RGB 图像为 30hz，应用从 SDK 获取的 Depth 和 IR 是与 RGB 对齐之后的图像。
- 通过 setWorkMode(Code)切换工作模式进入 CodeMode，此模式下 TOF 默认关闭，只有 RGB 图像，RGB 图像的分辨率为 480x640，可以通过 setRgbResolution 接口切换到其他分辨率，也可以通过 setTofFrameEnabled 开启或者关闭 TOF 图像数据。
- 调用 setWorkMode(Standby)切换进入到待机模式，此模式下 TOF 和 RGB 均默认关闭，只能通过 setWorkMode 切换回人脸或扫码模式来获取图像。

4. SDK 介绍

4.1. Enum 数据类型

4.1.1. FrameType

功能：

图像数据流类型

枚举值：

- **DepthFrame** 表示 16 位深度图像流
- **IRFrame** 表示 16 位 IR 灰度图像流
- **RGBFrame** 表示 24 位 3 通道 RGB 图像流

4.1.2. PixelFormat

功能：

图像的像素类型

枚举值：

- **PixelFormatDepthMM16** 表示每像素数据为 16 位的深度值，以毫米为单位
- **PixelFormatGray16** 表示每像素数据为 16 位的灰度值
- **PixelFormatGray8** 表示每像素数据为 8 位的灰度值
- **PixelFormatRGB888** 表示每像素数据为 24 位的 RGB 值
- **PixelFormatBGR888** 表示每像素数据为 24 位的 BGR 值
- **PixelFormatRGBA8888** 表示每像素数据为 32 位的 RGBA 值

4.2. Struct 数据类型

4.2.1. FrameReady

功能：

图像可获取的状态

枚举值：

参数	说明
depth	可获取 Depth 图像数据
ir	可获取 IR 图像数据
rgb	可获取 RGB 图像数据

4.2.2. Frame

功能：

图像的像素类型

成员：

参数	说明
frameIndex	帧号
frameType	图像数据类型
pixelFormat	像素类型
frameData	图像数据
dataLength	数据长度，以字节为单位
timeStamp	时间戳，单位为 ms
fps	帧率
width	图像的宽度
height	图像的长度
bytePerPixel	每个像素点的字节数

4.2.3. CameraParameter

功能：

相机内参和畸变系数，说明见下面表格

成员：

参数	说明
fx，fy，cx，cy	相机的内参
k1，k2，k3，p1，p2	相机的畸变参数

4.2.4. CameraExtrinsicParameter

功能：

相机的外参

成员：

参数	说明
rotation[1-9]	TOF 相机到 RGB 相机的旋转矩阵
translation[1-3]	TOF 相机到 RGB 相机的平移矩阵
e[1-9]	输出本征矩阵
f[1-9]	输出基础矩阵

4.3. API

SDK 接口说明如下：

API	int init()
说明	SDK 的初始化，在启动的时候必须最先调用此接口 **返回值:**0：成功；其他：失败

API	int destroy()
说明	SDK 的资源释放，在 stop 之后调用 **返回值:**0：成功；其他：失败

API	int start()
说明	开始图像采集，在 init 之后调用 **返回值:**0：成功；其他：失败

API	int stop()
说明	停止图像采集 **返回值:**0：成功；其他：失败

API	int readNext(FrameReady &ready)
说明	采集指定设备的下一帧图像，在 getFrame 之前调用 **返回值:**0：成功；其他：失败

API	int getFrame(FrameType frameType, Frame* pFrame)
说明	获取图像数据 **参数:** frameType：图像类型 pFrame：图像数据

	返回值:0: 成功；其他: 失败

API	int setGmmGain(int gmmGain)
说明	设置 IR 图像的 gmmgain 值，用于调整 IR 图像的亮度 **参数:** gmmGain: IR 图像的 gmmGain 值，取值范围为 0-4095 **返回值:**0: 成功；其他: 失败

API	int getGmmGain()
说明	获取当前 IR 图像的 gmmgain 值 **返回值：**当前 IR 图像的 Gmmgain 值

API	int setRgbResolution(int resolutionIndex)
说明	设置 RGB 图像分辨率，参数取值范围为 0-3 **参数:** resolutionIndex: 0 : 1080x1920 1 : 720x1280 2 : 480x640 3 : 360x640 **返返回值:**0: 成功；其他: 失败

API	int getDepthCameraParameter(CameraParameter* pDepthParameter)
说明	获取 TOF 相机内参，详细说明参见 4.2.3 **返回值:**0：成功；其他：失败

API	int getRgbCameraParameter(CameraParameter* pRgbParameter)
说明	获取 RGB 相机内参，详细说明参见 4.2.3 **返回值:**0：成功；其他：失败

API	int getCameraExtrinsicParameter(CameraExtrinsicParameter* pExtrinsicParameter)
说明	获取相机外参，详细说明参见 4.2.4 **返回值:**0：成功；其他：失败

API	int getSn(char* sn, int length)
说明	获取设备的序列号，例如 PD3051AGD5130013M **返回值:**0：成功；其他：失败

API	int getFWVersion(char* fw, int length)
说明	获取设备的固件版本号，例如 DCAM305_c086_pc_sv0.01_R2_20190518_b02 **返回值:**0：成功；其他：失败

API	int getHWVersion(char* hw, int length)
说明	获取设备的硬件版本号，例如 R2 **返回值:**0：成功；其他：失败

API	int getDeviceName(char* name, int length)
说明	获取设备的名称，例如 PicoZense RGBD DCAM **返回值:**0：成功；其他：失败

API	int setWorkMode(int workMode)
说明	设置工作模式，可以设置的工作模式有三种,取值分别为 1,2,3 **参数:** workMode: 1：人脸模式，此模式下默认为 TOF 数据 15hz，RGB 数据 30hz，TOF 和 RGB 分辨率都为 640x480，可以通过 setTofFrameEnabled 来打开或者关闭 TOF 数据，TOF 图像与 RGB 图像是同步以及对齐的 2：扫码模式，此模式下默认只有 RGB 数据，RGB 图像的分辨率为 640x480,用户可以通过 setTofFrameEnabled 来打开或者关闭 TOF 数据,如果打开了 TOF,TOF 图像与 RGB 图像是同步以及对齐的 3：待机模式，此模式下 TOF 和 RGB 均默认关闭，只能通过 setWorkMode 切换回人脸或扫码模式来获取图像 **返回值:**0：成功；其他：失败

API	int setTofFrameEnabled(bool bEnabled)
说明	设置是否获取 TOF 图像数据，扫码模式下默认关闭 TOF，如果想获取 TOF 数据，需要调用此接口设置 TOF 状态为 true **参数:** bEnabled: true: 打开 TOF 数据 false: 关闭 TOF 数据 **返回值:**0: 成功；其他: 失败

API	int setImageMirror(int mirrorValue)
说明	设置图像的镜像处理 **参数:** mirrorValue: 0：不做镜像处理 1：左右镜像 2：上下镜像 3：上下及左右镜像(旋转 180 度) **返回值:**0: 成功；其他: 失败

API	int setRgbFrameEnabled(bool bEnabled)
说明	设置是否获取 RGB 图像数据。Standby 模式下设置无效。 **参数:**

	bEnabled: true: 打开 RGB 数据 false: 关闭 RGB 数据 **返回值:**0: 成功；其他: 失败