PicoZense TOF RGBD Camera

Android SDK 开发者指南

Android

2019.3

Pico Technology Co., Ltd.

关于本指南

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

文档结构

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

版本发布记录

日期	版本	发布说明
2019.3.15	V1.3.6	初始版本

目 录

1 概述

PicoZense TOF RGBD Camera（型号：DCAM710）是一款 Pico 公司采用飞行时间测距技术（TOF: Time of Flight）研发的 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、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。

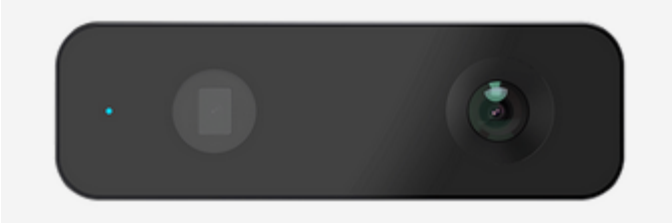

图 1.1 PicoZense TOF RGBD Camera：DCAM710

PicoZense Android SDK 是基于 PicoZense TOF RGBD Camera 开发的软件开发工具包，该开发包目前适用于 Android 系统的平板或者智能手机，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 安装

2.1 推荐系统配置

配置项	推荐配置
开发环境	Android API Android-21 以上 JDK: jdk1.7.0_01 及以上
运行环境	Android OS 5.0 及以上 ARMv7a/ARMv8a @ 1.8ghz+ 512M RAM USB 2.0(支持 Host 功能)

2.2 安装说明

2.2.1 硬件安装

USB 连接线一端连接模组，另一端连接单板或者智能手机的 USB 接口，如图 2.1 所示。

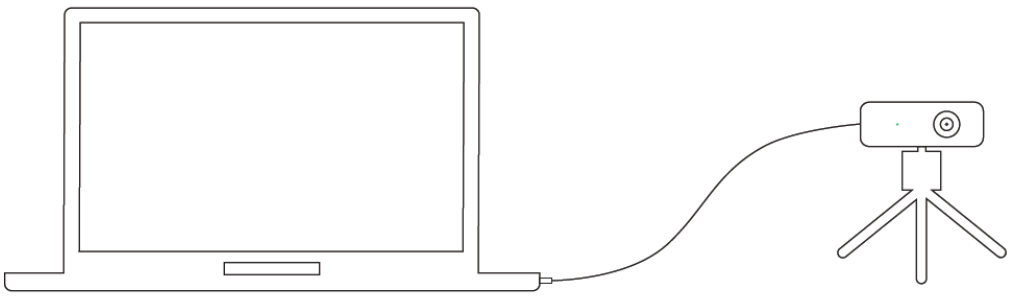

图 2.1 硬件模组安装示意图

3 SDK 使用说明

3.1 SDK 目录结构

PicoZense Android SDK 包含 SDK，Sample，安装包，说明文档等。目录结构如下图所示：

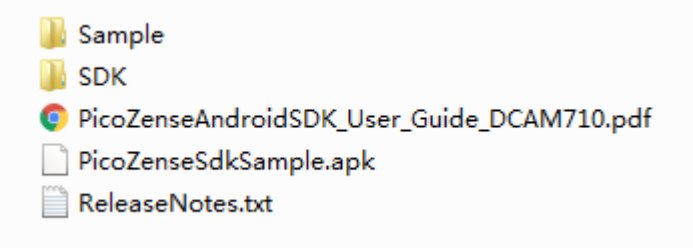

图 3.1 Android SDK 目录结构

SDK 目录主要包含 PicoZense Android SDK 的 jar 文件以及 so 库

Samples 主要包含使用 PicoZense Android SDK 开发的例程。

APK 安装包是 release 版本的，可直接安装到 Android 设备上运行。

ReleaseNotes.txt 主要是介绍本次更新的主要内容。

PDF 文档详细描述了 Android 系统上的开发说明。

3.2 应用程序安装以及运行效果

将 PicoZense 深度摄像头连接到 Android 设备的 USB 接口，把 apk 文件拷贝到设备上，双击安装，安装完成打开程序，会启动一个包括图像预览以及菜单按钮的界面，如下图所示，默认显示 Depth 图像以及中心点的深度值。可以点击界面上的 Spinner 来切换显示数据以及设置各种模式。

图 3.2 PicoZenseSdkSample.apk 运行效果

3.3 开发流程

3.3.1 导入 jar 文件

新建一个 AndroidStudio 工程，拷贝 PicoCamera.jar 文件到 app/libs 目录，点击 File，在下拉里选择 Project Structure，弹出工程组件界面，如下图所示，选择 Modules 下面的工程，点击右边的 Dependencies 菜单，然后点击右上角的”+”，在弹出的菜单中选择 jar dependency，选择上面导入的 PicoCamera.jar，点击确定。

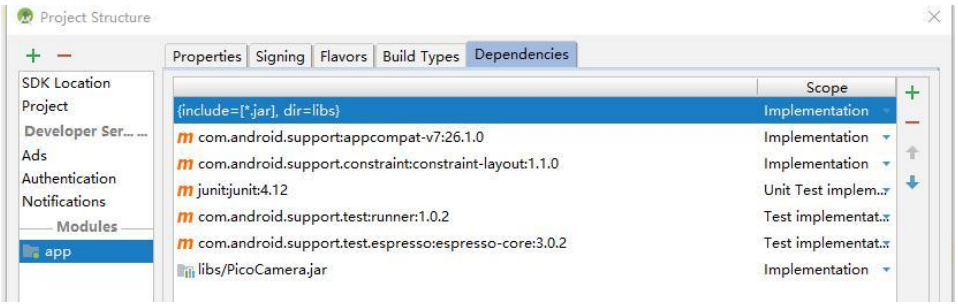

图 3.3 导入 jar 文件

3.3.2 导入 so 库文件

把 so 库拷贝到 app/libs 目录下，打开工程的 build.gradle 文件，添加以下代码，导入 so。导入库完成之后就可以在工程中调用接口进行开发，如下图 demo 示例：

```
android {
    sourceSets {
        main {
            jniLibs.srcDirs = ['libs']
        }
    }
}
```

图 3.4 导入 so 文件

3.3.3 接口调用流程

1． 导入接口类

```
import com.picozense.sdk.PsCamera;
```

2． 创建 PsCamera 类，进行初始化

```
mPicoCamera = new PsCamera();
if (mPicoCamera != null) {
    mPicoCamera.init(this);
}
```

3． 创建回调

```
mFrameCallback = new FrameCallback();
```

4． 打开 camera，设置参数，回调

```
mPicoCamera.setFrameCallback(mFrameCallback);
mPicoCamera.setBGThresdhold(20);
mPicoCamera.setDepthRange(0);
dataType = DataType.DATA_TYPE_DEPTH_RGB_30;
mPicoCamera.setDataType(dataType.ordinal());
mPicoCamera.setRgbResolution(2);
mPicoCamera.start(this);
```

5． 在重载的回调函数里取出数据

```
public class FrameCallback implements IFrameCallback {
    @Override
    public void onFrame(PsFrame DepthFrame,PsFrame IrFrame,PsFrame
RgbFrame) {
            //DataProcess
        }
}
```

4 SDK 接口介绍

4.1 Enum 数据类型

4.1.1 DepthRange

功能：

Depth Range 模式

枚举值：

- **NearRange** 表示设置为 Near Range 模式，Range0
- **MidRange** 表示设置为 Middle Range 模式，Range1
- **FarRange** 表示设置为 Far Range 模式，Range2
- **XNearRange** 表示设置为 XNear Range 模式，Range3
- **XMidRange** 表示设置为 XMiddle Range 模式，Range4
- **XFarRange** 表示设置为 XFar Range 模式，Range5
- **XXNearRange** 表示设置为 XXNear Range 模式，Range6
- **XXMidRange** 表示设置为 XXMiddle Range 模式，Range7
- **XXFarRange** 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。

4.1.2 DateType

功能：

数据类型设置，设置输出图像帧的类型和帧数

枚举值：

- **DATA_TYPE_DEPTH_30**： 表示以 30fps 输出 Depth 单路图像

支持分辨率：

Depth ：640*480

- **DATA_TYPE_IR_30**：表示以 30fps 输出 IR 单路图像

支持分辨率：

IR : 640*480

- **DATA_TYPE_DEPTH_RGB_30**： 表示以 30fps 同时输出 Depth 和 RGB 两路图像

支持分辨率：

Depth ：640*480

RGB ：1920*1080/1280*720/640*480/640*360

RGB 图像分辨率可通过 setRgbResolution 接口设置

- **DATA_TYPE_IR_RGB_30**：表示以 30fps 同时输出 IR 和 RGB 两路图像

支持分辨率：

IR ：640*480

RGB ：1920*1080/1280*720/640*480/640*360

RGB 图像分辨率可通过 setRgbResolution 接口设置

- **DATA_TYPE_DEPTH_IR_30**：表示以 30fps 同时输出 Depth 和 IR 两路图像

支持分辨率：

Depth ：640*480

IR ：640*480

- **DATA_TYPE_DEPTH_60**： 表示以 60fps 输出 Depth 单路图像

支持分辨率：

Depth ：640*360

- **DATA_TYPE_IR_60**：表示以 60fps 输出 Ir 单路图像

支持分辨率：

IR : 640*360

- **DATA_TYPE_DEPTH_IR_RGB_30**：表示以 30fps 输出 Depth/IR/RGB 三路图像

支持分辨率：

Depth ：640*360

IR ：640*360

RGB ：1920*1080/1280*720/640*480/640*360

RGB 图像分辨率可通过 setRgbResolution 接口设置

- **DATA_TYPE_DEPTH_IR_15_RGB_30** 表示以15fps输出Depth/IR图像 ,以30fps 输出 RGB 图像

支持分辨率：

Depth ：640*480

IR ：640*480

RGB ：1920*1080/1280*720/640*480/640*360

RGB 图像分辨率可通过 setRgbResolution 接口设置

4.1.3 FrameType

功能：

图像数据流类型

枚举值：

- **DepthFrame** 表示深度图像流
- **IRFrame** 表示 IR 灰度图像流
- **RGBFrame** 表示彩色图像流

4.1.4 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 主要 Class 介绍

4.2.1 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

4.2.2 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

4.2.3 PsFrame

功能：

图像信息，说明见下面表格

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

4.2.4 IFrameCallback

功能：

图像数据的回调接口

说明：

上层应用需要创建一个此接口对象，通过 setFrameCallback 接口设置到 native，native 会通过接口类的 OnFrame 方法把数据 callback 到应用层。

4.2.5 PsCamera

功能：

接口类，用户可以通过此类来进行 camera 打开，关闭，设置参数，获取数据等操作，详细接口信息见下表

说明：

API	void init(Context context)
说明	SDK 的初始化

API	void destroy()
说明	SDK 的资源释放

API	void start(Context context)
说明	开始图像采集

API	void stop()
说明	停止图像采集

API	void setFrameCallback(final IFrameCallback callback)
说明	设置图像数据的回调函数

API	void setDepthRange(int depthRange)
说明	设置 depth range 的值

API	int getDepthRange()
说明	获取当前 depth range 的值

API	void setGmmGain(int gmmGain)
说明	设置 IR 图像的 gmmgain 值

API	int getGmmGain()
说明	获取当前 IR 图像的 gmmgain 值

API	void setBGThresdhold(int threshold)
说明	设置 Depth 图像的背景滤波阈值

API	int getBGThresdhold()
说明	获取当前 Depth 图像的背景滤波阈值

API	void setRgbResolution(int resolutionIndex)
说明	设置 RGB 图像分辨率

API	void setDataType(int dataType)
说明	设置 datetype 的值

API	int getPulseCount()
说明	获取 TOF 相机当前的 pulse count 值

API	void getDepthCameraParameter(CameraParameter mDepthParameter)
说明	获取 TOF 相机内参

API	void getRgbCameraParameter(CameraParameter mRgbParameter)
说明	获取 RGB 相机内参

API	void getCameraExtrinsicParameter(CameraExtrinsicParameter mExtrinsicParameter)
说明	获取相机外参

API	void setDepthUndistortionEnabled(boolean bEnabled)
说明	设置 Depth 图像是否做反畸变处理

API	void setRgbUndistortionEnabled(boolean bEnabled)
说明	设置 RGB 图像是否做反畸变处理

API	void setIrUndistortionEnabled(boolean bEnabled)
说明	设置 IR 图像是否做反畸变处理

API	void setFilterEnabled(boolean bEnabled)
说明	设置 Depth 图像是否做滤波处理

API	void setTimeFilterEnabled(boolean bEnabled)
说明	设置 Depth 图像是否做时域滤波处理

API	void setRemoveEdgeEnabled(boolean bEnabled)
说明	设置 Depth 图像是否做去边缘处理

API	void setMapperEnabledDepthToRGB(boolean bEnabled)
说明	设置是否做 Depth 到 RGB 的图像对齐处理

API	void setMapperEnabledRGBToDepth(boolean bEnabled)
说明	设置是否做 RGB 到 Depth 的图像对齐处理

API	void setMapperEnabledRGBToIR(boolean bEnabled)
说明	设置是否做 RGB 到 IR 的图像对齐处理