Metoak Stereo Camera Driver Documentation

发布 1.1.0

Metoak

2021 年 12 月 8 日

元橡科技（北京）有限公司
地址：北京海淀区学院路51号首享科技大厦12层 / 电话：010-86468956
www.metoak.com

目录

公司简介

元橡科技是全球领先的双目立体觉解决方案提供商，公司成立于 2017 年，专注双目立体视觉领域，是软硬件一体的解决方案提供商，国家高新技术企业。公司立足于核心技术研发，掌握自主知识产权，实现双目立体视觉领域 0-1 多项突破。

双目立体视觉基于视差原理，依据成像设备从不同位置获取的被测物体的图像，匹配对应点的位置偏移，得到视差数据，进而计算物体的空间三维信息。不同的空间测量技术，有其各自的特点和适用范围。双目立体视觉相对于其他立体深度感知技术，具有高分辨率、低功耗、远距离等优点。

元橡科技（北京）有限公司
地址：北京海淀区学院路51号首享科技大厦12层 / 电话：010-86468956 www.metoak.com

一、 SDK 安装

1.1相机安装

不同相机的安装方式参考具体相机介绍，下面以 U60 相机为例进行说明。U60 是一款通过 USB3.0 接口进行供电和数据通信的相机，外观见下图：

图 1.1 U60 相机正面

图 1.2 U60 相机反面

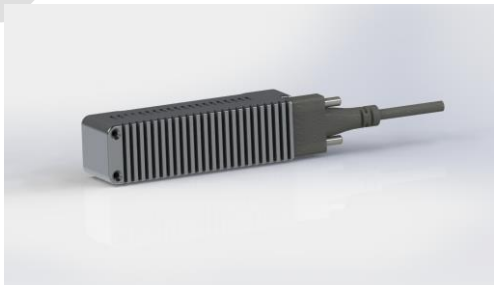

图 1.3 U60 相机通过自带 USB3.0 数据线进行连接

请使用相机自带 USB3.0 数据线将相机与系统连接。

相机支持不同的操作系统，具体参见相机说明。本手册后面示例都以 Ubuntu16.04 64 作为基础。相机支持标准的 V4L2 协议，所以不需要安装额外的驱动就可以在系统中看到。在 Ubuntu16.04 系统中打开终端程序并执行如下查询命令：

```
$ ls /dev/v4l/by-id/

usb-Metoak._METOAK_Depth_Camera_5504k-video-index0 -> ../../video0
```

查询结果中看见 METOAK_Depth_Camera_5504k 文本，说明相机已被 Ubuntu16.04 64 位系统识别并已准备就绪可以使用了。

提示：
查询结果中的 ../../video0 指的是 /dev/video0 这是 UVC 相机的真实路径地址。
/dev/v4l/by-id/usb-Metoak._METOAK_Depth_Camera_5504k-video-index0 只是符号链接。

1.2 SDK 包解压

将已经获得的相机 SDK 解压，使用命令：

```
$ tar -zxf mo_stereo_camera_driver_vx.x.x.tar.gz
```

解压缩该文件，可以得到一个 mo_stereo_camera_driver_vx.x.x 目录。X.x.x 代指具体的 SDK 版本。

1.3 SDK 目录介绍

解压后的 mo_stereo_camera_driver_vx.x.x 目录结构如下：

```
| -- doc                                          # 本文档所在目录
      | -- metoak_stereo_camera_driver_doc_zh_cn.pdf # 本文档
| -- include                                      # SDK 接口头文件
      | -- mo_stereo_camera_driver_c.h             # 基础功能接口头文件
      | -- mo_stereo_camera_driver_c_utilities.h   # 实用功能接口头文件
      | -- mo_stereo_camera_driver_macro_define.h  # 基础宏定义头文件
      | -- mo_stereo_camera_driver_type_define.h   # 基础类型定义头文件
| -- lib                                          # SDK 库文件所在目录
      | -- libmoStereoCameraDriver.so              # 无版本号符号连接
      | -- libmoStereoCameraDriver.so.1.0          # 两位版本号符号连接
      | -- libmoStereoCameraDriver.so.1.0.0        # 三位版本号库文件
| -- sample                                       # 示例代码所在目录
```

```
    | -- include                                  # 示例代码头文件目录
        | -- common_function.h                   # 示例代码共用功能头文件
    | -- source                                  # 示例代码实现文件目录
        | -- common_function.cpp                 # 示例代码共用功能实现文件
        | -- get_point_cloud_sample.cpp          # 示例代码点云数据实现文件
        | -- get_raw_sample.cpp                  # 示例代码原始图像实现文件
        | -- get_rectify_sample.cpp              # 示例代码校正图像实现文件
        | -- get_rgbd_sample.cpp                 # 示例代码 RGBD 图像实现文件
        | -- set_camera_sample.cpp               # 示例代码相机参数设置实现文件
| -- Makefile                                   # 示例代码编译 Makefile 文件
```

1.4 编译示例代码

打开 mo_stereo_camera_driver_vx.x.x 目录下的 Makefile 文件：

$ cd mo_stereo_camera_driver_vx.x.x

$ gedit Makefile

（如下为 Makefile 内容片断）
开启示例代码使用 OpenCV 来展示图像，就需要
修改

```
WANNA_USE_OPENCV := NO
#WANNA_USE_OPENCV := YES
```

为

```
#WANNA_USE_OPENCV := NO
WANNA_USE_OPENCV := YES

ifeq ($(WANNA_USE_OPENCV),YES)
    EXTRA_CFLAGS := -DWANNA_USE_OPENCV
    LIBS += -lopencv_highgui -lopencv_core -lopencv_imgproc -lopencv_imgcodecs
    # 设置 OpenCV 头文件(如：/usr/local/include)与库文件(如：/usr/local/lib)的实际路径
    INCPATH      += -I/usr/local/include
    LIBPATH      += -L/usr/local/lib
endif
```

保存后，执行：

$ make

示例程序将创建在 bin 目录下：

```
mo_stereo_camera_driver_vx.x.x
 | -- bin
       | -- get_point_cloud_sample
       | -- get_raw_sample
       | -- get_rectify_sample
       | -- get_rgbd_sample
       | -- set_camera_sample
```

运行示例程序测试相机功能：

```
mo_stereo_camera_driver_vx.x.x/bin$ ./get_point_cloud_sample  # 获取点云数据
mo_stereo_camera_driver_vx.x.x/bin$ ./get_raw_sample          # 获取原始图像
mo_stereo_camera_driver_vx.x.x/bin$ ./get_rectify_sample      # 获取校正图像
mo_stereo_camera_driver_vx.x.x/bin$ ./get_rgbd_sample         # 获取 RGBD 图像
mo_stereo_camera_driver_vx.x.x/bin$ ./set_camera_sample       # 设置相机参数
```

二、 SDK 示例代码

2.1获取 RGBD 图像

参考代码片段：

```
/**< 1. Open specific camera */
s32Result = moOpenUVCCameraByPath(caCameraPath, &hCameraHandle);

/**< 2. Get current video mode */
s32Result = moGetVideoMode(hCameraHandle, &eVideoMode);

/**< 3. Set RGBD mode : frame = Disparity + YUV_I420 */
if (MVM_RGBD != eVideoMode) {
    s32Result = moSetVideoMode(hCameraHandle, MVM_RGBD);
}

/**< 4. Get current video frame */
s32Result = moGetCurrentFrame(hCameraHandle, &u64ImageFrameNum, &pu8FrameBuffer);

/**< 5. Get left and right image */
s32Result = moGetRGBDImage(hCameraHandle, pu8FrameBuffer,
                           &pu16RGBDDisparityData, &pu8RGBDYUVI420Img);

/**< 6. Close specific camera */
moCloseCamera(&hCameraHandle);
```

```
#ifdef WANNA_USE_OPENCV
    waitKey(FETCH_AND_DISPLAY_TIME_LENGTH);
    Mat matYUV2BGR; // 12bits / 8bits
    Mat matYUVI420(video_frame_height * 1.5, video_frame_width, CV_8UC1, pu8RGBDYUVI420Img);
    cvtColor(matYUVI420, matYUV2BGR, COLOR_YUV2BGR_I420);
    imshow("RGB", matYUV2BGR);

    Mat matDisparity(video_frame_height, video_frame_width, CV_8UC3);
    GetDisparityImage(matDisparity.rows, matDisparity.cols, matDisparity.step,
                      pu16RGBDDisparityData, matDisparity.data);
    imshow("Disparity", matDisparity);
```

```
#endif // WANNA_USE_OPENCV
```

完整代码详见：sample/source/get_rgbd_sample.cpp

moGetRGBDImage　　API 同时获得视差数据和 RGB 图像。

以下两接口详见：include/mo_stereo_camera/mo_stereo_camera_driver_c.h

moGetRGBDDisparityData　API 只获得视差数据。

moGetRGBDYUVI420Image　API 只获得 RGB 图像。

2.2 获取点云数据

参考代码片段：

```
/**< 1. Open specific camera */
s32Result = moOpenUVCCameraByPath(caCameraPath, &hCameraHandle);

/**< 2. Get current video mode */
s32Result = moGetVideoMode(hCameraHandle, &eVideoMode);

/**< 3. Set RGBD mode : frame = Disparity + YUV_I420 */
if (MVM_RGBD != eVideoMode) {
    s32Result = moSetVideoMode(hCameraHandle, MVM_RGBD);
}

/**< 4. Get current video frame */
s32Result = moGetCurrentFrame(hCameraHandle, &u64ImageFrameNum, &pu8FrameBuffer);

/**< 5. Get left and right image */
s32Result = moGetRGBDImage(hCameraHandle, pu8FrameBuffer,
                           &pu16RGBDDisparityData, &pu8RGBDYUVI420Img);

/**< 6. Get point cloud data */
float*   pfXAxisArray    = NULL;
float*   pfYAxisArray    = NULL;
float*   pfZAxisArray    = NULL;
uint32_t u32AxisArraySize = 0;
s32Result = moConvertDisparity2PointCloud(hCameraHandle, pu16RGBDDisparityData,
                                          &u32AxisArraySize,
                                          &pfXAxisArray, &pfYAxisArray, &pfZAxisArray);
/**< 7. Close specific camera */
moCloseCamera(&hCameraHandle);
```

完整代码详见：sample/source/get_point_cloud_sample.cpp

2.3设置相机参数

参考代码片段：

```
/**< 1. Open specific camera */
s32Result = moOpenUVCCameraByPath(caCameraPath, &hCameraHandle);

/**< 2. Set fill light type */
s32Result = moSetFilllightType(hCameraHandle, MFT_ON);

/**< 3. Get fill light type */
mo_filllight_type eFilllightType = MFT_OFF;
s32Result = moGetFilllightType(hCameraHandle, &eFilllightType);

/**< 4. Close specific camera */
moCloseCamera(&hCameraHandle);
```

完整代码详见：sample/source/set_camera_sample.cpp
更多设置相机参数接口详见：include/mo_stereo_camera/mo_stereo_camera_driver_c.h

三、 SDK 接口说明

3.1 接口简介

SDK 里包含有获取数据和相机设置的各种接口。这里介绍的 SDK 覆盖多种接口，各相机能够使用的接口和相机规格有关。

3.2 接口说明

3.2.1 moGetSdkVersion

得到当前 SDK 版本

参数

无

返回值

char* 版本号字符串

3.2.2 moGetUVCCameraInfoList

得到识别到的元橡 UVC 相机信息列表

参数

*mo_camera_info_node*** *ppstCameraInfoList* 元橡 UVC 相机信息列表

注意：

ppstCameraInfoList 指向的内存需要调用 moReleaseUVCCameraInfoList API 释放

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 没找任何元橡 UVC 相机

3.2.3 moReleaseUVCCameraInfoList

释放元橡 UVC 相机信息列表内存

参数

*mo_camera_info_node*** *ppstCameraInfoList* 元橡 UVC 相机信息列表

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

3.2.4 moOpenUVCCameraByPath

通过设备路径打开 UVC 相机

参数

char*	*pcPath*	UVC 相机的设备路径，例如：/dev/video0
*MO_CAMERA_HANDLE**	*phCameraHandle*	得到 UVC 相机的句柄

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 打开相机失败

3.2.5 moOpenUVCCameraByNumber

通过设备索引号打开 UVC 相机

参数

uint8_t	*u8Number*	UVC 相机的设备索引号，取值范围：0 ～ 126
MO_CAMERA_HANDLE	*hCameraHandle*	得到 UVC 相机的句柄

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 打开相机失败

3.2.6 moCloseCamera

关闭指定相机

参数

MO_CAMERA_HANDLE hCameraHandle 指定相机的句柄

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

3.2.7 moSuspendCamera

暂停指定相机的视频流

注意： 会影响图像获取，调用 moGetCurrentFrame/moGetSafeCurrentFrame API 前，先调用 moResumeCamera API 。

参数

MO_CAMERA_HANDLE hCameraHandle 指定相机的句柄

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.8 moResumeCamera

恢复指定相机的视频流

参数

MO_CAMERA_HANDLE hCameraHandle 指定相机的句柄

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.9 moGetBxfAndBase

得到指定相机的 BxF(基线乘以焦距的积)和基线值

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*float**	*pfBxf*	*基线乘以焦距的积*
*float**	*pfBase*	*基线值*

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.10 moGetCurrentFrame

得到指定相机的指定视频模式的视频帧原始数据

注意：

数据内存由 SDK 管理，一定时间后数据将被刷新！

调用 moGetSafeCurrentFrame API 得到的视频帧原始数据则无此顾虑！

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint64_t**	*pu64ImageFrameNum*	视频帧序号
*uint8_t***	*ppu8FrameBuffer*	视频帧原始数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 得到当前视频帧失败

3.2.11 moGetSafeCurrentFrame

得到指定相机的指定视频模式的视频帧原始数据

注意：

数据内存由 SDK 管理，一定时间后数据不会被刷新，直到此 API 再次被调用！

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint64_t**	*pu64ImageFrameNum*	视频帧序号
*uint8_t***	*ppu8FrameBuffer*	视频帧原始数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 得到当前视频帧失败

3.2.12 moGetRGBDImage

得到指定相机的 RGBD 视频模式的视差数据和YUVI420图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint16_t***	*ppu16DisparityData*	视差数据
*uint8_t***	*ppu8YUVI420Img*	YUVI420 图像数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.13 moGetRGBDDisparityData

只得到指定相机的 RGBD 视频模式的视差数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint16_t***	*ppu16DisparityData*	视差数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.14 moGetRGBDYUVI420Image

只得到指定相机的 RGBD 视频模式的YUVI420图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8YUVI420Img*	YUVI420 图像数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.15 moGetRawImage

得到指定相机的原始视频模式的左Bayer图像数据和右Bayer图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8LeftBayerImg*	左 Bayer 图像数据
*uint8_t***	*ppu8RightBayerImg*	右 Bayer 图像数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.16 moGetRawLeftBayerImage

只得到指定相机的原始视频模式的左Bayer图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8LeftBayerImg*	左 Bayer 图像数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.17 moGetRawRightBayerImage

只得到指定相机的原始视频模式的右Bayer图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8RightBayerImg*	右 Bayer 图像数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.18 moGetRectifiedImage

得到指定相机的校正视频模式的左灰度图像数据和右YUVI420图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8LeftGrayImg*	左灰度图像数据
*uint8_t***	*ppu8RightYUVI420Img*	右 YUVI420 图像数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.19 moGetRectifiedLeftGrayImage

只得到指定相机的校正视频模式的左灰度图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8LeftGrayImg*	左灰度图像数据

返回值 int32_t

0 - 成功，负数 - 失败
-1 - 无效参数
-2 - 命令执行失败

3.2.20 moGetRectifiedRightYUVI420Image

只得到指定相机的校正视频模式的右YUVI420图像数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8FrameBuffer*	视频帧原始数据
*uint8_t***	*ppu8RightYUVI420Img*	右 YUVI420 图像数据

返回值 int32_t

0 - 成功，负数 - 失败
-1 - 无效参数
-2 - 命令执行失败

3.2.21 moSetVideoMode

设置指定相机当前的视频模式

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
mo_video_mode	*eVideoMode*	要设置的视频模式

返回值 int32_t

0 - 成功，负数 - 失败
-1 - 无效参数
-2 - 命令执行失败

3.2.22 moGetVideoMode

得到指定相机当前的视频模式

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
mo_video_mode	*peVideoMode*	已设置的视频模式

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

3.2.23 moQuerySupportedVideoParam

查询指定相机支持的视频参数

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*mo_video_param***	*ppstVideoFrameParamArray*	视频帧参数数组
*uint8_t**	*pArraySize*	数组元素个数

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.24 moSetVideoParam

设置指定相机当前的视频参数

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
uint8_t	*u8VideoParamIndex*	支持的视频参数索引 （由 moQuerySupportedVideoParam 返回）

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.25 moGetVideoParam

获得指定相机当前的视频参数

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint8_t**	*pu8VideoParamIndex*	当前视频参数的索引

返回值 int32_t

0 - 成功，负数 - 失败
-1 - 无效参数
-2 - 命令执行失败

3.2.26 moGetVideoResolution

获得指定相机当前的视频参数

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint16_t**	*pu16ResolutionWidth*	*当前分辨率宽度*
uint16_t*	pu16ResolutionHeight	*当前分辨率高度*

返回值 int32_t

0 - 成功，负数 - 失败
-1 - 无效参数
-2 - 命令执行失败

3.2.27 moGetRealTimeFPS

得到统计得出的每秒帧率

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*double**	*pd8FPS*	当前的视频帧参数

返回值 int32_t

0 - 成功，负数 - 失败
-1 - 无效参数
-2 - 失败通常是由于开始时没有足够的时间来统计

3.2.28 moSetFilllightType

设置指定相机的补光灯类型

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
mo_filllight_type	*eFilllightType*	*想要设置的*补光灯类型

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.29 moGetFilllightType

获得指定相机的当前补光灯类型

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*mo_filllight_type**	*peFilllightType*	当前*设置的*补光灯类型

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.2.30 moConvertDisparity2PointCloud

获得视差数据对应的点云数据

参数

MO_CAMERA_HANDLE	*hCameraHandle*	指定相机的句柄
*uint16_t**	*pu16RGBDDisparityData*	视差数据
uint32_t*	pu32AxisArraySize	X/Y/Z 轴数据个数
float**	ppu32XAxisArray	点云 X 轴数据
float**	ppu32YAxisArray	点云 Y 轴数据
float**	ppu32ZAxisArray	点云 Z 轴数据

返回值 int32_t

0 - 成功，负数 - 失败

-1 - 无效参数

-2 - 命令执行失败

3.3 结构体说明

3.3.1 mo_camera_info_node

```
typedef struct _mo_camera_info_node_s
{
    mo_camera_type    eCameraType;      // 相机类型
    char              caSN[24];         // e.g. "2021072318520001"
    char              caFriendName[32]; // e.g. Depth Camera 5504k
    char              caAddress[16];    // e.g. /dev/video0
    struct _mo_camera_info_node_s* pstPrevNode;
    struct _mo_camera_info_node_s* pstNextNode;
}mo_camera_info_node;
```

3.3.2 mo_video_param

```
typedef struct _mo_video_param_s
{
    uint8_t  u8VideoParamIndex;   // 视频参数的索引 - 不能修改
    uint16_t u16ResolutionWidth;  // 视频图像宽度
    uint16_t u16ResolutionHeight; // 视频图像高度
    uint8_t  u8DefaultFPS;        // 默认的视频帧率/秒
    uint8_t  u8MaxFPS;            // 最大的视频帧率/秒(零为固定帧率)
}mo_video_param;
```

3.4 枚举说明

3.4.1 mo_camera_type

```
typedef enum _mo_camera_type_e
{
    MCT_UVC      = 0, // UVC  camera
    MCT_MIPI     = 1, // MIPI camera
    MCT_ETHERNET = 2  // Ethernet camera
}mo_camera_type;
```

3.4.2 mo_video_mode

```
typedef enum _mo_video_mode_e
{
    MVM_RAW          = 0, // Bayer + Bayer
    MVM_RECTIFIED    = 1, // Gray(Only Y) + YUV_I420
    MVM_RGBD         = 2, // Disparity + YUV_I420
    MVM_RGBD_DENSE   = 3  // Dense disparity + YUV_I420
}mo_video_mode;
```

3.4.3 mo_filllight_type

```
typedef enum _mo_filllight_type_e
{
    MFT_OFF                 = 0, // Fill light off
    MFT_ON                  = 1, // Fill light on
    MFT_EXPOSURE_SYNC       = 2, // Fill light synchronizes with exposure
    MFT_ON_OFF_ALTERNATION = 3  // Fill light alternates between on and off
}mo_filllight_type;
```

四、 常见问题答疑

4.1 程序是否可以在虚拟机运行

确保 PC 机为真实物理机器（不支持虚拟机，如：Virtualbox、VMWare 等），且支持 USB3.0(Super Speed USB)标准接口,安装有 Ubuntu16.04 64 位操作系统。

4.2 无法获得图像数据

确保 PC 机支持 USB3.0(Super Speed USB)标准接口,且 USB3.0 数据线为相机自带或兼容 USB3.0 标准。

图三 蓝色 USB 接口为 USB3.0 接口

图四 附有 SS 标识的非蓝色 USB3.0 接口