Linux 开发指南

1、简介

MoveSense DVPU 双目深度相机支持 Linux 环境开发， 在 Linux 下 MoveSense DVPU 作为 UVC 标准设备免去驱动安装的步骤，自动识别为视频设备。人加目前为 MoveSense DVPU 提供了 C++版本的 SDK，使得用户能在 Linux 平台下开发基于双目深度相机的应用。

2、快速使用说明

在 Linux 平台下提供了快速测试的软件程序，将设备插入 Linux 系统的主机，打开用于测试的 Linux 可执行程序即可对设备进行快速测试，具体步骤如下：

（1）首先安装以下依赖库：

sudo apt-get install libusb-1.0-0-dev libusb-dev libudev-dev libv4l-dev

（2）然后终端进入 lib 文件夹下，键入以下命令：

g++ -I/usr/local/include -I/usr/local/include/opencv -I/usr/local/include/opencv2 -L/usr/local/lib

-o"MoveSense_Dvpu" ./src/Camera.cpp ./src/MoveSenseCamera.cpp ./src/VL4IF.cpp ./src/Sample_Cap_OpenCV.cpp -lusb-1.0 -lopencv_highgui -lopencv_core -lopencv_imgproc -lopencv_calib3d -lopencv_contrib -lopencv_features2d -lopencv_flann -lopencv_imgproc -lopencv_legacy -lopencv_ml -lopencv_objdetect -lopencv_video -ludev -lusb

（3）执行完后会在 lib 文件夹下，生成一个名为"MoveSense_Dvpu"的可执行程序，运行即可。

3、工作模式说明

MoveSense DVPU 支持三种工作模式，三种工作模式参数设置如下，可根据需要自行选择：

D_VPU_320X240_LR_30FPS 显示左右图

D_VPU_320X240_LD_30FPS 显示左图和视差图

D_VPU_320X240_LRD_30FPS 显示左右图和视差图

```
int sel = D_VPU_320X240_LD_30FPS;
movesense::MoveSenseCamera c(sel);
```

4、可设置接口说明

（1）void OpenCamera():打开相机。

（2）void CloseCamera():关闭相机。

（3）void EnableCam(bool val):可在相机打开的过程中使能和停止相机流，一般使用不到（true-使能，false-禁止）。

（4）void SetUndistort(bool val)：打开/关闭矫正开关（true-打开；false--关闭）。

（5）int GetCameraID():获取相机的 ID 号，默认会打印 ID 号（4 个字节），并有一个整形的返回值。

（6）void GetCamParas(MsgPkg& p):获取相机参数 t_P2(在示例程序中有示例代码),关于相机参数 t_P2 的使用请看“相机参数/相机参数说明.pdf”

（7）int GetImageData(unsigned char * &data,int &len):获取相机的图像数据。

(注：函数的声明在 MoveSenseCamera.h 中)

**************如需自己开发 SDK 要看以下部分*************

5、Set/Get 接口的具体实现

具体命令的实现过程是：采用 SATURATION 通道去实现 I2C 寄存器地址的传输；

```
void VL4_IF::SendI2CAddr(int addr){
    v4l2_control c;
    c.id    = V4L2_CID_SATURATION;
    c.value = addr;
    //printf("SendAddr--is--%d-\n",addr);
    xioctl(m_dev_fd, VIDIOC_S_CTRL, &c);
}
```

此外用 WHITE_BALANCE_TEMPERATURE 通道去更新该寄存器地址对应的数值（其中 VIDIOC_S_CTRL 用来实现寄存器的写，VIDIOC_G_CTRL 用来实现寄存器的读）。

```
void VL4_IF::SetI2CValue(int val){
    v4l2_control c;
    c.id    = V4L2_CID_WHITE_BALANCE_TEMPERATURE;
    c.value = val;
    //printf("SetVal--is--%d-\n",val);
    xioctl(m_dev_fd, VIDIOC_S_CTRL, &c);
}
```

```
unsigned char VL4_IF::GetI2CValue(){
    v4l2_control c;
    c.id     = V4L2_CID_WHITE_BALANCE_TEMPERATURE;
    ioctl(m_dev_fd, VIDIOC_G_CTRL, &c);
    //printf("GetVal--is--%d--\n",c.value);
    return (unsigned char)(c.value);
}
```

Example:

以 void SetUndistort(bool val)为例，这个命令是校正开关的设置，其对应的 I2C 寄存器是 0xB4。

①假如我们想关闭矫正开关，即：SetUndistort(false)，该函数的具体实现过程是：

```
int addr = 0xB4;
Bool val = 0;
v4l2_control c;

c.id      = V4L2_CID_SATURATION;
c.value = addr;
xioctl(m_dev_fd, VIDIOC_S_CTRL, &c);
Sleep(1);//等待 1ms
c.id      = V4L2_CID_WHITE_BALANCE_TEMPERATURE;
c.value = val;
xioctl(m_dev_fd, VIDIOC_S_CTRL, &c);
```

②假如我们想获取矫正开关对应寄存器的数值，具体做法是

```
int addr = 0xB4;
v4l2_control c;

c.id      = V4L2_CID_SATURATION;
c.value = addr;
xioctl(m_dev_fd, VIDIOC_S_CTRL, &c);
Sleep(1);//等待 1ms
c.id      = V4L2_CID_WHITE_BALANCE_TEMPERATURE;
xioctl(m_dev_fd, VIDIOC_G_CTRL, &c);//返回值在 c.value 中
printf("the register value is %d \n",c.value);
```