OpenNI2 环境配置与使用

一．环境设置

1. 去Github中搜索RealSense,下载ZIP包，找到“[librealsense](https://github.com/IntelRealSense/librealsense)/[wrappers](https://github.com/IntelRealSense/librealsense/tree/master/wrappers)/**openni2**/”,在使用说明处下载OpenNI2 SDK 和RealSense2 SDK，或者到官网下载OpenNI2 SDK (<https://structure.io/openni>),此网址提供各个版本的SDK，
2. 不要改动目录直接安装两个SDK文件，然后根据使用说明创建SDK 系统环境目录
3. 下载CMake程序（桌面版），将下载的ZIP包解压，找到openni2下的CMkeLists.txt文件，使用CMake程序编译出vs工程
4. 使用编译出的vs工程(Rs2driver.sln)，编译所需驱动的动态库（Rs2driver.DLL）,
5. 安装好的OpenNI2 会有一个工程文件夹“OpenNI2\Samples\”，将动态库放置在所需位置(按照1中的使用说明来)

二．OpenNI2 API 调用过程

1.使用OpenNI2提供的initialize()函数加载API，如果要结束OpenNI2的调用需要使用shutdown()函数来关闭

Status rc = openni::OpenNI:: initialize()；

Status rc = openni::OpenNI::shutdown()

2.使用传感器与物理设备通信，通过Device类连接设备并获取设备的信息，当使用完设备时可以通过close()函数来断开设备

Device device

device.open(ANY\_DEVICE);

device.close();

当只有一个设备处于连接状态时，直接使用上方函数可以很好获取数据，当有多个设备处于连接时，可以使用OpenNI2提供的数组Array和enumerateDevices()获取所有设备的信息，根据设备信息（如URL）选取使用设备。

Device device

const char\* deviceURI = openni::ANY\_DEVICE;

openni::Array<openni::DeviceInfo> deviceInfoList;

openni::OpenNI::enumerateDevices(&deviceInfoList);

Device.open(deviceURI)或者Device.open(deviceInfoList[i].geturl());

device.close();

3.使用Open提供的VideoStream类来获取相机的数据流，不同的数据流需要创建不同的流对象，然后使用流对象启动相应流，摧毁流对象和停止流分别使用destroy()和stop()函数，本文提供深度流和彩色流示例

Depth:

VideoStream depth

rc = depth.create(device, openni::SENSOR\_DEPTH)

rc = depth.start();

depth.stop();

depth.destory();

Color：

VideoStream color

rc = color.create(device, openni::SENSOR\_COLOR)

rc = color.start();

color.stop();

color.destory();

4.检测最新的数据帧使用VideoStream,为了去掉延迟需要调用waitForAnyStream()函数，然后用VideoFrameRef创建一个新的帧对象，用readFrame()来获取新帧并对VideoFrameRef对象初始化，用getData()函数获取帧中深度数据。

VideoFrameRef frame;

int changedStreamDummy;

VideoStream\* pStream = &depth;

rc= OpenNI::waitForAnyStream(&pStream,1,&changedStreamDummy,SAMPLE\_READ\_WAIT\_TIMEOUT);

rc = depth.readFrame(&frame);

DepthPixel\* pDepth = (DepthPixel\*)frame.getData();

以上使用过程代码在OpenNI2工程文件夹“OpenNI2\Samples\“ 下。

三．其他说明

# 1. 在“OpenNI2\Documentation\”下有一个API说明文档

# 2. 《OpenNI2 开发者指南》 转载 <https://blog.csdn.net/LinkageWorld/article/details/52621156?spm=1001.2101.3001.6650.3&utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-3-52621156-blog-26505393.pc_relevant_multi_platform_whitelistv1&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-3-52621156-blog-26505393.pc_relevant_multi_platform_whitelistv1&utm_relevant_index=5>

# 3. 《浅析OpenNI2---Driver开发（1）》转载<https://blog.csdn.net/magicfrog2009/article/details/74062894>

# 4.《浅析OpenNI2---Driver开发（2）》转载<https://blog.csdn.net/magicfrog2009/article/details/75153937>

# 四．示例Demo

#include <stdio.h>

#include <OpenNI.h>

#include "OniSampleUtilities.h"

#define SAMPLE\_READ\_WAIT\_TIMEOUT 2000 //2000ms

using namespace openni;

int main()

{

Status rc = OpenNI::initialize();

if (rc != STATUS\_OK)

{

printf("Initialize failed\n%s\n", OpenNI::getExtendedError());

return 1;

}

printf("Initialize success");

Device device;

rc = device.open(ANY\_DEVICE);

if (rc != STATUS\_OK)

{

printf("Couldn't open device\n%s\n", OpenNI::getExtendedError());

return 2;

}

printf(" open device success\*-\*-\*-\*-\*-\*-\*-\*-\*\n");

VideoStream depth;

//传感器类型

if (device.getSensorInfo(SENSOR\_DEPTH) != NULL)

{

rc = depth.create(device, SENSOR\_DEPTH);

if (rc != STATUS\_OK)

{

printf("Couldn't create depth stream\n%s\n", OpenNI::getExtendedError());

return 3;

}

}

printf("depth.create success\*-\*-\*-\*-\*-\*-\*\n");

rc = depth.start();

if (rc != STATUS\_OK)

{

printf("Couldn't start the depth stream\n%s\n", OpenNI::getExtendedError());

return 4;

}

printf("depth.start success\*-\*-\*-\*-\*-\*-\*\n");

VideoFrameRef frame;

while (!wasKeyboardHit())

{

int changedStreamDummy;

VideoStream\* pStream = &depth;

rc = OpenNI::waitForAnyStream(&pStream, 1, &changedStreamDummy, SAMPLE\_READ\_WAIT\_TIMEOUT);

if (rc != STATUS\_OK)

{

printf("Wait failed! (timeout is %d ms)\n%s\n", SAMPLE\_READ\_WAIT\_TIMEOUT, OpenNI::getExtendedError());

continue;

}

//最大最小深度值

int maxDepth = pStream->getMaxPixelValue();

int minDepth = pStream->getMinPixelValue();

rc = depth.readFrame(&frame);

if (rc != STATUS\_OK)

{

printf("Read failed!\n%s\n", OpenNI::getExtendedError());

continue;

}

if (frame.getVideoMode().getPixelFormat() != PIXEL\_FORMAT\_DEPTH\_1\_MM && frame.getVideoMode().getPixelFormat() != PIXEL\_FORMAT\_DEPTH\_100\_UM)

{

printf("Unexpected frame format\n");

continue;

}

//获得深度数据数组

DepthPixel\* pDepth = (DepthPixel\*)frame.getData();

printf("frame.getHeight() %d frame.getWidth() %d \t", frame.getHeight(), frame.getWidth());

//二位图片一维化后找到中心点

int middleIndex = (frame.getHeight()+1)\*frame.getWidth()/2;

int point = 0;

//打印一维化后图片的各点深度值

for (int i = 0; i < frame.getWidth(); i++)

{

for (int j = 0; j < frame.getHeight(); j++)

{

printf("point%d %8d \t", point, pDepth[point++]);

}

printf("\n");

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\*\* [%08llu] %8d %d %d %d \*\*\n", (long long)frame.getTimestamp(), pDepth[middleIndex],maxDepth,minDepth, middleIndex);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

}

depth.stop();

depth.destroy();

device.close();

OpenNI::shutdown();

system("pause");

return 0;

}

# 基于Mv3d SDK的悉灵相机OpenNI2框架开发

# 五，OpenNI2底层调用原理

# OpenNI2通过上文提到过的rs2Driver的接口来进行过程调用，从而让开发者只是用接口调用进行开发工作。

# rs2Driver底层实现了四个类，相关的类都放在命名空间oni::driver中，主要的类有DriverBase、DeviceBase、StreamBase；各个设备Driver都需要实现这些类，同时将DriverClass通过宏ONI\_EXPORT\_DRIVER导出，而这三个类都继承自父类Base，具体介绍请看《三，其他说明》中的3，4。

# Intel在实现rs2Driver的三个类时，利用IntelRealSense SDK进行了再封装。从而实现了简单易懂的OpenNI框架，所以我们的开发目标就是在原有OpenNI的基础上，利用Mv3d SDK替换掉IntelRealSense SDK，按功能需求重新实现相关函数即可。

# 六，OpenNI2 示例开发

链接：https://github.com/Huankongya/OpenNI.git