Orbbec-OpenNI2 Extended API Instruction

Table of Contents

目录

1. Overview

Orbbec SDK extends OpenNI2 to implement some functions, which including reading serial number, obtaining device type, saving by Flash, obtaining and setting camera parameter, IR setting, LDP control, LDM modular control, etc. This document describes those functions and demonstrate basic procedures.

2. Scope

Device: Astra Series Camera which including Astra, Astra Pro, Astra Mini, Astra Embedded, Astra Stereo etc.
Platform: Windows 7 and above; Ubuntu 14.04 and above.

3. Introduction

Orbbec OpenNI2 Extended API supports Windows x86/x64, Linux x86/x64, ARM32/64 and Android. OpenNI2 should be initialized and device should be opened successfully before utilizing the extended API.
Here are the instructions for all extended APIs.

4. Obtain Serial Number

```
char serNumber[12];
int dataSize = sizeof(serNumber);
memset(serNumber, 0, dataSize);
g_Device.getProperty(openni::OBEXTENSION_ID_SERIALNUMBER, (uint8_t *)&serNumber,
&dataSize);
```

5. Obtain Device Type

```
char devType[32];
int dataSize = sizeof(devType);
memset(devType, 0, dataSize);
g_Device.getProperty(openni::OBEXTENSION_ID_DEVICETYPE, (uint8_t *)&devType, &dataSize);
```

6. Obtain Camera Parameters

```
typedef struct OBCameraParams
{
    float l_intr_p[4];        //[fx,fy,cx,cy]
    float r_intr_p[4];        //[fx,fy,cx,cy]
    float r2l_r[9];           //[r00,r01,r02;r10,r11,r12;r20,r21,r22]
    float r2l_t[3];           //[t1,t2,t3]
    float k[5];               //[k1,k2,k3,p1,p2]
    int is_mirror;
}OBCameraParams;

OBCameraParams    m_CamParams;
int dataSize = sizeof(OBCameraParams);
g_Device.getProperty(openni::OBEXTENSION_ID_CAM_PARAMS,    (uint8_t    *)&m_CamParams,
&dataSize);
```

7. Set Camera Parameters to Device

```
typedef struct OBCameraParams
{
   float l_intr_p[4];              //[fx,fy,cx,cy]
   float r_intr_p[4];              //[fx,fy,cx,cy]
   float r2l_r[9];                 //[r00,r01,r02;r10,r11,r12;r20,r21,r22]
   float r2l_t[3];                 //[t1,t2,t3]
   float k[5];                     //[k1,k2,k3,p1,p2]
   int is_mirror;
} OBCameraParams;

OBCameraParams m_CamParams = { 0 };
int dataSize = sizeof(OBCameraParams);

m_CamParams.l_intr_p[0] = 577.318970;
m_CamParams.l_intr_p[1] = 577.318970;
m_CamParams.l_intr_p[2] = 308.729004;
m_CamParams.l_intr_p[3] = 269.143005;

m_CamParams.r_intr_p[0] = 517.447998;
m_CamParams.r_intr_p[1] = 517.447998;
m_CamParams.r_intr_p[2] = 305.432007;
m_CamParams.r_intr_p[3] = 250.410995;

m_CamParams.r2l_r[0] = 0.999972;
m_CamParams.r2l_r[1] = -0.005735;
m_CamParams.r2l_r[2] = 0.004735;
m_CamParams.r2l_r[3] = 0.005736;
m_CamParams.r2l_r[4] = 0.999983;
m_CamParams.r2l_r[5] = -0.000298;
m_CamParams.r2l_r[6] = -0.004733;
m_CamParams.r2l_r[7] = 0.000325;
m_CamParams.r2l_r[8] = 0.999989;

m_CamParams.r2l_t[0] = -25.147900;
m_CamParams.r2l_t[1] = 0.015202;
m_CamParams.r2l_t[2] = -0.648167;

m_CamParams.k[0] = -0.077348;
m_CamParams.k[1] = 0.208761;
m_CamParams.k[2] = -0.196780;
m_CamParams.k[3] = 0.000617;
m_CamParams.k[4] = 0.001059;

m_CamParams.is_mirror = 0;
g_Device.setProperty(openni::OBEXTENSION_ID_CAM_PARAMS, (uint8_t *)&m_CamParams,
dataSize);
```

8. Obtain IR Gain Value

```
int gain = 0;
int dataSize = 4;
g_Device.getProperty(openni::OBEXTENSION_ID_IR_GAIN, (uint8_t*)&gain, &dataSize);
printf("ir gain value : 0x%x\n", gain);
```

9. Set IR Gain Value

```
int gain = 0;
int dataSize = 4;
g_device.getProperty(openni::OBEXTENSION_ID_IR_GAIN, (uint8_t *)&gain, &dataSize);
printf("ir gain value : 0x%x\n", gain);
```

```
gain++;
g_device.setProperty(openni::OBEXTENSION_ID_IR_GAIN, (uint8_t *)&gain, dataSize);
```

10. Obtain IR Exposure Value

```
int exposure = 0;
int dataSize = 4;
g_device.getProperty(openni::OBEXTENSION_ID_IR_EXP, (uint8_t*)&exposure, &dataSize);
printf("ir exposure value : 0x%x\n", exposure);
```

11. Set IR Exposure Value

```
int exposure = 0;
int dataSize = 4;
g_device.getProperty(openni::OBEXTENSION_ID_IR_EXP, (uint8_t *)&exposure, &dataSize);
printf("ir exposure value : 0x%x\n", exposure);
exposure += 256;
g_device.setProperty(openni::OBEXTENSION_ID_IR_EXP, (uint8_t *)&exposure, dataSize);
```

12. LDP Switch, Camera Re-plug is Required

```
int dataSize = 4;
int ldp_en = enable;
g_Device.setProperty(openni::OBEXTENSION_ID_LDP_EN, (uint8_t *)&ldp_en, dataSize);
```

13. Laser Switch

```
int dataSize = 4;
int laser_en = enable;
g_Device.setProperty(openni::OBEXTENSION_ID_LASER_EN, (uint8_t *)&laser_en, dataSize);
```

14. Fan Swicth(immediately,temporary)

This type interface work immediately, but it become useless after restart.

14.1 Obtain fan status

```
//value = 1: open
//value = 0: close
//value = 2: default
unsigned int nValue = 0;
openni::Status rc = device.getProperty(XN_MODULE_PROPERTY_FAN_STATUS, &nValue);
```

14.2 Set fan status

```
openni::Device device
openni::Status rc = device.setProperty(XN_MODULE_PROPERTY_FAN_ENABLE,1);//Open
openni::Status rc = device.setProperty(XN_MODULE_PROPERTY_FAN_ENABLE,0);//Close
openni::Status rc = device.setProperty(XN_MODULE_PROPERTY_FAN_ENABLE,2);//Default value
```

15. Fan Swicth(flash,perpetual)

To take effect, you need to turn off the power and restart device.

15.1 Obtain fan flash status

```
//value = 1: open
//value = 0: close
//value = 2: default
unsigned int nValue = 0;
openni::Status rc = device.getProperty(XN_MODULE_PROPERTY_FAN_F_STATUS, &nValue)
```

15.2 Set fan flash status

```
openni::Device device
openni::Status rc = device.setProperty(XN_MODULE_PROPERTY_FAN_F_ENABLE,1);//Open
openni::Status rc = device.setProperty(XN_MODULE_PROPERTY_FAN_F_ENABLE,0);//Close
openni::Status rc = device.setProperty(XN_MODULE_PROPERTY_FAN_F_ENABLE,2);//Default value
```