

女娲星-XB40/XB40L/X100 (Nuwa-XB40/XB40L/X100) Windows SDK 开发规范

修订历史

版本号	时间	修定记录	修订部门
V1.0.0	2022/04/22		软件部

The copyright of this manual belongs to Shenzhen Angstrong Technology Co., Ltd. No part of this manual may be translated into other languages or reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording. Information in this document is subject to change without notice and does not represent a commitment on the part of Shenzhen Angstrong Technology Co., Ltd. Please contact us for the latest product information. The manual is only for customers who have purchased the product.

本手册版权归深圳市安思疆科技有限公司所有，未经许可，任何单位和个人都不得以电子的、机械的、磁性的、光学的、手工的等形式复制、传播、转录和保存该出版物，或翻译成其他语言版本。一经发现，将追究法律责任。深圳市安思疆科技有限公司保留更改本说明书的权利，届时恕不另行通知，请谅解。请在订购时联系我们以获得产品的最新信息。本使用说明仅面向已购买产品的顾客。在编写说明内容时，仅针对产品使用者。本产品说明书可能无法满足非产品购买者的疑问，请予以谅解。

目录

1. 简介.....	4
1.1 产品概述.....	5
1.2 环境要求.....	6
1.2.1 运行环境.....	6
1.2.2 系统要求.....	6
1.3 SDK 功能介绍.....	6
1.3.1 获取深度.....	6
1.3.2 获取点云.....	6
2.SDK 集成指南.....	7
2.1 包结构说明.....	7
2.2 示例程序运行说明.....	7
2.3 调用流程图.....	8
3 ANG SDK 核心 API 说明.....	8
3.1 获取深度图像接口.....	8
3.1.1 获取摄像头列表.....	8
3.1.2 释放资源.....	9
3.1.3 打开摄像头.....	9
3.1.4 关闭摄像头.....	9
3.1.5 打开深度数据流.....	9
3.1.6 关闭深度数据流.....	10
3.1.7 读取深度数据.....	10
3.1.8 注册深度数据回调函数.....	10
3.1.9 注册点云数据回调函数.....	10
3.1.10 设置分辨率.....	11
3.1.11 深度转点云.....	11
3.1.12 旋转数据流.....	11
3.1.13 配置相机参数.....	12
3.1.14 设置帧率.....	12
3.1.15 设置投射器.....	12
3.2 扩展 API 接口.....	13
3.2.1 获取 SDK 版本号.....	13
3.2.4 获取摄像头 SN 号.....	13
3.2.4 获取内参.....	13
4. 获取深度视频流和点云代码实现.....	13

图表

图 1 女娲星产品实物图.....6

图 2 调用流程图.....8

表 1 深度图格式.....6

1.简介

1.1 产品概述

安思疆 Linux-SDK API 为第三方应用集成摄像头模组功能提供了一站式服务。基于 SDK 可快速为客户提供二次开发服务,防止在使用 Angstrong-Linux SDK API 的过程中由于不规范的调用而引起其他问题。

SDK 用于驱动安思疆科技女娲星系列模组,提供实时获取深度图像、点云图等功能





图 1 女娲星产品实物图

1.2 环境要求

1.2.1 运行环境

Windows 平台

1.2.2 系统要求

Windows 平台

1.3 SDK 功能介绍

1.3.1 获取深度

支持驱动安思疆女娲星系列模组，实时获取深度图；
深度数据的空间格式如下

表 1 深度图格式

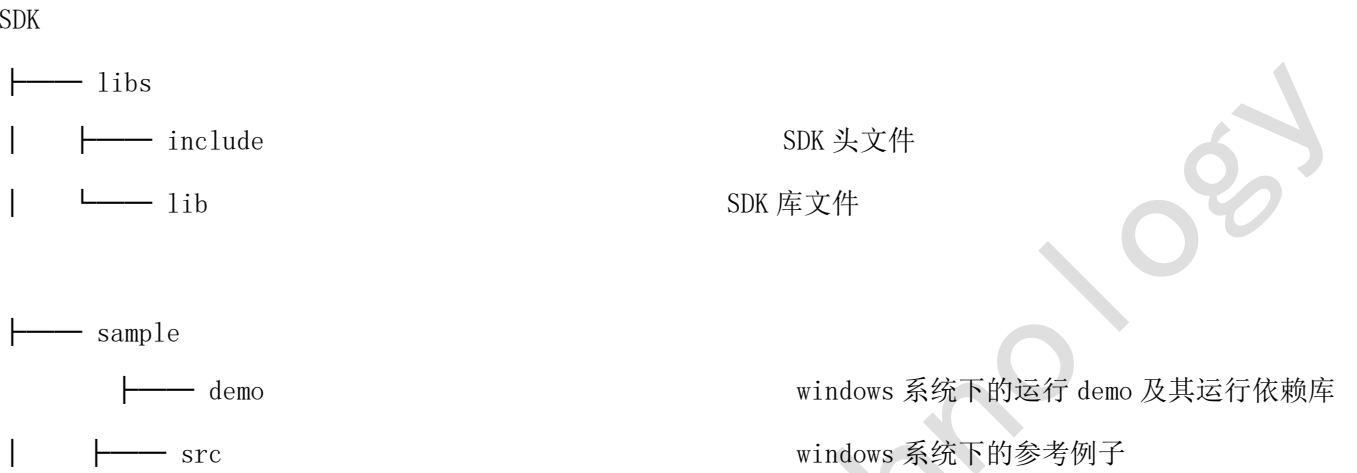
位宽度	16
单位	mm
类型	Short
深度范围	27cm~4m

1.3.2 获取点云

提供接口对深度图进行转化点云图；也可注册回调，直接得到点云图数据

2.SDK 集成指南

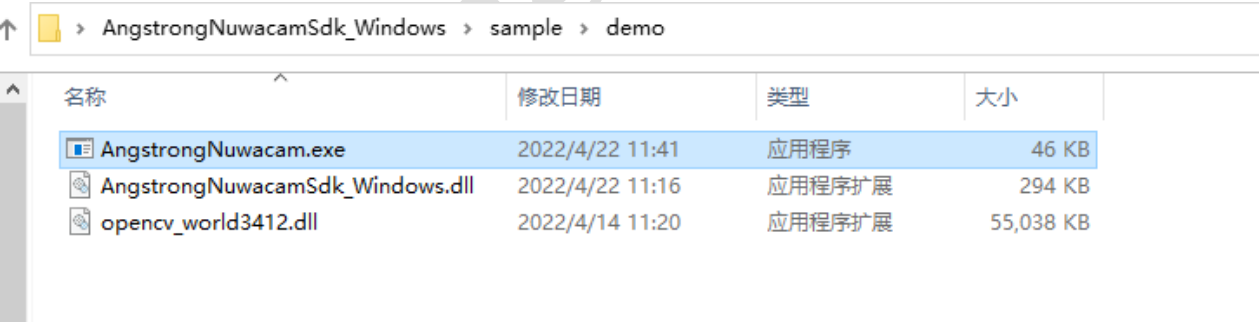
2.1 包结构说明



2.2 示例程序运行说明

2.2.1 运行 window 系统示例程序

进入 SDK 包的 sample 文件夹，然后再进入 demo 文件夹，可以看到示例的 exe，和其运行需要的的 dll 动态库。如下图所示



2.3 调用流程图

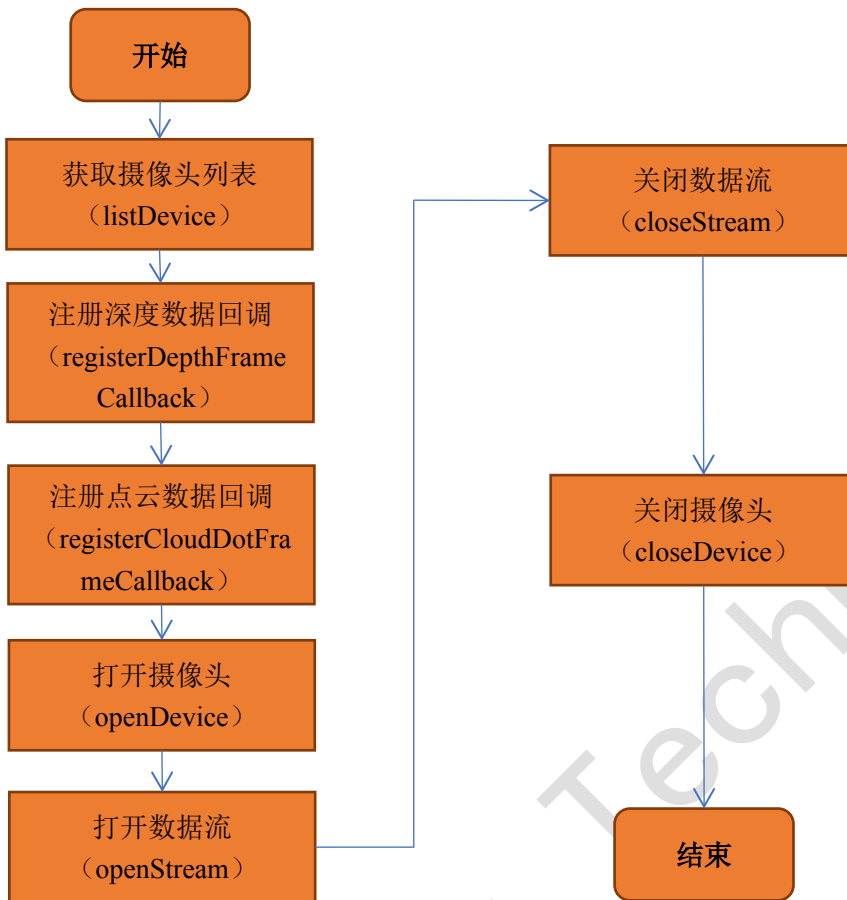


图 2 调用流程图

3 ANG SDK 核心 API 说明

3.1 获取深度图像接口

3.1.1 获取摄像头列表

```

1.  /**
2.     * 获取摄像头列表
3.     * 参数: [out] deviceIDs 设备号的集合, 里面设备号唯一, 但不一定连续
4.     * 返回值: 连接的设备个数
5.     */
6.  int listDevice(std::vector<int>& deviceIDs);
    
```


3.1.2 释放资源

```
1. /**
2.     * 释放资源
3.     * 参数: 无
4.     * 返回值 : 0 success ; != 0 fail
5.     */
6. int deinitCapture();
```

3.1.3 打开摄像头

```
1. /**
2.     * 打开摄像头
3.     * 参数:[in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.     * 返回值 : 0 success ; != 0 fail
5.     */
6. int openDevice(int deviceID);
```

3.1.4 关闭摄像头

```
1. /**
2.     * 关闭摄像头
3.     * 参数:[in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.     * 返回值 : 0 success ; != 0 fail
5.     */
6. int closeDevice(int deviceID);
```

3.1.5 打开深度数据流

```
1. /**
2.     * 打开深度数据流
3.     * 参数:[in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.     * 返回值 : 0 success ; != 0 fail
5.     */
6. int openStream(int deviceID);
```

3.1.6 关闭深度数据流

```
1. /**
2.  * 关闭深度数据流
3.  * 参数: [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  * 返回值 : 0 success ; != 0 fail
5.  */
6. int closeStream(int deviceID);
```

3.1.7 读取深度数据

```
1. /**
2.  * 读取深度数据 (用 3.1.8 注册深度数据回调函数代替)
3.  * 参数: [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *          [out] angFrame 数据帧
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int readFrame(int deviceID, AngFrame *angFrame);
```

3.1.8 注册深度数据回调函数

```
1. /**
2.  * 注册深度数据回调函数
3.  * 参数 [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  * 参数 : [in] pstCallback 深度数据回调函数
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int registerDepthImageCallback(int deviceID, const
    AS_DEPTHIMAGE_Callback *pstCallback);
```

3.1.9 注册点云数据回调函数

```
1. /**
2.  * 注册点云数据回调函数
3.  * 参数 [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  * 参数 : [in] pstCallback 点云数据回调函数
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int registerPointCloudCallback(int deviceID, const
    AS_POINTCLOUD_Callback *pstCallback);
```

3.1.10 设置分辨率

```
1. /**
2.  * 设置分辨率
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *          [in] zoom 缩小 DEFAULT (640*400) X2 (320*200)
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int setDownSample(int deviceID, Zoom zoom);
```

3.1.11 深度转点云

```
1. /**
2.  * 深度转点云
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *          [in] angFrame 深度数据帧
5.  *          [out] cloudDot 点云数据
6.  * 返回值 : cloudDot 点云数据大小
7.  */
8. int convertDepthToCloudDot(int deviceID, AngFrame angFrame, float*
    cloudDot);
```

3.1.12 旋转数据流

```
1. /**
2.  * 旋转数据流
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *          [in] rotate 旋转角度 ANGLE_0(0) CLOCKWISE_90(-90)
5.  *          *CLOCKWISE_180(-180) CLOCKWISE_270(-270) ANTICLOCKWISE_90(90)
6.  *          *ANTICLOCKWISE_180(180) ANTICLOCKWISE_270(270)
7.  * 返回值 : 0 success ; != 0 fail
8.  */
9. int setRotateAngle(int deviceID, eAngle rotate);
```

3.1.13 配置相机参数

```
1. /**
2.  * 配置相机参数
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *      [in] pstCamCfg 参数配置,
5.  *      isAtcOn: 是否开启温度补偿, 温漂功能需固件支持, 如固件不支持, 则配置
        false
6.  *      isDenoisesOn: 是否开启降噪功能;
7.  *      isFillHolesOn: 是否开启深度图补洞功能;
8.  *      isAntiDistortion: 是否开启深度图抗畸变功能;
9.  * 返回值 : 0 success ; != 0 fail
10. */
11. int setAsCamCfg(int deviceID, const AS_CAM_CFG_S *pstCamCfg);
```

3.1.14 设置帧率

```
1. /**
2.  * 设置深度图帧率, 此功能需固件支持切换帧率
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *      [in] enFps 帧率大小,
5.  * 返回值 : 0 success ; != 0 fail
6.  */
12. int setFrameRate(int deviceID, AS_FPS_E enFps);
```

3.1.15 设置投射器

```
1. /**
2.  * 设置摄像头图像投射器开关
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *      [in] enable true 打开 false 关闭
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int setVCSEL(int deviceID, bool enable);;
```

3.2 扩展 API 接口

3.2.1 获取 SDK 版本号

```

1. /**
2.  * 获取 SDK 版本号
3.  * 参数: [out] version 版本号
4.  * 返回值 : 0 success ; != 0 fail
5.  */
6. int getSDKSoftwareVersion(std::string &version);

```

3.2.4 获取摄像头 SN 号

```

1. /**
2.  * 获取 SN 号
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *          [out] SN sn 号
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int getSN(int deviceID, std::string &SN);

```

3.2.4 获取内参

```

1. /**
2.  * 获取内参
3.  * 参数 : [in] deviceID 设备号 取值范围:(listDevice 中容器 deviceIDs 的设备号)
4.  *          [out] irRgbparameter 内参结构体
5.  * 返回值 : 0 success ; != 0 fail
6.  */
7. int getIrRgbParameter(int deviceID, IrRgbParameter *irRgbParameter);

```

4. 获取深度视频流和点云代码实现

```

1. /*深度数据回调函数*/
2. static void AS_SDK_DepthImageCallback(int deviceId, const
3.     AS_DEPTHIMAGE_Data *pstFrame, void *privateData)
4. {
5.
6. }
7.

```

```
8. /*点云数据回调函数*/
9. static void AS_SDK_PointCloudCallback(int deviceId, const
10.     AS_POINTCLOUD_Data *pstPointCloud, void *privateData)
11. {
12.
13. }
14. std::vector<int> deviceIDs;
15. /* Main program for angstrong nuwa camera sdk demo */
16. int main(int argc, char *argv[])
17. {
18.     int ret = 0;
19.     Capture capture;
20.     std::string sdkVersion;

21.     LOG(INFO) << "Hello, angstrong NUWA camera" << std::endl;
22.     capture.getSDKSoftwareVersion(sdkVersion); /*获取摄像头 sdk*/
23.     int deviceNum = capture.listDevice(deviceIDs); /*获取连接的摄像头列表
    */
24.     if (deviceNum <= 0) {
25.         LOG(INFO) << "no device!\n";
26.         return 0;
27.     }
28.     LOG(INFO) << "Found " << deviceNum << " nuwa camera" << std::endl;
29.
30.     for (int idx = 0; idx < deviceNum; idx++) {
31.         AS_CAM_CFG_S stCamCfg;
32.         /*register callback function*/
33.         AS_DEPTHIMAGE_Callback stDepthImageCallback = {
34.             .callback = AS_SDK_DepthImageCallback,
35.             .privateData = (void *)0x1234,
36.         };
37.         AS_POINTCLOUD_Callback stPointCloudCallback = {
38.             .callback = AS_SDK_PointCloudCallback,
39.             .privateData = (void *)0x1234,
40.         };
41.
42.         /* 注册深度数据回调函数 */
43.         capture.registerDepthImageCallback(deviceIDs.at(idx),
44.             &stDepthImageCallback);
45.         /* 注册点云数据回调函数 */
46.         capture.registerPointCloudCallback(deviceIDs.at(idx),
47.             &stPointCloudCallback);
48.         /* 打开相机 */
49.         if (capture.openDevice(deviceIDs.at(idx)) != 0) {
50.             LOG(ERROR) << "device id " << deviceIDs.at(idx) << " open
                device failed!\n";
```

```

51.             continue;
52.         }
53.         LOG(INFO) << "Open camera " << deviceIDs.at(idx) << " success" <<
std::endl;
54.
55.         /* 获取默认参数 */
56.         capture.getAsCamCfg(deviceIDs.at(idx), &stCamCfg);
57.         LOG(INFO) << " default camera cfg: Atc[" << stCamCfg.isAtcOn <<
"]" \
58.             << " Denoises[" << stCamCfg.isDenoisesOn << "]" \
59.             << " FillHoles[" << stCamCfg.isFillHolesOn << "]" \
60.             << " AntiDistortion[" << stCamCfg.isAntiDistortion <<
"]" \
61.             << std::endl;
62.         // stCamCfg.isAtcOn = false; /*温飘功能需模组固件支持*/
63.         // stCamCfg.isDenoisesOn = true; /*降噪功能*/
64.         // stCamCfg.isFillHolesOn = false; /*补洞功能*/
65.         // stCamCfg.isAntiDistortion = false; /*抗畸变功能*/
66.         /* 配置相机参数 */
67.         // capture.setAsCamCfg(deviceIDs.at(idx), &stCamCfg);
68.
69.         /* 设置帧率, 此功能需模组固件支持 */
70.         // capture.setFrameRate(deviceIDs.at(idx), AS_FPS_15);
71.         /* 打开深度数据流 */
72.         if (capture.openStream(deviceIDs.at(idx)) != 0) {
73.             capture.closeDevice(deviceIDs.at(idx));
74.             LOG(ERROR) << "device id " << deviceIDs.at(idx) << " open
stream failed!\n";
75.             continue;
76.         }
77.
78.         /* 获取相机序列号 */
79.         std::string SN;
80.         capture.getSN(deviceIDs.at(idx), SN);
81.         LOG(INFO) << "SN: " << SN << std::endl;
82.
83.         /* 获取相机内参 */
84.         IrRgbParameter irRgbParameter;
85.         capture.getIrRgbParameter(deviceIDs.at(idx), &irRgbParameter);
86.         LOG(INFO) << "fx: " << irRgbParameter.irFx << std::endl;
87.         LOG(INFO) << "fy: " << irRgbParameter.irFy << std::endl;
88.         LOG(INFO) << "cx: " << irRgbParameter.irCx << std::endl;
89.         LOG(INFO) << "cy: " << irRgbParameter.irCy << std::endl;
90.     }
91.
92.     system("pause");

```

```
93.  
94.     return 0;  
95. }
```