YDLIDAR HP60C-SDK

Development And Use Manual

CONTENTS

1 INTRODUCTION

1.1 product Overview

The HP60C SDK API provides a one-stop service for third-party applications to integrate camera module functions. Based on the SDK, it can quickly provide customers with secondary development services to prevent other problems caused by non-standard calls in the process of using the HP60C SDK API.

Used to drive the HP60C SDK, provide real-time depth of image figure, point cloud image, IR, RGB figure and PEAK figure, etc.

1.2 Environmental Requirement

1.2.1 Operating Environment

ARM or X86 platform

1.2.2 System Requirement

Linux: Ubuntu16.04/Ubuntu18.04/Ubuntu20.04/Ubuntu22.04 (recommend Ubuntu18.04)

ROS1: Kinetic Kame/Melodic Morenia/Noetic Ninjemys (recommend Melodic Morenia)

ROS2: Foxy Fitzroy/Galactic Geochelone/Humble Hawksbill (recommend Humble Hawksbill)

1.3 SDK function Introduction

1.3.1 Acquire Depth

Support to drive HP60C, real-time depth map acquisition;

The space of depth data format is as follows:

Chart 1 Depth chart format

Bit width	16
Unit	mm
Type	unsigned short

1.3.2 Obtain Point Cloud

Support to drive HP60C, real-time acquisition of point cloud image;

Get point cloud data correction to the image.

1.3.3 Obtain IR

Support to drive HP60C, real-time IR map acquisition;

In the IR image data correction chart.

1.3.4 obtain RGB

Support to drive HP60C, real-time access to RGB map;

In RGB image data correction chart.

1.3.5 Acuire PEAK

Support to drive HP60C, real-time access to PEAK graph;

In image data callback PEAK figure.

2 SDK Integration Guide

2.1 Packet Structure Specification

```
SDK
├─── docs                                  SDK Development and ROS instructions
├─── sample
|      ├─── linux                          The reference example for linux
|           ├─── configurationfiles          Directory for storing configuration
files
|           └─── libs                       SDK Header files and library files
|                ├─── include              SDK header file
|                └─── lib                  SDK library file
|                     ├─── aarch64-linux-gnu     aarch64-linux-gnu-g++ compiler SDK
library
|                     └─── x86_64          x86_64 SDK library
|      ├─── linux_listener                 Example of hot plugging with listening
function in linux
|      ├─── ros                            ROS1 reference example under ros
system
|      ├─── ros2                           ROS2 reference example under ros
system
|      ├─── win                            The reference example for windows
|      ├─── win_listener                   Example of hot plugging with listening
```

```
function windows
|       ├──── upgrade                                  Reference example for upgrading under
Linux system
|           ├──── configurationfiles                   Directory for storing configuration
files
|           ├──── linux_x86                            Upgrade program executable files
under Linux system
|           └──── win_msvc                             Upgrade program executable files
under Windows system
```

2.2 Example Program Running Instructions

2.2.1 Camera Configuration File Description

Camera configurationfile in the directory "configurationfiles", you need to upload the configurationfile path when you call the "Open camera" interface.

2.2.2 Run Ubuntu (Linux) System Sample Program

Go to the sample/Linux or sample/linux_listener (listening function) directory, where build.sh is the compilation script and run_camera.sh is the script for running the sample program. Firstly, execute build. sh for compilation, and then execute the run_camera. sh script to run the program. Executing programs requires root privileges, and the script will run in sudo mode.

Compile:
./build.sh
Run:
./run_ascamera.sh

2.2.3 Run ROS/ROS2 System Sample Program

Go to the sample/ros or ros2 directory, where build.sh is the compilation script and run.ascamera. sh is the script for starting the node. Firstly, execute build. sh for compilation, and then execute the run_camera. sh script to run the program. Starting a node requires root privileges, and the script will run in sudo mode.

Compile:
./build.sh
Start node:
./run_ascamera.sh

2.2.4 How To cross-compile

Cross compilation is the process of generating executable code on one platform from another. Compile executable files that can run on Linux systems with arm/arch64 architecture on x86_64 architecture. In the sample code of the Linux SDK, the corresponding library files will be automatically

linked based on the host's architecture. Therefore, the SDK will be placed on an arm development board that supports compilers. The compilation method is the same as in the x86_64 architecture, and only the build.sh script in the directory needs to be executed. To cross compile and generate executable files for arm/arch64 in Linux on x86_64, please refer to the following chapters. The currently compiled arm/arch64 version of the dynamic link library in the SDK can be viewed in the/libs/lib directory, corresponding to the name of the compiler. If the required arm/arch64 version of the dynamic link library is not available, a cross compilation toolchain for Linux on x86_64 needs to be provided to recompile and generate the relevant dynamic link library.

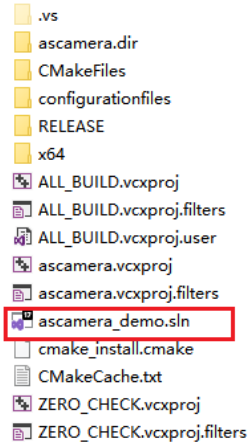
2.2.5 Run Windows System Sample Program

2.2.5.1 Directly Run The Compiled Program

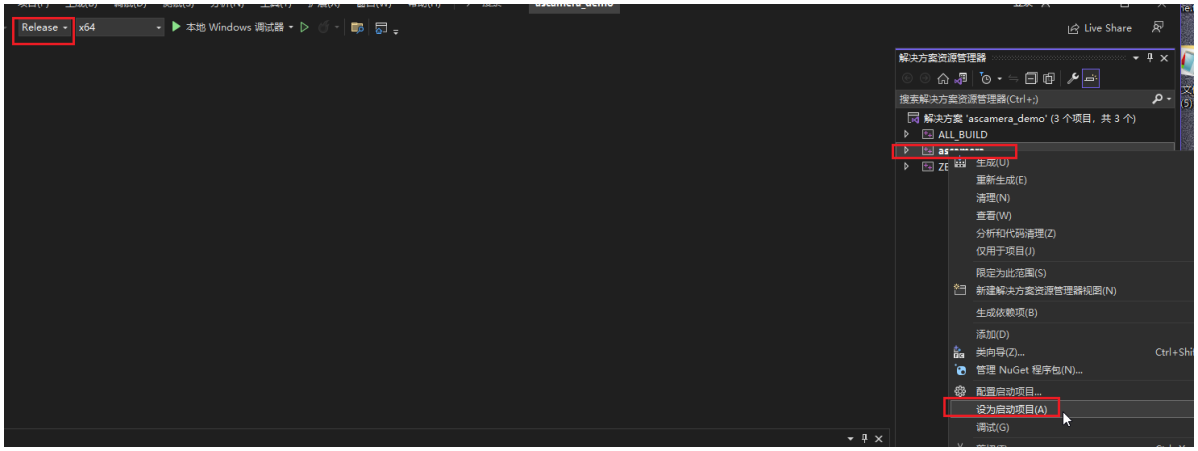
Go to the sample/win/build/RELEASE directory and double-click SdkTest.exe to run the program.

2.2.5.2 Compile Runner

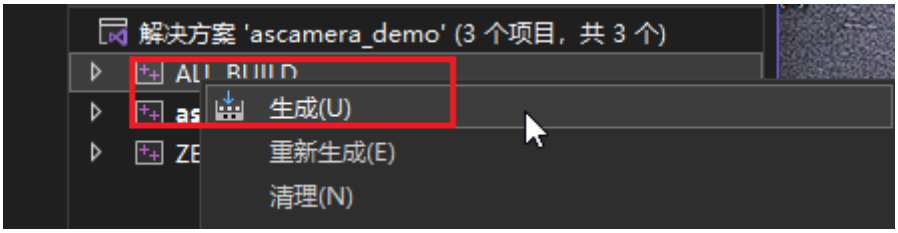
1. Go to the sample/win/build directory and run the cmake .. Compile.
2. Double-click the SLNS suffix detected file.



3. After selecting Release, right-click on the ascamera on the right and set it to start the project.



4. Right-click ALL_BUILD on the right and click Build.



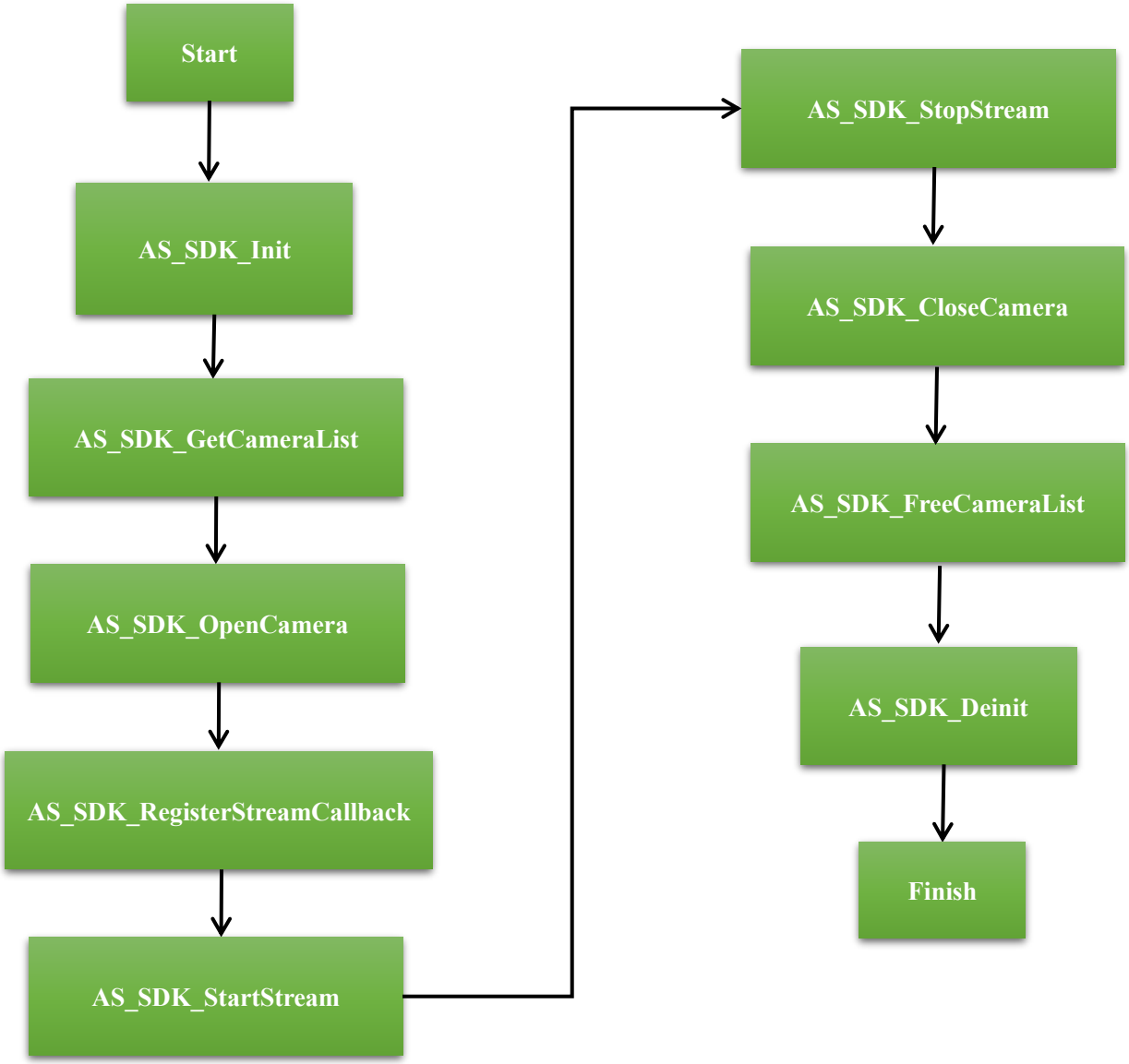
5. Go to the sample/win/build/RELEASE directory and double-click ascamera.exe to run the program.

2.2.6 Use Of Upgrading Sample Programs

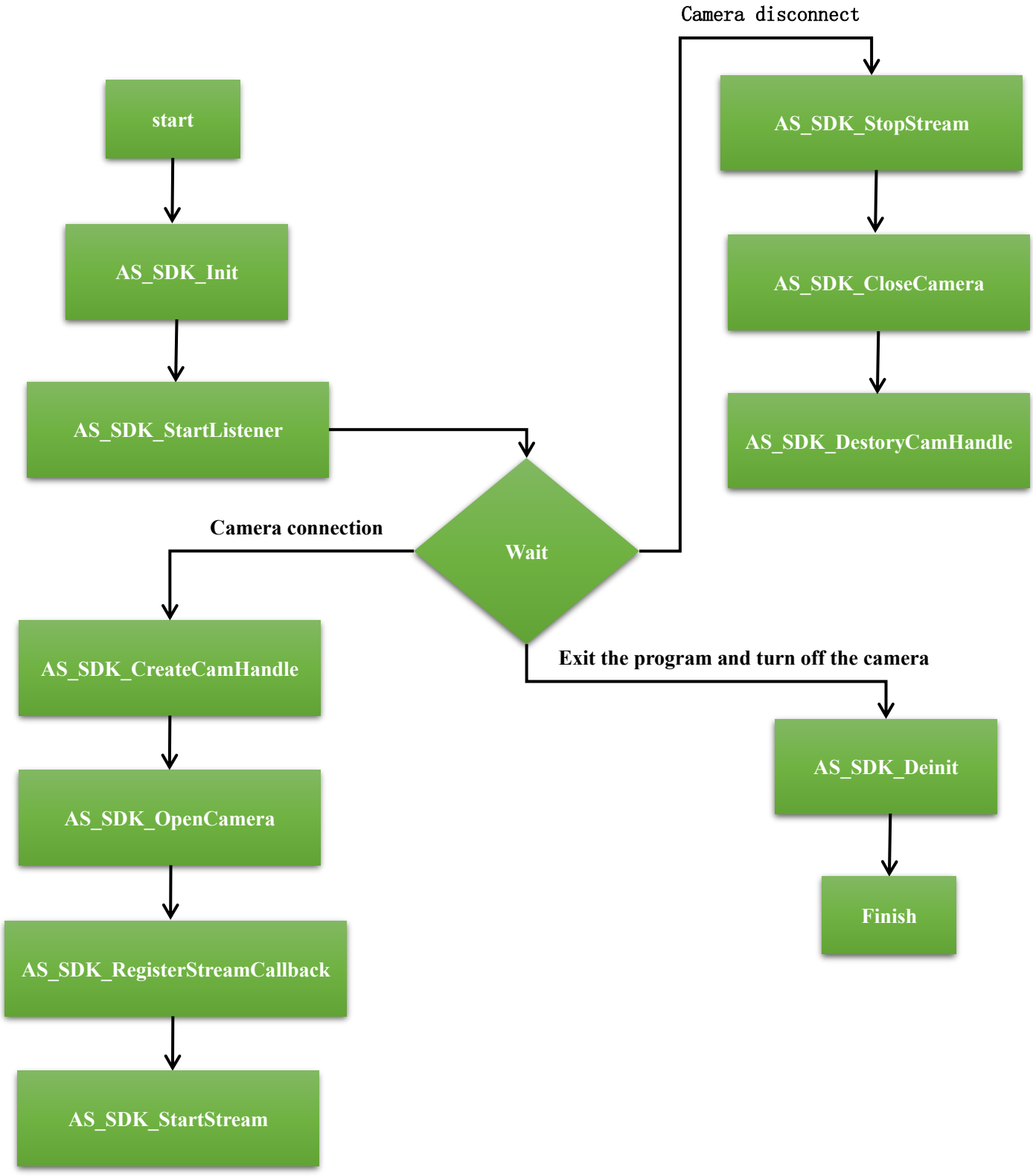
Upgrade the example program in the sample/upgrade directory. Enter the directory and find readme.md to quickly view the usage method. This directory provides upgrade programs for Linux systems. Additionally. This section provides compiled executable files for Linux x86 architecture and Windows system. The instructions for use can be found in the readme.txt file in the sample/upgrade/executable directory.

2.3 Call Flowchart

2.3.1 Actively Obtaining A Camera List



2.3.2 Camera Management Based On Hot Plugging



3 DESCRIPTION OF IMPORTANT DATA STRUCTURES

3.1 Stream Interface Data Type

3.1.1 AS_CAM_PTR

[Description]

Camera handle

[Definition]

typedef void * AS_CAM_PTR;

3.1.2 AS_FRAME_Type_e

[Description]

Data frame type

[Definition]

```
typedef enum FRAME_TYPE_E {
    AS_FRAME_TYPE_DEPTH = 0,
    AS_FRAME_TYPE_RGB,
    AS_FRAME_TYPE_IR,
    AS_FRMAE_TYPE_POINTCLOUD,
    AS_FRAME_TYPE_YUYV,
    AS_FRAME_TYPE_PEAK,
    AS_FRAME_TYPE_BUTT
} AS_FRAME_Type_e;
```

[Parameters]

Member	Description
AS_FRAME_TYPE_DEPTH	Depth data frame type
AS_FRAME_TYPE_RGB	Color image frame type
AS_FRAME_TYPE_IR	IR image frame type
AS_FRMAE_TYPE_POINTCLOUD	Point cloud data frame type
AS_FRMAE_TYPE_YUYV	YUYV data
AS_FRAME_TYPE_PEAK	PEAK image frame type
AS_FRAME_TYPE_BUTT	invalid value

3.1.3 AS_Frame_s

[Description]

Frame structure

[Definition]

```
typedef struct CAM_FRAME {
    AS_FRAME_Type_e type;
    unsigned int width;
    unsigned int height;
```

```
    void *data;
    unsigned int size;
    unsigned int bufferSize;
    unsigned int frameId;
    unsigned long long ts;
} AS_Frame_s;
```

[Parameters]

Member	Description
type	Data frame type
width	Image width
height	Image height
data	Data cache address
size	The size of valid data, in bytes
bufferSize	The total size of data cache space, in bytes
frameId	The sequence number of this frame
ts	The timestamp of this frame is taken from the system time of the system running the SDK, in milliseconds

3.1.4 AS_SDK_Data_s

[Description]

Image data structure

[Definition]

```
typedef struct CAM_DATA {
    AS_Frame_s depthImg;
    AS_Frame_s rgbImg;
    AS_Frame_s irImg;
    AS_Frame_s pointCloud;
    AS_Frame_s yuyvImg;
    AS_Frame_s peakImg;
    std::vector<AS_POINTCLOUD_s> pointCloud2;
} AS_SDK_Data_s;
```

[Parameters]

Member	Description
depthImg	Depth data, unsigned short type, in millimeters
rgbImg	Color image data, BGR format, unsigned char type
irImg	Infrared image data, unsigned char type
pointCloud	Point cloud data corresponding to unaligned depth images, float type, arranged in xyz order, invalid points discarded
yuyvImg	YUYV data, unsigned char type
peakImg	PEAK image data, unsigned char type
pointCloud2	Polar point cloud data (only supports KONDYOR usage, please ignore other types of modules)

3.1.5 AS_SDK_MERGE_s

[Description]

Fusing image data structures

[Definition]

```
typedef struct CAM_MERGE {
    AS_Frame_s depthImg;
    AS_Frame_s pointCloud;
} AS_SDK_MERGE_s;
```

[Parameters]

Member	Description
depthImg	Depth data
pointCloud	Point cloud data

3.1.6 AS_CAM_StreamCallback

[Description]

Image data callback function

[Definition]

typedef void(*AS_CAM_StreamCallback)(AS_CAM_PTR pCamera, const AS_SDK_Data_s *pstData, void *privateData);

[Parameters]

Member	Description
pCamera	Camera handle
pstData	Image data
privateData	User Private Data

3.1.7 AS_CAM_Stream_Cb_s

[Description]

Image data callback data structure

[Definition]

```
typedef struct STREAM_CALLBACK_S {
    AS_CAM_StreamCallback callback;
    void *privateData;
} AS_CAM_Stream_Cb_s;
```

[Parameters]

Member	Description
callback	Data callback function
privateData	User Private Data

3.1.8 AS_CAM_MergeFrameCallback

[Description]

Fusion image data callback function

[Definition]

typedef void(*AS_CAM_MergeFrameCallback)(AS_CAM_PTR pCamera, const AS_SDK_Merge_s *pstData, void *privateData);

[Parameters]

Member	Description
pCamera	Camera handle
pstData	Image data
privateData	User Private Data

3.1.9 AS_CAM_Merge_Cb_s

[Description]

Fusion image data callback data structure

[Definition]

typedef struct MERGE_CALLBACK_S {
    AS_CAM_MergeFrameCallback callback;
    void *privateData;
} AS_CAM_Merge_Cb_s;

[Parameters]

Member	Description
callback	Data callback function
privateData	User Private Data

3.1.10 AS_Listener_Callback

[Description]

Listening function callback function

[Definition]

typedef void(*AS_Listener_Callback)(AS_CAM_ATTR_S *attr, void *privateData);

[Parameters]

Member	Description
attr	Camera information
privateData	User Private Data

3.1.11 AS_LISTENER_CALLBACK_S

[Description]

The listening function calls back the data structure

[Definition]

typedef struct AS_LISTENER_CALLBACK {
    AS_Listener_Callback onAttached;
    AS_Listener_Callback onDetached;
    void *privateData;
} AS_LISTENER_CALLBACK_S;

[Parameters]

Member	Description
onAttached	Camera connection callback
onDetached	Camera disconnection callback
privateData	User Private Data

3.1.12 AS_LISTENER_TYPE_E

[Description]

Listening type

[Definition]

```
typedef enum AS_LISTENER_TYPE {
    AS_LISTENNER_TYPE_USB = 0,
    AS_LISTENNER_TYPE_NET,
    AS_LISTENNER_TYPE_BUTT
} AS_LISTENER_TYPE_E;
```

[Parameters]

Member	Description
AS_LISTENNER_TYPE_USB	USB type
AS_LISTENNER_TYPE_NET	Network type
AS_LISTENNER_TYPE_BUTT	Invalid value

3.1.13 IMG_TYPE_FLG

[Description]

Image type flag

[Definition]

```
#define DEFAULT_IMG_FLG         0x00000000
#define DEPTH_IMG_FLG           0x00000001
#define RGB_IMG_FLG             0x00000002
#define IR_IMG_FLG              0x00000004
#define POINTCLOUD_IMG_FLG      0x00000008
#define YUYV_IMG_FLG            0x00000010
#define PEAK_IMG_FLG            0x00000020
```

[Parameters]

Member	Description
DEFAULT_IMG_FLG	Default Image Type
DEPTH_IMG_FLG	Depth Image Type
RGB_IMG_FLG	Color image types
IR_IMG_FLG	IR image type
POINTCLOUD_IMG_FLG	Point cloud image types
YUYV_IMG_FLG	Yuyv Image Type
PEAK_IMG_FLG	PEAK image type

3. 1. 14 AS_POINTCLOUD_s

typedef struct AS_POINTCLOUD {

double angle;

double distance;

double quality;

} AS_POINTCLOUD_s;

[Parameters]

Member	Description
angle	Polar point cloud angle
distance	Polar point cloud distance
quality	Polar point cloud quality

[Note]

This structure is only supported by KONDYOR, other types of modules should be ignored.

3.1.15 AS_CONFIG_INFO_s

typedef struct AS_CONFIG_INFO {

bool is_Registration;

bool is_pclOrganized;

} AS_CONFIG_INFO_S;

[Parameters]

Member	Description
is_Registration	Are depth data and RGB data aligned
is_pclOrganized	Is the output an ordered point cloud

[Note]

This structure is used to query RGB camera configuration information. Please ignore other types of modules.

3.2 Communication Interface Data Type

3.2.1 AS_CAM_Parameter_s

[Description]

Camera internal and external parameters

[Definition]

typedef struct CAM_Parameter {

float fxir;

float fyir;

float cxir;

float cyir;

float fxrgb;

```
    float fyrgb;
    float cxrgb;
    float cyrgb;

    float R00;
    float R01;
    float R02;
    float R10;
    float R11;
    float R12;
    float R20;
    float R21;
    float R22;
    float T1;
    float T2;
    float T3;

  float K1ir;
  float K2ir;
  float K3ir;
  float P1ir;
  float P2ir;
  float K1rgb;
  float K2rgb;
  float K3rgb;
  float P1rgb;
  float P2rgb;
  } AS_CAM_Parameter_s;
```

[Parameters]

Member	Description
fxir	Horizontal focal length of infrared cameras
fyir	Vertical focal length of infrared cameras
cxir	Infrared camera optical center horizontal axis
cyir	Infrared camera optical center ordinate
fxrgb	Horizontal focal length of color camera
fyrgb	Vertical focal length of color camera
cxrgb	Color camera optical center horizontal axis
cyrgb	Color camera optical center vertical axis
R00	Infrared and color camera extrinsic rotation matrix elements

Member	Description
R01	Infrared and color camera extrinsic rotation matrix elements
R02	Infrared and color camera extrinsic rotation matrix elements
R11	Infrared and color camera extrinsic rotation matrix elements
R12	Infrared and color camera extrinsic rotation matrix elements
R20	Infrared and color camera extrinsic rotation matrix elements
R21	Infrared and color camera extrinsic rotation matrix elements
R22	Infrared and color camera extrinsic rotation matrix elements
T1	Elements of the extrinsic translation matrix for infrared and color cameras
T2	Elements of the extrinsic translation matrix for infrared and color cameras
T3	Elements of the extrinsic translation matrix for infrared and color cameras
K1ir	IR distortion parameters
K2ir	IR distortion parameters
K3ir	IR distortion parameters
P1ir	IR distortion parameters
P2ir	IR distortion parameters
K1rgb	RGB distortion parameters
K2rgb	RGB distortion parameters
K3rgb	RGB distortion parameters
P1rgb	RGB distortion parameters
P2rgb	RGB distortion parameters

3.2.2 AS_STREAM_Param_s

[Description]

Camera internal and external parameters

[Definition]

```
typedef struct STREAM_PARAM_S {
    int width;
    int height;
int fps;
} AS_STREAM_Param_s;
```

[Parameters]

Member	Description
width	Image width
height	Image height
fps	Frame rate

3.2.3 AS_MEDIA_TYPE_E

[Description]

Stream Type

[Definition]

```
typedef enum MEDIA_TYPE_E {
```

AS_MEDIA_TYPE_DEPTH,
AS_MEDIA_TYPE_RGB,
AS_MEDIA_TYPE_IR,
AS_MEDIA_TYPE_PEAK,
AS_MEDIA_TYPE_BUTT
} AS_MEDIA_TYPE_E;

[Parameters]

Member	Description
AS_MEDIA_TYPE_DEPTH	Depth type
AS_MEDIA_TYPE_RGB	RGB type
AS_MEDIA_TYPE_IR	IR type
AS_MEDIA_TYPE_PEAK	PEAK type
AS_MEDIA_TYPE_BUTT	Invalid value

3.2.4 AS_CAM_UpGradeCallback

[Description]

Firmware upgrade status callback function

[Definition]

typedef void(*AS_CAM_UpGradeCallback)(AS_CAM_PTR pCamera, void *privateData,float fProcess);

3.2.5 AS_CAM_Upgrade_Dev_s

[Description]

Device firmware upgrade status callback data structure

[Definition]

typedef struct CAM_ UPGRADE_DEV_S {
AS_CAM_UpGradeCallback callback;
void *privateData;
} AS_CAM_Upgrade_Dev_s;

[Parameters]

Member	Description
callback	Device firmware upgrade status callback function
privateData	User Private Data

3.2.6 AS_TOFMODE_E

[Description]

Firmware mode

[Definition]

typedef enum TOFMode {
odd_mode = 0,

even_mode = 1,

merge_mode = 2,

mode_butt

} AS_TOFMODE_E;

[Parameters]

Member	Description
odd_mode	Individual odd frame mode
even_mode	Individual even frame mode
merge_mode	Parity alternating frame mode
mode_butt	Invalid value

3.2.7 AS_SDK_CAM_MODEL_E

[Description]

Camera model

[Definition]

typedef enum CAMERA_MODEL_E {

AS_SDK_CAM_MODEL_UNKNOWN,

AS_SDK_CAM_MODEL_NUWA_XB40,

AS_SDK_CAM_MODEL_NUWA_X100,

AS_SDK_CAM_MODEL_NUWA_HP60,

AS_SDK_CAM_MODEL_NUWA_HP60V,

AS_SDK_CAM_MODEL_KUNLUN_A,

AS_SDK_CAM_MODEL_KUNLUN_C,

AS_SDK_CAM_MODEL_KONDYOR,

AS_SDK_CAM_MODEL_KONDYOR_NET,

AS_SDK_CAM_MODEL_HP60C,

AS_SDK_CAM_MODEL_HP60CN,

AS_SDK_CAM_MODEL_VEGA,

AS_SDK_CAM_MODEL_CHANGA,

AS_SDK_CAM_MODEL_BUTT

} AS_SDK_CAM_MODEL_E;

[Parameters]

Member	Description
AS_SDK_CAM_MODEL_UNKNOWN	Unknown type
AS_SDK_CAM_MODEL_NUWA_XB40	XB40
AS_SDK_CAM_MODEL_NUWA_X100	X100
AS_SDK_CAM_MODEL_NUWA_HP60	HP60
AS_SDK_CAM_MODEL_NUWA_HP60V	HP60V
AS_SDK_CAM_MODEL_KUNLUN_A	KUNLUN-A
AS_SDK_CAM_MODEL_KUNLUN_C	KUNLUN-C

Member	Description
AS_SDK_CAM_MODEL_KONDYOR	KONDYOR(USB)
AS_SDK_CAM_MODEL_KONDYOR_NET	KONDYOR(NET)
AS_SDK_CAM_MODEL_HP60C	HP60C
AS_SDK_CAM_MODEL_HP60CN	HP60CN
AS_SDK_CAM_MODEL_VEGA	VEGA
AS_SDK_CAM_MODEL_CHANGA	CHANG-A

3.2.8 AS_CAM_ATTR_TYPE_E

[Description]

Camera model

[Definition]

```
typedef enum AS_CAM_ATTR_TYPE {
    AS_CAMERA_ATTR_LNX_USB,
    AS_CAMERA_ATTR_WIN_USB,
    AS_CAMERA_ATTR_NET,
    AS_CAMERA_ATTR_BUTT
} AS_CAM_ATTR_TYPE_E;
```

[Parameters]

Member	Description
AS_CAMERA_ATTR_LNX_USB	Linux USB type
AS_CAMERA_ATTR_WIN_USB	Windows USB type
AS_CAMERA_ATTR_NET	Network type
AS_CAMERA_ATTR_BUTT	Invalid value

3.2.9 AS_USB_DEV_ATTR_S

[Description]

Linux USB type structure

[Definition]

```
typedef struct AS_USB_DEV_ATTR {
    int vid;
    int pid;
    char serial[64];
    int bnum;
    int dnum;
    int port_number;
    char port_numbers[64];
} AS_USB_DEV_ATTR_S;
```

[Parameters]

Member	Description
vid	Supplier identification code
pid	Product identification code
serial	SN serial number
bnum	Bus number
dnum	Equipment number
port_number	Port number
port_numbers	Port path

3.2.10 AS_NETWORK_DEV_ATTR_S

[Description]

Network type structure

[Definition]

```
typedef struct AS_NETWORK_DEV_ATTR {
    char ip_addr[64];
    int port;
    AS_SDK_CAM_MODEL_E model;
} AS_NETWORK_DEV_ATTR_S;
```

[Parameters]

Member	Description
ip_addr	IP address
port	Port number
model	Camera model

3.2.11 AS_USB_WIN_DEV_ATTR_S

[Description]

Windows USB type structure

[Definition]

```
typedef struct AS_USB_WIN_DEV_ATTR {
    int vid;
    int pid;
    int deviceID;
    char symbol_link[260];
    char location_path[260];
} AS_USB_WIN_DEV_ATTR_S;
```

[Parameters]

Member	Description
vid	Supplier identification code
pid	Product identification code
deviceID	Device ID
symbol_link	Device unique identification code

Member	Description
location_path	Device path

3.2.12 AS_CAM_ATTR_S

[Description]

Camera information structure

[Definition]

```
typedef struct AS_CAM_ATTR {
    AS_CAM_ATTR_TYPE_E type;
    union {
        AS_USB_DEV_ATTR_S usbAttrs;
        AS_NETWORK_DEV_ATTR_S netAttrs;
        AS_USB_WIN_DEV_ATTR_S winAttrs;
    } attr;
} AS_USB_WIN_DEV_ATTR_S;
```

[Parameters]

Member	Description
type	Camera type
usbAttrs	Linux USB Information Architecture
netAttrs	Network Information Structure
winAttrs	Windows USB Information Architecture

3.2.13 AS_NET_DEV_Attrs_s

[Description]

Network device information

[Definition]

```
typedef struct NET_DEV_ATTRS {
    char ip_addr[64];
    char mac_addr[64];
    int is_dhcp;
} AS_NET_DEV_Attrs_s;
```

[Parameters]

Member	Description
ip_addr	IP address
mac_addr	MAC address
is_dhcp	Is DHCP enabled. 0 off, 1 on, -1 not set

3.2.14 AS_TIME_STAMP_TYPE_E

[Description]

Timestamp type

[Definition]

```
typedef enum TIME_STAMP_TYPE {
    AS_TIME_STAMP_TYPE_STEADY_CLOCK,
    AS_TIME_STAMP_TYPE_SYSTEM_CLOCK,
    AS_TIME_STAMP_TYPE_BUTT
} AS_TIME_STAMP_TYPE_E;
```

[Parameters]

Member	Description
AS_TIME_STAMP_TYPE_STEADY_CLOCK	The system's monotonous clock starts counting from the start time of 0
AS_TIME_STAMP_TYPE_SYSTEM_CLOCK	system time
AS_TIME_STAMP_TYPE_BUTT	invalid value

4 ANG SDK CORE API DESCRIPTION

4.1 Stream Interface Definition

4.1.1 Initialize SDK

[Function]

SDK initialization

[Format]

```
int AS_SDK_Init();
```

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.2 Deinitializate SDK

[Function]

SDK deinitialization

[Format]

```
int AS_SDK_Deinit();
```

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.3 Get camera list

[Function]

Gets a list of connected cameras

[Format]

int AS_SDK_GetCameraList(std::list<AS_CAM_PTR> &devList);

[Parameters]

Type	Name	Description	IN/OUT
std::list<AS_CAM_PTR>	devList	Camera list	OUT

[Returned value]

Type	Description
int	Number of cameras greater than 0; Other abnormal values

[Note]

N/A

4.1.4 Release Camera List

[Function]

Release camera list resources

[Format]

int AS_SDK_FreeCameraList(std::list<AS_CAM_PTR> &devList);

[Parameters]

Type	Name	Description	IN/OUT
std::list<AS_CAM_PTR>	devList	Camera list	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

After the call, devList will be cleared and the camera pointer inside will be released. Subsequent actions may result in unknown errors.

4.1.5 Turn On Camera

[Function]

Open the specified camera

[Format]

int AS_SDK_OpenCamera(AS_CAM_PTR pCamera, const char *pParaFilePath);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
char*	pParaFilePath	Camera configuration file path	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

The camera configuration file is explained in section 2.2.1 above this document.

Before repeating the call, the camera needs to be turned off first, otherwise a negative value will be returned.

4.1.6 Turn Off The Camera

[Function]

Turn off the specified camera

[Format]

int AS_SDK_CloseCamera(AS_CAM_PTR pCamera);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.7 Register The Image Data Callback Function

[Function]

Register image data callback function

[Format]

int AS_SDK_RegisterStreamCallback(AS_CAM_PTR pCamera, AS_CAM_Stream_Cb_s *pstCallback);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_CAM_Stream_Cb_s	pstCallback	callback function	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.8 Register Image Fusion Data Callback Function

[Function]

Register image fusion data callback function

[Format]

int AS_SDK_RegisterMergeFrameCallback(AS_CAM_PTR pCamera, AS_CAM_Merge_Cb_s *pstCallback);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_CAM_Merge_Cb_s	pstCallback	Callback function	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Only supports kunlunA.

4.1.9 Open Camera Data Stream

[Function]

Opens the image for the specified camera

[Format]

int AS_SDK_StartStream(AS_CAM_PTR pCamera, int type);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	Camera handle, obtained by the AS-SDK_GetCameraList interface or AS_CreateCamHandle interface	IN

Type	Name	Description	IN/OUT
int	type	The image type IMG-TYPEFLG is one or a combination of depth, point cloud, etc., and its effective parameters need to be determined based on the camera type. When it is 0, the SDK will output according to the default value. Combining through | operations, such as DEPTH_IMG_FLG | POINTL_CLOUD_FLG	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.10 Turn Off Camera Data Stream

[Function]

Close the image of the specified camera

[Format]

int AS_SDK_StopStream(AS_CAM_PTR pCamera, int type);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS-SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
int	type	The image type IMG-TYPEFLG is one or a combination of depth, point cloud, etc., and its effective parameters need to be determined based on the camera type. When it is 0, close all images	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.11 Enable The Device Listening Function

[Function]

Activate device listening function

[Format]

int AS_SDK_StartListener(const AS_LISTENER_CALLBACK_S &callback, AS_LISTENER_TYPE_E type, bool enumerate = true);

[Parameters]

Type	Name	Description	IN/OUT
AS_LISTENER_CALLBACK_S	callback	Device listening callback function	IN
AS_LISTENER_TYPE_E	type	listening type	IN
bool	enumerate	Do you want to enumerate devices	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Windows version SDK and network type devices are not supported.

4.1.12 Stop Device Monitoring Function

[Function]

Stop device monitoring function

[Format]

int AS_SDK_StopListener(void);

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.13 Create A Camera Handle

[Function]

Create a camera handle

[Format]

int AS_SDK_CreateCamHandle(AS_CAM_PTR &pCamera,AS_CAM_ATTR_S *attr);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	OUT
AS_CAM_ATTR_S	attr	Camera Properties	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.14 Destroy The Camera Handle

[Function]

Destroy the camera handle

[Format]

int AS_SDK_DestoryCamHandle(AS_CAM_PTR pCamera);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.1.15 Get Camera Properties

[Function]

Get camera properties

[Format]

int AS_SDK_GetCameraAttrs(AS_CAM_PTR pCamera, AS_CAM_ATTR_S &attr);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_CAM_ATTR_S	attr	Camera Properties	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2 Communication Interface Definition

4.2.1 Get The Camera Firmware Version Number

[Function]

Get the camera firmware version number

[Format]

int AS_SDK_GetFwVersion(AS_CAM_PTR pCamera, char *pszVersion, int size);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
char	pszVersion	Camera firmware version number	OUT
int	size	PszVersion maximum space value	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.2 Obtain SDK Version

[Function]

Obtain SDK version

[Format]

int AS_SDK_GetSwVersion(char *pszVersion, unsigned int size);

[Parameters]

Type	Name	Description	IN/OUT
char	pszVersion	SDK version number	OUT
unsigned int	size	PszVersion maximum space value	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.3 Get The Type Of The Stream

[Function]

Get the type of the stream

[Format]

int AS_SDK_GetStreamType(AS_CAM_PTR pCamera, int *type);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
int*	type	Get Stream Image Type (IMG-TYPE.FLG)	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Not accepted for the moment.

4.2.4 Get Camera SN

[Function]

Obtain serial number

[Format]

int AS_SDK_GetSerialNumber(AS_CAM_PTR pCamera, char *pszSerialNo, int size);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	Camera handle	IN
char*	pszSerialNo	Space for storing serial numbers	OUT
int	size	Space size	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.5 Upgrade Camera Firmware

[Function]

Update Firmware

[Format]

int AS_SDK_UpgradeDev(AS_CAM_PTR pCamera, char *pszFilePath, AS_CAM_Upgrade_Dev_s *pstCallback);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
char*	pszFilePath	Firmware path to be upgraded	IN
AS_CAM_Upgrade_Dev_s	pstCallback	Upgrade status callback function	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.6 Obtain Camera Internal And External Parameters

[Function]

Obtain the internal and external parameters of the specified camera

[Format]

int AS_SDK_GetCamParameter(AS_CAM_PTR pCamera, AS_CAM_Parameter_s *pstParam);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_CAM_Parameter_s	pstParam	Camera internal and external reference	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

To obtain the correct inner and outer parameters, it is necessary to call AS-SDK StartStream to start the stream.

4.2.7 Get Camera Firmware Mode

[Function]

Mode for obtaining camera firmware

[Format]

int AS_SDK_GetTofMode(AS_CAM_PTR pCamera, AS_TOFMODE_E &mode);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_TOFMODE_E	mode	Firmware mode	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Only supports KUNLUN series cameras.

4.2.8 Obtain Resolution Capability Set

[Function]

Obtain Resolution Capability Set

[Format]

int AS_SDK_GetCapability(AS_CAM_PTR pCamera, AS_MEDIA_TYPE_E type,std::vector<AS_STREAM_Param_s> &capability);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_MEDIA_TYPE_E	type	image type	IN
std::vector<AS_STREAM_Param_s>	capability	Camera image parameters	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Only supports hp60c and hp60cn.

4.2.9 Set Image Parameters

[Function]

Set image parameters

[Format]

int AS_SDK_SetStreamParam(AS_CAM_PTR pCamera, AS_MEDIA_TYPE_E type,AS_STREAM_Param_s *pstStreamParam);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_MEDIA_TYPE_E	type	Flow type	IN
AS_STREAM_Param_s	pstStreamParam	Camera image parameters	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

The supported resolution capability set can be obtained through the AS-SDK_GetCapability interface.

4.2.10 Obtain Image Parameters

[Function]

Obtain image parameters

[Format]

int AS_SDK_GetStreamParam(AS_CAM_PTR pCamera, AS_STREAM_Param_s *pstStreamParam);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_STREAM_Param_s	pstStreamParam	Camera image parameters	OUT

[Returned value]

Type	Description
int	0 0 successful Other abnormal values

[Note]

Only surpport kunlunA 和 kunlunC.

4.2.11 Get Camera Model

[Function]

Get camera model

[Format]

int AS_SDK_GetCameraModel(AS_CAM_PTR pCamera, AS_SDK_CAM_MODEL_E &model);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_SDK_CAM_MODEL_E	model	Camera model	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.12 Set Camera Network Information

[Function]

Set camera network information

[Format]

int AS_SDK_SetDevNetAttrs(AS_CAM_PTR pCamera, AS_NET_DEV_Attrs_s attrs);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_NET_DEV_Attrs_s	attrs	Including IP address, MAC address, and whether DHCP is enabled or not	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.13 Get Camera Network Information

[Function]

Get camera network information

[Format]

int AS_SDK_GetDevNetAttrs(AS_CAM_PTR pCamera, AS_NET_DEV_Attrs_s &attrs);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN

Type	Name	Description	IN/OUT
AS_NET_DEV_Attrs_s	attrs	Including IP address, MAC address, and whether DHCP is enabled or not	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

N/A

4.2.14 Set Timestamp Type

[Function]

Set timestamp type

[Format]

int AS_SDK_SetTimeStampType(AS_CAM_PTR pCamera, AS_TIME_STAMP_TYPE_E type);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_TIME_STAMP_TYPE_E	type	Timestamp type	IN

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Only supports Linux version hp60c

4.2.15 Obtain Camera Configuration Information

[Function]

Obtain camera configuration information

[Format]

int AS_SDK_GetConfigInfo(AS_CAM_PTR pCamera, AS_CONFIG_INFO_S &config_info);

[Parameters]

Type	Name	Description	IN/OUT
AS_CAM_PTR	pCamera	The camera handle is obtained by the AS_SDK_GetCameraList interface or the AS_CreateCamHandle interface	IN
AS_CONFIG_INFO_S	config_info	configuration information	OUT

[Returned value]

Type	Description
int	0 successful Other abnormal values

[Note]

Only supports RGB cameras, this interface is used to obtain whether RGB and depth are aligned, as well as the ordered/unordered state of point clouds.

5 Assistance

5.1 Cautions

The ROS package provided by this SDK does not need to be moved to the ROS workspace, and can be compiled and run directly according to the methods provided in section 2.2.

5.1.1 How To Run Linux Programs Without Root Privileges

In Linux systems, accessing devices requires root privileges. For programs that operate devices by operating device nodes in the/dev directory, one way is to modify the permissions of device nodes through the chmod command. However, this is only temporary and cannot be changed through chmod for programs that do not operate devices through the/dev device node. At this point, device permissions can be managed through Linux's udev and rules rules rules.

For Linux systems using the HP60C camera module, access to the HP60C camera can be achieved by running Linux programs with normal permissions by executing the create_udev_rules.sh script in our SDK package sample/Linux (linux_listener)/scripts directory.

5.1.2 How To Run Ros Nodes Without Root Privileges

In Linux systems, accessing devices requires root privileges. For programs that operate devices by operating device nodes in the/dev directory, one way is to modify the permissions of device nodes through the chmod command. However, this is only temporary and cannot be changed through chmod for programs that do not operate devices through the/dev device node. At this point, device permissions can be managed through Linux's udev and rules rules rules.

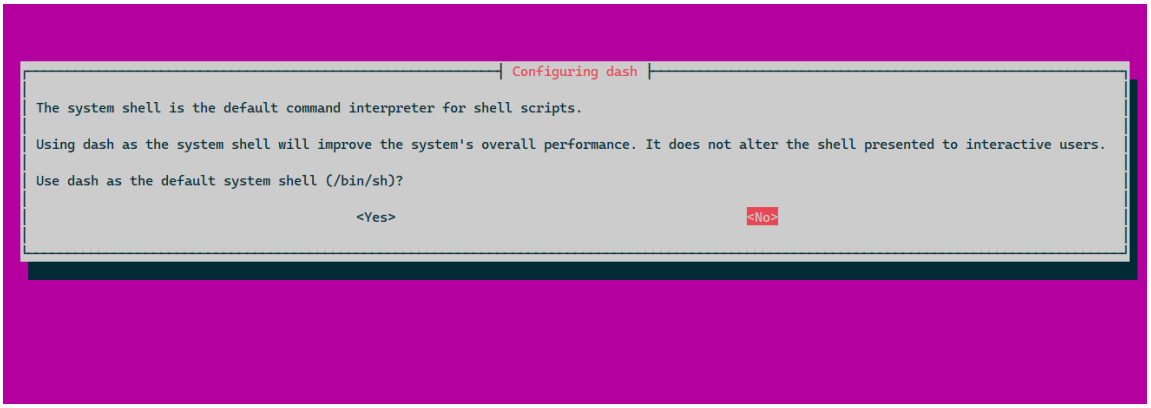
For Linux systems using the HP60C camera module, you can access the HP60C camera with normal permissions by executing the create_udev_rules.sh script in the sample/ros/src/ascamera (ascamera_listener)/scripts directory of our SDK package. Modify the run_camera_mode.sh script in the sample/ros/directory to annotate the statement that checks and applies for root permission. As shown in the following figure

```
CUR_DIR="$(dirname "$(realpath "${BASH_SOURCE[0]}")")"
# check for whitespace in $CUR_DIR and exit for safety reasons
grep -q "[[:space:]]" <<<"${CUR_DIR}" && { echo "\"${CUR_DIR}\" contains whitespace. Not supported. Aborting." >&2 ; exit 1 ; }

# if [ $EUID -ne 0 ]; then
#     echo -e "${RED}---This script requires root privileges, trying to use sudo${NORMAL}"
#     sudo "$CUR_DIR/run_ascamera_node.sh" "$@"
#     exit $?
# fi

if [ -f /opt/ros/melodic/setup.bash ]; then
    source /opt/ros/melodic/setup.bash
elif [ -f /opt/ros/kinetic/setup.bash ]; then
    source /opt/ros/kinetic/setup.bash
elif [ -f /opt/ros/noetic/setup.bash ]; then
    source /opt/ros/noetic/setup.bash
else
    echo "Error,Can't not found ros in /opt/"
fi

if [ "$(ps -aux | grep rosmaster | grep -v grep | wc -l)" -eq "0" ]; then
    roscore &
    sleep 3
fi
```

5.1.3 Executing Script Error Provided By SDK

Execute the script provided by the SDK with errors such as Bad substitution or syntax error: redirect unexpected. This is because the shell script provided by the SDK is a bash shell, while the default shell parser of the Ubuntu system is dash. Bash is more powerful than Dash, and some Bash syntax may not be parsed by Dash. You can change the default shell of ubuntu to bash by following these steps.

Execute sudo dpkg configure dash on the terminal and select [NO] in the pop-up window.



5.1.4 Compilation Error On Ubuntu 20.04 ROS

The following error was reported during ROS (Noetic) compilation in Ubuntu 20.04.

```
                 from /opt/ros/noetic/include/ros/time.h:58,
                 from /opt/ros/noetic/include/ros/ros.h:38,
                 from /home/boocax/roswork/src/as_nuwacam/src/as_nuwacam_node.cpp:29:
/usr/include/boost/mpl/aux_/preprocessed/gcc/minus.hpp:68:8: note:    'boost::mpl::minus'
   68 | struct minus
      |        ^~~~~
In file included from /usr/include/pcl-1.10/pcl/point_types.h:44,
                 from /usr/include/pcl-1.10/pcl/common/impl/copy_point.hpp:41,
                 from /usr/include/pcl-1.10/pcl/common/copy_point.h:58,
                 from /usr/include/pcl-1.10/pcl/common/impl/io.hpp:45,
                 from /usr/include/pcl-1.10/pcl/common/io.h:586,
                 from /usr/include/pcl-1.10/pcl/io/file_io.h:41,
                 from /usr/include/pcl-1.10/pcl/io/pcd_io.h:44,
                 from /opt/ros/noetic/include/pcl_conversions/pcl_conversions.h:70,
                 from /home/boocax/roswork/src/as_nuwacam/src/as_nuwacam_node.cpp:38:
/usr/include/pcl-1.10/pcl/point_types.h:698:1: error: 'minus' is not a member of 'pcl::traits'
  698 | POINT_CLOUD_REGISTER_POINT_STRUCT (pcl::_PointDEM,
      | ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
/usr/include/pcl-1.10/pcl/point_types.h:698:1: note: suggested alternatives:
In file included from /usr/include/c++/9/string:48,
                 from /usr/include/c++/9/bits/locale_classes.h:40,
                 from /usr/include/c++/9/bits/ios_base.h:41,
                 from /usr/include/c++/9/ios:42,
                 from /usr/include/c++/9/ostream:38,
                 from /usr/include/c++/9/iostream:39,
                 from /home/boocax/roswork/src/as_nuwacam/src/as_nuwacam_node.cpp:21:
/usr/include/c++/9/bits/stl_function.h:177:12: note:    'std::minus'
  177 |     struct minus : public binary_function<_Tp, _Tp, _Tp>
      |            ^~~~~
```

Due to the fact that our ROS node example program is developed based on the Melodic version of ROS on Ubuntu 18.04, when ported to ROS Noetic, the - std=c++11 in the ROS/src/ascamera/CMakeLists.txt file can be changed to - std==c++14.

sample > ros > src > ascamera > CMakeLists.txt

```
1  cmake_minimum_required(VERSION 3.0.2)
2  project(ascamera)
3
4  ## Compile as C++11, supported in ROS Kinetic and newer
5  add_compile_options(-std=c++11)
6  add_definitions(-std=c++11)
7  # get gcc -v target
```

5.1.5 Explanation Of Usb Device Matching Mechanism For Depth Cameras

Due to the HP60C camera being composed of 2 USB devices combined into 1 camera (1 USB communication device+1 UVC device), matching is required; When running on a virtual machine, USB devices may not pair properly due to incorrect device topology. Therefore, when using a virtual machine to run a program, only one HP60C camera can be connected, and more than one camera may cause matching failure due to the above reasons.