Shining 3D EXScan H SDK User Manual

Version 2.0

Shining 3D Tech Co., Ltd.

Revision History		
Version	Date	Revision Content
V2.0.0.0	20230522	First release.

Content

1. SDK Description

Development Related Files

SDK List

Item	Description	Note
Shining 3D EXScan H SDK User Manual 2.0	Software interfaces description manual.	
Include	Header files for ZMQ development; API header files helping analyze shared memory.	
Lib	Lib files For ZMQ development; API lib file helping analyze shared memory.	
Bin	Software execution package.	
ScannerDemo	SDK demo software source code.	
calibration_tool_h1.1.exe	Calibration tool for version 1.1.	
calibration_tool_h1.2.exe	Calibration tool for version 1.2.	

Operation Requirements

1, Use vs2017+Qt5.12.8
2, The output of ScannerDemo.exe after compilation is in the bin file (the demo install will copy it).
3, Copy zmq_sdk_config.ini to the working directory and modify isRemote=1 (demo install will copy it).
4, Copy the sn3d_smc.dll output in the bin file (the demo install will copy it).
5, Copy the sn3dE3H2SDK.dll output in the bin file (the demo install will copy it).

Main Software Components

List of Documents

Item	Function	Description
Bin\zmq_sdk_config.ini	SDK configuration file	isRemote: 0 does not support SDK access 1 Support SDK access Pub: public messaging port, Default value 11398 Rep: Request messaging port Default value 11399
Bin\scanservice.exe	Scan service	Scans and processes data.

ScannerDemo.exe	Scan client demo	Call scanner service to scan.
Bin\Sn2dH2SDK.dll	SDK link library	
qttunnel.3.0.9.dll tunnel_encrypt.dll	process communication	Enable communication between scanservice.exe and ScannerDemo.exe

Operation Requirements

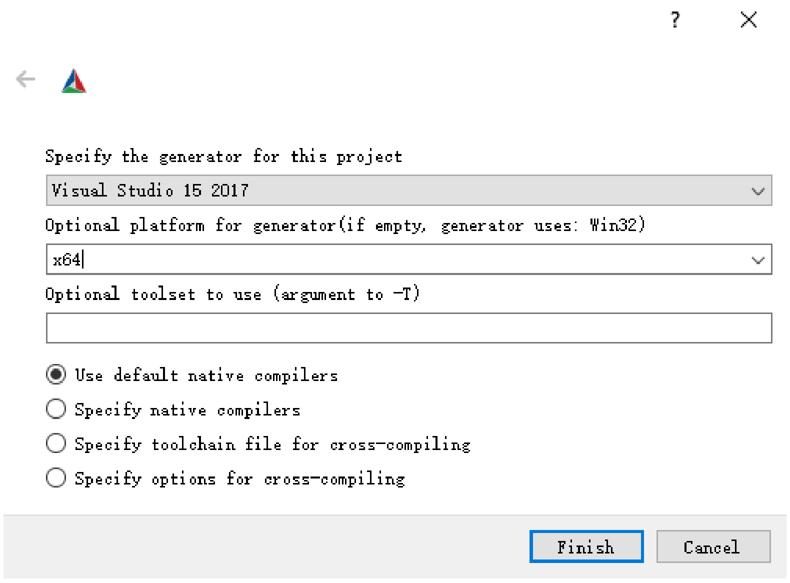
1 Modify interface in zmq_sdk_config.ini when communication port 11398 or 11399 is used by other software in your PC.

SDK Header File Description

File Name	Description	Note
Include\Sn3DE3H2Sdk.h	Contains sdk interfaces declaration	
Include\Sn3DErrorCode.h	Contains error codes information	
Include\Sn3DPublic.h	Custom structure included	
Include\PublicClass.h	Public definitions	

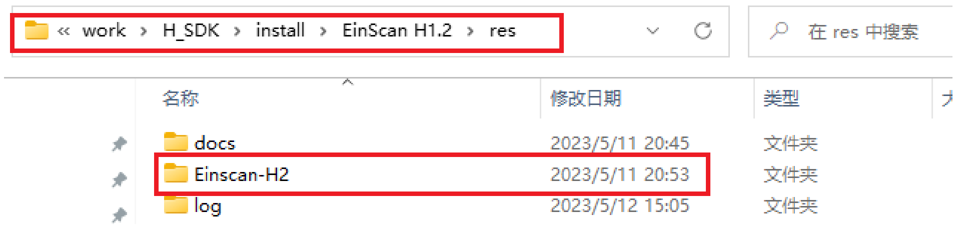
SDK Demo

1. Install CMake3.17.2 and above, install QT version 5.12.8;
2. Build solutions using cmake, Visual Studio 15 2017 + QT 5.12.8;


Specify the generator for this project
Visual Studio 15 2017
Optional platform for generator(if empty, generator uses: Win32)
x64
Optional toolset to use (argument to -T)

[✓ Use default native compilers]
Specify native compilers
Specify toolchain file for cross-compiling
Specify options for cross-compiling

3. vs 2017 builds solutions under release x64;
4. Copy ScannerDemo.exe, zmq_sdk_config.ini and all dynamic libraries under the folder bin to the root directory of the H1.1 or H1.2 release software (the default is C:/Program Files (x86)/EinScan H)

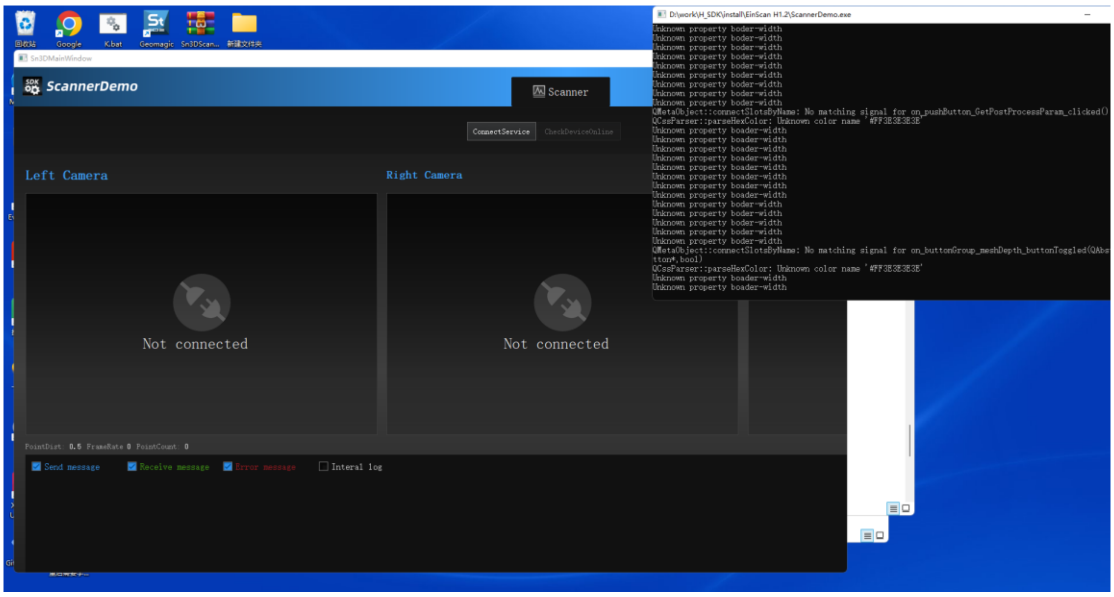


```
« work › H_SDK › install › EinScan H1.2 › res
```

名称	修改日期	类型
docs	2023/5/11 20:45	文件夹
Einscan-H2	2023/5/11 20:53	文件夹
log	2023/5/12 15:05	文件夹

a)

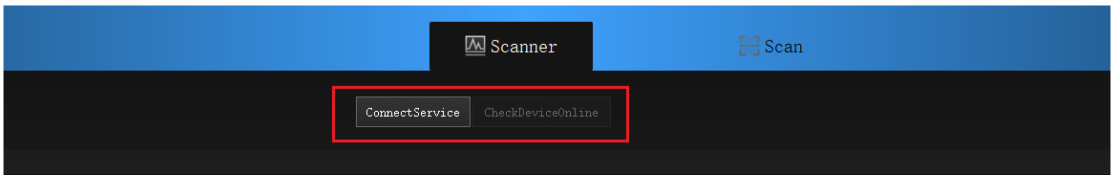
5. Execute scannerDemo.exe under the root directory of the release software.



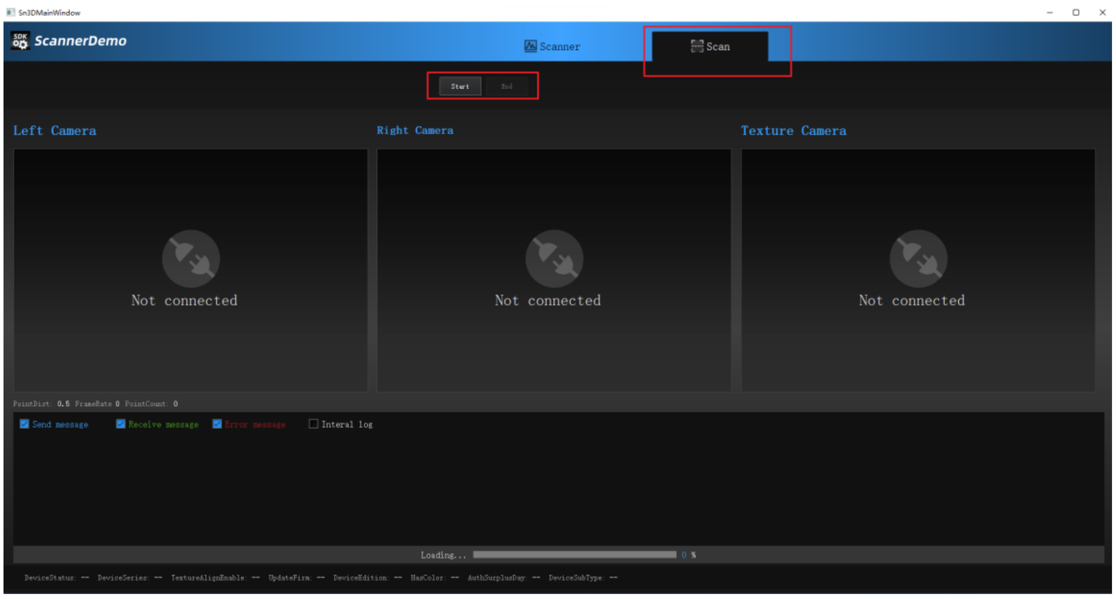
```
D:\work\H_SDK\install\EinScan H1.2\ScannerDemo.exe
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
QMetaObject::connectSlotsByName: No matching signal for on_pushButton_GetPostProcessParam_clicked()
QCssParser::parseHexColor: Unknown color name '#FF3E3E3E'
Unknown property boader-width
Unknown property boader-width
Unknown property boader-width
Unknown property boader-width
Unknown property boader-width
Unknown property boader-width
Unknown property boader-width
Unknown property boader-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
Unknown property boder-width
QMetaObject::connectSlotsByName: No matching signal for on_buttonGroup_meshDepth_buttonToggled(QAbs
tton*,bool)
QCssParser::parseHexColor: Unknown color name '#FF3E3E3E'
Unknown property boader-width
Unknown property boader-width
```

a)

6. Execute connectService (sdk initialization) and connectdevice (connection device) in turn.

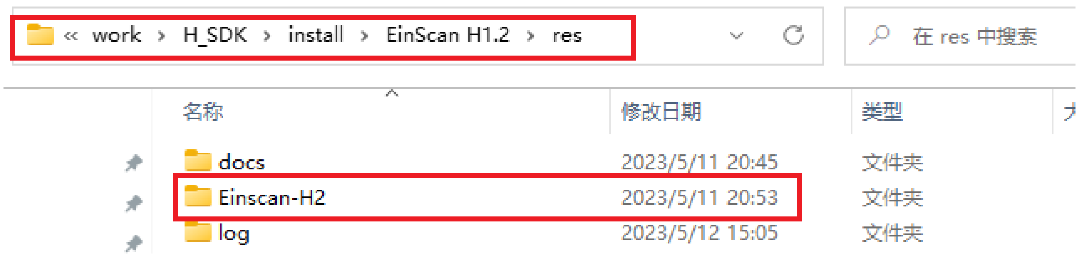


7. Go to the scanner tab page, click start (start scan) and end (generate data).



Calibration Tool

1. Copy calibration_tool_h1.1.exe to the H1.1 release software directory, and copy calibration_tool_h1.2.exe to the H1.2 release software directory
2. Copy the correct ple file to the release directory (./res/Einscan-H or ./res/Einscan-H2).



« work › H_SDK › install › EinScan H1.2 › res

在 res 中搜索

名称	修改日期	类型
docs	2023/5/11 20:45	文件夹
Einscan-H2	2023/5/11 20:53	文件夹
log	2023/5/12 15:05	文件夹

3. Connect the device, start it directly, and calibrate according to the prompt;
4. After the calibration is completed, a calibration file will be generated in the directory res/Einscan-H/200x150(H) or res/Einscan-H2/200x150(H2);

Note:

When starting the calibration program, make sure that the ple file exists and the device is connected normally, otherwise it cannot be calibrated normally.

SDK Known Issues

1. Garbled characters will appear in the new project under the Chinese path.

2. Data Structures and Macro Definitions

Data Structures

Image Data

SN3D_IMAGE_DATA

```
typedef struct tag SN3D_IMAGE_DATA
{
        int                 width;
        int                 height ;
        int                 channel ;
        int                 length ;
        unsigned char*      data;
} SN3D_IMAGE_DATA, *LPSN3D_IMAGE_DATA;
```

Members

width

The width of the image;

height

The height of the image;

channel

The number of image channels;

length

image data length = width*height;

data

Image data.

Remarks

Point Data

SN3D_POINT_DATA

```
typedef struct
{
        float                   x;                          //X coordinate value
        float                   y;                          //Y coordinate value
        float                   z;                          //Z coordinate value
}Sn3dPointData, *LPSn3dPointData;
```

Members

x

X coordinate.

y

Y coordinate.

z

Z coordinate.

Remarks

Point Cloud

SN3D_CLOUD_POINT

```
typedef struct
{
        int                     vertex_count;           //The number of vertex .
        LPSn3dPointData         vertex_data;            //The data of vertex.
        int                     norma_count;            //The number of vertex normal.
        LPSn3dPointData         norma_data;             //The data of vertex normal.
        int                     vertex_color_count; //The number of vertex color.
        LPSn3dPointData         vertex_color_data;  //The data of vertex color.
}Sn3DPointCloud,*LPSn3dPointCloud;
```

Members

vertex_count

Number of vertex data.

vertex_data

Vertex data.

norma_count

Number of vertex normal data.

norma_data

Vertex normal data.

vertex_ color _count

The number of vertex color data.

vertex_color_data

Vertex color data.

Remarks

Increase Point Cloud Data

SN3D_INCREASE_POINT_CLOUD

```
typedef struct
{
        int                    vertex_count;              //The number of vertex .
        LPSn3dPointData        vertex_data;               //The data of vertex.
        int                    norma_count;               //The number of vertex normal.
        LPSn3dPointData        norma_data;                //The data of vertex normal.
        int                    vertex_color_count;        //The number of vertex color.
        LPSn3dPointData        vertex_color_data;         //The data of vertex color.
        int                    index_count;               //The number of index
        unsigned int*          index;                     //The data of index
}Sn3DIncreasePointCloud,*LPSn3dIncreasePointCloud;
```

Members

vertex_count

Number of vertex data.

vertex_data

Vertex data.

norma_count

Number of vertex normal data.

norma_data

Vertex normal data.

vertex_ color _count

The number of vertex color data.

vertex_color_data

Vertex color data.

index_count

Index count.

index

Index.

Remarks

Camera Position

SN3D_CAMERA_POSITION

```
typedef struct
{
        QVector3D       position;
        QVector3D       center;
        QVector3D       up;
}Sn3DCameraPosition,*LPSn3dCameraPosition;
```

Members

position

The position of the observer(camera viewpoint).

center

The position of the observed(the center of view).

up

The vector of the top of view.

Remarks

Face and Texture Index

SN3D_FACE_ID

```
typedef struct
{
        int x;
        int y;
        int z;
}Sn3DFaceId, *LPSn3dFaceId;
```

Members

x, y, z: face index.

Remarks

Texture UV Coordinate

SN3D_VEC2F

```
typedef struct
{
        float x;
        float y;
}Sn3DVec2F, *LPSn3dVec2F;
```

Members

x, y: the coordinate of the face.

Remarks

Texture Image

SN3D_IMAGE

```
typedef struct
{
	int width;
	int height;
	int channel;
	uchar* data;
	//The format defaults to RGB
}Sn3DImage;
```

Members

width: The width of the texture image.

height: The height of the texture image.

channel: The number of the texture image channels.

data: Texture image data.

Remarks

Mesh Data

SN3D_MESHDATA

```
typedef struct
{
        //The number of point clouds and point cloud data
        int                     meshpoint_count;
        LPSn3dPointData     meshpoint;
        // The number of normal and normal data
        int                     meshnormal_count;
        LPSn3dPointData     meshnormal;
        //Point cloud index
        int                     meshtrifaceid_count;
        LPSn3dFaceId        meshtrifaceid;
        //Texture index
        int                     meshtextureid_count;
        LPSn3dId            meshtextureid;
        //Texture UV coordinate
        int                     textureUV_count;
        LPSn3dVec2F         textureUV;
        //Texture imgae
        Sn3DImage           image;
}Sn3DMeshData, *LPSn3dMeshData
```

Members

meshpoint_coun: the number of point cloud.

meshpoint: point cloud data.

meshnormal_count: the number of normal.

meshnormal: normal data of the point cloud.

meshtrifaceid_count: the number of point cloud index IDs.

meshtrifaceid: point cloud index ID data.

meshtextureid_count: the number of texture index IDs.

meshtextureid: texture index ID data.

textureUV_count: The number of texture image UV coordinates.

textureUV: texture image UV coordinates.

image: texture image.

Remarks

Return Value Definition

Macro definition	Value	Description
EC_SUCCESS	0	No error
EC_NOTINITIALIZED	1	No initialization
EC_INITIALIZEFIAILZED	2	Initialization failed.
EC_AlREADYINITIALIZED	3	Initialization repeat.
EC_SAVEFAILED_TYPEERROR	4	Format error, failed to save files.
EC_CHECKDEVICEFAILED_PLENOTRIGHT	5	PLE error, failed to connect device.
EC_CHECKDEVICEFAILED_NODEVICEFOUND	6	No device found, connection failed.
EC_OPENORCREATSLNFAILED	7	Failed to create or open a project.
EC_ENTERSCANFAILED	8	Failed to enter the scan mode.
EC_SCANFAILED	9	Failed to scan.
EC_ENDSCANFAILED	10	Failed to generate.
EC_MESHFAILED	11	Failed to mesh.
EC_EXITSCANFAILED	12	Failed to exit scanning.
EC_CANCELSCANFAILED	13	Failed to cancel scanning.
EC_SAVEFAILED	14	Failed to save projects.
EC_CREATNEWPROJECTFAILED	15	Failed to create a new project.

Alignment Type

Type definition	Value	Description
AT_FEATURES	0	Feature alignment
AT_MARKERS	1	Markers alignment
AT_HYBRID	2	Hybrid alignment
AT_GLOBLE_POINT	7	Global markers alignment
AT_TEXTURE	8	Texture alignment

Scanning Type

Type definition	Value	Description
ST_E3_H_NORMAL	5	Infrared ray
ST_E10_NORMAL	10	White light

Device Event Type

Type definition	Value	Description
DE_NULL	-1	null
DE_DOUBLECLICK	0	Double click the start button
DE_CLICK	1	Click the start button
DE_UP_CLICK	4	Click the up button
DE_DOWN_CLICK	5	Click the down button
DE_RIGHT_CLICK	6	Click the right button
DE_LEFT_CLICK	7	Click the left button

Callback Function Declaration

Initialize the Callback

Sn3DScanServiceWatcherCallBack

Initialize the device setup callback.

```
typedef    void (CALLBACK *Sn3DScanServiceWatcherCallBack)(int);
```

Remarks

Register the server monitors the callback and the server is notified of the exit.

Point Cloud Callback

Sn3DWholePointCloudCallBack

Get whole point cloud callback.

```
typedef          void(CALLBACK          *Sn3DWholePointCloudCallBack)(LPSn3dPointCloud
wholePointCloud);
```

Parameter

wholePointCloud

[out] point cloud

Remarks

1. Get the point cloud set at different stages by setting sn3dEndscan (end scan) and sn3dMesh (meshing) callback.
2. The callback needs to be reset when the callback is successfully triggered.

Current Point Cloud Callback

Sn3DCurrentPointCloudCallBack

Get the current point cloud callback.

```
typedef void(CALLBACK *Sn3DCurrentPointCloudCallBack)(LPSn3dPointCloud
currentPointCloud, void* owner);
```

Parameter

currentPointCloud

[out] current point cloud

owner

[out] The Point to the owner of the callback function

Remarks

1.Get the current point cloud set throughout the whole scanning process by setting Sn3DCurrentPointCloudCallBack after Sn3DInitialize.

Increase Point Cloud Callback

Sn3DIncreasePointCloudCallBack

Get the increase point cloud callback.

```
typedef void(CALLBACK *Sn3DIncreasePointCloudCallBack)(LPSn3dIncreasePointCloud
increasePointCloud, void* owner);
```

Parameter

increasePointCloud

[out] increase point cloud

owner

[out] The Point to the owner of the callback function

Remarks

1. Get the increase point cloud set throughout the whole scanning process by setting Sn3DIncreasePointCloudCallBack after Sn3DInitialize.

Camera Position Callback

Sn3DCameraPositionCallBack

Get the camera position callback.

typedef void(CALLBACK *Sn3DCameraPositionCallBack)(LPSn3dCameraPosition cameraPosition, void* owner);

Parameter

cameraPosition

[out] camera position

owner

[out] The Point to the owner of the callback function

Remarks

1. Get the camera position set throughout the whole scanning process by setting Sn3DCameraPositionCallBack after Sn3DInitialize.

Track Lost Status Callback

Sn3DTrackLostStatusCallBack

Get the track lost status callback.

typedef void(CALLBACK *Sn3DTrackLostStatusCallBack)(bool trackLostStatus, void* owner);

Parameter

trackLostStatus

[out] track lost status (true: lost; false: not lost).

owner

[out] The Point to the owner of the callback function

Remarks

1. Get the track lost status throughout the whole scanning process by setting Sn3DTrackLostStatusCallBack after Sn3DInitialize.

Scanning Distance Callback

Sn3DScanDistCallBack

Get the scanning distance callback.

```
typedef void(CALLBACK *Sn3DScanDistCallBack)(double scanDist, void* owner);
```

Parameter

scanDist

[out] scan distance

owner

[out] The Point to the owner of the callback function

Remarks

1. Get the scanning distance throughout the whole scanning process by setting Sn3DScanDistCallBack after Sn3DInitialize.

2. When scanDist = -1, it means the distance is too short; when scanDist = 100, it means the distance is too far; when scanDist = -2, it means the distance is invalid (point cloud may fail to be reconstructed); any other value (1 ~ 12) means the distance is normal.

Mesh Data Callback

Sn3DMeshDataCallBack

Get the mesh data callback.

```
typedef void(CALLBACK *Sn3DMeshDataCallBack)(LPSn3dMeshData meshData, void* owner);
```

Parameter

meshData

[out] mesh data

owner

[out] The Point to the owner of the callback function

Remarks

1. It needs to be called after the Sn3DEndScan

Device Event Callback

Sn3DDeviceEventCallBack

Get the device event callback.

```
typedef void(CALLBACK *Sn3DDeviceEventCallBack)(DeviceEvent event, void* owner);
```

Parameter

event

[out] device event type

owner

[out] The Point to the owner of the callback function

Remarks

1.It needs to set the callback after Sn3DInitialize (initialize).

Point Count Callback

Sn3DPointCountCallBack

Get the point cloud count callback.

```
typedef void(CALLBACK *Sn3DPointCountCallBack)(int pointCount, void* owner);
```

Parameter

pointCount

[out] point cloud count

owner

[out] The Point to the owner of the callback function

Remarks

1. It needs to set the callback after Sn3DInitialize (initialize).

Frame Rate Callback

Sn3DFrameRateCallBack

Get the frame rate callback.

```
typedef void(CALLBACK *Sn3DFrameRateCallBack)(int frameRate, void* owner);
```

Parameter

frameRate

[out] frame rate

owner

[out] The Point to the owner of the callback function

Remarks

1. It needs to set the callback after Sn3DInitialize (initialize).

Frame Callback

Sn3DFrameCountCallBack

Get the frame callback.

```
typedef void(CALLBACK *Sn3DFrameCountCallBack)(int frameCount, void* owner);
```

Parameter

frameCount

[out] frame

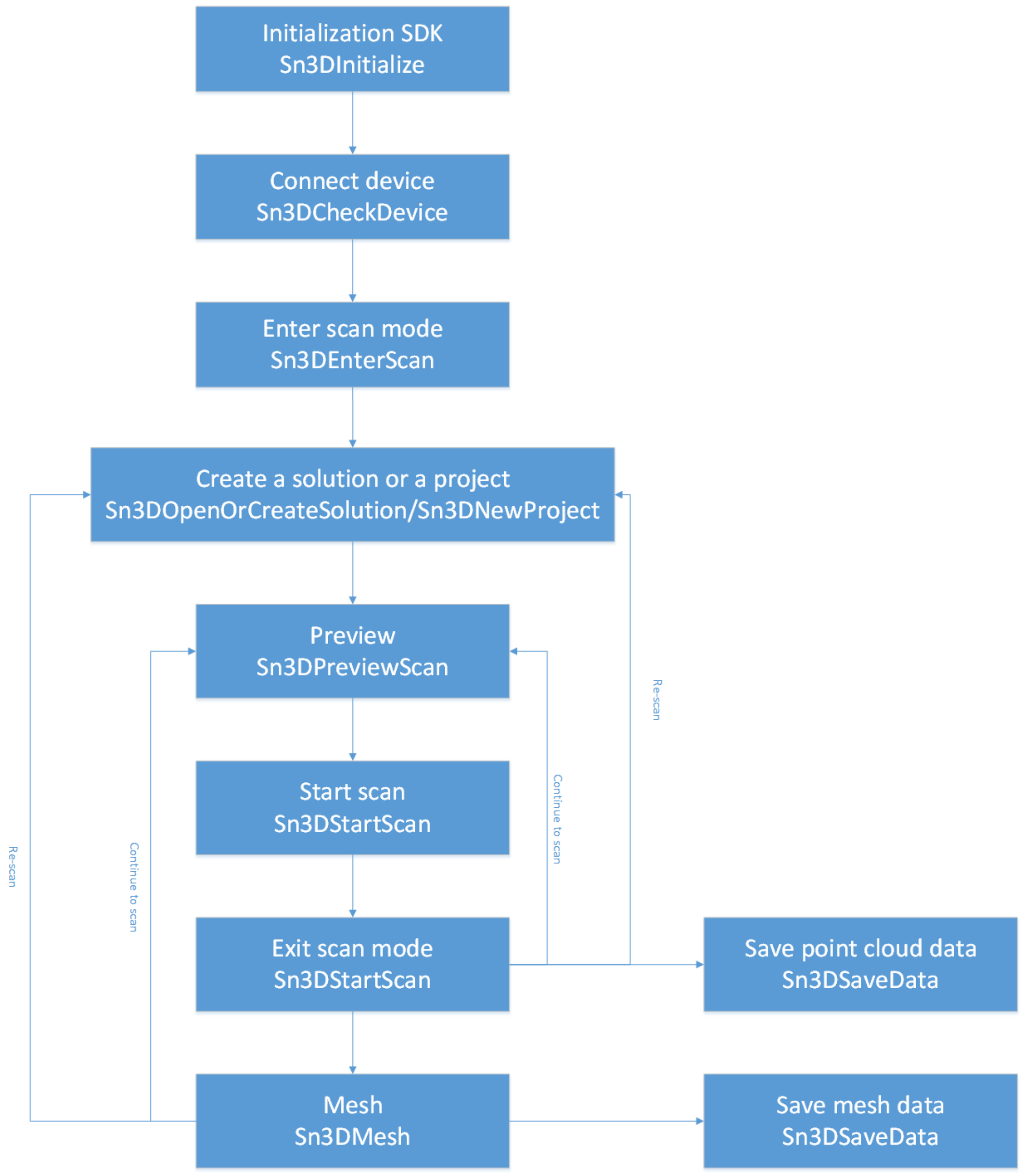
owner

[out] The Point to the owner of the callback function

Remarks

1. It needs to set the callback after Sn3DInitialize (initialize).

3. Workflow


Initialization SDK
Sn3DInitialize
Connect device
Sn3DCheckDevice
Enter scan mode
Sn3DEnterScan
Create a solution or a project
Sn3DOpenOrCreateSolution/Sn3DNewProject
Preview
Sn3DPreviewScan
Start scan
Sn3DStartScan
Exit scan mode
Sn3DStartScan
Mesh
Sn3DMesh
Save point cloud data
Sn3DSaveData
Save mesh data
Sn3DSaveData
Re-scan
Continue to scan
Continue to scan
Re-scan

4. API Definition

Base Function API

Sn3DInitialize

SDK Initialization

```
void* Sn3DInitialize(
        Sn3DScanServiceWatctherCallBack watcher
);
```

Parameters

Sn3DScanServiceWatctherCallBack

[in] Registration service monitors the callback and receives notice when the server ended.

Return Values

For macro definition, refer to return values.

Remarks

1 Just initialize SDK once.

2 After initializing SDK, if it is not in use, it must be released by calling Sn3DRelease.

SDK release

Sn3DRelease

SDK release

```
int Sn3DRelease();
```

Return Values

For macro definition, refer to return values.

Remarks

1 The function is to release the SDK resource.

Set up the SDK output log.

Sn3DInitialLog

```
int Sn3DInitialLog (
char* exeName,
char* path
);
```

Parameters

exeName

[in] Call the name of SDK proceeding.

path

[in] Set up the directory of log in.

Return Values

Refer to the error codes.

Save data

Sn3DSaveData

```
int Sn3DSaveData (
char* absolutePrjName,
char*savePath,
char*saveType
);
```

Parameters

absolutePrjName

[in] The complete path of project files.

savePath

[in] The path and the name of the file required to be saved (without a suffix).

saveType

[in] The format of the file required to be saved (.asc, .ply, .stl or .obj).

Return Values

Refer to the error codes.

Device Control API

Connect the device

Sn3DCheckDevice

int Sn3DConnectDevice ()

Return Values

For macro definition, refer to return values.

Reconnect the device interface

Sn3DReConnectDevice

int Sn3DReConnectDevice()

Return Values

For macro definition, refer to return values.

Remarks

This function can only be used after initializing the device.

Device status notification interface

Sn3DGetDeviceIsOnline

int Sn3DGetDeviceIsOnline**()**

Return Values

For macro definition, refer to return values.

Remarks

This function can only be used after initializing the device.

Get brightness range

Sn3DGetBrightnessRange

```
SN3DSDKE10API int Sn3DGetBrightnessRange(int& min, int& max);
```

Parameters

min

[out] Reserved, the minimum brightness level is set to 0 (default). By default, this function is disable.

max

[out] The maximum brightness of the camera.

Return Values

Refer to the error codes.

Remarks

It needs to be used after creating a solution interface.

Set brightness

Sn3DSetBrightness

```
int Sn3DSetBrightness (
int    brightness
)
```

Parameters

brightness

[in] The brightness level of the camera. The value range is determined by the value returned by Sn3DGetBrightnessRange. The larger the value, the brighter it is.

Return Values

Refer to the error codes.

Remarks

It needs to be used after creating a solution interface.

Get the device brightness

Sn3DGetCurrentBrightness

```
int Sn3DGetCurrentBrightness (
int&    brightness
)
```

Parameters

brightness

[out]   The brightness level of the camera. The value range is determined by the value returned by Sn3DGetBrightnessRange. The larger the value, the brighter it is.

Return Values

Refer to the error codes.

Remarks

It needs to be used after creating a solution interface.

Set the LED brightness interface

Sn3DSetCurrentLEDDutyCycle

```
int Sn3DSetCurrentLEDDutyCycle(
int    ledDutyCycle
)
```

Parameters

ledDutyCycle

[in]   Camera LED brightness value. Value range [0, 100].

Return Values

Refer to the error codes.

Remarks

Get the LED brightness interface

Sn3DGetCurrentLEDDutyCycle

```
int Sn3DGetCurrentLEDDutyCycle(
int&    ledDutyCycle
)
```

Parameters

ledDutyCycle

[out]   Camera LED brightness value. Value range [0, 100].

Return Values

Refer to the error codes.

Remarks

Get video stream

Sn3DIntallGetImagasCallBack

```
int Sn3DIntallGetImagasCallBack (
        Sn3DGetImagesCallBack imageCallback,
        void* owner,
);
```

Parameters

imageCallback

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Set the scanning distance callback

Sn3DSetScanDistCallBack

```
int Sn3DSetScanDistCallBack(
        Sn3DScanDistCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Set the device event callback

Sn3DSetDeviceEventCallBack

```
int Sn3DSetDeviceEventCallBack(
        Sn3DDeviceEventCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Project Management API

Enter scan

Sn3DEnterScan

```
SN3DSDKE10API int CALLMETHOD Sn3DEnterScan(ScanType scanType);
```

Parameters

Scanning mode: white light or infrared ray.

Return Values

Error code.　For macro definition, refer to return values.

Remarks

1 This function can only be called after detecting that the device is online.

2 Perform the function of initializing scanning.

3 Different suffixes correspond to different scan type projects.

Create new solutions

Sn3DOpenOrCreateSolution

```
SN3DSDKE10API int Sn3DOpenOrCreateSolution(
    const char* slnDirPath,
    bool isCreate,
    int scanMode,
    bool hasTexture,
    float pointDis);
```

Parameters

slnDirPath

[in] Solution path

isCreate

[in] “true”- new, “false”-open

scanMode

[in] scan mode, 0 portrait, 1 object, 2 smallobject. Reference: ScanMode

hasTexture

[in] true texture scan, false scan without texture

pointDis

[in] point distance

Return Values

Error code. For macro definition, refer to return values.

Remarks

1 It needs to be called after Sn3DEnterScan.

Create new projects

Sn3DNewProject

```
SN3DSDKE10API int CALLMETHOD Sn3DNewProject(
        const char* slnDirPath,
        int scanMode,
        bool hasTexture,
        float pointDis,
        int alignType,
        const char* globalMarkerPath);
```

Parameters

slnDirPath

[in] Solution path

scanMode

[in] scan mode, 0 portrait, 1 object, 2 smallobject. Reference: ScanMode

hasTexture

[in] true, texture scan; false, scan without texture.

pointDis

[in] point distance

alignType

[in] alignment mode. Reference: For Align Types, see appendix

globalMarkerPath

[in] Frame point path. Non-frame point scan mode is set to null.

Return Values

Error code. For macro definition, refer to return values.

Remarks

1 It needs to be called after Sn3DOpenOrCreateSolution.

Open a project

Sn3DOpenProject

```
SN3DSDKE10API int CALLMETHOD Sn3DOpenProject(
        const char* projFileName);
```

Parameters

projFileName

[in] Project path

Return Values

Error code. For macro definition, refer to return values.

Remarks

1 It needs to be called after Sn3DOpenOrCreateSolution.

Scanning API

Preview

Sn3DPreviewScan

```
SN3DSDKE10API int CALLMETHOD Sn3DPreviewScan();
```

Parameters

Return Values

Error code. For macro definition, refer to return values.

Remarks

1. Data can be reconstructed by previewing and the data cannot be integrated into the overall point cloud.
2. Call this function after creating or opening a project.

Scanning

Sn3DStartScan

```
SN3DSDKE10API int CALLMETHOD Sn3DStartScan();
```

Parameters

Return Values

Error code. For macro definition, refer to return values.

Remarks

1. Call this function after Sn3DPreviewScan.
2. Reconstruction, tracking, fusion to the overall point cloud

Pause scanning

Sn3DPauseScan

```
SN3DSDKE10API int CALLMETHOD Sn3DPauseScan ();
```

Parameters

Return Values

Error code. For macro definition, refer to return values.

Remarks

1. Call this function after Sn3DStartScan.

End scanning

Sn3DEndScan

```
SN3DSDKE10API int CALLMETHOD Sn3DEndScan(
    bool globalOptimize,
    double pointDist,
    Sn3DWholePointCloudCallBack callback = nullptr);
```

Parameters

globalOptimize

[in]“true”-optimize; “false” -not optimized

pointDist

[in] point distance setting.

callback

[in] Set up the callback to get the point cloud data after finishing scanning.

Return Values

Error code. For macro definition, refer to return values.

Remarks

1. Call this function after Sn3DStartScan or Sn3DPauseScan.

2. Generate the overall point cloud data.
3. Get the point cloud data generated at this stage by setting this callback. There are examples in the demo.

Empty the scan data

Sn3DCancelScan

```
SN3DSDKE10API int    Sn3DCancelScan(
        bool isCancelCurrentProjectFramerMark);
```

Parameters

isCancelCurrentProjectFramerMark

[in] “true”- Empty the global markers information.

Return Values

Error code.   For macro definition, refer to return values.

Remarks

1. Empty the currently scanned point cloud data.

Quit scanning

Sn3DExitScan

```
SN3DSDKE10API int CALLMETHOD Sn3DExitScan();
```

Parameters

Return Values

Error code.   For macro definition, refer to return values.

Remarks

1. Quit scanning.

Enable pseudo color function

Sn3DSetEnablePseudoColor

```
int Sn3DSetEnablePseudoColor(
        bool enable
);
```

Parameters

enable

[in]] “true”-open; “false”-off.

Return Values

Refer to the error codes.

Current frame point cloud callback

Sn3DSetCurrentPointCloudCallBack

```
int Sn3DSetCurrentPointCloudCallBack (
        Sn3DCurrentPointCloudCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Set increase point cloud callback

Sn3DSetIncreasePointCloudCallBack

```
int Sn3DSetIncreasePointCloudCallBack(
        Sn3DIncreasePointCloudCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Set camera position callback

Sn3DSetCameraPositionCallBack

```
int Sn3DSetCameraPositionCallBack(
        Sn3DCameraPositionCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Set track lost status callback

Sn3DSetTrackLostStatusCallBack

```
int Sn3DSetTrackLostStatusCallBack(
        Sn3DTrackLostStatusCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Get point count callback

Sn3DGetPointCountCallBack

```
int Sn3DGetPointCountCallBack(
        Sn3DPointCountCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Get frame rate callback

Sn3DGetFrameRateCallBack

```
int Sn3DGetFrameRateCallBack(
        Sn3DFrameRateCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

Get frames callback

Sn3DGetFrameCountCallBack

```
int Sn3DGetFrameCountCallBack(
        Sn3DFrameCountCallBack callBackFunc,
        void* owner,
);
```

Parameters

CallbackFunc

[in] Set the callback function.

owner

[in] The Point to the owner of the callback function.

The Mesh Function API

Generate mesh data

Sn3DMesh

```
SN3DSDKE10API int CALLMETHOD Sn3DMesh(
        int meshType,
        int filterLevel,
        int SmoothLevel,
        float PointDis,
        bool fillSmallHole,
        double smellHolePerimeter,
        bool fillMarkerHole,
        double spikeSensitivity,
        int faceLimit,
        Sn3DWholePointCloudCallBack callback = nullptr);
```

Parameters

meshType.

[in] Reference: MeshType

filterLevel.

[in] 0-3 level, different degree of the filter.

SmoothLevel.

[in] 1-3 level. Set up smooth degree in filter.

PointDis.

[in]point distance

fillSmallHole.

[in]“true”-Fill the small hole

smellHolePerimeter.

[in] Fill the circumference of the small hole

fillMarkerHole.

[in] “true”-filling markers.

spikeSensitivity.

[in] Remove the spikes

faceLimit.

[in] Maximum number of faces

callback.

[in] Set up the callback for getting meshed point cloud data.

Return Values

Refer to the error codes.

Remarks

1. Call this function after Sn3DEndScan.
2. Meshed data.
3. Set up the callback for getting the meshed overall point cloud set.

Generate mesh data

Sn3DMeshEx

```
SN3DSDKE10API int CALLMETHOD Sn3DMesh(
        int meshType,
        int filterLevel,
        int SmoothLevel,
        float PointDis,
        bool fillSmallHole,
        double smellHolePerimeter,
        bool fillMarkerHole,
        double spikeSensitivity,
        int faceLimit,
       void* owner
        Sn3DMeshDataCallBack callback = nullptr);
```

Parameters

meshType.

[in] Reference: MeshType

filterLevel.

[in] 0-3 level, different degree of the filter.

SmoothLevel.

[in] 1-3 level. Set up smooth degree in the filter.

PointDis.

[in]point distance

fillSmallHole.

[in] “true”-Fill the small hole

smellHolePerimeter.

[in] Fill the circumference of the small hole

fillMarkerHole.

[in]”true”-filling markers.

spikeSensitivity.

[in] Remove the spikes

faceLimit.

[in] Maximum number of faces

owner

[in] The Point to the owner of the callback function.

callback.

[in] Set up the callback for getting meshed point cloud data.

Return Values

Refer to the error codes.

Remarks

1. Call this function after Sn3DEndScan.
2. Meshed data.
3. Set up the callback for getting the meshed overall point cloud set.