# HPS-3D160 Solid-state Depth camera

# SDK manual

# Contents

# 1. SDK introduction

The SDK provides the application interface of the HPS3D160 Solid-State Depth camera, which is currently available on the Linux platform, the Windows platform, the ROS platform, and most of the microcontrollers that do not run the operating system; Please read the user manual carefully before using the SDK;

# 2. Integrate SDK into IDE

In order to facilitate cross-platform and cross-language use of the SDK, the API interface defined by basic data types is currently provided. Refer to HPS3DBase_IF.h. Currently, this API has a secondary package in C/C++ and C#, which is convenient for users to integrate into the project. Languages such as Java and Python will be further expanded according to demand in the future;

## 2.1 SDK used in C/C++

In order to facilitate users to integrate into the project, we re-encapsulate the basic API, among which HPS3DBase_IF.h is the basic API interface, HPS3DUser_IF.c and HPS3DUser_IF.h are secondary encapsulation; the basic steps are as follows:

1. Copy HPS3DBase_IF.h, HPS3DUser_IF.C, HPS3DUser_IF.h to the project and include the HPS3DBase_IF.h, HPS3DUser_IF.C, HPS3DUser_IF.h three files in the project;

2、Define global parameters and initialize, Example:

```
int g_handle = -1;
static HPS3D_MeasureData_t g_measureData;

HPS3D_StatusTypeDef ret = HPS3D_RET_ERROR;
ret = HPS3D_MeasureDataInit(&g_measureData);
if (ret != HPS3D_RET_OK)
{
    printf("MeasureDataInit failed,Err:%d\n", ret);
}
```

Note: For the convenience of memory control, a dynamic memory allocation method is adopted here. When the program exits, memory release needs to be executed, and the HPS3D_MeasureDataFree interface is called;

3. Select the connection method according to the device model. Example:

```
HPS3D_HandleTypeDef handle;
HPS3D_StatusTypeDef ret = HPS3D_RET_ERROR;
ret = HPS3D_USBConnectDevice((char *)"/dev/ttyACM0",&g_handle); //USB Connect
//ret = HPS3D_EthernetConnectDevice((char *)"192.168.0.10", 12345, &g_handle);
if (ret != HPS3D_RET_OK)
{
```

```
    printf("connect failed,Err:%d\n", ret);
}
```

Note: When connecting a USB device under linux, you need to manually modify the device permissions by executing chmod 777 /dev/ttyACM0;

When connecting a USB device under Windows, enter the port number. In order to solve the encoding problem, it is recommended to add _T before the port number. If the port number is above COM10, you need to add "\\\\.\\" before the port number , Such as "\\\\.\\ COM15";

```
ret = HPS3D_USBConnectDevice((char *)_T("COM3"), &g_handle);
ret = HPS3D_USBConnectDevice((char *)_T("\\\\.\\COM15"), &g_handle);
```

4. Register a callback function for event notification. Example:

```
void EventCallBackFunc(int handle, int eventType, uint8_t *data, int dataLen, void *userPara)
{
    switch ((HPS3D_EventType_t)eventType)
    {
        case HPS3D_SIMPLE_ROI_EVEN:
        case HPS3D_FULL_ROI_EVEN:
        case HPS3D_FULL_DEPTH_EVEN:
        case HPS3D_SIMPLE_DEPTH_EVEN:
            printf("Measure date!");
            HPS3D_ConvertToMeasureData(data,&g_measureData,eventType);
            break;
        case HPS3D_SYS_EXCEPTION_EVEN:
            printf("SYS ERR :%s\n", data);
            break;
        case HPS3D_DISCONNECT_EVEN:
            printf("Device disconnected!\n");
            HPS3D_CloseDevice(handle);
            break;
        case HPS3D_NULL_EVEN:
        default:
            break;
    }
}

HPS3D_StatusTypeDef ret = HPS3D_RET_ERROR;
ret = HPS3D_RegisterEventCallback(EventCallBackFunc, NULL);
if (ret != HPS3D_RET_OK)
{
    printf("RegisterEventCallback failed,Err:%d\n", ret);
}
```

5、Start capture， Example: ：

```
/*Continuous capture*/
HPS3D_StartCapture(g_handle);

/*single capture*/
HPS3D_EventType_t type = HPS3D_NULL_EVEN;
ret = HPS3D_SingleCapture(g_handle, &type, &g_measureData);
if (ret != HPS3D_RET_OK)
{
      printf("SingleCapture failed,Err:%d\n", ret);
}
```

**Note：**

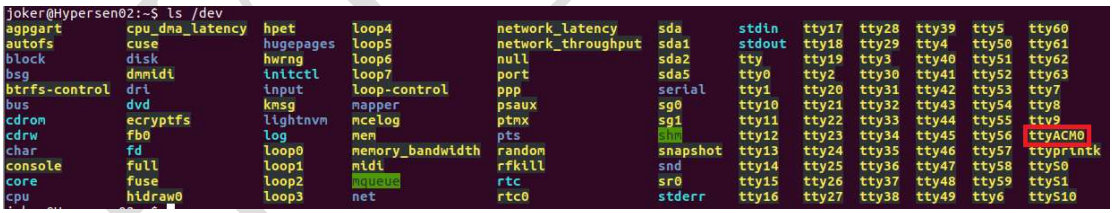
- **The continuous mode measurement result is obtained in the callback function. The callback function is used for data notification. It is not recommended to perform more time-consuming operations in the callback function, otherwise the acquisition frame rate may be affected;**
- **After calling HPS3D_StartCapture, only executing HPS3D_StopCapture is valid, and all other interfaces return error values;**
- **The measurement result in single mode is returned immediately, and the return value needs to be checked to determine whether the acquisition is successful;**

## 2.1.1 SDK used in Linux

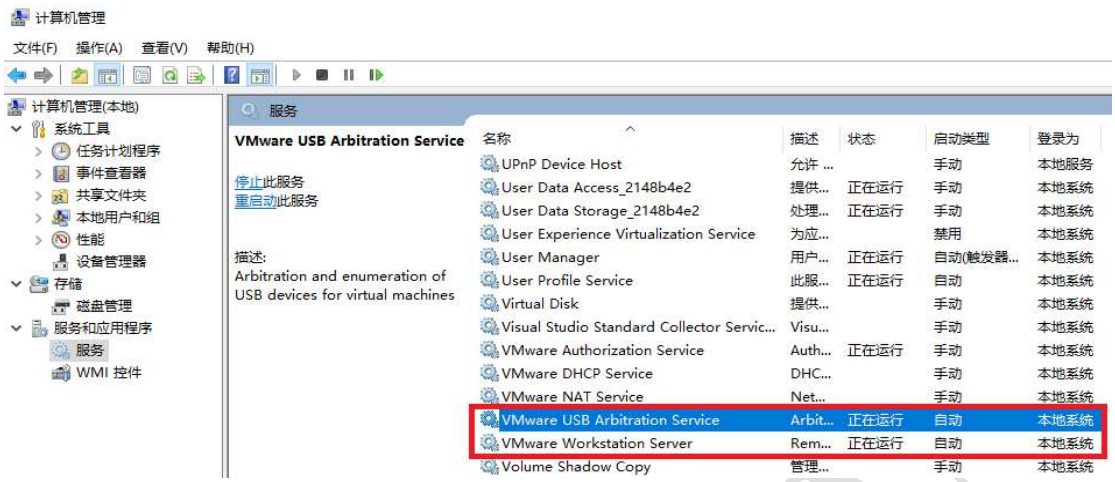
xxx.so is suitable to use on Linux operation system, take Ubuntu as example. This example is based on the SDK with API version number V1.8.0.

- Device connection

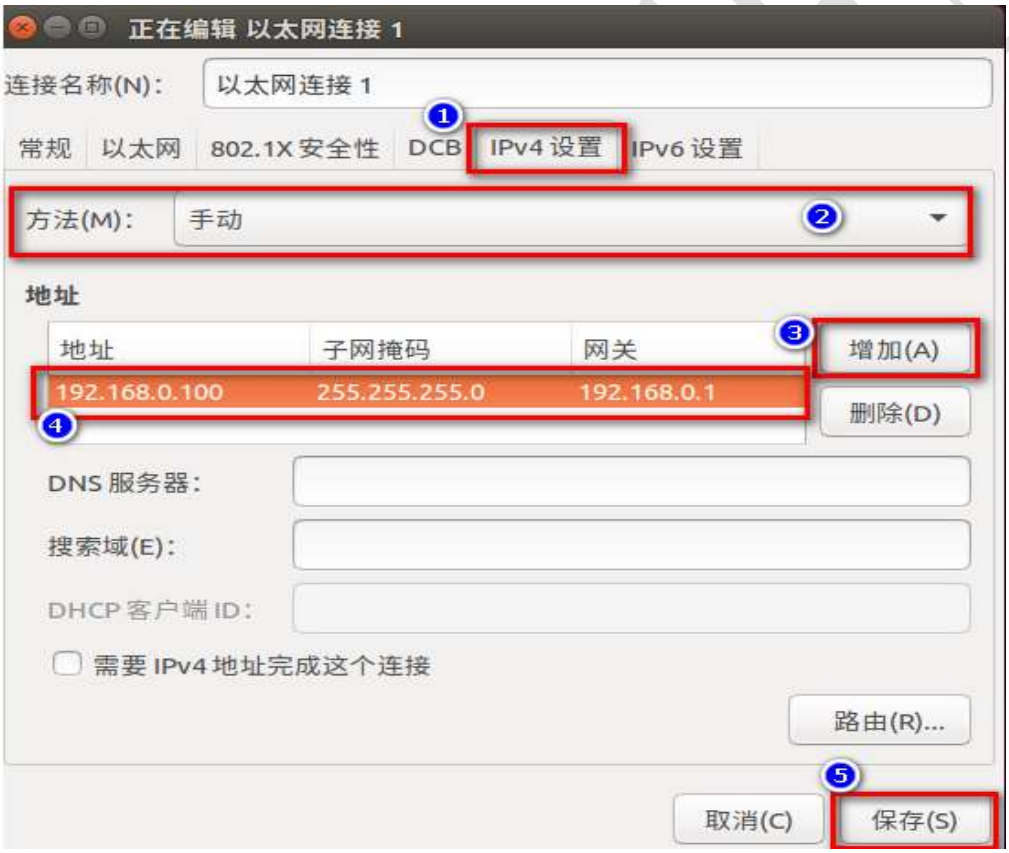
Connect the HPS3D160 device to your computer, open the terminal and type ls /dev to view the device ttyACM*, as shown below:




If you do not see the ttyACM* device name, you need to re-plug and view it again. If not, go to "Computer Management -> Services and Applications -> Services" to see if the VMware USB Arbitration Service is running. If disabled, then turn on the operation, then re-plug the device; if you don't want to start the USB device service every time you log in to the virtual machine, you can set the VMware Workstation Server and VMware USB Arbitration Service to run and auto, restart the computer.




The connection steps of the network device are: connect the device to ububtu, and configure the Ethernet connection IPV4 address, taking the default IP address of the sensor 192.168.0.10 as an example, the configuration steps are shown in the following figure:




- Use SDK under Ububtu, run Demo program

  1、 Copy the HPS3D160-Linux-C Demo program to any directory in ubuntu, open the terminal and enter make to compile the demo program. The execution result is as follows, the compilation is passed;

```
kevin@kevin-virtual-machine:~/HPS3D160-SDK/demo/HPS3D160-Linux-C_Demo$ make
gcc -Wall -O -g -c HPS3DUser_IF.c -o HPS3DUser_IF.o
g++ HPS3DUser_IF.o main.o -o ./app -Wl,-rpath=./ -L./ -lHPS3DSDK
chmod a+x ./app
kevin@kevin-virtual-machine:~/HPS3D160-SDK/demo/HPS3D160-Linux-C_Demo$
```

2、 Enter ./app in the terminal to connect to the device and start measurement. After the connection is successful, the current device parameters are automatically exported as shown below. According to the prompt, enter 1 for single measurement, and enter 2 for continuous measurement;

```
kevin@kevin-virtual-machine:~/HPS3D160-SDK/demo/HPS3D160-Linux-C_Demo$ ./app
HPS3D160 C/C++ Demo (Linux)

SDK Ver:V1.8.0 21-6-10
Dev Ver:V1.8.0 21-6-10
SN:SN:HL21M03173D1901663

resolution:160 X 60
max_roi_group_number:16  cur_group_id: 0
max_roi_number:8
max_multiCamera_code:15, cur_multiCamera_code:0
user_id: 0
optical_path_calibration: 0

select capture mode: SingleCapture(1)  ContinuousCapture(2)  Exit(...)
```

3、 Different types of sensor need to modify the connection method, the modification method is shown in the figure below

```
ret = HPS3D_USBConnectDevice((char *)"/dev/ttyACM0",&g_handle); //USB Connect

//ret = HPS3D_EthernetConnectDevice((char *)"192.168.0.10", 12345, &g_handle);
if (ret != HPS3D_RET_OK)
{
        printf("connect failed,Err:%d\n", ret);
        break;
}
printf("Dev Ver:%s\n", HPS3D_GetDeviceVersion(g_handle));
printf("SN:%s\n\n", HPS3D_GetSerialNumber(g_handle));

HPS3D_RegisterEventCallbackFunc(EventCallBackFunc, NULL);
```

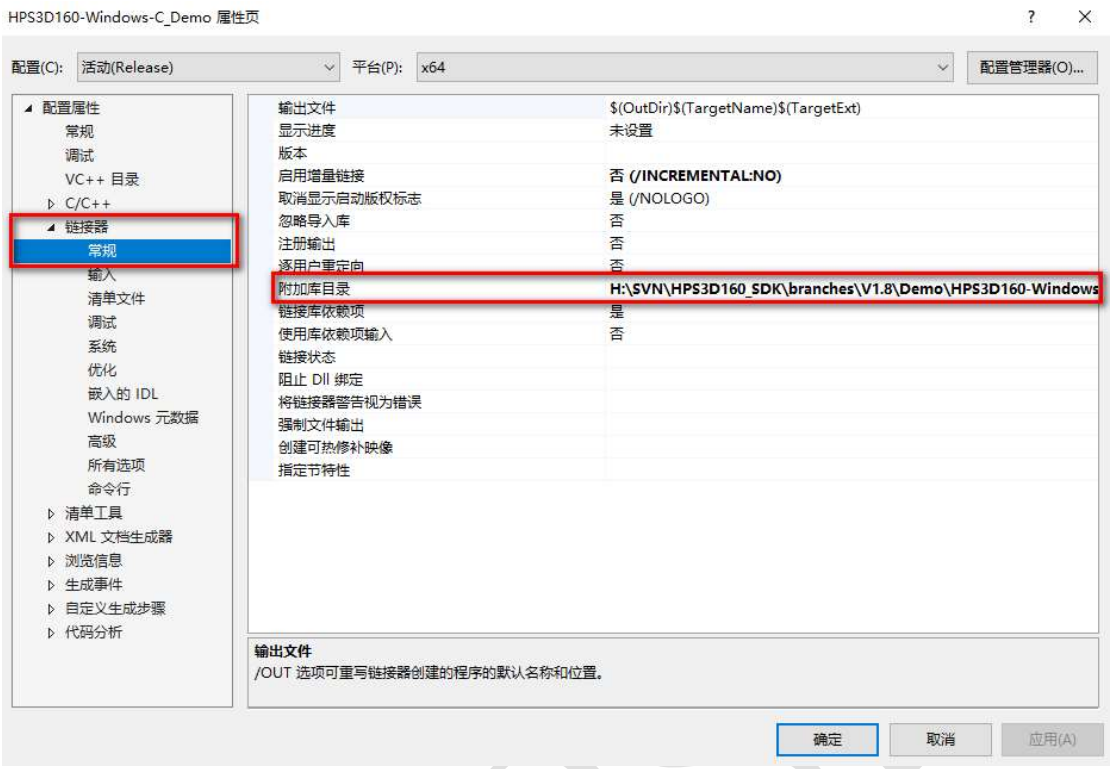
Note: The USB device connection failed. The reason may be that the device is No permission. You can execute the command sudo chmod 777 /dev/ttyACM0 and try again;

## 2.1.2 SDK used in Windows

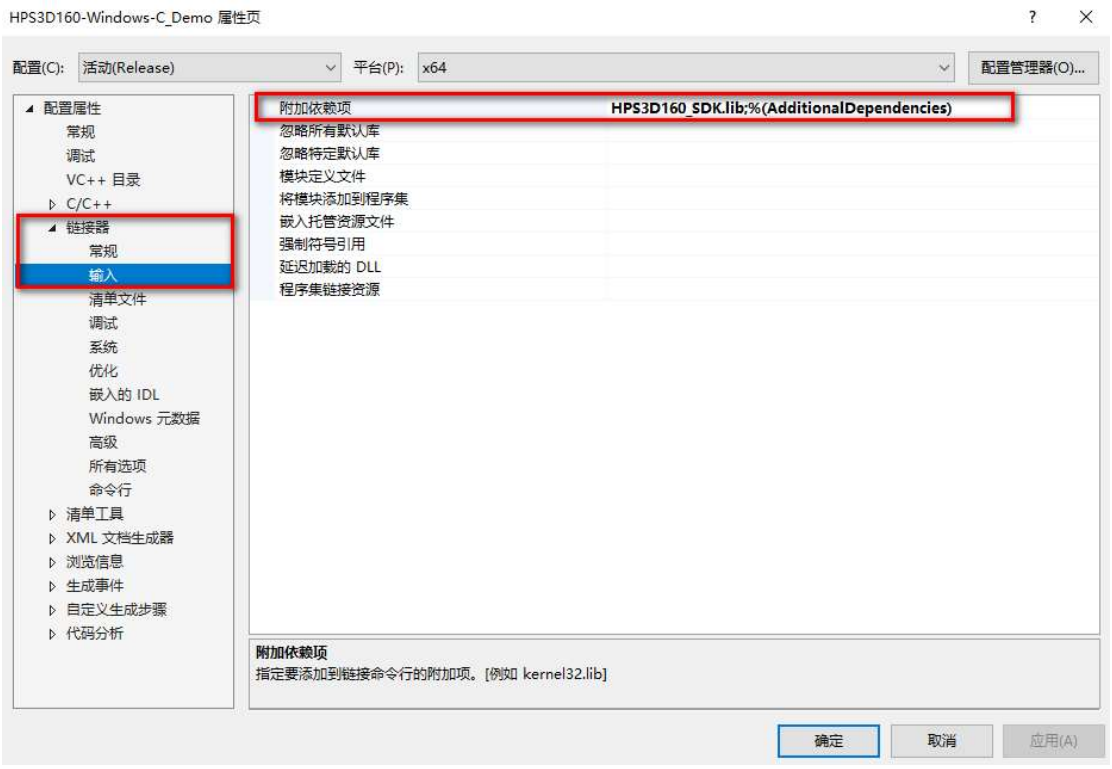
Xxx.dll is suitable for use on the Windows operating system platform， Here is an example of Microsoft Visual Studio 2017。This example is written based on the SDK with API version number V1.8.0。

1、 Open the HPS3D160-Windows-C_Demo project and check the project configuration as follows：

Right-click the item-Properties-Linker-General. Specify the path of HPS3D160_SDK.dll in the additional library directory.




Right-click the item-Properties-Linker-Input. Fill in HPS3D160_SDK.lib in the additional dependency option.




Copy HPS3D160_SDK.dll to the program running directory, you can run the Demo program normally;

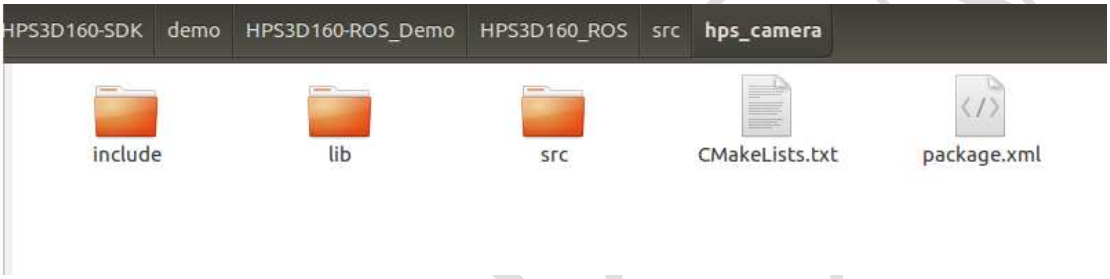
Note: due to encoding problems when USB devices are may faild. You can add _T before the port number to solve this problem. If the port number is above COM10, you need to add "\\\\.\\" before the port number, such as \\\\.\\COM10;

```
ret = HPS3D_USBConnectDevice((char *)_T("COM3"), &g_handle);
ret = HPS3D_USBConnectDevice((char *)_T("\\\\.\\COM10"), &g_handle);
```

## 2.1.3 SDK used in ROS

Here is an example of Ubuntu 16.04.6 LTS and kinetic。This example is written based on the SDK with API version number V1.8.0。

Refer to Demo example: enter the source code directory HPS3D160-ROS_Demo/HPS3D160_ROS/src/hps_camera.




- Copy HPS3DBase_IF and HPS3DUser_IF in the include directory to the include directory of the project directory
- Copy libHPS3DSDK.so in the lib directory to the lib directory in the project directory.
- Copy HPS3DUser_IF.c in the src directory to the src directory in the project directory, and refer to hps_camera.cpp to reference the relevant code to the project;
- Modify the content of the CMakeLists.txt file as follows:

  Add related dependency libraries

```
## Find catkin macros and libraries
## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  roscpp
  rospy
  std_msgs
  message_generation
  std_srvs
  sensor_msgs
  cv_bridge
  image_transport
  pcl_conversions
  pcl_ros
)
```

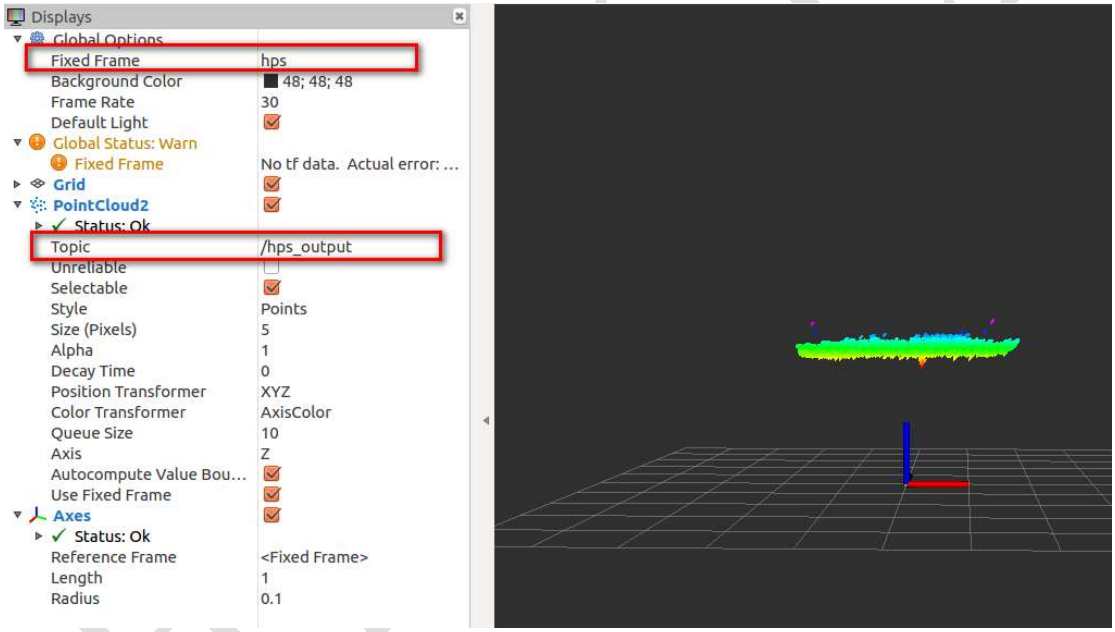
Specify include header file and library path

```
## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
 include
  ${catkin_INCLUDE_DIRS}
)
link_directories(
 lib
  ${catkin_LIB_DIRS}
)
```

Add the dependent library name HPS3DSDK, which must be consistent with the library name in the lib directory.

```
# add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_E
find_package(PCL REQUIRED)
include_directories(include${PCL_INCLUDE_DIRS})
add_executable(hps_camera src/hps_camera.cpp)
target_link_libraries(hps_camera ${catkin_LIBRARIES}  ${PCL_LIBRARIES} HPS3DSDK)
```

After the modification is completed, you can compile and run; the point cloud data can be displayed in real time through RVIZ, as shown below:

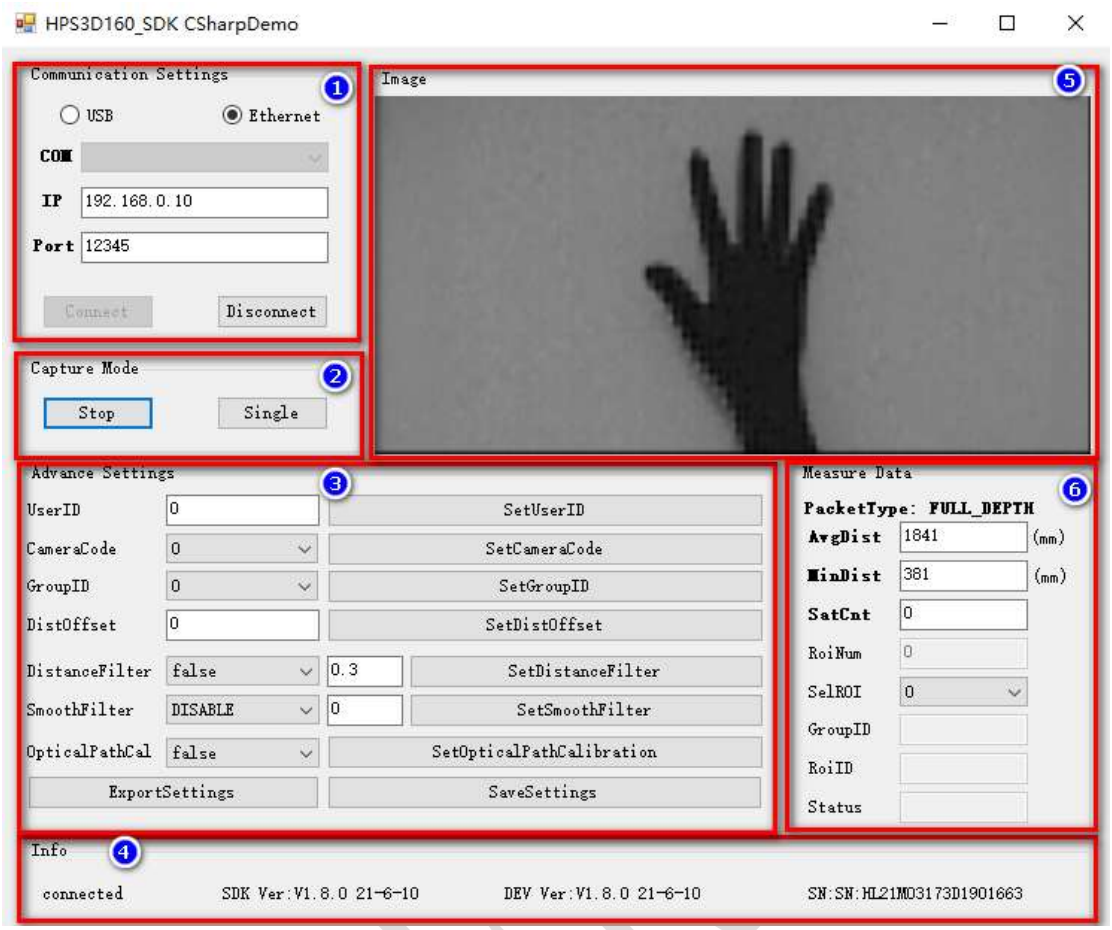



## 2.2 SDK Used in C#

In order to facilitate users to integrate into the project, we re-encapsulate the basic API. In the C# environment, we encapsulate the HPS3D160_Device class, which greatly facilitates the secondary development of the HPS3D160 device in the C# environment;

Here is an example of Microsoft Visual Studio 2017, based on .NET Framework 4.6.1。This example is written based on the SDK with API version number V1.8.0。

- Open the HPS3D160_CSharpDemo project and check whether the software running directory contains HPS3D160_SDK.dll；
- Build and run the project, if there is no error, you can connect normally and perform measurement；




Integrate HPS3D160_SDK in the C# project, just copy the HPS3D160_Device device class and HPS3D160_SDK.dll dynamic library in the Demo program to the program running directory;

# 3. API function interface

## 3.1 USB Connection

### HPS3DAPI_USBConnectDevice

| | |
|---|---|
| **brief** | USB device connection |
| **define** | int HPS3DAPI_USBConnectDevice(__IN char* portName,__OUT int* deviceHandler); |
| **param** | portName : device name(eg:COM3 、/dev/ttyACM0)<br>deviceHandler: Return device ID |
| **retval** | Successfully returned 1 |
| **note** | When the port number is greater than 10, \\\\.\\ characters must be added before the port number, such as \\\\.\\COM15 |

Example:

```
int handle = -1;
int ret = 0;
ret = HPS3DAPI_USBConnectDevice ("COM3", &handle);
if (ret != 0)
{
    printf("USBConnectDevice failed,err:%d\r\n",ret);
    return;
}
```

## 3.2 Ethernet Connection

### HPS3DAPI_EthernetConnectDevice

| | |
|---|---|
| **brief** | Ethernet device connection |
| **define** | int HPS3DAPI_EthernetConnectDevice(__IN char* controllerIp, __IN uint16_t controllerPort, __OUT int* deviceHandler); |
| **param** | controllerIp : default 192.168.0.10<br>controllerPort : default 12345<br>deviceHandler : Return device ID |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
int handle = -1;
int ret = 0;
ret = HPS3DAPI_EthernetConnectDevice("192.168.0.10",12345, &handle);
if (ret != 0)
{
```

```
        printf("EthernetConnectDevice failed,err:%d\r\n",ret);
        return;
    }
```

## 3.3 Close device

### HPS3DAPI_CloseDevice

| | |
|---|---|
| **brief** | Close device |
| **define** | int HPS3DAPI_CloseDevice(__IN int handle); |
| **param** | handle : device ID |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
    int ret = 0;
    ret = HPS3DAPI_CloseDevice(handle);
    if (ret != 0)
    {
        printf("CloseDevice failed,err:%d\r\n",ret);
        return;
    }
```

## 3.4 Device is connected

### HPS3DAPI_IsConnect

| | |
|---|---|
| **brief** | Device is connected |
| **define** | int HPS3DAPI_IsConnect(__IN int handle); |
| **param** | handle : device ID |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
    int ret = 0;
    ret = HPS3DAPI_IsConnect(handle);
    if (ret != 1)
    {
        printf("Device is disconnected\r\n");
        return;
    }
```

## 3.5 Device is Start Capture

### HPS3DAPI_IsStart

| | |
|---|---|
| **brief** | device is start capture |
| **define** | int HPS3DAPI_IsStart(__IN int handle); |
| **param** | handle : device ID |
| **retval** | Continuous measurement mode returns 1 |
| **note** | |

Example:

```
int ret = 0;
ret = HPS3DAPI_IsStart(handle);
if (ret != 1)
{
    printf("Device is standby.\r\n");
    return;
}
```

## 3.6 Start continuous capture

### HPS3DAPI_StartCapture

| | |
|---|---|
| **brief** | Start continuous capture |
| **define** | int HPS3DAPI_StartCapture(__IN int handle); |
| **param** | handle : device ID |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
int ret = 0;
ret = HPS3DAPI_StartCapture(handle);
if (ret != 1)
{
    printf("StartCapture err:%d\r\n",ret);
    return;
}
```

## 3.7 Stop Capture

### HPS3DAPI_StopCapture

| | |
|---|---|
| **brief** | Stop capture |
| **define** | int HPS3DAPI_StopCapture(__IN int handle); |
| **param** | handle : device ID |
| **retval** | Successfully returned 1 |
| **note** | |

**Example:**

```
int ret = 0;
ret = HPS3DAPI_StopCapture(handle);
if (ret != 1)
{
    printf("StopCapture err:%d\r\n",ret);
    return;
}
```

## 3.8 Single Capture

### HPS3DAPI_SingleCapture

| | |
|---|---|
| **brief** | Single Capture |
| **define** | int HPS3DAPI_SingleCapture(__IN int handle, __OUT int *type, __OUT uint8_t **data, __OUT int *dataLen); |
| **param** | handle : device ID<br>type : Acquisition events (0: no event, 1: simple ROI data package 2: complete ROI data package 3: complete deep data package 4: simple deep data package 7: system abnormal event 8: abnormal disconnection event 5-6 reserved)<br>data : The returned data pointer data analysis method can refer to the demo program<br>dataLen: returned data length |
| **retval** | Successfully returned 1 |
| **note** | A single measurement must determine the retval value and type type at the same time. Only when the retval value is 1 and the type is a valid data packet, the acquisition is successful, otherwise the acquisition fails; |

**Example:**

```
int ret = 0;
uint8_t *ret_data = NULL;
int dataLen = 0;
int type = 0;
```

```
ret = HPS3DAPI_SingleCapture(handle, (int *)type, (uint8_t **)&ret_data,&dataLen);
if (ret != 1)
{
    printf("SingleCapture failed err:%d\r\n",ret);
     return;
}
switch (*type)
{
     case 1:
     case 2:
     case 3:
     case 4:
            printf("SingleCapture succeed\r\n");
            break;
     default:
            printf("SingleCapture failed type:%d\r\n",*type);
            break;
}
```

## 3.7 Register callback function

### HPS3DAPI_RegisterEventCallback

| | |
|---|---|
| **brief** | Register callback function |
| **define** | int HPS3DAPI_RegisterEventCallback(__IN HPS3DAPI_EVENT_CALLBACK eventHandle, __IN void *userPara); |
| **param** | HPS3DAPI_EVENT_CALLBACK : Function pointer<br>原型 : void(*HPS3DAPI_EVENT_CALLBACK)(int handle, int eventType, uint8_t *data,int dataLen, void *userPara);<br>Consistent with single capture<br>userPara : user param, can be empty |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
void EventCallBackFunc(int handle, int eventType, uint8_t *data,int dataLen, void *userPara)
{
  switch (eventType)
  {
      case 1:
      case 2:
      case 3:
```

```
        case 4:
                printf("Is measure data\r\n");
                break;
        default:
                printf("SYS ERR\r\n");
                break;
    }
}
int ret = 0;
ret = HPS3DAPI_RegisterEventCallback(EventCallBackFunc,NULL);
if (ret != 1)
{
    printf("RegisterEventCallback err:%d\r\n",ret);
    return;
}
```

## 3.9 Unregister callback function

### HPS3DAPI_UnregisterEventCallBack

| | |
|---|---|
| **brief** | unregister callback function |
| **define** | int HPS3DAPI_UnregisterEventCallBack(); |
| **param** | |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
int ret = 0;
ret = HPS3DAPI_UnregisterEventCallBack();
if (ret != 1)
{
    printf("UnregisterEventCallBack err:%d\r\n",ret);
    return;
}
```

## 3.10 Get device version information

| | |
|---|---|
| **brief** | Get device version information |
| **define** | const uint8_t* HPS3DAPI_GetDeviceVersion(__IN int handle); |
| **param** | handle: device ID |

| | |
|---|---|
| retval | Return device version information , eg: V1.8.0 21-6-11 |
| note | |

### HPS3DAPI_GetDeviceVersion

Example:

```
printf("Device Version:%s\n", HPS3DAPI_GetDeviceVersion(handle);
```

## 3.11 Get SDK version information

### HPS3DAPI_GetSDKVersion

| | |
|---|---|
| brief | Get SDK version information |
| define | const uint8_t* HPS3DAPI_GetSDKVersion(__IN int handle); |
| param | handle: device ID |
| retval | Return SDK version information , eg: V1.8.0 21-6-11 |
| note | |

Example:

```
printf("SDK Version:%s\n", HPS3DAPI_GetSDKVersion(handle);
```

## 3.12 Get device serial number

### HPS3DAPI_GetSerialNumber

| | |
|---|---|
| brief | Get device serial number |
| define | const uint8_t* HPS3DAPI_GetSerialNumber(__IN int handle); |
| param | handle: device ID |
| retval | Return device serial number , eg: SN:HU21M06083D2100162 |
| note | |

Example:

```
printf("%s\n", HPS3DAPI_GetSerialNumber(handle);
```

## 3.13 Export Settings

### HPS3DAPI_ExportSettings

| brief | ExportSettings |
|---|---|
| define | int HPS3DAPI_ExportSettings(__IN int handle, __OUT uint8_t *settings); |
| param | Handle : device ID<br>settings :<br>typedef struct<br>{<br>int user_id; /* default:0*/<br>int max_resolution_X; /* default:160*/<br>int max_resolution_Y; /* default:60*/<br>int max_roi_group_number; /* default:16*/<br>int max_roi_number; /* default:8*/<br>int max_threshold_number; /* default:3*/<br>int max_multiCamera_code; /* default:16*/<br><br>int dist_filter_enable; /* default:false*/<br>float dist_filter_K;<br>int smooth_filter_type;<br>int smooth_filter_args;<br>int cur_group_id;<br>int cur_multiCamera_code; /* default: 0*/<br>int dist_offset;<br>int optical_path_calibration;<br>}HPS3D_DeviceSettings_t; |
| retval | Successfully returned 1 |
| note | |

Example:

```
int ret = 0;
HPS3D_DeviceSettings_t settings;
ret = HPS3DAPI_ExportSettings(handle,(uint8_t *)&settings);
if (ret != 1)
{
    printf("ExportSettings failed, err:%d\r\n",ret);
    return;
}
```

## 3.14 Save Settings

| brief | Save Settings |
|---|---|
| define | int HPS3DAPI_SaveSettings(__IN int handle); |

| | |
|---|---|
| **param** | handle: device ID |
| **retval** | Successfully returned 1 |
| **note** | |

### HPS3DAPI_SaveSettings

**Example:**

```
int ret = 0;
ret =HPS3DAPI_SaveSettings(handle);
if(ret != 1)
{
    printf("SaveSettings failed, err:%d\r\n",ret);
    return;
}
```

## 3.15 Set Device ID

### HPS3DAPI_SetDeviceUserID

| | |
|---|---|
| **brief** | Set device ID |
| **define** | int HPS3DAPI_SetDeviceUserID (__IN int handle, __IN uint8_t userID); |
| **param** | handle: device ID<br>userID: 0-255 |
| **retval** | Successfully returned 1 |
| **note** | |

**Example:**

```
int ret = 0;
ret =HPS3DAPI_SetDeviceUserID (handle,1);
if(ret != 1)
{
    printf("SaveDeviceUserID failed, err:%d\r\n",ret);
    return;
}
```

## 3.16 Set ROI Group ID

| | |
|---|---|
| **brief** | Set Group ID |
| **define** | int HPS3DAPI_SetROIGroupID (__IN int handle, __IN uint8_t groupID); |

| | |
|---|---|
| **param** | handle: device ID<br>group ID: 0-15 |
| **retval** | Successfully returned 1 |
| **note** | |

### HPS3DAPI_SetROIGroupID

Example:

```
int ret = 0;
ret =HPS3DAPI_SetROIGroupID (handle,1);
if(ret != 1)
{
    printf("SaveROIGroupID failed, err:%d\r\n",ret);
    return;
}
```

## 3.17 Set Camera Code

### HPS3DAPI_SetMultiCameraCode

| | |
|---|---|
| **brief** | Set Camera code |
| **define** | int HPS3DAPI_SetMultiCameraCode (__IN int handle, __IN uint8_t CameraCode); |
| **param** | handle: device ID<br>CameraCode: 0-15 |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
int ret = 0;
ret =HPS3DAPI_SetMultiCameraCode (handle,1);
if(ret != 1)
{
    printf("SaveMultiCameraCode failed, err:%d\r\n",ret);
    return;
}
```

## 3.18 Set distance filter

### HPS3DAPI_SetDistanceFilterConf

| | |
|---|---|
| **brief** | Set distance filter |
| **define** | int HPS3DAPI_SetDistanceFilterConf(__IN int handle, __IN int enable, __IN float K); |
| **param** | handle: device ID<br>enable: 1 or 0<br>K : 0-1 |
| **retval** | Successfully returned 1 |
| **note** | |

**Example:**

```
int ret = 0;
ret = HPS3DAPI_SetDistanceFilterConf(handle,1,0.3);
if(ret != 1)
{
    printf("SetDistanceFilterConf failed, err:%d\r\n",ret);
    return;
}
```

## 3.19 Set Smooth filter

### HPS3DAPI_SetSmoothFilterConf

| | |
|---|---|
| **brief** | Set Smooth filter |
| **define** | int HPS3DAPI_SetSmoothFilterConf(__IN int handle, __IN int type, __IN int args); |
| **param** | handle: device ID<br>type: 0:disable 1:average filter 2: guass filter<br>args : Filter parameter reference value is 2, 3 |
| **retval** | Successfully returned 1 |
| **note** | |

**Example:**

```
int ret = 0;
ret = HPS3DAPI_SetSmoothFilterConf (handle,1,2);
if(ret != 1)
{
    printf("SetSmoothFilterConf failed, err:%d\r\n",ret);
```

```
        return;
    }
```

## 3.20 Set the distance offset

### HPS3DAPI_SetDistanceOffset

| | |
|---|---|
| **brief** | Set the distance offset |
| **define** | int HPS3DAPI_SetDistanceOffset(__IN int handle, __IN int16_t offset); |
| **param** | handle: device ID<br>offset: |
| **retval** | Successfully returned 1 |
| **note** | |

Example:

```
    int ret = 0;
    ret = HPS3DAPI_SetDistanceOffset(handle,-100);
    if(ret != 1)
    {
        printf("SetDistanceOffset failed, err:%d\r\n",ret);
        return;
    }
```

## 3.21 Set optical path compensation

### HPS3DAPI_SetOpticalPathCalibration

| | |
|---|---|
| **brief** | Set optical path compensation |
| **define** | int HPS3DAPI_SetOpticalPathCalibration(__IN int handle, __IN int enbale); |
| **param** | handle: device ID<br>enable:  1 or 0 |
| **retval** | Successfully returned 1 |
| **note** | Optical path compensation can convert oblique line distance to vertical distance |

Example:

```
int ret = 0;
ret = HPS3DAPI_SetOpticalPathCalibration(handle,1);
if(ret != 1)
{
    printf("SetOpticalPathCalibration failed, err:%d\r\n",ret);
    return;
}
```

## 3.22 Device reconnect

### HPS3DAPI_EthernetReconnectDevice

| | |
|---|---|
| **brief** | Set the device to reconnect |
| **define** | int HPS3DAPI_EthernetReconnectDevice(__IN int deviceHandler); |
| **param** | handle: device ID |
| **retval** | Successfully returned 1 |
| **note** | You need to wait about 10 seconds for the connection to be successfully reconnected |

Example:

```
int ret = 0;
ret = (HPS3D_StatusTypeDef)HPS3D_EthternetReconnection(handle);
if (ret == HPS3D_RET_OK)
{
    HPS3D_StartCapture(handle);
    printf("Reconnect success: Device %d\n", handle);
    return ret;
}
```

# 4. Revision history

| Date | Revision | Description |
|---|---|---|
| 2021/06/17 | V2.1 | V1.8 SDK |
| 2022/06/16 | V2.2 | V1.8.5 SDK |
| | | |

**IMPORTANT NOTICE – PLEASE READ CAREFULLY**

Hypersen Technologies Co., Ltd. reserve the right to make changes, corrections, enhancements, modifications, and improvements to Hypersen products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on Hypersen products before placing orders. Hypersen products are sold pursuant to Hypersen's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of Hypersen products and Hypersen assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by Hypersen herein.

Resale of Hypersen products with provisions different from the information set forth herein shall void any warranty granted by Hypersen for such product.

Hypersen and the Hypersen logo are trademarks of Hypersen. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.