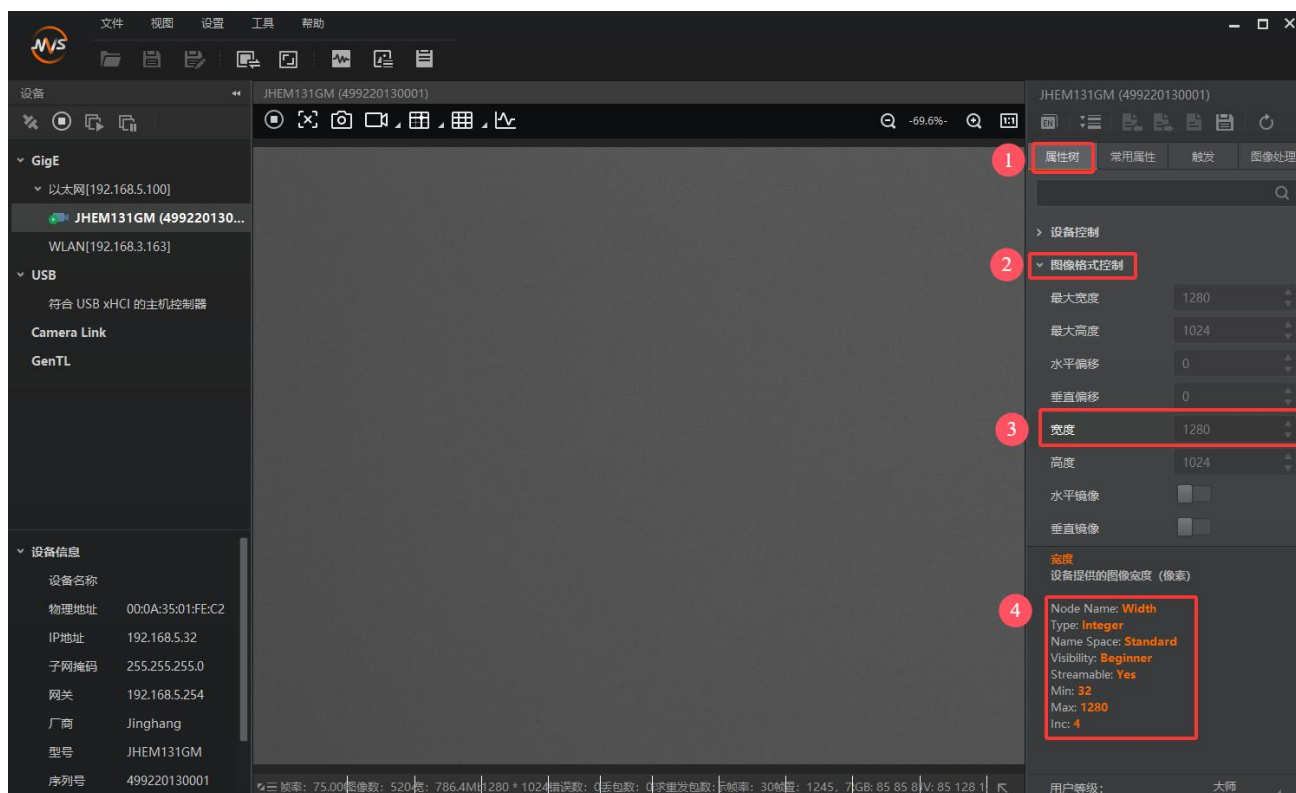


使用 SDK 设置相机参数

本手册描述 JHEM 系列网口和千兆网相机使用 SDK 设置参数的基本原则，是具体 API 调用方法。

1 相机参数规则

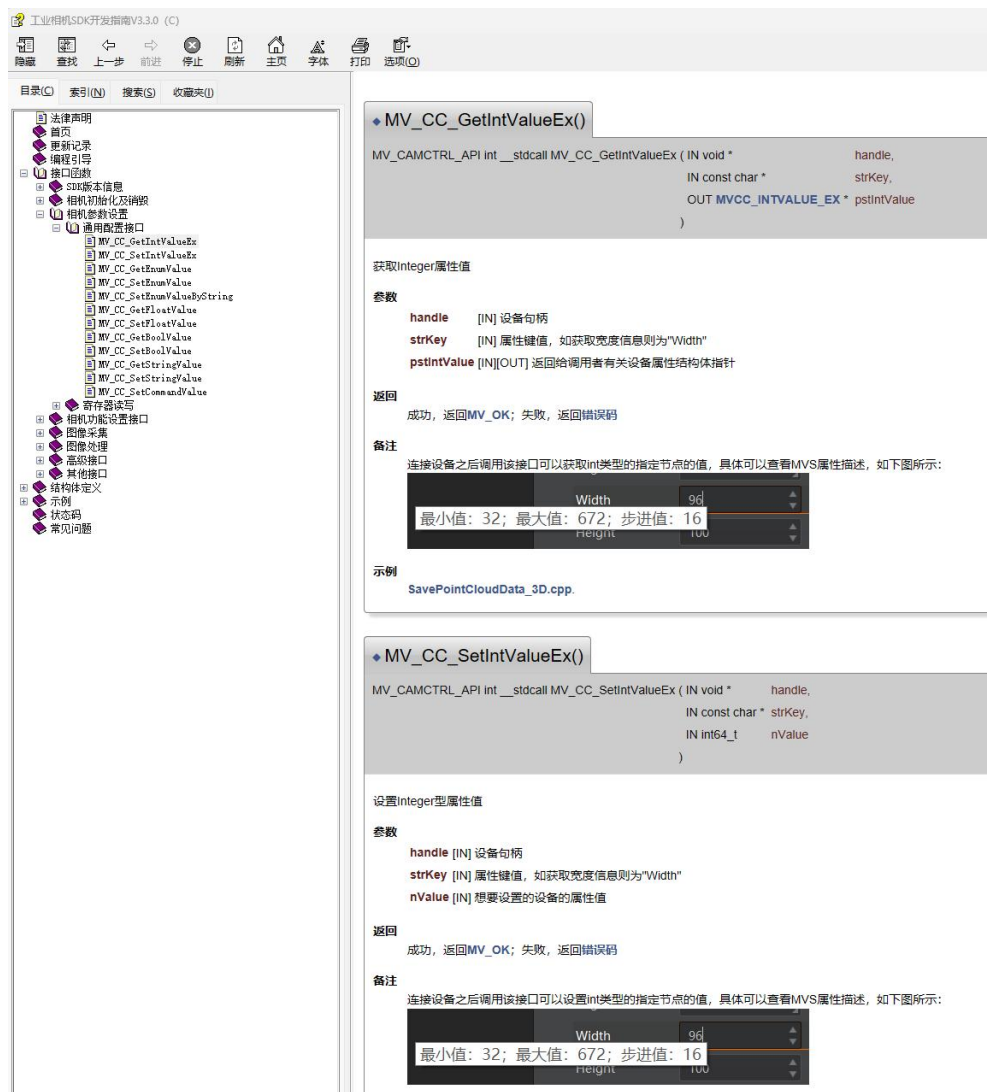
JHEM 系列相机是基于标准的 GigE Vision 和 USB3 Vision 开发，参数设置符合 Genicam 规则。每个相机携带 XML 文件定义了这台相机的基本功能和参数，所有参数都有取值范围。MVS 客户端软件，将 XML 以**属性树**的方式展现出来，以方便用户交互。如下图所示：1 属性树根节点，2 属性分类，3 属性节点，4 属性名称(Node Name)，属性类型(Type)，属性取值范围(Min, Max, Inc)。



2 参数设置方法

通过 MVS 查询到需要设置的参数以后，可以通过 SDK 调用对应的参数设置 API 来设置。参数类型有整形(Int)，枚举类型(Enum)，浮点型(Float)，布尔型(Bool)，字符串型(String)和命令型(Command)。每个类型有对应的 Get 和 Set 方法。

(查阅 C:\Program Files (x86)\MVS\Development\Documentations\工业相机 SDK 开发指南(C).chm)



The screenshot displays the MVS SDK development guide interface. On the left, a tree view shows the '相机参数设置' (Camera Parameter Settings) section, with '通用配置接口' (General Configuration Interface) expanded. The main content area shows the 'MV_CC_GetIntValueEx()' function signature and its details. The function signature is: `MV_CAMCTRL_API int __stdcall MV_CC_GetIntValueEx ( IN void * handle, IN const char * strKey, OUT MVCC_INTVALUE_EX * pIntValue )`. Below the signature, it describes the function as '获取Integer属性值' (Get Integer attribute value). The parameters are: `handle` [IN] 设备句柄 (Device handle), `strKey` [IN] 属性键值, 如获取宽度信息则为"Width" (Attribute key value, e.g., "Width" for width information), and `pIntValue` [IN|OUT] 返回给调用者有关设备属性结构体指针 (Return pointer to the device attribute structure to the caller). The return value is: 成功, 返回MV_OK; 失败, 返回错误码 (Success, return MV_OK; Failure, return error code). The备注 (Remarks) section states: 连接设备之后调用该接口可以获取int类型的指定节点的值, 具体可以查看MVS属性描述, 如下图所示: (After connecting the device, this interface can be used to get the value of the specified node of the int type. For specific details, see the MVS attribute description, as shown in the figure below:). The figure shows a camera property window with 'Width' set to 96, 'Height' set to 100, and 'Step' set to 16. The example code is 'SavePointCloudData_3D.cpp'.

The second function shown is 'MV_CC_SetIntValueEx()' with the signature: `MV_CAMCTRL_API int __stdcall MV_CC_SetIntValueEx ( IN void * handle, IN const char * strKey, IN int64_t nValue )`. It is described as '设置Integer属性值' (Set Integer attribute value). The parameters are: `handle` [IN] 设备句柄 (Device handle), `strKey` [IN] 属性键值, 如获取宽度信息则为"Width" (Attribute key value, e.g., "Width" for width information), and `nValue` [IN] 想要设置的设备的属性值 (Attribute value to be set for the device). The return value is: 成功, 返回MV_OK; 失败, 返回错误码 (Success, return MV_OK; Failure, return error code). The备注 (Remarks) section states: 连接设备之后调用该接口可以设置int类型的指定节点的值, 具体可以查看MVS属性描述, 如下图所示: (After connecting the device, this interface can be used to set the value of the specified node of the int type. For specific details, see the MVS attribute description, as shown in the figure below:). The figure shows the same camera property window with 'Width' set to 96, 'Height' set to 100, and 'Step' set to 16.

有些属性的设置是有条件的，比如设置**宽度(Width)**需要在预览停止后，否则会设置失败。所以需要先调用 **MV_CC_StopGrabbing**。再例如设置**线路 1(Line1)的线路翻转(LineInverter)**需要先将**线路选择器(LineSelector)**设置为 1。这些约束条件在 MVS 上均有体现，如果无法操作的时候，属性不会显示，或者即使显示也是灰色的无法修改的状态。

代码举例（C 语言）：

1 整形



```
int nRet = MV_CC_SetIntValueEx(handle, "Width", 1000);
if (MV_OK != nRet)
{
    printf("Set Width fail! nRet [0x%x]\n", nRet);
}
```

2 枚举型



```
// ch:设置触发模式为off | en:Set trigger mode as off
int nRet = MV_CC_SetEnumValue(handle, "TriggerMode", MV_TRIGGER_MODE_OFF);
if (MV_OK != nRet)
{
    printf("Set Trigger Mode fail! nRet [0x%x]\n", nRet);
    break;
}
```

3 浮点型



```
int nRet = MV_CC_SetFloatValue(handle, "ExposureTime", 200.0);
if (MV_OK != nRet)
{
    printf("Set ExposureTime fail! nRet [0x%x]\n", nRet);
    break;
}
```

4 布尔型

使能采集帧率控制
☒

使能采集帧率控制
使能手动控制相机的帧率

Node Name: **AcquisitionFrameRateEnable**
Type: **Boolean**
Name Space: **Standard**
Visibility: **Beginner**
Streamable: **Yes**

```
int nRet = MV_CC_SetBoolValue(handle, "AcquisitionFrameRateEnable", true);
if (MV_OK != nRet)
{
    printf("Set AcquisitionFrameRateEnable fail! nRet [0x%x]\n", nRet);
    break;
}
```

5 字符串型

设备用户ID

设备用户ID
用户定义的名称

Node Name: **DeviceUserID**
Type: **StringReg**
Name Space: **Standard**
Visibility: **Beginner**
Streamable: **No**

```
int nRet = MV_CC_SetStringValue(handle, "DeviceUserID", "CAM1");
if (MV_OK != nRet)
{
    printf("Set DeviceUserID fail! nRet [0x%x]\n", nRet);
    break;
}
```

6 命令型

触发模式

触发源

软触发

软触发
如果触发器源设置为软件，则生成内部触发器

Node Name: **TriggerSoftware**
Type: **Command**
Name Space: **Standard**
Visibility: **Beginner**
Streamable: **No**

```
int nRet = MV_CC_SetCommandValue(handle, "TriggerSoftware");
if (MV_OK != nRet)
{
    printf("Set TriggerSoftware fail! nRet [0x%x]\n", nRet);
    break;
}
```