Machine Vision Camera SDK Demo (VB)

User Manual

User Manual

About this Manual

This Manual is applicable to Machine Vision Camera SDK Demo (VB).

The Manual includes instructions for using and managing the product. Pictures, charts, images and all other information hereinafter are for description and explanation only. The information contained in the Manual is subject to change, without notice, due to firmware updates or other reasons. Please find the latest version in the company website.

Please use this user manual under the guidance of professionals.

Legal Disclaimer

REGARDING TO THE PRODUCT WITH INTERNET ACCESS, THE USE OF PRODUCT SHALL BE WHOLLY AT YOUR OWN RISKS. OUR COMPANY SHALL NOT TAKE ANY RESPONSIBILITIES FOR ABNORMAL OPERATION, PRIVACY LEAKAGE OR OTHER DAMAGES RESULTING FROM CYBER ATTACK, HACKER ATTACK, VIRUS INSPECTION, OR OTHER INTERNET SECURITY RISKS; HOWEVER, OUR COMPANY WILL PROVIDE TIMELY TECHNICAL SUPPORT IF REQUIRED.

Contents

Chapter 1 Overview 1
Chapter 2 Basic Demo 2
2.1 Interface Overview 2
2.2 Operation Procedure 2
2.3 Programming Guideline 3
Chapter 3 Reconnect Demo 4
3.1 Interface Overview 4
3.2 Operation Procedure 4
3.3 Programming Guideline 4
Chapter 4 Set IO Demo 6
4.1 Interface Overview 6
4.2 Operation Procedure 6
4.3 Programming Guideline 6
4.3.1 IO Property 6
4.3.2 APIs for Getting and Setting 7
4.3.3 IO Operation 7
Chapter 5 Force IP Demo 8
5.1 Interface Overview 8
5.2 Operation Procedure 8
5.3 Programming Guideline 8
Chapter 6 Multiple Demo 9
6.1 Interface Overview 9
6.2 Operation Procedure 9
6.3 Programming Guideline 10
6.3.1 Multiple Cameras 10
6.3.2 Total/Lost Frame Number and Picture Storage 10

Chapter 1 Overview

This manual mainly introduces the SDK (Software Development Kit) programming methods and procedure of machine vision camera based on VB.NET.

Thirteen VB.NET Demos are provided in the SDK directory, including five Form Demos and eight console Demos.

Form Demos: Basic Demo, Reconnect Demo, Set IO Demo, Force IP Demo, and Multiple Demo.

Console Demos: ConnectSpecCamera, ConvertPixelType , Events , Grab_Callback, GrabImage, MultiCast, ParametrizeCamera_FileAccess, and ParametrizeCamera_LoadAndSave.

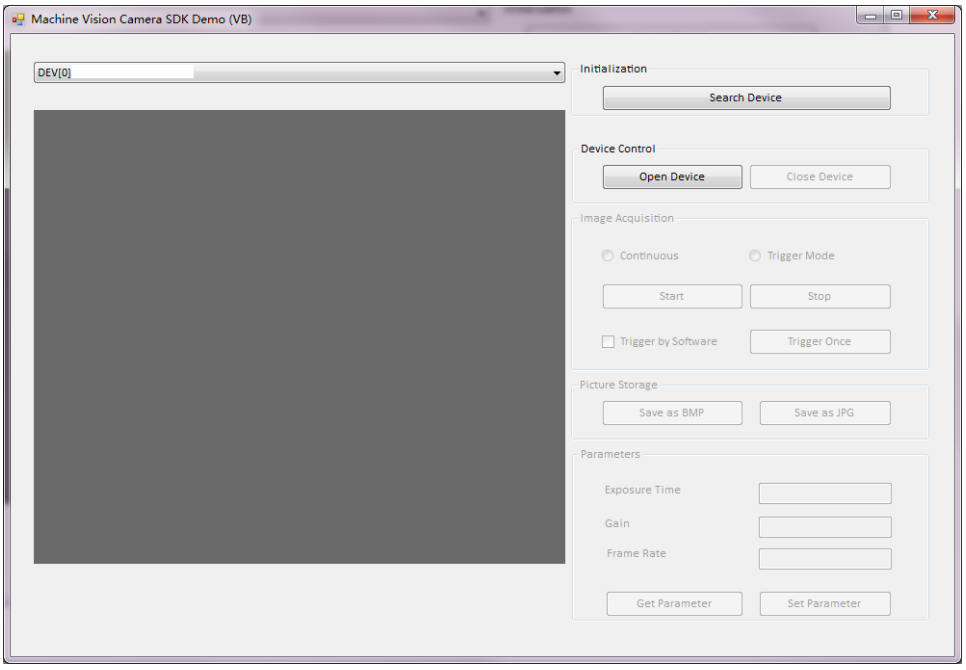
The Demos are developed by adopting MvCameraControl.Net.

To ensure the proper use of SDK, please refer to the contents below and read the manual carefully before operation and development.

Chapter 2 Basic Demo

2.1 Interface Overview

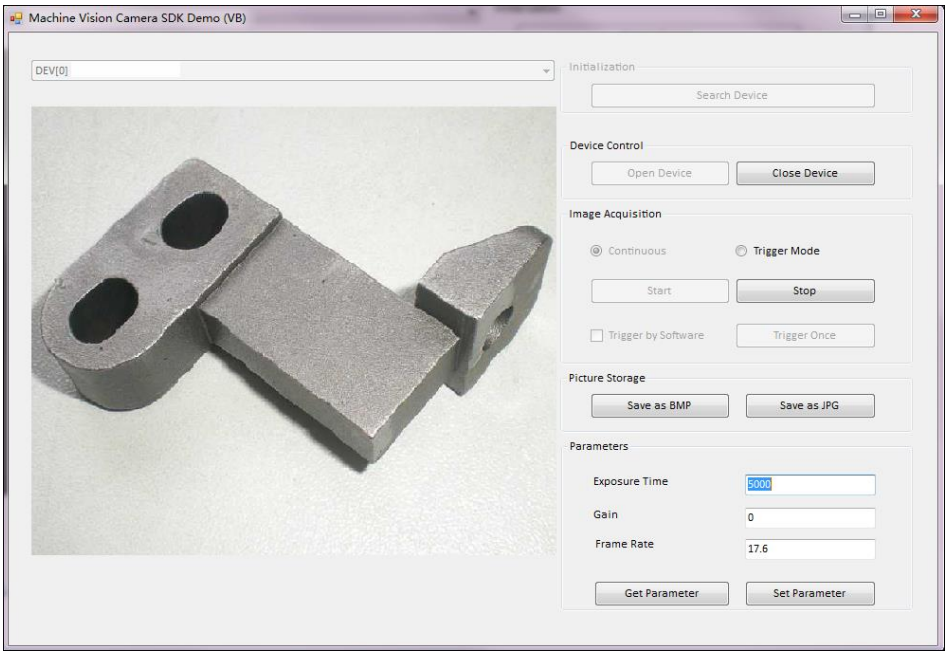
The Basic Demo based on VB.NET for machine vision camera can realize the function of image display, initialization, image acquisition, picture storage and parameter control.



2.2 Operation Procedure

Steps:

1. Click **Search Device** in the Initialization field to search the online device.
   The online devices will display in the drop-down list of the upper left corner field.
   Note: The devices will be displayed as “device ID” + “device name” + “IP address”.
2. Click to select a device in the drop-down list.
3. Click **Open Device** button in the Initialization field to active the Image Acquisition field.
4. Select image acquisition mode as **Continuous** or **Trigger Mode**.
   Notes:
   - The default image acquisition mode is **Continuous**.
   - When **Trigger Mode** is selected, you can check the **Trigger by Software** checkbox.
5. Click **Start** button in the Image Acquisition field to start image acquisition.
   The real-time image will display on the left display window if the **Continuous** mode is selected.
   You can also click **Trigger Once** button to realize software trigger for once if **Trigger by Software** checkbox is checked in Trigger mode.



6. Click **Save as BMP** or **Save as JPG** button in the Picture Storage field to save the current image, which is named by **.bmp* or **.jpg*, to the directory of .exe.
7. Set the value of exposure time, gain and frame rate in the Parameter field.
8. Click **Set Parameter** button to save the settings.
9. (Optional) You can click **Get Parameter** button in the Parameter field to refresh the value of exposure time, gain and frame rate.

   Note: If exception or error occurred during the procedure, the prompt dialog will pop up.

2.3 Programming Guideline

Steps:

1. Load DLL.

   The *.dll* file of 32-bit and 64-bit will be put into the directory of environment variables after installing the MVS.
2. Import *.NET* library.
3. Create class *MyCamera*R. It includes all the operations of camera in SDK.
4. (Optional) You can develop according to the function description.

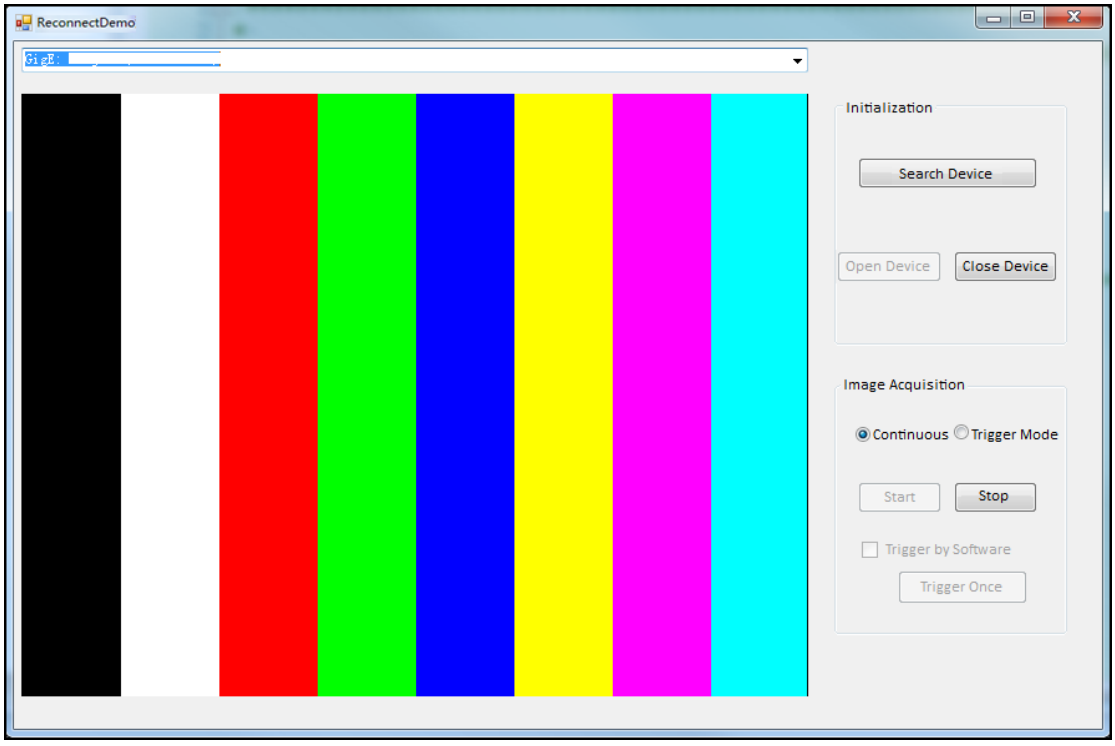
```
Imports System.Runtime.InteropServices
Imports System.Threading.Thread
Imports System.Net.IPAddress
Imports MvCamCtrl.NET


Public Class Thread
    Dim dev As MyCamera = New MyCamera
```

Chapter 3 Reconnect Demo

3.1 Interface Overview

The Reconnect Demo based on VB.NET for machine vision camera can realize the function of device search, device control, image acquisition and configuration trigger.



3.2 Operation Procedure

The operation procedure of Reconnect Demo is similar with that of Basic Demo, please refer to *Chapter 2.2 Operation Procedure* for details.

Note:

When the camera is disconnected, there will be callback exception, and the Reconnect Demo will attempt to connect the camera according to the selected camera information. And the online camera will be connected.

3.3 Programming Guideline

The programming guidance of Reconnection Demo is similar with that of Basic Demo, please refer to *Chapter 2.3 Programming Guidance* for details. Here we introduce the application method of callback function.

For VB, you should replace the function pointer of C language by delegate (proxy) method. So the exceptional disconnection callback proxy is *MyCamera. cbExceptiondelegate* in the machine vision

camera SDK (C#).

Steps:

1. Assign a variable for callback proxy member in *Form1* class.
2. Create an example.

   Example: Dim cbCallback As MyCamera.cbExceptiondelegate = New MyCamera.cbExceptiondelegate(AddressOf cbExceptiondelegateFunc);

   While, the cbExceptiondelegateFunc indicates the callback handling function, and it firstly makes the operations of CloseDevice and DestroyHandle, then continuously attempts to reconnect the camera.
3. Register callback function by calling the callback function registration API after open the camera.

   Example: dev.MV_CC_RegisterExceptionCallBack_NET(cbCallback, IntPtr.Zero);

Note:

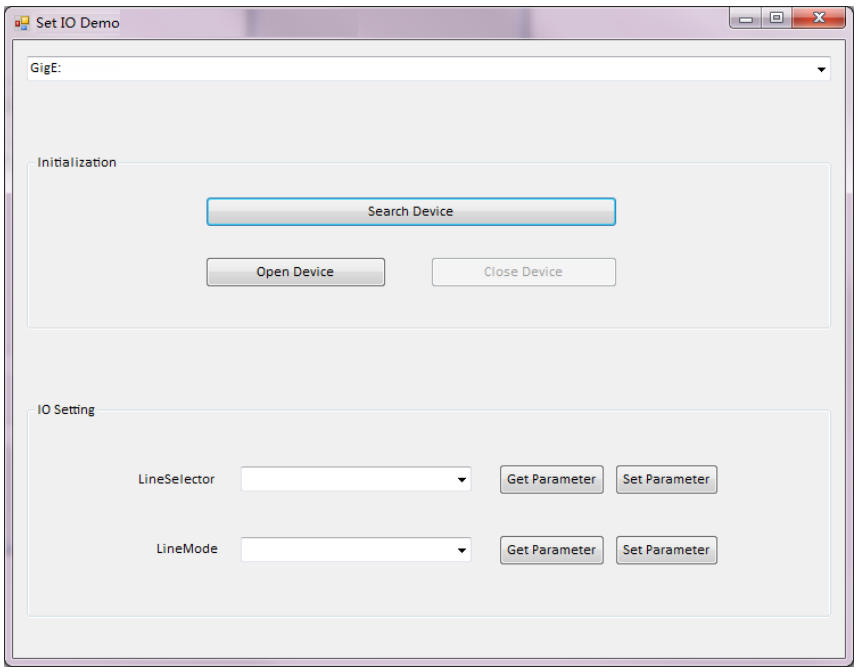
When the camera is disconnected, there will be callback exception, and you can reconnect the camera in exceptional callback.

Chapter 4 Set IO Demo

The Demo in this section mainly realizes the control of camera IO input and output.

4.1 Interface Overview

The Set IO Demo based on C++ language for machine vision camera can realize the function of device search, device control and IO settings.



4.2 Operation Procedure

The operation procedure of Set IO Demo is similar with that of Basic Demo, please refer to *Chapter 2.2 Operation Procedure* for details.

After opening a device, you can get and set the camera IO properties, e.g., LineSelector and LineMode. Click **Get Parameter** or **Set Parameter** to read or write the corresponding property.

4.3 Programming Guideline

4.3.1 IO Property

There are two IO properties, LineSelector or LineMode.

LineSelector: Select IO port. Three IO ports are available: Line0, Line1 and Line2. Line0 – Can be configured as input only; Line1 – Can be configured as output only; Line2 – Can be configured input or output.

LineMode: Input or output mode.

4.3.2 APIs for Getting and Setting

In the Demo, the APIs used to get and set IO are MV_CC_GetEnumValue_NET(ByVal strKey As String, ByRef pstValue As MyCamera.MVCC_ENUMVALUE) and MV_CC_SetEnumValue_NET(ByVal strKey As String, ByVal nValue As UInteger).

In the SDK, the API function which is similar with the format of *Set* or *Get + Data Type +Value* is a general API used to get or set any camera properties. The first parameter in the general API is property name, which is a *string* type string and can be searched in the file *CameraNode.xls*, while the second parameter is the obtained or configured property value.

4.3.3 IO Operation

The type of property nodes <LineSelector> and <LineMode> in the Demo is *Enumeration* type. Call general API can realize the property operations.

Get:

```
Dim stLineSelector As MyCamera.MVCC_ENUMVALUE =
        New MyCamera.MVCC_ENUMVALUE
        nRet = dev.MV_CC_GetEnumValue_NET("LineSelector", stLineSelector)
Dim stLineMode As MyCamera.MVCC_ENUMVALUE =
        New MyCamera.MVCC_ENUMVALUE
        nRet = dev.MV_CC_GetEnumValue_NET("LineMode", stLineMode)
```

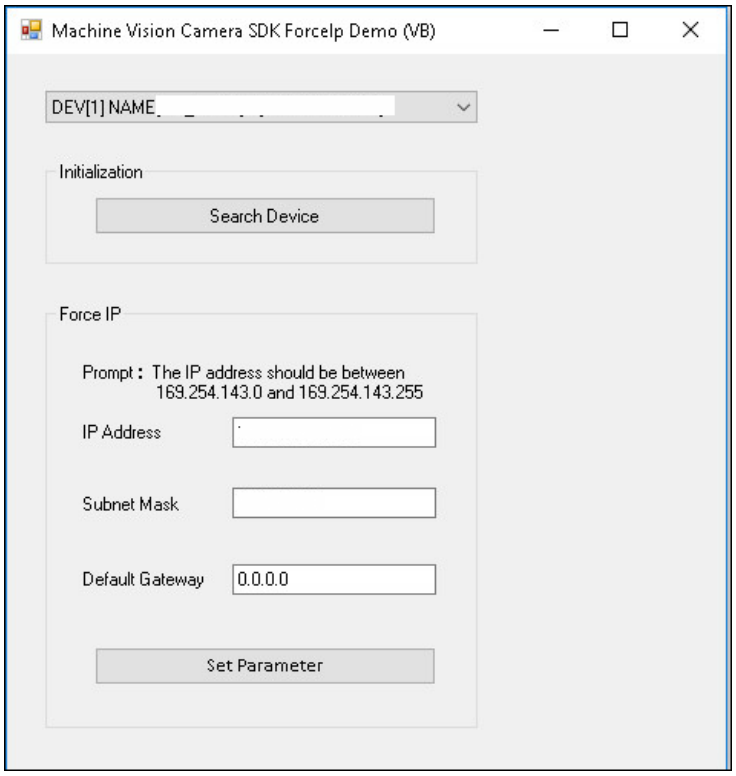
Set:

```
nRet = dev.MV_CC_GetEnumValue_NET("LineSelector ", stLineSelector)
nRet = dev.MV_CC_GetEnumValue_NET("LineMode", stLineMode)
```

Chapter 5 Force IP Demo

5.1 Interface Overview

The Force IP Demo based on VB.NET for machine vision camera can realize the function of device search and IP address settings.



5.2 Operation Procedure

Steps:

1. Click **Search Device** to enumerate the devices in the IP segment.
   Note: The first device item in the searched list will be selected automatically.
2. Select a device to configure IP address.
3. Input desired IP address in the text field.
   Note: In the Set IP field, the IP segment of local NIC and suggested IP range will be displayed in prompt information.
4. Click **Set Parameter** to set the IP address.

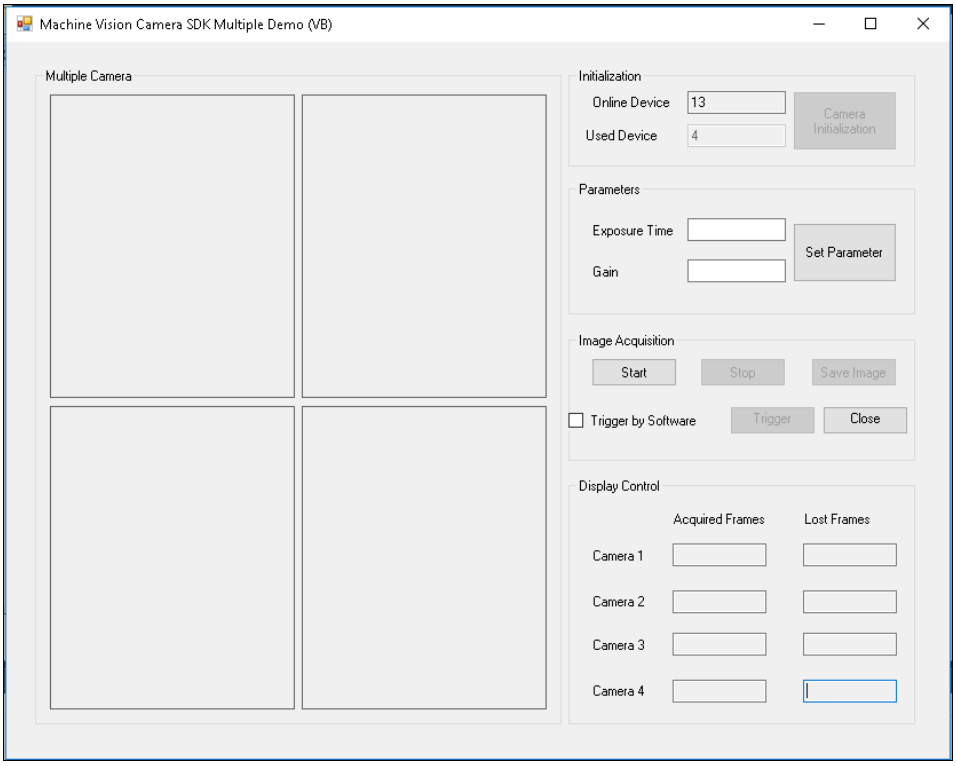
5.3 Programming Guideline

After setting the IP address, call API MV_GIGE_ForceIpEx_NET(ByVal nIP As UInteger, ByVal nSubNetMask As UInteger, ByVal nDefaultGateWay As UInteger) in the SDK.

Chapter 6 Multiple Demo

6.1 Interface Overview

The Multiple Demo based on VB.NET for machine vision camera can realize the function of initialization, parameter settings, image acquisition, image display and frame information display.



6.2 Operation Procedure

Steps:

1. The online device number will be enumerated automatically and display in the Online Device field after opening the Demo.
2. Input required number of cameras in the Used Device field.
3. Click **Camera Initialization** to open the devices in corresponding number continuously.
   Note: After initializing, the Parameter field and Image Acquisition field will be active.
4. Input parameters in Exposure and Gain field to edit.
5. Click **Set Parameter** to edit the corresponding parameters of all opened devices.
6. Select image acquisition mode as continuous or trigger mode.
7. Click **Start** to start the acquisition.
   The live image will display in the left display area.
   The acquired frame number and lost frame number will be refreshed (refresh per second).
8. Click **Save** to save the four camera's pictures as files named by *image1-image4.bmp* under the

directory of executing program.

9. (Optional) Click **Stop** and **Close** to end the operation.

Note: If exception or error occurred during the procedure, the prompt dialog will pop up.

6.3 Programming Guideline

6.3.1 Multiple Cameras

Based on the Basic Demo, the Multiple Demo added a member variable m_bEnabled array in the class. This variable is used to enable four cameras, and the m_bEnabled status (True or False) is determined by the used device number and opened device number when initializing. The following basic operations (do or not) of corresponding cameras are also determined by m_bEnabled.

6.3.2 Total/Lost Frame Number and Picture Storage

The total frame number (member variable) is calculated in the callback function.

Get the lost frame number by calling API MV_GIGE_GetNetTransInfo_NET(ByRef pstInfo As MyCamera.MV_NETTRANS_INFO).

The update period of total frame and lost frame is 1 second. Set the timer and get the lost frame number per second, and then update the total frame number and lost frame number.

The callback function can also save the pictures and judge whether to save the current frame as picture via the click of Save Picture button. After saving the picture, edit the flag bit to avoid saving picture repeatedly.