DVP2 SDK user guide

Table of Contents

1 Introduction and Overview

First of all,thank you very much for using the products.DVP2 SDK is the second generation industrial camera product software development package independently developed by the company. The latest industrial camera products(U3M series,U3S series,GM series)of our company can use this SDK to develop application software. It has the following characteristics:

(1)Efficient, concise and standardized API interface provides users with a simple and easy-to-use development environment; The powerful debugging function can easily locate the problems encountered by users in the use process.

(2)UI configuration interface is provided,Users do not need to implement camera configuration interface. Users can adjust various parameters of the camera by calling an API.

(3)Provide special conversion interfaces for various mainstream machine vision software(LabView,Halcon),so that cameras can run on these machine vision software more efficiently.

(4)Support GenICam interface, and support general camera application software using this interface.

(5)Support major programming languages and development environments: Visual C++, Visual C#, VB.net, Qt.

(6)Provide high-quality and practical reference sample code.

2 How to use the DVP2 API

This chapter mainly introduces how to use the DVP API for application development, involving the following aspects

1). Some related documents.
2). Setting up the development environment.
3). The basic flow of camera development.
4). Adjustment of relevant parameters in the running process.
5). Image acquisition methods.
6). Sample code list.

2.1 Introduction to relevant documents

This section mainly describes the files that need to be used during compilation and runtime.

2.1.1 Files required for compilation

File name	File function
DVPCamera. h	define enum, struct and API
DVPCamera.lib	static link file for DVPCamera.dll
DVPCamera64.lib	static link file for DVPCamera64.dll

Table1 Files required for compilation

2.1.2 Files required at runtime

The files required for running applications developed using the DVP2 API are shown in the following table.

file name	File function
DVPCamera.dll	API Interface Dynamic Library
CommonHZD.dll	Camera Drive Common Library
XXXXX.dscam.dll	Camera Device Driver

Table 2 Files required for runtime (32-bit)

file name	File function
DVPCamera64.dll	API Interface Dynamic Library
CommonHZD64.dll	Camera Drive Common Library
XXXXX.dscam64.dll	Camera Device Driver

Table 3 Files required for runtime (64 bit)

If the driver installation package provided by our company is used, the above runtime files will be copied to the installation directory (assuming the system directory is C disk):

When using the installation package,

32-bit files are installed to C:\Program Files(x86)\Camera\DVP2

64 bit files are installed to C:\Program Files(x86)\Camera\DVP2 x64

User can also copy the files to the application directory. If the files exist in both the system directory and the application directory, the .dll file in the application will be loaded first.

2.2 Set development environment

This section mainly introduces how to set the development environment with Visual C++, Visual C# and Visual Basic.net.

2.2.1 Visual C++ development environment configuration

(1)Add header file:

```
#Include "Header file directory/DVPCamera.h"
```

(2)Load LIB library:

```
#Ifdef _ M_ X64//64 bit              //Load 64 bit LIB library
#pragma comment (lib, "64 bit LIB file directory/DVPCamera64. lib")
# else                               //Load 32-bit LIB libraries
#pragma comment (lib, "32-bit LIB file directory/DVPCamera.lib")
# endif
```

2.2.2 Visual C # development environment configuration

Add Reference
32-bit Add DVPCameraCS.dll
64 bit Add DVPCameraCS64.dll
Namespace
Namespace - DVPCameraType
API function class - DVPCamera

2.2.3 Visual Basic.net development environment configuration

Add Reference
32-bit Add DVPCameraCS.dll
64 bit Add DVPCameraCS64.dll
Namespace
Namespace - DVPCameraType
API function class - DVPCamera

2.3 Basic process of camera operation

The general use process is divided into the following steps:

(1)Use dvpRefresh and dvpEnum to get camera information;

(2)After using dvpOpenByName/dvpOpen to successfully open/initialize the camera, user will get an ID corresponding to the camera (handle);

(3)After the camera is opened, you can configure various modes and adjust parameters of the camera;

(4)After the configuration mode and parameters are set, use dvpStart to start image streaming;

(5)After grabed image, you can use dvpStop to close image streaming;

(6)Use dvpClose to close (deinitialize) the camera and end the camera operation.

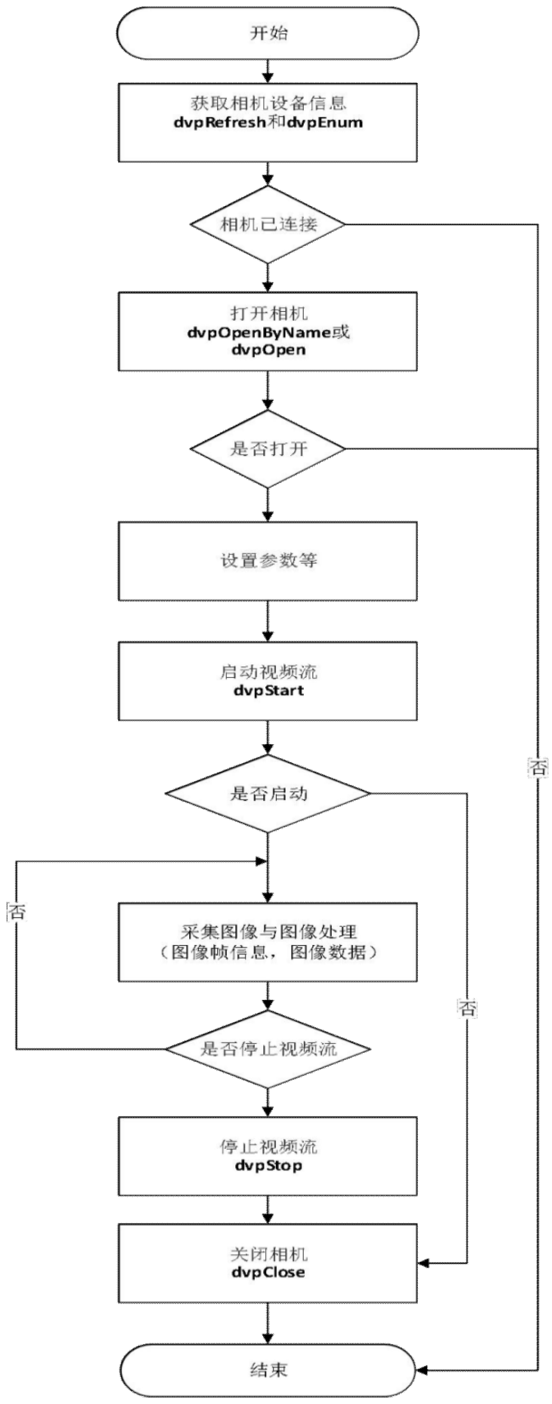


Figure 1 Camera operation flow chart

2.4 Set parameters during camera operation

After the camera is started or in the process of outputting images, it is allowed to adjust each function of the camera in real time, except for the following functions, which need to stop the camera (dvpStop) first.

function	Related interfaces
Network camera transmission packet length	dvpSetStreamPackSize
Acquired image format	DvpSetSourceFormat
Destination image format	DvpSetTargetFormat

Table 4 Functions that can be set only when the camera is stopped

2.5 Grab image method

There are two main methods to use the camera for image acquisition: callback function and directly mode, which can be used at the same time.

2.5.1 Callback function

Callback mode is the simplest application mode, which is suitable for video display, simple image processing and other applications. Register a user provided callback function through dvpRegisterStreamCallback to get image data.

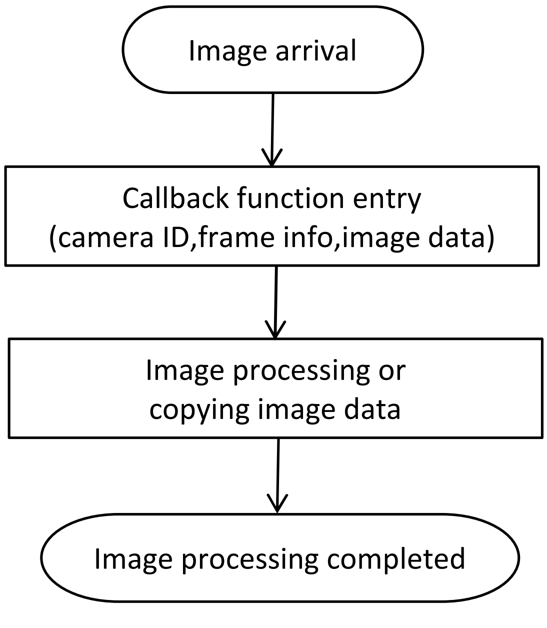


Figure 2 Flow chart of image acquisition and processing using callback method

2.5.2 Directly mode (blocking mode)

Main application scenario: The user calls dvpGetFrame in the image acquisition or processing thread to grab image data. These processes will be blocked before the image data is grabbed. To avoid long-term blocking, dvpGetFrame can set timeout to allow users to determine the waiting time.

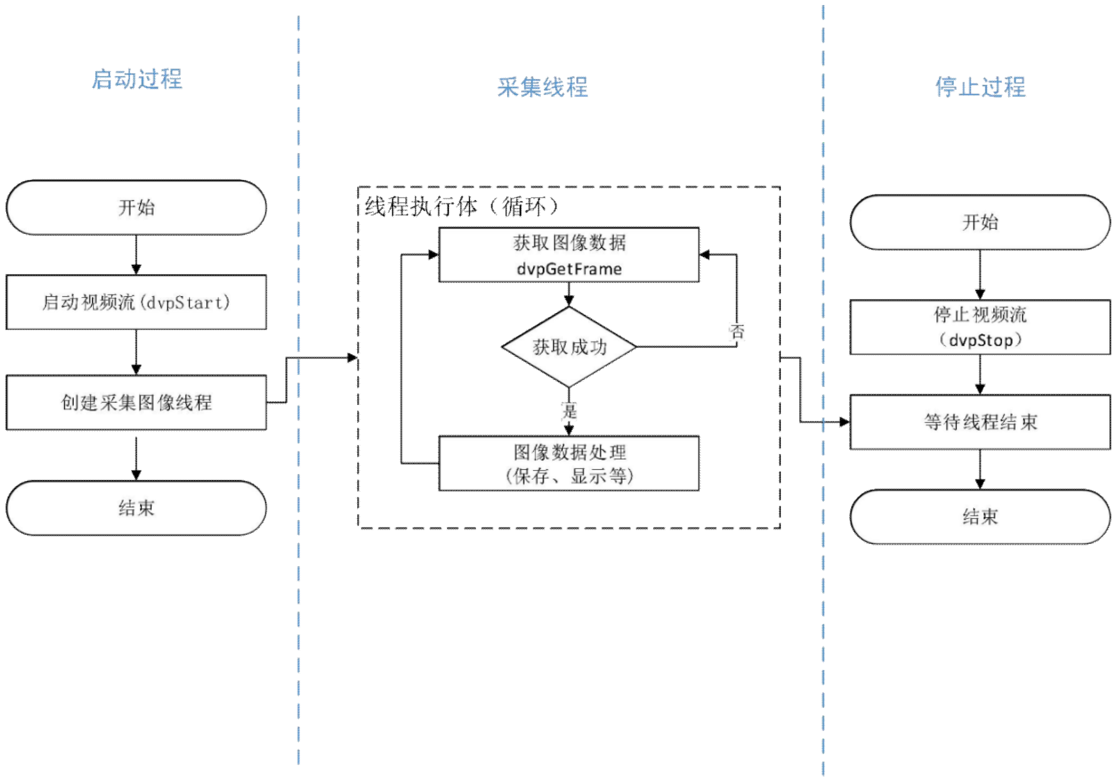
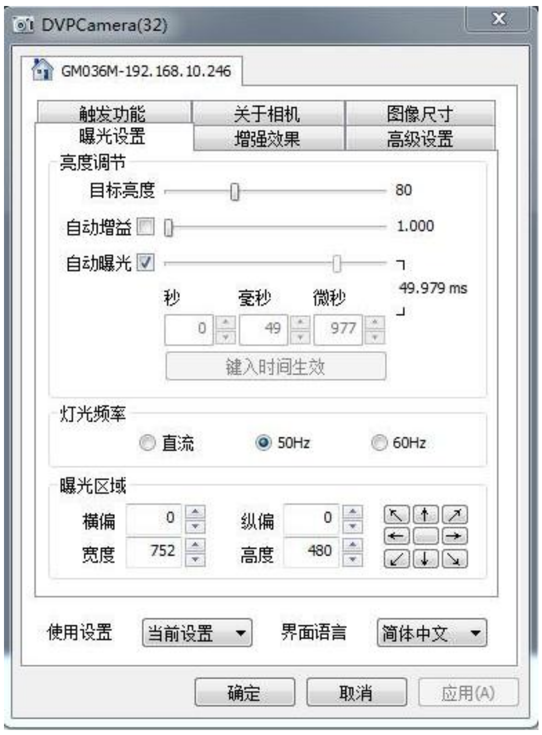


Figure 3 Image acquisition in directly mode

3 Camera property page operations

Use the API dvpShowPropertyModalDialog to pop up a property page as follows:



DVPCamera(32)

GM036M-192.168.10.246

触发功能 | 关于相机 | 图像尺寸
曝光设置 | 增强效果 | 高级设置

亮度调节
目标亮度 80
自动增益 ☐ 1.000
自动曝光 ☑ 49.979 ms
秒 0 毫秒 49 微秒 977
键入时间生效

灯光频率
○ 直流 ● 50Hz ○ 60Hz

曝光区域
横偏 0 纵偏 0
宽度 752 高度 480

使用设置 当前设置
界面语言 简体中文

确定 取消 应用(A)

Figure 4 Property page

Common parameter configuration operations can be completed through the property page. Click "OK" to save the camera parameters.

3.1 Save camera parameters

(1)If the user uses the property page function and clicks the "OK" button on the property page, an archive will be generated.

(2)If the user uses the driver installation program corresponding to the camera product of our company to install, the default parameter archive path may be C:\ ProgramData\Camera\DVP2 directory (assuming that disk C is the system disk).

(3)If the user directly copies the library files (DVPCamera.dll, XXX. dscam. dll) to the application directory, the default parameter archive path will be the directory where the EXE program is located.

(4)The general file name of the parameter archive file is: [Product Serial Number]. ini, so each camera will correspond to an archive file.

4 Reference sample code list

Sample project	Keyword	Sample introduction
BasicFunction	Auto exposure Exposure time Analog gain Anti - Flick Resolution switching Video stream callback function	Enumerate the connectable cameras, the opening and closing of camera devices, the starting and stopping of camera video streams, camera property settings and other basic functions. Save picture function. AE operation, AE mode selection, exposure time adjustment, Anti - Flick mode setting, analog gain adjustment and other functions. Camera resolution settings.
ImageAcquisition	Acquisition thread Save image Soft trigger Timing to Grab image Synchronization grab	Enumerate the connectable cameras, the opening and closing of camera devices, the starting and stopping of camera video streams, camera property settings and other basic functions. Capture images in a thread using synchronization and can display the captured images or save them to a file depending on the settings. Open a folder of saved images. You can use soft-trigger acquisition, which can save communication bandwidth and reduce system overhead without requiring image data; the soft-trigger method can also have the effect of acquisition synchronization. You can set the waiting time for acquisition to realize the effect similar to timed acquisition.
MultipleCamera	Multi camera User naming (User ID) Camera name binding	Enumerate the connectable cameras, the opening and closing of camera devices, the starting and stopping of camera video streams, camera property settings and other basic functions. User ID setting. Four cameras are working at the same time,

		bind each camera by user ID and save the binding relationship to realize that each video window corresponds to the specified camera at each startup.
Trigger	Soft trigger External trigger Cycle trigger Strobe signal	Adjustment of soft trigger and external trigger related parameters, such as trigger signal jitter filtering, delay, timer setting. Configuration of trigger input signals and strobe output signals.