Tucsen Camera Labview Development Guide

Contents

1. Read before use

This document and software sample code are TUCSEN's internal documents and publications to enable users to create applications using TUCSEN digital cameras.

This documentation and software sample code are disclosed for the above purposes only, and do not constitute a license, assignment, or any other right of the owner.

All risks and consequences of using the software documentation remain with the user.

This document may contain technical inaccuracies or typographical errors. and cannot guarantee any damages resulting from such errors or texts.

TUCSEN makes no commitment to update or maintain the current information contained in this document.

All brand and product names are trademarks or registered trademarks of their respective owners.

Instructions for this version:

Since the current version is not perfect, if you want to use it, you must copy the TucsenCamera folder in the compressed package to the user.lib file in the Labvie installation path, so that the corresponding VI module can be called normally. Subsequent versions will install the corresponding content in a packaged form, omitting the step of copying the folder.

The picture is an example path to store the corresponding folder

We apologize for any inconvenience caused to users, and we will make changes as soon as possible to meet user needs.

2. Introduction

This manual describes in detail the TULV_API specification operation using TUCSEN digital cameras. The TULV_API software development kit is called "SDK". The part of TULV_API that controls the digital camera is called a "module".

The SDK contains VI modules and a sample application, which shows how to carry out secondary development of digital cameras through VI modules. SDK users are free to use the software in any way they like, such as partially modifying VI construction or creating completely separate projects.

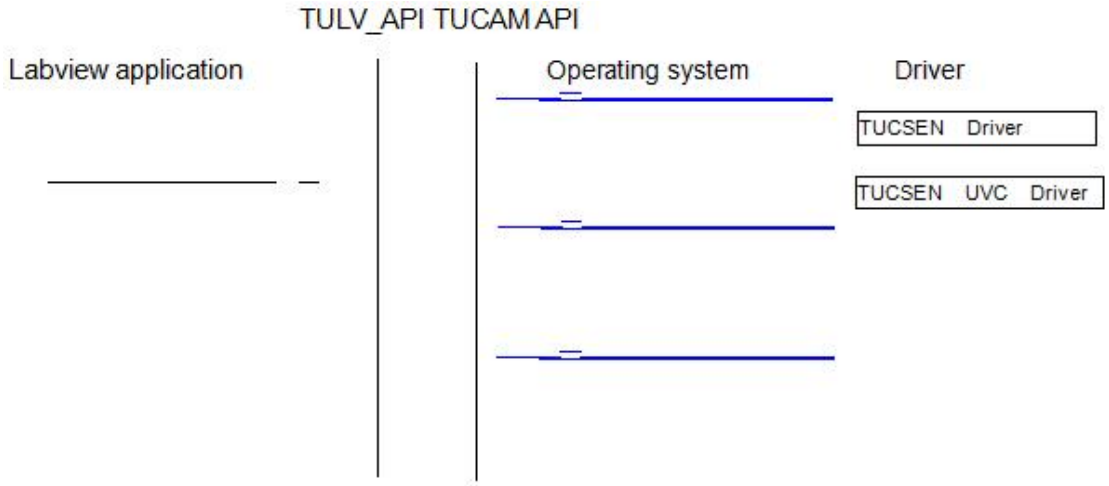
This SDK is designed to be particularly easy to understand. For this reason, the number of functional interfaces is limited to a minimum, and the calling format of functions is written in the C language.

Partially extended functions are additional functions available for certain digital cameras.

Values may vary for different digital cameras, depending on the model of digital camera used to capture the image. Numerical values should simply be regarded as guidelines, not exact values.

3. Overview

3.1 Layer structure



TUCSEN digital cameras connect digital camera drivers of different operating systems through SDK to control digital cameras and collect image data.

The current SDK only supports Windows systems .

3.2 Principles

The specific bus interface and library of the digital camera are encapsulated by TULV_API. The

interface calls of the VI parts of each module are all from the interface of TULV_API. You can refer to the LabviewDemo provided by us to understand the usage of the interface, and how to use it can refer to (6.VI Introduction and Application).

3.3 Interface Type

TULV_API functions can be divided into many types:
Start/stop processing
Camera information collection
Property/property get and set
Memory management
Capture control
Document Control
Extended control

3.4 Terminology

3.4.1 Capture mode:

The capture modes of the camera are divided into the following 2 categories:

Sequential Mode (Streaming Mode): Used to capture continuous image data.

Trigger Mode: The camera captures images through an external signal. We call this option "trigger mode" and you can call TU_SetTrigger() to configure this option. We also refer to external signals as "external triggers" .

3.4.2 Image unit:

Usually two-dimensional, with vertical and horizontal orientation.

Frame: is a unit used for image data. For a frame, a pixel's data is aligned left-to-right and top-to-bottom. This is a series of image data units.

3.4.3 Trigger Mode:

Standard mode (Standard): When the camera receives the level signal (determined by the active edge), it starts to capture one or more frames of images, and the number of captured frames is determined by the configuration parameters.
Synchronization mode: When the camera receives the level signal (determined by the active edge), the exposure starts. When the opposite level signal is received, the exposure ends and the image data is captured. That is, the exposure and readout of each frame are completely synchronized with the

external trigger signal.

Global mode (Global): pre-trigger before the camera is triggered, when the camera receives the level signal (determined by the activation edge) or when the exposure time set by the software ends the current reset operation, waiting to be exposed Image data is captured at the end and pre-triggering is restarted. This method is used to control the camera in the rolling shutter exposure mode to realize the global exposure mode.

Exposure Mode:
Exposure time: After receiving the trigger signal, it is determined by the exposure time set by TULVIDP_EXPTM
Level width: After receiving the trigger signal, the exposure time is determined by the width of the level
Note: These two options can be configured in Standard mode and Global mode. Synchronization can only be level width.

Excitation Level Type:
Rising Edge: Exposure starts when the received trigger level is on the rising edge
Falling Edge: Exposure starts when the received trigger level is on the falling edge

3.4.4 Camera Status:

The state of the camera determines which functions can be called. Some functions change the state of the camera. The 4 camera states are described below:

Unstable: parameter settings and other function calls, but they are not in the set state.

Stable: Parameters and functions are set, but capture cannot begin because no frame memory is created.

Preparation: The frame memory has been created and image capture can begin.

Busy: Image capture is being performed.

4. Version introduction

4.1 Development environment

Currently only supports Labview development under Windows system. Developers can choose the X86 or X64 installation package for development and application according to the actual situation.

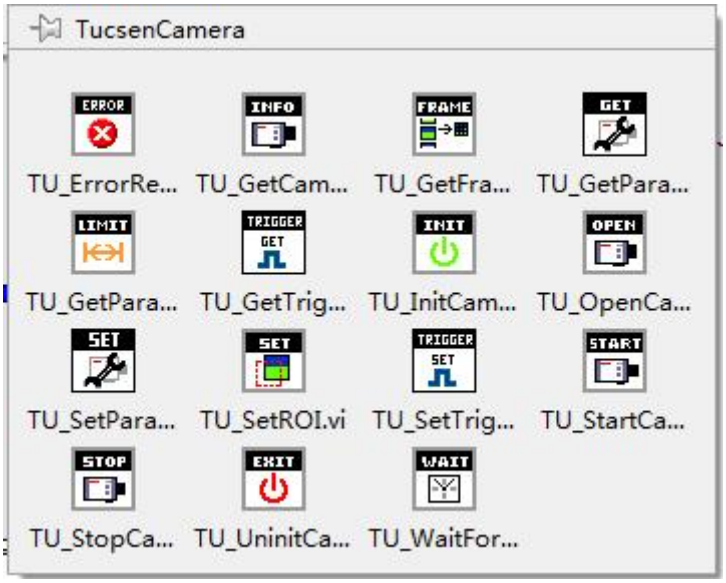
4.2 Performance optimization

4.2.1 Number of interfaces and call optimization

Version 2.0 systematically optimizes the interface functions. Without reducing the basic functions of version 1.0, the number of interface functions is reduced, which is convenient for users to call.

4.2.2 VI module and call optimization

Version 2.0 splits a single VI module in version 1.0 into multiple sub-VIs, which can be called directly in the Labview block diagram, which facilitates the developer's research on the sample program LabviewSample. For the introduction of VI module, see (6.VI Introduction and Application).



(Figure 4.2.2)

4.3 Function addition

4.3.1 Display of camera related information

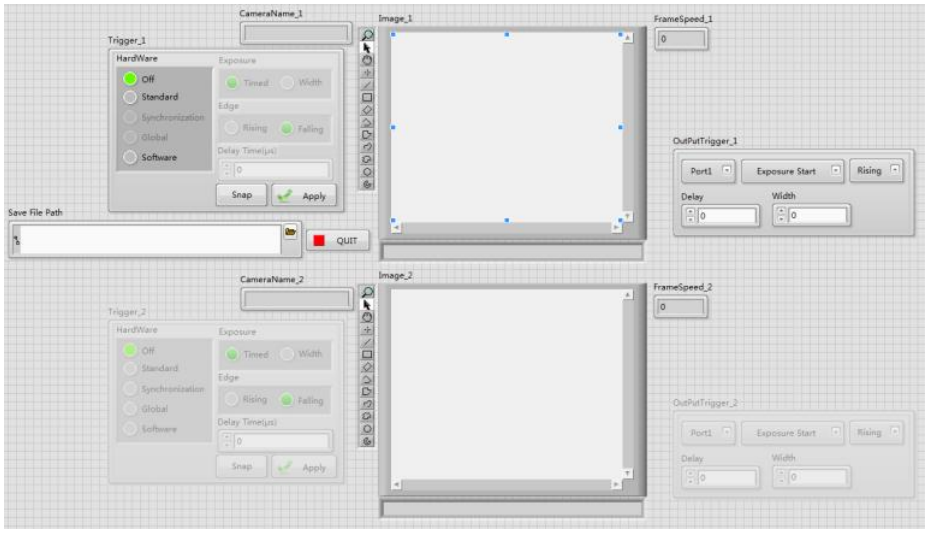
In order to accurately identify the camera, version 2.0 adds the display of camera name, TYPE , SN code, SDK version information, and firmware version number. (Get 0x210 is USB2.0, 0x03 is FireBird,

0x04 is Euresys, others are USB3.0. Cameralink does not display SN and firmware number)

(Fig. 4.3.1)

4.3.2 External trigger

According to the functions supported by the camera, in the sample program SetTriggerDemo.VI of version 2.0, an external trigger module and a trigger output module have been added . For the introduction of the trigger mode, see (3.4.3 Trigger Mode), and for the use of the trigger, see the introduction of TU_SetTrigger.VI. .



(Fig. 4.3.2)

4.3.3 Multi-Camera Operation

The Labview sample program of version 2.0 can support the simultaneous preview of multiple cameras. At the same time, you can set the parameters of the camera according to the camera name to change the preview effect, which is convenient for users to compare the effects of multiple cameras.


ImageB
2048x2044 0.17X 32-bit RGB image 136,135,144 (585,5)
Image A
1272x952 0.34X 32-bit RGB image 17,13,0 (93,585)

(Figure 4.3.3)

5. Reference

5.1 Types and Constants

5.1.1. TULVRET error code: for TU_ErrorReport.vi

```
TULVRET _SUCCESS          =0x00000001,        // no error, general success code
TULVRET _FAILURE          = 0x80000000,       // error

// initialization error
TULVRET _NO_MEMORY        = 0x80000101,       // not enough memory
TULVRET _NO_RESOURCE      = 0x80000102,       // not enough resources (excluding memory)
TULVRET _NO_MODULE        = 0x80000103,       // no submodule
TULVRET _NO_DRIVER        = 0x80000104,       // no driver
TULVRET _NO_CAMERA        = 0x80000105,       // no camera
TULVRET _NO_GRABBER       = 0x80000106,       // no image is taken
TULVRET _NO_PROPERTY      = 0x80000107,       // no alternate property ID

TULVRET _FAILOPEN_CAMERA     = 0x80000110,          // Failed to open camera
TULVRET _FAILOPEN_BULKIN          = 0x80000111,     // Failed to open bulk transfer input
endpoint
TULVRET _FAILOPEN_BULKOUT    = 0x80000112,     // Failed to open batch output endpoint
TULVRET _FAILOPEN_CONTROL    = 0x80000113,     // Failed to open control endpoint
TULVRET _FAILCLOSE_CAMERA    = 0x80000114,      // Failed to close the camera
TULVRET _FAILOPEN_FILE       = 0x80000115,     // fail to open the file

// status error
TULVRET _INIT             = 0x80000201,       // API needs initialization state.
TULVRET _BUSY             = 0x80000202,       // API is busy
TULVRET _NOT_INIT         = 0x80000203,       // API not initialized
TULVRET_EXCLUDED _        = 0x80000204,       // some resources are used exclusively
TULVRET _NOT_BUS          = 0x80000205,       // API is not busy
TULVRET _NOT_READ         = 0x80000206,       // API is not ready

// wait error
TULVRET _ABORT             = 0x80000207,      // Terminate processing
TULVRET _TIMEOUT          = 0x80000208,       // timeout
TULVRET _LOSTFRAME        = 0x80000209,       // frame lost
TULVRET _MISSFRAME        = 0x8000020A,       // frame lost but it is the underlying driver
problem
```

```
// calling error
TULVRET _INVALID_CAMERA     = 0x80000301,        // invalid camera
TULVRET _INVALID_HANDLE     = 0x80000302,        // invalid camera handle
TULVRET _INVALID_OPTION     = 0x80000303,        // invalid configuration value
TULVRET _INVALID_IDPROP     = 0x80000304,        // invalid property ID
TULVRET _INVALID_IDCAPA     = 0x80000305,        // Invalid Capability ID
TULVRET _INVALID_IDPARAM    = 0x80000306,        // invalid parameter ID
TULVRET _INVALID_PARAM      = 0x80000307,        // invalid parameter
TULVRET _INVALID_FRAMEIDX = 0x80000308,          // invalid frame number
TULVRET _INVALID_VALUE      = 0x80000309,        // invalid value
TULVRET _INVALID_EQUAL      = 0x8000030A,        // the value is equal, the parameter is invalid
TULVRET _INVALID_CHANNEL    = 0x8000030B,        // The attribute ID specifies the channel, but
the channel is invalid
TULVRET _INVALID_SUBARRAY = 0x8000030C,          // The subarray value is invalid
TULVRET _INVALID_VIEW       = 0x8000030D,        // invalid display window handle
TULVRET _INVALID_PATH       = 0x8000030E,        // invalid file path
TULVRET _NO_VALUETEXT       = 0x80000310,        // text for attribute without value
TULVRET _OUT_OF_RANGE     = 0x80000311,          // value out of range
TULVRET _NOT_SUPPORT        = 0x80000312,        // Unsupported function or property
TULVRET _NOT_WRITABLE     = 0x80000313,          // property is not writable
TULVRET _NOT_READABLE     = 0x80000314,          // attribute not readable
TULVRET _WRONG_HANDSHAKE          = 0x80000410,  // error occurred while getting error code
TULVRET _NEWAPI_REQUIRED          = 0x80000411,  // The old API does not support, only the
new API supports it
TULVRET_ACCESSDENY                = 0x80000412 , // When the camera is in a certain state
the property cannot be accessed
TULVRET _NO_CORRECTIONDATA        = 0x80000501,  // No color point correction data.

// camera or bus trouble
TULVRET _FAIL_READ_CAMERA         = 0x83001001,   // Failed to read from camera
TULVRET _FAIL_WRITE_CAMERA        = 0x83001002,   // Failed to write to camera
TULVRET _OPTICS_UNPLUGGED         = 0x83001003,  // for insert
```

5.1.2. TULV_IDINFO Product Information ID: for TU_GetCameraInfo.vi

```
TULVIDI_NAME          = 0x0001,      //Camera name
TULVIDI_SN            = 0x0002,      //SN code
TULVIDI_FS            = 0x0003,      //firmware version number
TULVIDI_USB           = 0x0004,      //USB interface type (USB2.0/USB3.0)
TULVIDI_SDKNO         = 0x0005,      //API version number
TULVIDI_RESOLUTION    = 0x0006,      //Resolution display
TULVIDI_PGAHIGH       = 0x0007,      //pga high display
TULVIDI_PGALOW      = 0x0008,        //pga low display
TULVIDI_IMGMODE       = 0x0009,      // Display of capture card mode
```

```
TULVIDI_FANMODE          = 0x000A,    // display fan information
TULVIDI_IMGTESTMODE      = 0x000B,    //Test image display
TULVIDI_GAINMODE         = 0x000C,    //Gain mode display
```

5.1.3. TULV_IDCAPA Capability ID: for TU_SetParameter.vi

```
TULVIDC_AUTOWB           = 0x0200,    //Auto white balance ID (DEC: 512)
TULCIDC_AUTOEXP          = 0x0205,    //Auto Exposure ID (DEC: 517)
TULVIDC_DEPTH            = 0x020A,    //Data width (8, 16) ID (DEC: 522)
TULVIDC_AUTOLEVEL        = 0x020F,    //Auto level ID (DEC: 527)
TULVIDC_RESOLUTION       = 0x0214,    // Resolution ID (DEC: 532)
TULVIDC_NOISELEVEL       = 0x0219,    // Noise reduction level ID (DEC: 537)
TULVIDC_IMGMODESELECT    = 0x021E,    //CMS(or 11bit)ID (DEC: 542)
TULVIDC_IMGSAVEPATH      = 0x0223,    //Image save path ID      (DEC: 547)
TULVIDC_LEDENBALE        = 0x0228,    // LED light control ID   (DEC: 552)
TULVIDC_PIENBALE         = 0x0229,    //PI heating film enable ID (DEC: 553)
TULVIDC_BLACKBALANCE     = 0x022D,    //Black balance ID       (DEC: 557)
TULVIDC_TESTIMGMODE      = 0x022E,    // Test pattern ID       (DEC: 558)
TULVIDC_SENSORRESET      = 0x022F,    // Sensor reset ID   (DEC: 559)
TULVIDC_SAVEFRAME        = 0x0300,    //Save image state ID (DEC: 768)
TULVIDC_IMGFORMAT        = 0x0301,    //Save the image format ID (DEC: 769)
TULVIDC_IMGCOUNT         = 0x0302,    //Save the number of pictures ID (DEC: 770)
TULVIDC_TRIGGERSAVE      = 0x0303,    //Trigger whether to save the image state ID (DEC:
771)
TULVIDC_CAMERASTATE      = 0x0305,    //Camera state (0, 1) ID (DEC: 773)
TULVIDC_FAN _            = 0x030 6 ,  //Camera fan gear ID       (DEC: 774)
TULVIDC_PGAHIGH          = 0x030B,    //Camera PGAHIGH ID (DEC: 779)
TULVIDC_PGALOW           = 0x0310,    //Camera PGALOW ID (DEC: 784)
TULVIDC_PIXCLK1_EN       = 0x0315,    //Camera PIXCLK1EN ID (DEC: 789)
TULVIDC_PIXCLK2_EN       = 0x031A,    //Camera PIXCLK2EN ID (DEC: 794)
TULVIDC_TRIOUT_EN        = 0x031B,    //Camera trigger output enable switch ID (DEC:
795)
TULVIDC_ROLLINGSCANMODE = 0x031C,     // camera light mode selection ID (DEC: 796)
TULVIDC_ROLLINGSCANLTD   = 0x031D,    //Camera light film mode line time delay ID (DEC:
797)
TULVIDC_ROLLINGSCANSLIT  = 0x031E,    // Camera light sheet mode slit height ID (DEC:
798)
TULVIDC_ROLLINGSCANDIR   = 0x031F,    //Camera light sheet mode scan direction ID (DEC:
799)
TULVIDC_ROLLINGSCANRESET= 0x0320,     //Camera light mode reset ID (DEC: 800)
TULVIDC_ROLLINGSCANINT   = 0x0321,    //Camera light film mode interval time ID (DEC:
801)
TULVIDC_TECENBALE        = 0x0322,    //Camera cooling switch ID (DEC: 802)
TULVIDC_CAMSTART         = 0x0323,    //Camera Start set the function ID of the number of
```

pictures taken (DEC: 803)

5.1.4. TULV_IDPROP Property ID: for TU_SetParameter.vi

```
TULVIDP_EXPTM          = 0x0100,   //Exposure time ID (DEC: 256)
TULVIDP_RGAIN          = 0x0105,   //Red channel (color camera)   ID (DEC: 261)
TULVIDP_GGAIN          = 0x010A,   //Greed channel (color camera) ID (DEC: 266)
TULVIDP_BGAIN          = 0x010F,   //Blue channel (color camera)  ID (DEC: 271)
TULVIDP_SATURATION     = 0x0114,   // Saturation (color camera)     ID (DEC: 276)
TULVIDP_GAMMA          = 0x0119,   // Gamma ID (DEC: 281)
TULVIDP_CONTRAST       = 0x011E,   //Contrast ID (DEC: 286)
TULVIDP_LFTLEVEL       = 0x0123,   //left level ID   (DEC: 291)
TULVIDP_RGTLEVEL       = 0x0128 ,  //Right level ID   (DEC: 296)
TULVIDP_GLOBGAIN       = 0x012D,   //Global gain ID   (DEC: 301)
TULVIDP_BLACKLEVEL     = 0x012E,   //Camera Offset ID (DEC: 302)
TULVIDP_GAINMODE       = 0x012F,   //Camera gain mode value ID (DEC: 303)
TULVIDP_SHARPNESS      = 0x0132,   //sharp ID   (DEC: 306)
TULVIDP_COLORTEMP      = 0x0137,   //Color temperature ID   (DEC: 311)
TULVIDP_ TRIKIND       = 0x013 C ,  // trigger mode ( 0: support sync and global, 1: do not
support sync global ) ID (DEC: 316)
TULVIDP_FRAMERATE      = 0x0141,   // Frame rate adjustable ID (DEC: 321)
TULVIDP_TEMPERATURE    = 0x0146,   // temperature get ID (DEC: 326)
```

5.1.5. TULV_CAPTURE_MODES capture mode ID: for TU_SetTrigger.vi

```
TULVCM_SEQUENCE              = 0x00,  //Use sequence mode (stream mode)
TULVCM_TRIGGER_STANDARD      = 0x01,  //Standard trigger mode
TULVCM_TRIGGER_SYNCHRONOUS   = 0x02,  //synchronous trigger mode
TULVCM_TRIGGER_GLOBAL        = 0x03,  // global trigger
TULVCM_TRIGGER_SOFTWARE      = 0x04,  //software trigger
```

5.1.6. TULVIMG_FORMATS image format ID: used to save the image

```
TULVFMT_RAW     = 0x01,    // RAW format
TULVFMT_TIF     = 0x02,    // TIFF format
TULVFMT_PNG     = 0x04,    // PNG format
TULVFMT_JPG     = 0x08,    // JPEG format
TULVFMT_BMP     = 0x10,    // BMP format
```

5.1.7. TULV_TRIGGER_EXP trigger exposure mode ID : for TU_SetTrigger.vi

```
TULVTE_EXPTM    = 0x00,    // trigger using exposure time mode
TULVTE_WIDTH    = 0x01,    // trigger using level width mode
```

5.1.8. TULV_TRIGGER_EDGE trigger trigger edge mode ID : for TU_SetTrigger.vi

```
TULVTD_RISING   = 0x01,    // trigger rising edge
```

TULVTD_FAILING = 0x00, // trigger falling edge

6. Introduction and application of VI

6.1 TU_InitCamera.vi

Input: error in
Output: error out, ReturnValue.

Function: This is the first VI module used before using all TUCSEN to control cameras, its function is to initialize all TUCSEN cameras that have been connected to the USB serial port and power supply. After calling this VI, an output value will be obtained. If the **ReturnValue** value is 1, the camera initialization is successful, and other VIs can be called after success. If the **ReturnValue** value is not 1, it indicates that the camera failed to initialize. For how to use this VI, please refer to the provided Demo. (Note: If the initialization module needs to be called repeatedly in a program, before the second call, TU_UninitCamera.vi must be called first to successfully call TU_InitCamera.vi).

Description of the output

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code .

6.2 TU_UninitCamera.vi

Input: error in, nCamCount.
Output: error out, ReturnValue.

Function: After calling this VI, the system will release all camera resources, and if the camera is still in the preview state, the camera will be forcibly closed . After the camera resource is released, TU_InitCamera.vi must be called to initialize the camera resource again. Whether the release of resources is successful can be judged according to the output value ReturnValue. If the value of ReturnValue is 1, it indicates that the release of camera resources is successful; if it is not 1, the release of camera resources fails.

Description of the input

nCamCount: The number of cameras that need to exit

Description of the output

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code .

6.3 TU_OpenCamera.vi

Input: error in, nCamCount
Output: error out, CameraCount, ReturnValue.

Function: Enable camera operation. After the camera resource is successfully initialized, TU_OpenCamera.vi will open multiple cameras according to the resource allocation during the initialization operation. The specific number of open cameras is controlled by the input terminal. Whether the operation of this step is successful or not can be judged according to the value of ReturnValue. If the value is 1, it means that the camera is successfully turned on, and if it is not 1, it fails to turn on. After the opening is successful, the value of CameraCount at the output end is the total number of cameras that have been successfully opened. Only when the camera is successfully turned on can you perform corresponding operations on the camera, such as parameter setting, parameter range acquisition, etc. Refer to the example on how to use this VI.

Description of the input

nCamCount: The number of cameras to be opened.

Description of the output

nCamCount: The number of cameras successfully opened.

nReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code.

6.4 TU_SetParameter.vi

Input: CameraIndex, Parameter, Value in, strInfo, error in.
Output: ReturnValue, Error out.

Function: Set camera parameters. Execute TU_SetParameter.vi to set a series of parameters for the

camera, including exposure time, Gamma value, gain value, contrast, etc. Whether the parameter is set successfully can be judged according to the ReturnValue value of the output terminal. If the ReturnValue value is 1, it means that the parameter setting is successful. If the ReturnValue value is not 1, it means that the parameter setting fails. The specific failure reason can be found according to the TULVRET error code corresponding to the ReturnValue value. Refer to the example SetParameterDemo.vi for how to use this VI .

Description of the input

CameraIndex: The serial number of the camera to be operated. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make a one-to-one correspondence between the camera serial number and the camera name, please refer to the example LabviewDemo.

Parameter: The ID corresponding to the parameter type to be set. For all parameters supported by the current version, please refer to TULV_IDCAPA and TULV_IDPROP. Additional parameters will be added gradually as needed.

Value in: The parameter value to be set. Refer to TU_GetParameterLimits.vi for the parameter range that can be set by the parameter type. For the meaning of the performance parameter values, please refer to the appendix.

strInfo: String parameters that need to be set, such as image storage address, etc.

Description of the output

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code .

6.5 TU_GetParameter.vi

Input: CameraIndex, ParameterID, error in.
Output: ParameterValue, ReturnValue, strInfo, error out.

Function: Get the parameter value. Execute TU_GetParameter.vi to obtain the current value of a series of parameters for the camera , including exposure time, Gamma value, gain value, contrast, etc.

Whether the parameter is successfully obtained can be judged according to the ReturnValue value of the output terminal . If the ReturnValue value is 1, it means that the parameter is successfully obtained. If the ReturnValue value is not 1, it means that the parameter acquisition fails. The specific failure reason can be determined according to the corresponding TULVRET error code of the ReturnValue value. Find.

How to use this VI can refer to the example Demo.

Description of the input

CameraIndex: The camera serial number corresponding to the parameter value needs to be obtained. The serial number starts at 0, where 0 is the first camera , 1 is the second, and so on. For how to make a one-to-one correspondence between the camera serial number and the camera name, please refer to the example LabviewDemo.

ParameterID: The parameter type ID to be obtained.

Description of the output

ParameterValue: The parameter value (numeric type) obtained according to the input parameter ID.

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code.

strInfo: The parameter value (string type) obtained according to the input parameter ID.

6.6 TU_GetParameterLimits.vi

Input: CameraIndex, ParameterID, error in.
Output: MaxValue, MinValue, StepValue, DefaultValue, ReturnValue, error out.

Function: Get the parameter value range. Including the maximum value, the minimum value, the default value, and the Step value. Whether the parameter range is successfully obtained can be judged according to the ReturnValue value of the output terminal. If the ReturnValue value is 1, it means that the parameter range is successfully obtained. If the ReturnValue value is not 1, it means that the parameter range is failed to be obtained. The specific failure reason can be based on the ReturnValue value corresponding to the TULVRET error code. to find. How to use this VI can refer to the example Demo.

Description of the input

CameraIndex: The camera serial number corresponding to the range of parameter values to be obtained. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make a one-to-one correspondence between the camera serial number and the camera name, please refer to the example LabviewDemo.

ParameterID: The parameter type ID to be obtained.

Description of the output

MaxValue: The maximum value corresponding to the input parameter type.

MinValue: The minimum value corresponding to the input parameter type.

StepValue: Enter the Step value corresponding to the parameter type.

DefaultValue: The default value corresponding to the input parameter type.

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code.

6.7 TU_GetCameraInfo.vi

Input: CameraIndex, TextIndex, TextID, error in.
Output: MaxValue, MinValue, StepValue, DefaultValue, ReturnValue, error out.

Function: Get camera related information. Including camera name, SN code (whether the camera supports it), firmware version, SDK version, USB interface type (2.0 or 3.0), and resolution name. Whether the information is obtained successfully can be judged according to the output value of ReturnValue. How to use this VI can refer to the example Demo.

Description of the input

CameraIndex: Need to obtain the camera serial number corresponding to the parameter information. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make the camera serial number correspond to the camera name one by one, please refer to the example Demo.

TextID: The ID corresponding to the camera information to be obtained (refer to TULV_IDINFO).

TextIndex: This input contains multiple pieces of camera information for one information ID. For example, the camera name is ISH130, and the camera has two resolutions. When the TextID is TULVIDI_RESOLUTION, to display all resolutions, you need to set the TextIndex to 0, respectively. 1. For details, please refer to the example Demo

Description of the output

: Camera-related information output according to TextID and TextIndex.

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code .

6.8 TU_SetTrigger.vi

Input: CameraIndex, CaptureMode, ExpourseMode, EdgeMode, DelayTime, Frame , nBufFrame, .
Output: ReturnValue.

Function: Trigger settings outside the camera (this function is only available for cameras that support external triggering, and the camera that does not support it will be invalid even if the module is called). For the introduction of external trigger, please refer to the document (3.4.1 Capture Mode) and (3.4.3 Trigger Mode). Refer to the example TriggerDemo.vi for how to use this VI.

Description of the input

CameraIndex: Need to set the camera serial number corresponding to the external trigger. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make the camera serial number correspond to the camera name, please refer to the example TriggerDemo.vi.

CaptureMode: This input contains 5 modes, namely TULVCM_SEQUENCE, TULVCM_TRIGGER_STANDERD,TULVCM_TRIGGER_SYNCHRONOUS,TULVCM_TRIGGER_GLOBAL, TULVCM_TRIGGER_SOFTWARE. For parameter description, please refer to 3.4.3 Trigger Mode Introduction.

ExpourseMode: This input contains two modes, TULVTE_EXPTM and TULVTE_WIDTH. For parameter description, please refer to 3.4.3 Trigger Mode Introduction.

EdgeMode: This input contains two modes, TULVTD_RISING and TULVTD_FAILING. For parameter description, please refer to 3.4.3 Trigger Mode Introduction.

DelayTime: When ExpourseMode selects TULV_TEPTM, the DelayTime value determines how long it takes to output the graph after receiving the trigger signal .

Frame: The total number of pictures to be drawn at one time (this parameter can not be set when using the trigger board to trigger).

nBufFrame : The number of buffered images in the SDK .

Description of the output

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code.

6.9 TU_GetTrigger.vi

Input: CameraIndex.
Output: ReturnValue.

Function: Get external trigger parameters and judge whether external trigger is supported. This VI determines whether the external trigger mode is supported based on the input camera serial number. Refer to the example LabviewDemo for how to use this VI.

(Note: TU_GetTrigger.vi can change this VI according to customer needs to obtain more output information, including current departure mode, exposure mode, delay time, and level trigger mode)

Description of the input

CameraIndex: Need to get the camera serial number corresponding to the external trigger value. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make the camera serial number correspond to the camera name, please refer to the example SetTriggerDemo.vi.

Description of the output

ReturnValue: A value of 1 indicates that the camera supports external trigger mode and the call is successful. If it is not 1, it indicates that the call fails and does not support external trigger. The specific failure reason is searched according to the TULVRET error code.

6.10 TU_SetROI.vi

Input: CameraIndex, StartX, StartY, Width, Height, SetROI.
Output: ReturnValue.

Function: Set ROI mode. This VI can select the desired area for data collection within the allowable range. Call example reference ROIDemo.vi

Description of the input

CameraIndex: The camera serial number corresponding to the ROI mode needs to be set. The serial number starts from 0, 0 represents the first camera, 1 represents the second camera, and so on. For how to make the camera serial number correspond to the camera name, please refer to the example ROIDemo.vi.

StartX: Horizontal offset.

StartY: Vertical offset.

Width: Horizontal width.

Heitht: Vertical height.

SetROI: Whether to set the ROI mode. 0, off, 1, on.

Description of the output

ReturnValue: A value of 1 indicates that the setting of the ROI mode is successful. If it is not 1, the setting of the ROI fails. The specific failure reason can be found according to the
TULVRET error code.

6.11 TU_StartCapture.vi

Input: CameraIndex, CaptureMode.
Output: CameraIndex, ImageType, Width, Height, Channel, Depth.

Function: After calling TU_InitCamera.vi and TU_OpenCamera.vi successfully, call TU_StartCapture.vi
to allocate the memory space, and start data capture after the allocation is successful. How to use this VI can refer to the example Demo.

Description of the input

CameraIndex: The serial number of the camera that needs to be captured.
CaptureMode: Data capture mode (streaming mode or trigger mode). For input type, see TULV_CAPTURE_MODE

Description of the output

CameraIndex: The serial number of the camera whose data was captured successfully **.**
ImageType: Output image type. Used in the IMAQ Create module. Please refer to the example Demo for usage details.

Width: Output image width.
Height: Output image height.
Channel: The number of output image channels (color image or black and white image).
Depth: Output image data bit width (8-bit or 16-bit).

6.12 TU_StopCapture.vi

Input: CameraIndex.
Output: RetuenValue.

Function: Stop data capture. The process of stopping data capture is performed according to the input camera serial number. Under the premise of TU_UninitCame, the data can be captured again by calling TU_StartCapture.vi. How to use this VI can refer to the example Demo.

Description of the input

CameraIndex: The serial number of the camera that needs to stop data capture.

Description of the output

ReturnValue: A value of 1 indicates that the data capture is stopped successfully. If the value is not 1, it fails to stop the data capture. The specific failure reason can be found according to the TULVRET error code.

6.13 TU_WaitForFrame.vi

Input: CameraIndex.
Output: nFrameIndex , dblTimeStamp, dblTimeLast, ReturnValue.

Function: Used to wait for the completion of data capture. For specific usage, please refer to Demo

Description of the input

CameraIndex: The serial number of the camera that needs to be processed.

Description of the output

ReturnValue: If the value is 1, the data capture is completed. If the value is not 1, the data capture fails.

The specific failure reason can be found according to the TULVRET error output.

nFrameIndex : Timestamp ID

d blTimeStamp : timestamp

dblTimeLast: the duration of the timestamp

6.14 TU_GetFrame.vi

Input: CameraIndex, ImageSRC_Average, ImageSRC_BKGD, ImageSRC_Normal, AverageCntIn, bSetBKGD, nFuncSelect.
Output: calBKGD, calAverage, Image_BKGD, Image_Average, Image_Normal.

Function: Real-time display of preview screen for obtaining frame data. For specific usage, please refer to the example Demo.

Description of the input

CameraIndex: The serial number of the camera that needs to obtain frame data.

ImageSRC_Average: Average input source image.

ImageSRC_BKGD: Background subtracted input source image.

ImageSRC_Normal: Normal input source image.

AverageCntIn: Find the average number of sheets (0-99).

BSetBKGD: Whether to subtract the background (0: No 1: Yes).

nFuncSelect: Function selection. (0: normal output 1: background subtraction 2: average)

Description of the output

calBKGD: Whether the calculation is done to subtract the background.

calAverage: Whether the calculation is completed for averaging.

Image_BKGD: Minus background output.

Image_Average: Average output.

Image_Normal: Normal output.

(Note: One GetFrame can only correspond to one output method)

6.15 TU_ErrorReport.vi

Functional perfection……

6.16 TU_OpenImageFile.vi

Input: NULL

Output: bCanceled, Image.

Function: Open the picture (used to reduce the background).

Description of the output

Image: The opened image (can be displayed directly).

bCanceled: Whether to open the picture (for the calculation when subtracting the background) (0: cancel 1: open the picture).

6.17 TU_StartRecorder.vi

Input: nCamIndex

Output: ReturnValue

Function description: Start recording video file

Description of the input

nCamIndex: The serial number of the camera that needs to acquire frame data.

Description of the output

ReturnValue: Whether to start recording video successfully (0: failure 1: success).

6.18 TU_StopRecorder.vi

Input: nCamIndex

Output: ReturnValue

Function description: Stop recording video files

Description of the input

nCamIndex: The serial number of the camera that needs to acquire frame data.

Description of the output

ReturnValue: Whether to stop recording video successfully (0: failure 1: success).

6.19 TU_Set OutPut Trigger.vi

Inputs: CameraIndex, OutPort , OutMode , EdgeMode, DelayTime, OutWidth .
Output: ReturnValue.

Function: Trigger output settings outside the camera (this function is only open for cameras that support external trigger output , and the camera that does not support it will be invalid even if the module is called). Refer to the example TriggerDemo.vi for how to use this VI.

Description of the input

CameraIndex: Need to set the camera serial number corresponding to the external trigger. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make the camera serial number correspond to the camera name, please refer to the example SetTriggerDemo.vi.

OutPort : The output port mode , which are Port1 , Port2 , Port3 port parameters for selection .

OutMode : This item is the trigger output mode. Different cameras have different trigger output modes. For details, please refer to the module examples of TriggerDemo.vi and LabviewSample.vi.

EdgeMode: This input contains two modes, TULVTD_RISING and TULVTD_FAILING.

DelayTime: When TULV_TEPTM is selected, the DelayTime value determines the delay in triggering the output signal .

OutWidth : The value determines the pulse width of the trigger output signal .

Description of the output

ReturnValue: If the value is 1, the call is successful. If it is not 1, the call fails. The specific failure reason is searched according to the TULVRET error code.

6.20 TU_Get OutPut Trigger.vi

Input: CameraIndex.
Output: ReturnValue.

Function: Get external trigger output parameters and judge whether external trigger output is supported . This VI determines whether the external trigger output mode is supported based on the input camera serial number. How to use this VI can refer to the example TriggerDemo.vi .

(Note: TU_Get OutPut Trigger.vi can change this VI according to customer needs to get more output information, including current output port , output mode, delay time, level trigger mode , pulse width time)

Description of the input

CameraIndex: Need to get the camera serial number corresponding to the external trigger value. The serial number starts at 0, where 0 is the first camera, 1 is the second, and so on. For how to make the camera serial number correspond to the camera name, please refer to the example SetTriggerDemo.vi.

Description of the output

ReturnValue: A value of 1 indicates that the camera supports the external trigger output mode and the call is successful. If it is not 1, the call fails and the external trigger output function is not supported . The specific failure reason is searched according to the TULVRET error code.

7. Appendix

Performance ID	Parameter value	Parameter meaning
TULVIDC_AUTOWB	0 , 1, 2	0: Disable auto white balance 1: Primary white balance 2: Auto white balance
TULCIDC_AUTOEXP	0 , 1, 2	0: Turn off auto exposure 1: Turn on auto exposure 2: One exposure
TULVIDC_DEPTH	0 , 1	0: 8-bit data width 1: 16-bit data width
TULVIDC_AUTOLEVEL	0 , 1, 2, 3	0: Disable auto level 1: Auto left level 2: Auto right level 3: Automatic left and right color levels
TULVIDC_RESOLUTION	0-n	Several resolutions depending on the camera, 0 is the first resolution, 1 is the second resolution, and so on
TULVIDC_NOISELEVEL	0 , 1, 2, 3	0: Off 1: Low 2: Medium 3: High
TULVIDC_IMGMODESELECT	0, 1 ,	0: Disable CMS 1: Enable CMS
TULVIDC_SAVEFRAME	n	To save pictures when taking pictures, how many pictures to save depends on the actual setting value
The TULVIDC_IMGSAVEPATH	string	To select the path to store the image
TULVIDC_IMGFORMAT	1, 2, 4, 8, 16	1: RAW 2: TIF 4: PNG

		8: JPG 16: BMP
TULVIDC_IMGCOUNT	0-n	Stores the number of pictures
TULVIDC_CAMERASTATE		This performance ID does not require specific parameter values, as long as TU_StartCapture.vi or TU_StopCapture.vi is called, the current camera state can be obtained by calling TU_GetParameter.vi. 0: Preview state 1: Suspend state

Attribute ID	**Parameter value range**
TULVIDP_EXPTM	MinValue-MaxValue
TULVIDP_RGAIN	MinValue-MaxValue
TULVIDP_GGAIN	MinValue-MaxValue
TULVIDP_BGAIN	MinValue-MaxValue
TULVIDP_SATURATION	MinValue-MaxValue
TULVIDP_GAMMA	MinValue-MaxValue
TULVIDP_CONTRAST	MinValue-MaxValue
TULVIDP_LFTLEVEL	MinValue-MaxValue
TULVIDP_RGTLEVEL	MinValue-MaxValue
TULVIDP_GLOBGAIN	MinValue-MaxValue
TULVIDP_SHARPNESS	MinValue-MaxValue
TULVIDP_COLORTEMP	MinValue-MaxValue

(Note: The precondition that the above parameters can be set is that the camera supports the above functions, and the MinValue and MaxValue values can be obtained by calling TU_GetParameterLimits.vi)