The LabVIEW Development Guide for Tucsen Cameras

Contents

1. Read before using

This document and software code sample are the internal files and published contents of Tucsen, which can enable users to create applications when using Tucsen digital cameras.

This document and software code sample are disclosed only for the above purposes, which shall not constitute the license, transfer or any other authority of owner.

All risks and consequences of using the software document still depend on users.

This document may contain technical inaccuracies or typographical errors. Further, there is no any guarantee for any damage caused by such mistakes or text.

Tucsen shall not commit to update or maintain the information contained in the current document.

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

Usage instructions for this version

1) Since the version is not perfect at present, you must copy Tucsen Camera folder in the compressed package and paste it to user.lib file under the LabVIEW installation path to be able to call the corresponding VI module if you want to use this version. For subsequent versions, the corresponding content will install in the form of packaging, which would omit the step of copying folder.

2) At present, the version may not be clear in the classification of some VI modules. We will carry out internal adjustments according to customers' feedback. Subsequent versions will make the classification more detailed to meet the needs of users.

2. Brief Introduction

This manual describes in detail the normative operation of the TULV_API when using TUCSEN digital cameras. The software development kit of TULV_API is called “SDK”. The part that T controls digital camera is called a [module].

The SDK includes VI module and a sample application, which demonstrates how to redevelop digital camera with VI module. SDK users use the software for free in any way that they like, such as partially modifying VI constructs or creating completely independent projects.

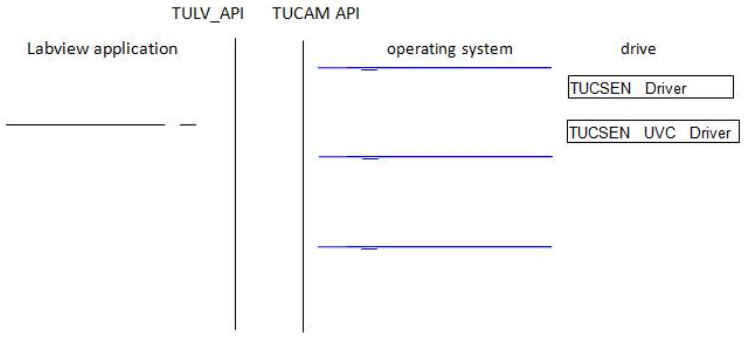
This SDK design is particularly easy to understand. For this case, the numbers of function interfaces are limited to a minimum, and the calling format of function is written in C language.

Partially extended functions are additional features that can be used by certain digital cameras.

The values of different digital cameras may vary, which depends on the type of digital camera used to capture the image. Value should be simply regarded as guidelines rather than exact values.

3. Summary

3.1. Layer structure



TUCSEN digital cameras connect drivers of digital cameras with different operating systems by the SDK to achieve the purposes of controlling the camera and capturing image data.

Current SDK is only supported in the Windows system.

3.2. Principles

The specific bus interface and library of digital camera are encapsulated by TULV_API. The interface calls of VI parts in each module are from the interface of T. You can refer to Demo provided by us to understand the usage of interface. Please refer to (6. VI introduction and application) for using it.

3.3. Interface type

TULV_API functions can be divided into many types:

Start/stop processing

Camera information collection

Performance/attribute acquisition and setup

Memory management

Capture control

Document control

Extended control

3.4. Terminology

3.4.1. Capture mode

The capture modes of cameras can be divided into the following two categories:

1) Sequential pattern (stream mode): It is used to capture continuous image data.
2) Trigger mode: The camera captures image with external signal. We call this option as “trigger mode”, and you can call TU_SetTrigger() to configure this option. We also call external signals as “external trigger”.

3.4.2. Image unit

Usually, it is two-dimensional, and it has vertical and horizontal directions.

Frame is a unit used for image data. For a frame, the data for one pixel is aligned from left to right and from top to bottom. This is a series of image data units.

3.4.3. Trigger mode

Standard mode: When the camera receives the level signal (determined by activation edge), it starts the image capture of one or more frames, the numbers of captured frames are determined by configuration parameters.

Synchronous mode: When camera receives the level signal (determined by activation edge), it begins to expose. When camera receives the opposite level signal, it ends the exposure and captures the image data. That is, the exposure and readout of each frame are completely synchronized with external trigger signal.

Global mode: Pre-trigger can be realized before camera is triggered. When the camera receives the level signal (determined by activation edge) or the exposure time is set by software, camera shall end the current reset operation, capture the image data at the end of exposure and restart the pre-trigger. This method is used to control the camera of shutter exposure mode for realizing the global exposure mode.

Exposure mode:

Exposure time: After receiving trigger signal, the exposure time is determined by TULVIDP_EXPTM setting.

Level width: After receiving trigger signal, the exposure time is determined by the width of level.

Note: These two options can be configured in standard mode and global mode. The

synchronization mode can only be suitable for the level width.

Excitation level type:

Rising Edge : The trigger level is on the rising edge when camera starts to expose.

Falling Edge : The trigger level is on the falling edge when camera starts to expose.

3.4.4. Camera state

The state of camera determines which functions can be called. Some functions change the state of camera. The following four kinds of camera states are described:

Unstable: You can carry out unstable parameter settings and other function calls, but they are not in the state of setting.

Stable: Stable parameters and functions are set, but the captured image cannot be started since no frame memory is created.

Ready: Preparation frame memory has been created, and image capture can be started.

Busy: Busy image capture is being executed.

4. Version introduction

4.1. Development environment

At present, only LabVIEW development under Windows system is supported. Developers can choose X86 or X64 installation package to develop and use according to actual conditions.

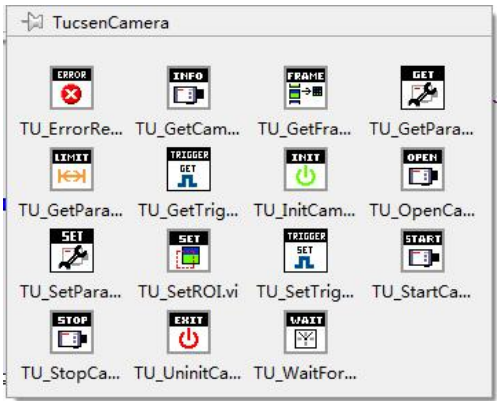
4.2. Performance optimization

4.2.1. Interface quantity and call optimization

In version 2.0, the interface functions have been optimized systematically. Without reducing the basic functions of version 1.0, interface functions are reduced in number, which are convenient for users to call.

4.2.2. VI module and call optimization

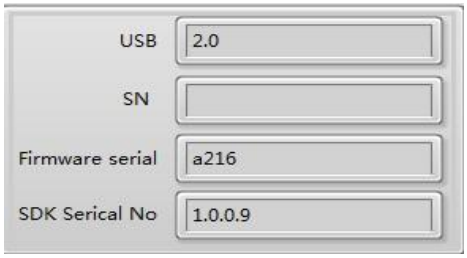
Version 2.0 splits single VI module in version 1.0 into several sub-VI modules and can be directly invoked in LabVIEW program block diagram, which facilitates the splitting research of sample program Demo for developers. VI module is described in detail in 6. VI introduction and application.



4.3. Function addition

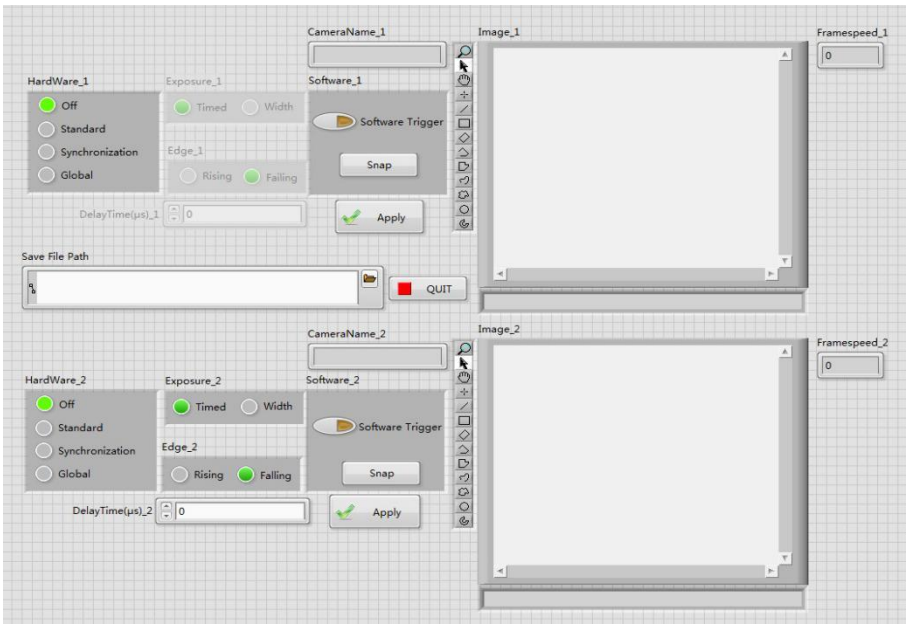
4.3.1. Display for the related information of camera

In order to accurately identify the camera, version 2.0 has been added with the display of camera name, USB type, SN code, SDK version and firmware version number. (Getting 0x210 is USB2.0, 0x03 is FireBird, 0x04 is Euresys, and the others are USB3.0. Cameralink does not display SN and firmware Numbers)



4.3.2. External trigger

According to the functions supported by camera, version 2.0 has been added with the external trigger module in the sample program TriggerDemo.VI. The trigger mode is described in detail (3.4.3 trigger mode). Please see TU_SetTrigger.VI for the use of trigger in detail.



4.3.3. The operation of multiple cameras

The LabVIEW sample program in version 2.0 can support multiple cameras to preview at the same time. Meanwhile, we can carry out parameter setting for this camera according to

camera name and change the preview effect to facilitate users to compare the effects of multiple cameras.

5. Reference

5.1. Types and constants

5.1.1. TULVRET error code: for TU_ErrorReport.vi

TULVRET_SUCCESS = 0x00000001, //Error-free, generally successful code

TULVRET_FAILURE = 0x80000000,//Error

//initialization error

TULVRET_NO_MEMORY = 0x80000101,//Not enough memory

TULVRET_NO_RESOURCE = 0x80000102,//Not enough resources (not including memory)

TULVRET_NO_MODULE = 0x80000103, //No submodules

TULVRET_NO_DRIVER = 0x80000104,//No drive

TULVRET_NO_CAMERA = 0x80000105,//No camera

TULVRET_NO_GRABBER = 0x80000106,//No picture taken

TULVRET_NO_PROPERTY = 0x80000107,//No substitute attribute ID

TULVRET_FAILOPEN_CAMERA = 0x80000110, //Failed to open camera

TULVRET_FAILOPEN_BULKIN = 0x80000111,//Failed to open the batch transport input endpoint

TULVRET_FAILOPEN_BULKOUT = 0x80000112,//Failed to open the batch transport output endpoint

TULVRET_FAILOPEN_CONTROL = 0x80000113,//Failed to open control endpoint

TULVRET_FAILCLOSE_CAMERA = 0x80000114,//Failed to close the camera

TULVRET_FAILOPEN_FILE = 0x80000115,//failed to open the file

// status error

TULVRET_INIT = 0x80000201,// API needs initialization status

TULVRET_BUSY = 0x80000202,// API is busy

```
TULVRET_NOT_INIT          = 0x80000203,//API not initialized
TULVRET_EXCLUDED          = 0x80000204,//Some resources are used exclusively
TULVRET_NOT_BUSY          = 0x80000205,//API is not busy
TULVRET_NOT_READY         = 0x80000206,//API is not ready

// wait error
TULVRET_ABORT             = 0x80000207, // Termination processing
TULVRET_TIMEOUT           = 0x80000208,//Time out
TULVRET_LOSTFRAME         = 0x80000209,//Frame loss
TULVRET_MISSFRAME         = 0x8000020A, //Frame loss 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 values
TULVRET_INVALID_IDPROP    = 0x80000304,//Invalid attribute ID
TULVRET_INVALID_IDCAPA    = 0x80000305,//Invalid performance 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 values
TULVRET_INVALID_EQUAL     = 0x8000030A,//The values are equal, but the
parameters are invalid
TULVRET_INVALID_CHANNEL   = 0x8000030B,//The attribute ID specifies the
channel, but the channel is invalid
TULVRET_INVALID_SUBARRAY  = 0x8000030C,//The value of the subarray is invalid

TULVRET_INVALID_VIEW      = 0x8000030D,//Invalid display window handle
TULVRET_INVALID_PATH      = 0x8000030E,//Invalid file path
```

TULVRET_NO_VALUETEXT = 0x80000310,//Attribute text that has no value

TULVRET_OUT_OF_RANGE = 0x80000311,//Value out of range

TULVRET_NOT_SUPPORT = 0x80000312,//Unsupported functionality or attributes

TULVRET_NOT_WRITABLE = 0x80000313,//Property unwritable

TULVRET_NOT_READABLE = 0x80000314,//Property unreadable

TULVRET_WRONG_HANDSHAKE = 0x80000410,//The error occurs when the error code is retrieved

TULVRET_NEWAPI_REQUIRED = 0x80000411,//The old API is not supported, only the new one

TULVRET_ACCESSDENY = 0x80000412,//When the camera is in a state property that 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 camera

TULVRET_OPTICS_UNPLUGGED = 0x83001003,//For insertion

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 (USB2.0/USB3.0) = 0x0004, //USB interface type

TULVIDI_SDKNO = 0x0005, //API version number

TULVIDI_RESOLUTION = 0x0006, //Resolution display

TULVIDI_PGAHIGH = 0x0007, //pga high

```
TULVIDI_PGALOW      = 0x0008,        //pga low
TULVIDI_IMGMODE     = 0x0009,        //Frame grabber mode
TULVIDI_FANMODE     = 0x000A,        //Fan mode
```

5.1.3. TULV_IDCAPA performance ID: for TU_SetParameter.vi

```
TULVIDC_AUTOWB          = 0x0200,    //Automatic white balance
TULCIDC_AUTOEXP         = 0x0205,    //Automatic exposure
TULVIDC_DEPTH           = 0x020A,    //Data width（8、16）
TULVIDC_AUTOLEVEL       = 0x020F,    //Automatic color scale
TULVIDC_RESOLUTION      = 0x0214,    //Resolution
TULVIDC_NOISELEVEL      = 0x0219,    //Noise reduction level
TULVIDC_IMGMODESELECT   = 0x021E,    //CMS
TULVIDC_IMGSAVEPATH     = 0x0223,    //Image save path
TULVIDC_LEDENBALE       = 0x0228,    //LED control
TULVIDC_SAVEFRAME       = 0x0300,    //Save the picture
TULVIDC_IMGFORMAT       = 0x0301,    //Save image format
TULVIDC_IMGCOUNT        = 0x0302,    //Save image number
TULVIDC_CAMERASTATE     = 0x0305,    //Camera stste
TULVIDC_FAN             = 0x0306,    //Fan grade
TULVIDC_PGAHIGH         = 0x030B,    //Camera PGAHIGH
TULVIDC_PGALOW          = 0x0310,    // Camera   PGALOW
TULVIDC_PIXCLK1_EN      = 0x0315,    // Camera   PIXCLK1EN
TULVIDC_PIXCLK2_EN      = 0x031A,    // Camera   PIXCLK2EN
```

5.1.4. TULV_IDPROP attribute ID: for TU_SetParameter.vi

```
TULVIDP_EXPTM       = 0x0100,        // Exposure time
TULVIDP_RGAIN       = 0x0105,        // Red channel (color camera)
TULVIDP_GGAIN       = 0x010A,        // Greed channel (color camera)
```

TULVIDP_BGAIN = 0x010F, // Blue channel (color camera)

TULVIDP_SATURATION = 0x0114, // Saturation (color camera)

TULVIDP_GAMMA = 0x0119, // Gamma

TULVIDP_CONTRAST = 0x011E, // Contrast

TULVIDP_LFTLEVEL = 0x0123, // Left color scale

TULVIDP_RGTLEVEL = 0x0128， // Right color scale

TULVIDP_GLOBGAIN = 0x012D, // Global gain

TULVIDP_SHARPNESS = 0x0132, //sharpen

TULVIDP_COLORTEMP = 0x0137, //Color temperature

TULVIDP_TRIKIND = 0x013C, //Trigger mode（0 supports synchronize and global, and 1 cannot support）

TULVIDP_FRAMERATE = 0x0141, // Frame adjustable

TULVIDP_TEMPERATURE = 0x0146, // Get temperature

5.1.5. TULV_CAPTURE_MODES capture mode ID: for TU_SetTrigger.vi

TULVCM_SEQUENCE = 0x00, // 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: for saving images

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 excitation edge mode ID: for TU_SetTrigger.vi

```
TULVTD_RISING       = 0x01,     // Stimulate rising edge
TULVTD_FAILING      = 0x00,     // Stimulate falling edg
```

6. Introduction and application of VI

6.1. TU_InitCamera.vi

Input: error in

Output: error out、ReturnValue

Function: This is the first used VI module before using the whole Tucsen control cameras. Its function is to initialize the whole Tucsen control cameras that have been connected to USB serial port and power supply. After the VI is called, an output value will be obtained. If ReturnValue value is 1, it indicates that the camera is successfully initialized. After succeeding, other VI can be called. If ReturnValue value is not 1, it shows that the initialization of camera fails. Please see example Demo for how to use this VI.

(**Note:** If the initialization module needs to be called repeatedly in a program, you must call TU_UninitCamera.vi before you can call TU_InitCamera.vi successfully before carrying out the second call).

Output description

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the call fails. The specific cause for failure can be found according to TULVRET error code.

6.2. TU_UninitCamera.vi

Input: error in、nCamCount

Output: error out、ReturnValue

Function: After calling the VI, the system releases all camera resources. If the camera is still in preview, the camera is forced to close. Once the camera resource is released, TU_InitCamera.vi must be called before initializing camera resource again. Whether releasing resources is successful or not, it can be judged according to output value R. If ReturnValue

value is 1, it indicates that the camera resource is successfully released. If not, the releasing of camera resource is unsuccessful.

Input Description

nCamCount: Number of cameras to exit.

Output Description

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the call fails. The specific cause for failure can be found according to TULVRET error code.

6.3. TU_OpenCamera.vi

Input: error in, nCamCount

Output: error out, CameraCount、ReturnValue

Function: Turn on camera operation. After the initialization of camera resources is successful, TU_OpenCamera.vi will turn on multiple cameras according to the allocation of resources during initialization operation. Whether the operation of this step is successful or not, it can be judged according to ReturnValue value. If the value is 1, it indicates that the camera is successfully turned on. If not, it shows that the opening fails.. After successful opening, the value of output CameraCount is the total value that opens camera successfully. Only when the camera is turned on successfully can the camera be operated accordingly, such as parameter setting, parameter range acquisition, etc. Please refer to the example Demo for how to use this VI.

Input Description

nCamCount: Number of cameras to open.

Output Description

nCamCount: Number of cameras successfully turned on.

NReturnValue: If the value is 1， it indicates that the call was successful, and the value is not

1, it indicates that the call failed. The specific reason for failure is found according to TULVRET error code.

6.4. TU_SetParameter.vi

Input: CameraIndex、Parameter、Value in、strInfo、error in

Output: ReturnValue、Error out

Function: Set camera parameters. Executing TU_SetParameter.vi can set a series of parameters for camera, which includes exposure time, Gamma value, gain value and contrast. (Specific parameters can be set in TULV_IDCAPA performance ID and TULV_IDPROP attribute ID). Whether the parameter is set successfully or not, it can be judged according to output ReturnValue value. If ReturnValue value is 1, it indicates that the parameter is set successfully. If ReturnValue is not 1, it shows that the parameter setting fails. The specific failure causes can be found according to corresponding TULVRET error code for ReturnValue value. Please refer to the example Demo for how to use this VI.

Input Description

CameraIndex: The serial number of camera to be operated. The serial number starts from 0. 0 represents the first camera, 1 represents the second camera, and so on. Please refer to the example Demo for how to make camera serial number corresponding to camera name one by one.

Parameter: The ID corresponding to parameter type that needs to be set. Please refer to TULV_IDCAPA and TULV_IDPROP for the whole parameters supported by the current version. You can add other parameters step by step according to the requirements.

Value in: The parameter value to be set. Please refer to TU_GetParameterLimits.vi for the parameter range that can be set by parameter type. The meaning of performance parameter values is detailed in the appendix.

StrInfo: string type parameter that needs to be set, such as image storage address, etc.

Output Description

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the

call fails. The specific cause for failure can be found according to TULVRET error code.

6.5. TU_GetParameter.vi

Input: CameraIndex、ParameterID、error in

Output: ParameterValue、strInfo、error out

Function: Obtain the parameter value. Executing TU_GetParameter.vi can carry out obtaining current values of a series of parameters, which include exposure time, Gamma value, gain value and contrast. (For specific parameters that can be set, see TULV_IDCAPA performance ID and TULV_IDPROP attribute ID). Whether the parameter is obtained successfully or not, it can be judged according to output ReturnValue. If ReturnValue value is 1, it indicates that the parameter is obtained successfully. If not, it shows that the obtaining of parameter fails. The specific failure causes can be found according to corresponding TULVRET error code for ReturnValue. Please refer to the example Demo for how to use this VI.

Input Description

CameraIndex: The serial number of camera of corresponding parameter value to be obtained. The serial number starts from 0. 0 represents the first camera, 1 represents the second camera, and so on. Please refer to the example LabviewSample.vi for how to make camera serial number corresponding to camera name one by one.

ParameterID: The parameter type ID that needs to be obtained.

Output Description

ParameterValue: The obtained parameter values based on input parameter ID.

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the call fails. The specific cause for failure can be found according to TULVRET error code.

strInfo: Get the parameter value (string type) from the input parameter ID.

6.6. TU_GetParameterLimits.vi

Input: CameraIndex、ParameterID、error in

Output: MaxValue、MinValue、StepValue、DefaultValue、ReturnValue、error out

Function: Obtain the range of parameter values. It includes maximum, minimum, default and Step values. Whether the parameter range is successful or not, it can be judged according to the ReturnValue value of output. If ReturnValue value is 1, it indicates that the parameter range is successfully obtained. If not, it shows that the parameter range fails to be obtained. The specific failure causes can be found according to corresponding TULVRET error code for ReturnValue. Please refer to the example Demo for how to use this VI.

Input Description

CameraIndex: The serial number of camera corresponding to parameter value range needs to be obtained. The serial number starts from 0. 0 represents the first camera, 1 represents the second camera, and so on. Please refer to the example LabviewSample.vi for how to make camera serial number corresponding to camera name one by one.

ParameterID: The parameter type ID needs to be obtained.

Output Description

MaxValue: Enter the maximum value corresponding to the parameter type.

MinValue: Enter the minimum value corresponding to the parameter type.

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

DefaultValue: Enter the default value corresponding to the parameter type.

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the call fails. The specific cause for failure can be found according to TULVRET error code.

6.7. TU_GetCameraInfo.vi

Input: CameraIndex、TextIndex、TextID、error in

Output: MaxValue、MinValue、StepValue、DefaultValue、ReturnValue、error out

Function: Obtain the related information of camera. It includes camera name, SN code (depending on 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. Please refer to the example Demo for how to use this VI.

Input Description

CameraIndex: The serial number of camera corresponding to parameter information that needs to be obtained. The serial number starts from 0. 0 represents the first camera, 1 represents the second camera, and so on. Please refer to the example Demo for how to make camera serial number corresponding to camera name one by one.

TextID: ID of corresponding camera information that needs to be obtained (Refer to TULV_IDINFO).

TextIndex: The input end contains multiple camera information for one information ID. For example, the camera name is ISH130, and the camera has two resolutions. When the Text ID is TULVIDI_RESOLUTION, it is necessary to respectively set TextIndex to 0 and 1 if you want to display all resolutions. Please refer to the example Demo in detail.

Output Description

CameraText: Output camera related information based on TextID and TextIndex.

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the call fails. The specific cause for failure can be found according to TULVRET error code.

6.8. TU_SetTrigger.vi

Input: CameraIndex、CaptureMode、ExpourseMode、EdgeMode、DelayTime、Frame

Output: ReturnValue

Function: External trigger settings for camera (This feature is only available for cameras that

support external triggers, and cameras that are not supported are invalid even if the module is called). The introduction of external triggers can be detailed in document (3.4.1 capture mode) and (3.4.3 trigger mode). Please refer to the example SetTriggerDemo.vi for how to use this VI.

Input Description

CameraIndex: The serial number of camera corresponding to external trigger that needs to be set. The serial number starts from 0. 0 represents the first camera, 1 represents the second camera, and so on. Please refer to the example SetTriggerDemo.vi for how to make camera serial number corresponding to camera name one by one.

CaptureMode: The input contains 5 modes, which are ①TULVCM_SEQUENCE、②TULVCM_TRIGGER_STANDERD、③TULVCM_TRIGGER_SYNCHRONOUS、④TULVCM_TRIGGER_GLOBAL、⑤TULVCM_TRIGGER_SOFTWAR, The parameter description can be referred to (3.4.3 trigger mode).

ExpourseMode: The input contains two modes, TULVTE_EXPTM and TULVTE_WIDTH. The parameter description can be referred to (3.4.3 trigger mode).

EdgeMode: The input contains two modes, TULVTD_RISING and TULVTD_FAILING. The parameter description can be referred to (3.4.3 trigger mode).

DelayTime: When ExpourseMode selects TULV_TEPTM, the DelayTime value determines how long the image will finish after receiving trigger signal.

Frame: The total number of images triggered at a time.

Output Description

ReturnValue: If the value is 1, it indicates that the call is successful. If not, it shows that the call fails. The specific cause for failure can be found according to TULVRET error code.

6.9. TU_GetTrigger.vi

Input: CameraIndex

Output: ReturnValue

Function: Obtain the external trigger parameter and determine whether the external trigger is supported. The VI determines whether the external trigger mode is supported based on the serial number of input camera. Please refer to the example Demo for how to use this VI.

(**Note:** TU_GetTrigger.vi can change the VI to get more output information according to the requirements of customers, which includes current departure mode, exposure mode, delay time and level trigger mode)

Input Description

CameraIndex: The serial number of camera corresponding to external trigger value that needs to be obtained. The serial number starts from 0. 0 represents the first camera, 1 represents the second camera, and so on. Please refer to the example SetTriggerDemo.vi for how to make camera serial number corresponding to camera name one by one.

Output Description

If the value is 1, it indicates that the camera supports external triggering mode and the call is successful. If not, it shows that the call fails and the external trigger is not supported. The specific failure causes can be found according to corresponding TULVRET error code.

6.10.TU_SetROI.vi

Input: CameraIndex、StartX、StartY、Width、Height、SetROI

Output: ReturnValue

Function: Set the ROI mode. The VI can select the desired area for data collection within the allowable range. Please refer to the example ROIDemo.vi for calling.

Input Description

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

corresponding to camera name one by one.

StartX: Horizontal offset

StartY: Vertical offset

Width: Horizontal width

Height: Vertical height

SetROI: Whether to set the ROI mode. 0: OFF, 1: ON

Output description

ReturnValue: If the value is 1, it indicates that the ROI mode was set successfully. If not, it shows that the setting of ROI fails. The specific failure causes can be found according to corresponding TULVRET error code.

6.11. TU_StartCapture.vi

Input: CameraIndex、CaptureMode

Output: CameraIndex、ImageType、Width、Height、Channel、Depth

Function: After TU_InitCamera.vi and TU_OpenCamera.vi are successfully called, TU_StartCapture.vi is called to allocate memory space. After the allocation is successful, the data capture can be started. Please refer to the example L for how to use this VI.

Input Description

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

CaptureMode: Data capture mode (stream mode or trigger mode). Please see TULV_CAPTURE_MODE for input types in detail.

Output Description

CameraIndex: The serial number of camera for which the data capture is successful.

ImageType: Output image type. It is used for IMAQ Create module. Please see LabviewSample.vi for usages in detail.

Width: The width of output image

Height: The height of output image

Channel: The number of channels of output image (Color image or monochrome image)

Depth: The Data width of output image (8bit or 16bit)

6.12.TU_StopCapture.vi

Input: CameraIndex

Output: RetuenValue

Function: Stop data capture. The process of stopping data capture is performed based on the input camera serial number. Under the premise of TU_UninitCame, data can be captured again by calling TU_StartCapture.vi. Please refer to the example LabviewSample.vi for how to use this VI.

Input Description

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

Output Description

ReturnValue: If the value is 1, it indicates that the stop data capture is successful. If not, the stop data capture fails. The specific failure causes can be found according to corresponding TULVRET error code.

6.13.TU_WaitForFrame.vi

Input: CameraIndex

Output: ReturnValue

Function: It is used to wait for the completion of data capture. The specific usage methods

can refer to Demo.

Input Description

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

Output Description

ReturnValue: If the value is 1, it indicates that the data capture is successful. If not, the data capture fails. The specific failure causes can be found according to corresponding TULVRET error code.

6.14.TU_GetFrame.vi

Input: ameraIndex, ImageSRC_Average, ImageSRC_BKGD, ImageSRC_Normal, AverageCntIn, bSetBKGD, nFuncSelect

Output: calBKGD, calAverage, Image_BKGD, Image_Average, Image_Normal

Function: Obtain real-time display for the preview image of frame data. Please refer to Demo - The specific usage methods.

Input Description

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

ImageSRC_Average: Average input source image.

ImageSRC_BKGD: Subtract the background input source image.

ImageSRC_Normal: Normally enters the source image.

AverageCntIn: Average number of sheets (0-99).

BSetBKGD: Whether to subtract background (0: no, 1: yes).

NFuncSelect: functionally selected. (0: normal output, 1: subtract background, 2: average.)

Output Description

CalBKGD: whether background subtraction is completed.

CalAverage: whether the calculation is completed to find the average.

Image_BKGD: subtract background output.

Image_Average: average output.

Image_Normal: normal output.

(Note: there can only be one output mode in a GetFrame.)

6.15.TU_ErrorReport.vi

Functional perfection……

6.16.TU_ OpenImageFile.vi

Input: NULL

Output: bCanceled、Image

Function: Open the image.

Output Description

Image: an open Image that can be displayed directly.

bCanceled: whether to open the image (for the calculation to subtract the background) (0: cancel, 1: open the picture).

6.17.TU_StartRecorder.vi

Input: nCamIndex

Output: ReturnValue

Function: start recording the video.

Input Description

NCamIndex: camera serial number needed to get frame data.

Output Description

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

6.18.TU_ StopRecorder.vi

Input: nCamIndex

Output: ReturnValue

Function: stop recording the video.

Input Description

NCamIndex: camera serial number needed to get frame data.

Output Description

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

6.19.TU_SetOutPutTrigger.vi

Input: CameraIndex, OutPort, OutMode, EdgeMode, DelayTime, OutWidth

Output: ReturnValue

Function: Camera external trigger output Settings (this feature is only open for cameras that support external trigger output, and the module is not valid for cameras that do not support it). See the example Triggerdemo.vi for how to use this VI.

Input Description

CameraIndex: The camera serial number corresponding to the external trigger needs to be set.The serial number starts at 0, 0 for the first camera, 1 for the second camera, and so on.See the example setTriggerdemo.vi for how to match the camera sequence number to the camera name.

OutPort: The output port mode is Port1, Port2 and Port3.

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 of the output signal.

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

Output Description

ReturnValue: When the value is 1, it indicates that the call is successful, and if it is not 1, it indicates that the call failed. The specific cause of the failure is found according to the TULVRET error code.

6.20.TU_GetOutPutTrigger.vi

Input: CameraIndex

Output: ReturnValue

Function: Gets external trigger output parameters and determines whether external trigger output is supported. The VI determines whether external trigger output mode is supported based on the input camera serial number. See the example Triggerdemo.vi for how to use this VI.

Input Description

CameraIndex: The camera serial number corresponding to the external trigger needs to be set.The serial number starts at 0, 0 for the first camera, 1 for the second camera, and so on.See the example setTriggerdemo.vi for how to match the camera sequence number to the camera name.

Output Description

ReturnValue: When the value is 1, it indicates that the call is successful, and if it is not 1, it indicates that the call failed. The specific cause of the failure is found according to the TULVRET error code.

7. Appendix

Performance ID	Set parameter values	Parameter meaning
TULVIDC_AUTOWB	0, 1, 2	0: Turn off automatic white balance 1: One white balance 2: Automatic 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: Turn off automatic color scale 1: Automatic left color scale 2: Automatic right color scale 3: Automatic left and right color scale
TULVIDC_RESOLUTION	0-n	There are several resolutions depending on 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: Hight
TULVIDC_IMGMODESELECT	0, 1,	0: Turn off CMS 1: Turn on CMS
TULVIDC_SAVEFRAME	n	It is used to save pictures when taking pictures. How many of them are stored according to the actual settings.
TULVIDC_IMGSAVEPATH	String	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	The number of stored images
TULVIDC_CAMERAS		This performance ID does not require a

TATE		specific setting of 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: Suspended state

Attribute ID	**Set parameter values**
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 for the above parameters is that camera supports the above functions, and both Min Value and Max Value values can be obtained by calling TU_Get Parameter Limits.vi)