



Testo LabVIEW™ Library

for Testo IR cameras

Testo 885

Testo 890

Version 1.0

11.11.2015

LabVIEW™ software was developed by National Instruments

Contents

VERSION 1.0 (11.11.2015)	3
RANGE OF FUNCTIONS IN THE TESTO LABVIEW LIBRARY	4
SUPPORTED INFRARED CAMERAS	4
SUPPORTED OPERATING SYSTEMS	5
SUPPORTED LABVIEW VERSIONS	5
REQUIRED SOFTWARE	5
INSTALLING THE TESTO LABVIEW LIBRARY	5
DOCUMENTATION OF FUNCTION CALLS	5
FUNCTIONS CONTAINED IN THE TESTO LABVIEW LIBRARY	6
SUBVI_GETCAMDEVICES.VI	6
SUBVI_OPENCAM.VI	6
SUBVI_GETEMISSION.VI	6
SUBVI_SETEMISSION.VI	6
SUBVI_GETHUMIDITY.VI	7
SUBVI_SETHUMIDITY.VI	7
SUBVI_GETNUMMEASRANGES.VI	7
SUBVI_GETMEASUREMENTRANGE.VI	7
SUBVI_SETMEASUREMENTRANGE.VI	8
SUBVI_GETREFLECTEDTEMPERATURE.VI	8
SUBVI_SETREFLECTEDTEMPERATURE.VI	8
SUBVI_GETATMOSPHERECORRECTION.VI	8
SUBVI_SETATMOSPHERECORRECTION.VI	8
SUBVI_CAPTUREIR.VI	9
SUBVI_CAPTUREVIS.VI	11
SUBVI_GET2DTEMPERATUREARRAY.VI	11
SUBVI_RGBCONVERTTO1DARRAY.VI	12
SUBVI_STOPSTREAMIR.VI	12
SUBVI_STOPSTREAMVIS.VI	12
DEMOS	13
1. MAIN IRVISION	13

<i>Start group</i>	13
<i>Status Indicator group</i>	13
<i>Parameter group</i>	14
<i>IR Image group</i>	14
2. VISIMAGE.....	15
<i>Connection status</i>	15
<i>Start group</i>	15
<i>VIS Image group</i>	15
3. 3DSURFACE	16
<i>Connection status</i>	16
<i>Start group</i>	16
<i>3D Mesh</i>	16
<i>Temperature Array</i>	16
4. GETSETPARAMETER	17
<i>Connection status</i>	17
<i>Start group</i>	17
IMPORTANT INFORMATION	18
COMMUNICATION WITH MULTIPLE CAMERAS	18



Version 1.0 (11.11.2015)

FIRST VERSION

Range of functions in the Testo LabVIEW library

The library lets the LabVIEW programs communicate with the supported Testo infrared cameras t890 and t885. It allows for the reading and setting of parameters. It also enables reading of the VIS or IR image and the corresponding temperature values. To use the library, you must have a LabVIEW developer environment installed on your computer.

The package includes the following:

- Testo Toolbox for supported devices.
- VI files containing functions for working directly in LabVIEW.
- A VI file with a GUI that works with the entire range of functions contained in the library in order to access the camera and to read or display an IR image.
- Three demo VI files showing examples of using the various VI functions.
- This manual describing the range of functions contained in the LabVIEW library.

Supported infrared cameras

The following IR cameras are supported:

- t885
- t890

The IR cameras are connected to the computer via USB port. In order to control the cameras, you must have the proper USB drivers installed.



Supported operating systems

The following operating systems are supported:

- Microsoft Windows 7 (SP1 and higher), 32- and 64-bit
- Microsoft Windows 8, 8.1, 32- and 64-bit

Supported LabVIEW versions

The following 32-bit LabVIEW versions are supported:

- LabVIEW 2011 SP1
- LabVIEW 2012 SP1
- LabVIEW 2013 SP1
- LabVIEW 2014 SP1

Required software

Microsoft Visual Studio 2013 or Microsoft Visual C++ Redistributable Packages for Visual Studio 2013 (<https://www.microsoft.com/de-de/download/details.aspx?id=40784>)

testo IIRSoft 3.8 or higher
(<http://www.testo.com/iirsoft>)

Installing the Testo LabVIEW library

Installation is performed using the included Setup program.


Documentation of function calls

The calls in the Testo LabVIEW library are found in "LabVIEW_cwrapper_1_0_0.dll" and implemented as C interfaces. For each function, there exists a .vi file that calls the relevant function.

Functions contained in the Testo LabVIEW library


SubVI_GetCamDevices.vi

This function checks what kind of cameras are connected to the computer and returns the number of cameras, an array with their serial numbers, and an array with a description of the cameras. Or it returns an error code. As shown in the demos, these values can be used to establish a connection to a particular camera.

Symbol	Parameter	Type	Description
	u32NumberOfCameras [out]	U32	Number of connected cameras
	au32CameraSerials [out]	U32 1D array	Array containing serial numbers of the connected cameras
	strCameraDeviceType [out]	string 1D Array	Array with the description of the cameras
	U8Error [out]	U8	Error code 0: Success 1: Error 2: no cameras connected


SubVI_OpenCam.vi

This function opens a connection to the camera defined by the relevant serial number. The connection is automatically interrupted when another camera is connected or when the program is terminated.

Symbol	Parameter	Type	Description
	u32SerialNumber [in]	U32	Serial number of the camera with which to establish a connection
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_GetEmissivity.vi

This function reads the emissivity value currently set in the camera.

Symbol	Parameter	Type	Description
	fEmissivity [out]	SGL	Currently set emissivity value
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_SetEmissivity.vi

This function sets the camera's emissivity value to the value passed by the function.

Symbol	Parameter	Type	Description
	fEmissivity [in]	SGL	Desired emissivity value
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_GetHumidity.vi

This function reads the humidity value currently set in the camera.

Symbol	Parameter	Type	Description
	fHumidity [out]	SGL	Current humidity value
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_SetHumidity.vi

This function sets the humidity value to the value passed by the function.

Symbol	Parameter	Type	Description
	fHumidity [in]	SGL	Desired humidity value
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_GetNumMeasRanges.vi

This function determines the number of possible measurement ranges. In order to select a particular measurement range, use the function **SetMeasurementRange.vi**, which expects an input value of measurement range index < U32NumOfMeasRanges-1.

Symbol	Parameter	Type	Description
	U32NumMeasRanges [out]	U32	Number of possible measurement ranges
	U8Error [out]	U8	Error code 0: Success 1: Error

SubVI_GetMeasurementRange.vi


This function reads the index of the currently used measurement range.

Symbol	Parameter	Type	Description
	u32MeasRangeIndex [out]	U32	Index of the currently used measurement range
	U8Error [out]	U8	Error code 0: Success 1: Error

SubVI_SetMeasurementRange.vi


This function selects the measurement range to be used. This is done by passing the index of the measurement range. The number of measurement ranges and which can be selected is determined by the function **SubVI_GetNrOfMeasRanges.vi**.

This function expects an input value of measurement range index < U32NumOfMeasRanges-1.

Symbol	Parameter	Type	Description
	U32MeasRangeIndex [in]	SGL	Index of measurement range to be used.
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_GetReflectedTemperature.vi

This function reads the reflected temperature value currently set in the camera.

Symbol	Parameter	Type	Description
	fReflectedTemp [out]	SGL	Currently set reflected temperature value
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_SetReflectedTemperature.vi

This function sets the reflected temperature value to the value passed by the function.

Symbol	Parameter	Type	Description
	fReflectedTemp [in]	SGL	Desired reflected temperature value
	U8Error [out]	U8	Error code 0: Success 1: Error


SubVI_GetAtmosphereCorrection.vi

This function reads the currently set atmospheric correction status.

Symbol	Parameter	Type	Description
	bAtmosCorrEnable [out]	Boolean	Currently used status of the atmospheric correction value: TRUE – on; FALSE – off
	U8Error [out]	U8	Error code 0: Success 1: Error

SubVI_SetAtmosphereCorrection.vi

This function sets the status of the atmospheric correction.

Symbol	Parameter	Type	Description
	u8AtmosCorrEnable [in]	U8	Status of the atmospheric correction: 1 to turn on, 0 to turn off
	U8Error [out]	U8	Error code 0: Success 1: Error

SubVI_CaptureIrr.vi

This function reads the IR images recorded by the camera. To this end, three pointers to allocated *U8 1D arrays* are passed to the interface. These arrays are used to store the IR image's RGB values. Using the RGB converter, this information can be used to display the IR image (see *ExampleVI_3DSurface.vi*). The color palette is chosen via the parameter *u8PaletteType*, which corresponds to the index of the chosen color palette:


Index Description of color palette

- 1 Iron
- 2 Rainbow
- 3 BlueRed
- 4 Testo
- 5 Sepia
- 6 Dewpoint
- 7 RainbowHC

The choice of color palette relates only to the image's display in LabVIEW and has no influence on how it is displayed in the camera.


In addition, the function is passed an *SGL 1D array* in which the temperature values measured in the image are stored. The temperature values may be read via an *SGL 2D array* at the output.

NOTE: This VI **cannot** be executed at the same time as *SubVI_Get2DTemperatureArray.vi*.

Symbol	Parameter	Type	Description
	au8ChannelB [in]	U8 1D array	Allocated 1D array for the IR image's B-channel
	au 8ChannelG [in]	U8 1D array	Allocated 1D array for the IR image's G-channel
	au8ChannelR [in]	U8 1D array	Allocated 1D array for the IR image's R-channel
	af1DTempMat [in]	SGL 1D array	Allocated 1D buffer array for the IR image's temperature values
	u8PaletteType [in]	U8	Index of the chosen color palette that is to be used to calculate the IR image. This influences only how the image is displayed in LabVIEW, not how it is displayed in the camera.
	au8ChannelB [out]	U8 1D array	1D array with the values for the IR image's B-channel
	au8ChannelG [out]	U8 1D array	1D array with the values for the IR image's G-channel
	au8ChannelR [out]	U8 1D array	1D array with the values for the IR image's R-channel
	af2DTempMat [out]	SGL 2D array	2D array with the temperature values for each pixel of the IR image
	i32Height [out]	I32	Height of IR image
	i32Width [out]	I32	Width of IR image
	U8Error [out]	U8	Error code 0: Success 1: Error

SubVI_CaptureVIS.vi


This function reads the VIS images recorded by the camera. To this end, pointers to three allocated *U8 1D arrays* are passed to the interface. These arrays are used to store the VIS image's RGB values. Using the RGB converter, this information can be used to display the VIS image (see **ExampleVI_VISImage.vi**).

Symbol	Parameter	Type	Description
	au8ChannelB [in]	U8 1D array	Allocated 1D array for the VIS image's B-channel
	au8ChannelG [in]	U8 1D array	Allocated 1D array for the VIS image's G-channel
	au8ChannelR [in]	U8 1D array	Allocated 1D array for the VIS image's R-channel
	au8ChannelB [out]	U8 1D array	1D array for the VIS image's B-channel
	au8ChannelG [out]	U8 1D array	1D array for the VIS image's G-channel
	au8ChannelR [out]	U8 1D array	1D array for the VIS image's R-channel
	i32VisHeight [out]	I32	Height of VIS image
	i32VisWidth [out]	I32	Width of VIS image
	u8Error [out]	U8	Error code 0: Success 1: Error

SubVI_Get2DTemperatureArray.vi


This function reads the 2D temperature array recorded by the camera. To this end, an *SGL 1D array* for storing the values is passed to the interface. The temperature values may be read as a 2D array at the output. In addition, the function returns the current number of rows and columns in the 2D temperature array.

NOTE: This VI **cannot** be executed at the same time as **SubVI_CaptureIRImage.vi**. If you wish to read the IR image as well as the temperature array, you must use **SubVI_CaptureIRImage.vi**.

Symbol	Parameter	Type	Description
	af1DTempMat [in]	SGL 1D array	Allocated 1D buffer array for the temperature values.
	af2DTempMat [out]	SGL 2D array	2D array containing the current image's temperature values
	i32Height [out]	I32	Number of rows in the 2D temperature array
	i32Width [out]	I32	Number of columns in the 2D temperature array
	u8Error [out]	U8	Error code 0: Success 1: Error

SubVI_RGBConvertTo1DArray.vi


This function uses the individual 1D arrays (R, G, B) to create a single 1D RGB array that is used to display the recorded images. The use of this function can be seen in the demos.

Symbol	Parameter	Type	Description
	au8ChannelB [in]	U8 1D array	1D array containing the image's B-channel
	au8ChannelG [in]	U8 1D array	1D array containing the image's G-channel
	au8ChannelR [in]	U8 1D array	1D array containing the image's R-channel
	au8SerializedImage [in]	U8 1D array	Allocated 1D array for 1D RGB image to be converted
	i32PixelAmount [in]	I32	Total number of pixels in the image, width times height
	au8SerializedImage [out]	U8 1D array	1D Array with converted RGB image

SubVI_StopStreamIR.vi

This function stops the streaming of IR images. In combination with **CaptureIRImage.vi**, it makes it possible to take individual pictures without the delays caused by the caching of previous images. The use of this VI can be seen in **Example_3DSurface.vi**.


NOTE: Use this function **only** when recording individual images.

Symbol	Parameter	Type	Description
	U8Error [out]	U8	Error code 0: Success 1: Error

SubVI_StopStreamVIS.vi

This function stops the streaming of IR images. In combination with **CaptureVISImage.vi**, it makes it possible to take individual pictures without the delays caused by the caching of previous images. The use of this VI can be seen in **Example_VISImage.vi**.

NOTE: Use this function **only** when recording individual images.

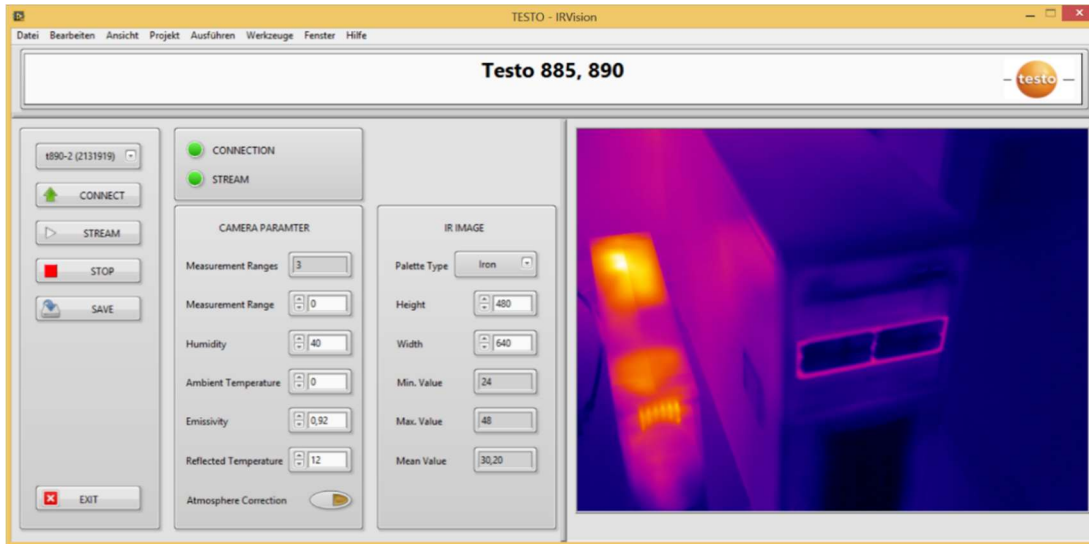
Symbol	Parameter	Type	Description
	U8Error [out]	U8	Error code 0: Success 1: Error

Demos

The demos on the following pages are intended more for developing your own VIs and to facilitate the learning process.

1. Main IRVision

This demo shows almost all possibilities provided by the LabVIEW library. In this way, you can test how the camera can be controlled in LabVIEW.



Start group

Camera selection

The combo box shows all connected cameras, including their serial numbers and descriptions. This field determines which camera is to be used. At startup, the program selects the first camera on the list.

CONNECT – establish connection

Opens a connection to the selected camera.

STREAM – transfer/playback

Starts the IR images' real-time stream.

STOP – end stream

Click this button to end the stream. The display area shows the last image to be transferred.

SAVE – save image

Saves the current IR image as a JPG file.

EXIT – exit mode

Ends the application.

Status Indicator group

- CONNECTION
Shows the connection status. The indicator is illuminated green if a connection exists.
- STREAM
Shows the stream status. The indicator is illuminated green if streaming is active.

Parameter group

Shows those camera parameters that can be changed. At startup, the program reads the camera's current parameter values. For more information on the individual parameters and their possible settings, see your camera manual.

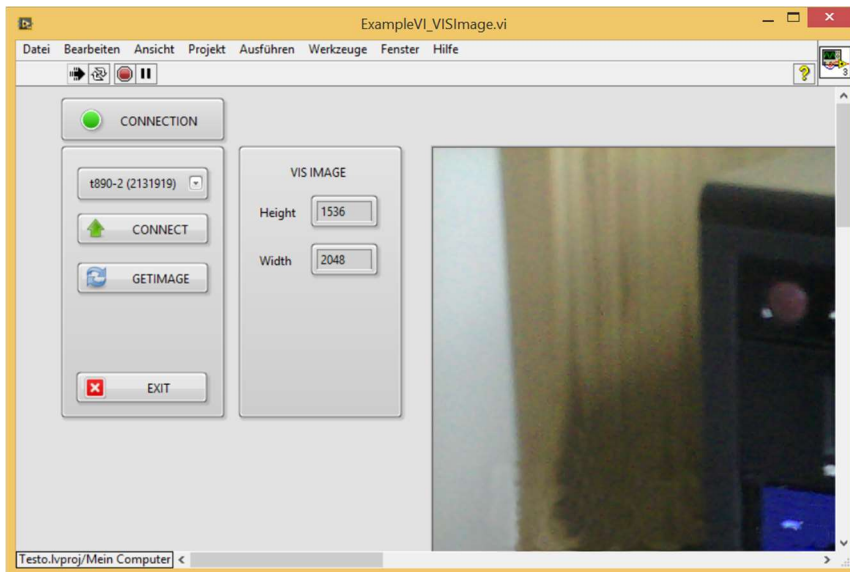
Note: Some parameters require time for the changes to be enacted. No further IR images can be read during this time, and the camera's operation is limited as well.

IR Image group

This group contains the parameters of the IR image. The combo box is used to select the color palette. For more information on the individual color palettes, see your camera manual.

2. VISImage

This demo shows how to retrieve and display VIS images.



Connection status

The indicator is illuminated light green if a connection exists.

Start group

Camera selection

The combo box shows all connected cameras, including their serial numbers and descriptions. This field determines which camera is to be used. At startup, the program selects the first camera on the list.

CONNECT – establish connection

Opens a connection to the selected camera.

GETIMAGE

Reads the data of the current VIS image and displays it in the display area.

EXIT – exit mode

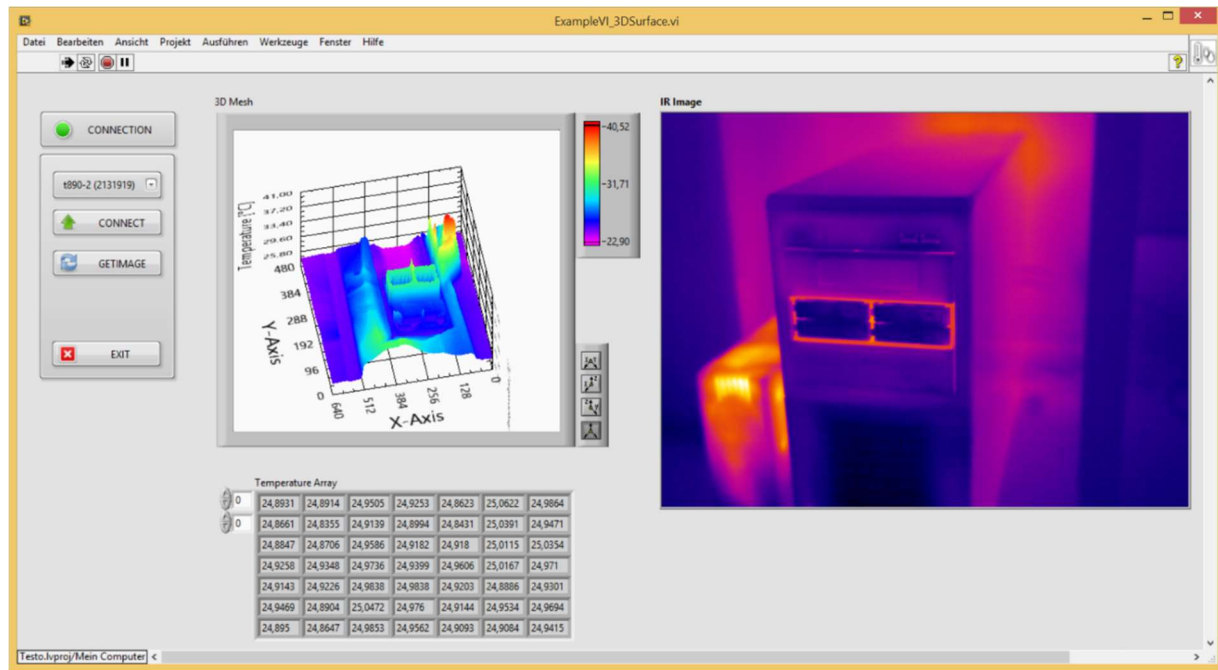
Ends the application.

VIS Image group

Shows the VIS image's dimensions in pixels.

3. 3DSurface

This demo shows how LabVIEW functions like 3D Mesh are used to work with the temperature matrix and its corresponding IR image retrieved from the camera.



Connection status

The indicator is illuminated light green if a connection exists.

Start group

Camera selection

The combo box shows all connected cameras, including their serial numbers and descriptions. This field determines which camera is to be used. At startup, the program selects the first camera on the list.

CONNECT – establish connection

Opens a connection to the selected camera.

GETIMAGE

Reads the data of the current IR image and displays it in the display area.

EXIT – exit mode

Ends the application.

3D Mesh

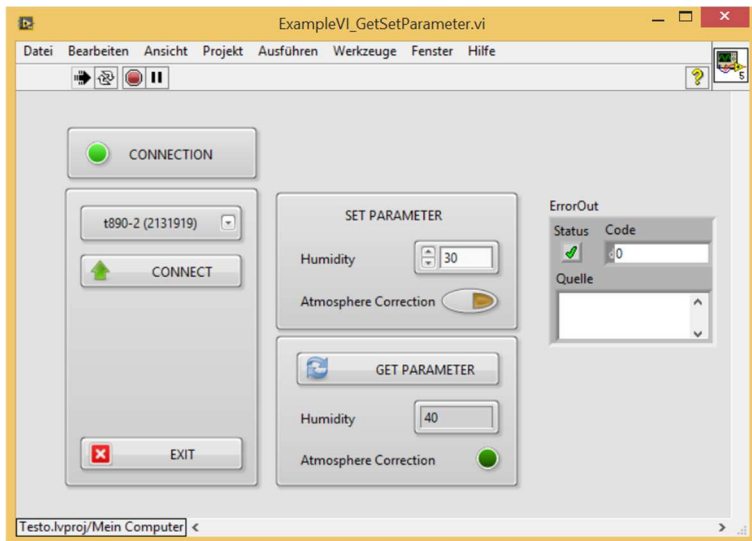
LabVIEW graphs are used to show the measured temperature values as 3D plot. The Z axis shows the temperature values. The higher a measured value, the higher the amplitude displayed in the graph.

Temperature Array

The matrix displays a part of the retrieved temperature matrix. In the upper left corner, (x, y) indices can be chosen to get the IR image's corresponding temperature values at a certain position.

4. GetSetParameter

This simple demo shows how to retrieve and set the camera's parameters.



Connection status

The indicator is illuminated light green if a connection exists.

Start group

Camera selection

The combo box shows all connected cameras, including their serial numbers and descriptions. This field determines which camera is to be used. At startup, the program selects the first camera on the list.

CONNECT – establish connection

Opens a connection to the selected camera.

EXIT – exit mode

Ends the application.



Important information

Communication with multiple cameras

When using the included VIs, always make sure that you are communicating with only one camera. Parallel communication with multiple cameras is not supported.