ImageSourcePy Documentation

Release 1.0

A. Parenti

Apr 29, 2025

Contents

Contents:

CHAPTER 1

The ImageSource Utils

imageSource.utils contains functions which are of common use for image source devices. This is the list of the currently provided ones:

imageSourcePy.utils.**unpack_mono12_packed**()

Unpack the input MONO12PACKED data to MONO12.

In MONO12PACKED pixel data format, every 3 bytes contain data for 2 pixels, according to the following table:

Byte	Pixel - Data bits
B0	P0 11…4
B1	P1 3…0 | P0 3…0
B2	P1 11…4
…	…
Bm	Pn 11…4

Parameters

- **in_buffer** – the input packed data
- **lock** – an optional lock

Returns the output unpacked data

imageSourcePy.utils.**unpack_mono_xx_p**()

Unpack the input MonoXXp data to MONO12, where XX is usually 10 or 12.

Use with caution! The function can be quite slow on large images.

In MONOXXp pixel data format, XX-bit pixel data are packed, with no padding bits in between. Padding 0s are added to the MSB if needed. For example Mono10p pixels are packed this way:

Byte	Pixel - Data bits
B0	P0 7…0
B1	P1 5…0 | P0 9…8
B2	P2 3…0 | P1 9…6
B3	P4 1…0 | P3 9…4
…	…

Parameters

- **data** – the input packed data
- **bpp** – the bits-per-pixel, normally 10 or 12
- **lock** – an optional lock

Returns the output unpacked data

imageSourcePy.utils.**unpack_mono10_p**()
Specialization of the 'unpack_mono_xx_p' function for the MONO10p format

imageSourcePy.utils.**unpack_mono12_p**()
Specialization of the 'unpack_mono_xx_p' function for the MONO12p format

CHAPTER 2

The ImageSource class

The *ImageSource* class provides a base class to be used for all kind of image providing devices, for example image processors.

For the cameras a more specific class is provided, see *here*.

The advantage of using this class, is that it can take care of creating the necessary output channels in the schema, and it provides functions to output images and End-of-Stream signals to them.

The class to be used in Python-bound devices is this one

class imageSourcePy.ImageSource.**ImageSource**(*conf*)

Base class for image sources.

It provides two output channels - 'output' and 'daqOutput' - for sending out images, and three functions - 'update_output_schema', 'write_channels' and 'signal_eos'.

The function 'update_output_schema' will update the schema for the output channels and make it fit for the DAQ.

The function 'write_channels' will write the input data to both the output channels, taking care of reshaping them for the DAQ.

The function 'signal_eos' will send an end-of-stream signal to both the output channels.

signal_eos()

Send an end-of-stream signal to 'output' and 'daqOutput' channels

Returns

update_output_schema(*shape*, *encoding*, *k_type*)

Update the schema of 'output' and 'daqOutput' channels

Parameters

- **shape** – the shape of image, e.g. (height, width)
- **encoding** – the encoding of the image. e.g. Encoding.GRAY
- **k_type** – the data type, e.g. Types.UINT16

Returns

write_channels(*data*, *binning=None*, *bpp=None*, *encoding=None*, *roi_offsets=None*, *timestamp=None*, ***deprecated*)
Write an image to 'output' and 'daqOutput' channels

Parameters

- **data** – the image data as numpy.ndarray
- **binning** – the image binning, e.g. (1, 1)
- **bpp** – the bits-per-pixel, e.g. 12
- **encoding** – the image encoding, e.g. Encoding.GRAY
- **roi_offsets** – the ROI offset, e.g. (0, 0)
- **timestamp** – the image timestamp - if none the current timestamp will be used

Returns

whereas for middle-layers ones the class is

class imageSourcePy.ImageSourceMdl.**ImageSource**(*configuration*)
Base class for image sources.

It provides two output channels - 'output' and 'daqOutput' - for sending out images, and three functions - 'update_output_schema', 'write_channels' and 'signal_eos'.

The function 'update_output_schema' will update the schema for the output channels and make it fit for the DAQ.

The function 'write_channels' will write the input data to both the output channels, taking care of reshaping them for the DAQ.

The function 'signal_eos' will send an end-of-stream signal to both the output channels.

signal_eos()
Send an end-of-stream signal to 'output' and 'daqOutput' channels

Returns

update_output_schema(*shape*, *encoding*, *dtype*)
Update the schema of 'output' and 'daqOutput' channels

Parameters

- **shape** – the shape of image, e.g. (height, width)
- **encoding** – the encoding of the image. e.g. EncodingType.GRAY
- **dtype** – the data type, e.g. UInt16

Returns

write_channels(*data*, *binning=None*, *bpp=None*, *encoding=None*, *roi_offsets=None*, *timestamp=None*)
Write an image to 'output' and 'daqOutput' channels

Parameters

- **data** – the image data as numpy.ndarray
- **binning** – the image binning, e.g. (1, 1)
- **bpp** – the bits-per-pixel, e.g. 12
- **encoding** – the image encoding, e.g. EncodingType.GRAY

- **roi_offsets** – the ROI offset, e.g. (0, 0)
- **timestamp** – the image timestamp - if none the current timestamp will be used

Returns

CHAPTER 3

The CameraImageSource class

The CameraImageSource class provides a base class to be used for all camera devices. In addition to the functionalities provided by its *base class*, this class provides a default scene for cameras.

The class to be used in Python-bound devices is this one

class imageSourcePy.CameraImageSource.**CameraImageSource**(*configuration*)
Base class for camera devices.

It is derived from the ImageSource class, and provides a default scene.

requestScene(*params*)
Fulfill a scene request from another device.

NOTE: Required by Scene Supply Protocol, which is defined in KEP 21. The format of the reply is also specified there.

Parameters **params** – A *Hash* containing the method parameters

whereas for middle-layers ones the class is

class imageSourcePy.CameraImageSourceMdl.**CameraImageSource**(*configuration*)
Base class for camera devices.

It is derived from the ImageSource class, and provides a default scene.

requestScene(*params*)
Fulfill a scene request from another device.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index