
senseye-cameras

Apr 22, 2022

Contents:

1	Introduction	1
2	Camera Support	3
3	Modules	5
4	Input Types	7
5	Using Inputs	9
6	Output Types	11
7	Using Outputs	13
8	Streams	15
9	Error handling	17
10	Indices and tables	19

CHAPTER 1

Introduction

A generic camera interface that allows Python developers to easily grab frames from multiple types of cameras and write them to disk.

CHAPTER 2

Camera Support

- usb cameras
- compressed video files
- raw video files
- **basler pylon cameras** [pylon SDK](#) and [pypylon package](#) must be installed.
- **ids ueye cameras** [ueye SDK](#) must be installed.

CHAPTER 3

Modules

`Input` objects obtain data from a data source, such as a usb camera, pylon camera, or video.

`Output` objects write data to a file, such as a raw video file or a compressed video file.

`Stream` objects link a single input and output.

CHAPTER 4

Input Types

Supported input types can be found under the `create_input` function docs.

CHAPTER 5

Using Inputs

Input objects obtain data from a data source. They can be created by using the `create_input` function.

All Inputs support the `open` and `read` functions.

The **`open`** function initializes the Input object.

The **`read`** function returns data from the Input in a tuple with the format: **(data, timestamp)**.

data is typically a numpy array obtained from the Input object.

timestamp is a unix timestamp that records when the frame was obtained.

Here's a simple example that opens and reads from a usb camera.

```
from senseye_cameras import create_input

# initialize and open a usb camera
camera = create_input(type='usb', id=0, config={})

# print documentation for the CameraUsb module
# docs will show what key/value pairs can go into the config dictionary
print(camera.__doc__)

camera.open()
frame, timestamp = camera.read()

print(frame)
```

More examples can be found in the `senseye_cameras/examples/input` folder.

CHAPTER 6

Output Types

Supported output types can be found under the `create_output` function docs.

CHAPTER 7

Using Outputs

Output objects put data into a file. They can be created by using the `create_output` function.

All Outputs support the `write` and `close` functions.

The **write** function writes data, typically in the form of numpy arrays, to the Output object.

The **close** function gracefully closes the Output object.

Here's a simple example that creates and writes to a file Output object.

```
import numpy as np
from senseye_cameras import create_output

# supported extensions
# FILE_PATH = './tmp/usb.avi'
# FILE_PATH = './tmp/usb.mp4'
# FILE_PATH = './tmp/usb.mkv'
# FILE_PATH = './tmp/usb.yuv'
FILE_PATH = './tmp/usb.raw'

data = np.random.rand(256, 256, 3)

# create a raw video Output object that points to the path1.raw file.
output = create_output('file', path=FILE_PATH)

output.write(data=data)
output.close()
```

More examples can be found in the `senseye_cameras/examples/output` folder.

The `Stream` module is a high level module that links a single Input and Output.

Here's a simple example that creates a stream that opens a usb camera and writes frames to a file:

```
import time
from senseye_cameras import Stream

'''
Takes a usb input and writes to a file.
The input_type can be easily switched to ffmpeg/pylon/ueye/raw_video.
'''

SLEEP_TIME = 5
CAMERA_ID = 0

# supported file extensions
# FILE_PATH = './tmp/usb.avi'
# FILE_PATH = './tmp/usb.mp4'
FILE_PATH = './tmp/usb.mkv'
# FILE_PATH = './tmp/usb.yuv'
# FILE_PATH = './tmp/usb.raw'

s = Stream(
    input_type='usb', id=CAMERA_ID,
    output_type='file', path=FILE_PATH,
    reading=True,
    writing=True,
)
time.sleep(SLEEP_TIME)

s.stop()
```

More examples can be found in the `senseye_cameras/examples/stream` folder.

CHAPTER 9

Error handling

Inputs/Outputs will fail loudly when initialization fails.

If initialization succeeds but a read/write fails, an Input/Output will typically catch the error and log it instead of erroring out.

CHAPTER 10

Indices and tables

- `genindex`
- `modindex`
- `search`