**REFERENCES:**

* <https://docs.opencv.org/3.4/d0/da7/videoio_overview.html>

**General Information**

The OpenCV Video I/O module is a set of classes and functions to read and write video or images sequence and the module provides the cv::VideoCapture and cv::VideoWriter classes as 2-layer interface to many video I/O APIs used as backend.

**Video I/O with OpenCV**

Some backbend’s such as

* (DSHOW) Direct Show
* Video for Windows (VFW)
* Microsoft Media Foundation (MSMF)
* Video 4 Linux (V4L)

This is the interfaces of the video I/O library provided by the operating system.

Some others backbend’s like

* OpenNI2 for Kinect
* Intel Perceptual Computing SDK
* GStreamer
* XIMEA Camera API

These are interfaces to proprietary drivers or to external library.

Each backend supports devices properties (cv::VideoCaptureProperties) in a different way or might not support any property at all.

**Select the backend at runtime**

OpenCV automatically selects and uses first available backend (apiPreference=cv::CAP\_ANY).

As advanced usage you can select the backend to use at runtime. Currently this option is available only with VideoCapture.

For example to grab from default camera using Direct Show as backend

//declare a capture object

cv::VideoCapture cap(0, cv::CAP\_DSHOW);

//or specify the apiPreference with open

cap.open(0, cv::CAP\_DSHOW);

If you want to grab from a file using the Direct Show as backend:

//declare a capture object

cv::VideoCapture cap(filename, cv::CAP\_DSHOW);

//or specify the apiPreference with open

cap.open(filename, cv::CAP\_DSHOW);

**Enable backends**

Backbend’s are available only if they have been built with your OpenCV binaries.

Check in opencv2/cvconfig.h to know which APIs are currently available (e.g. HAVE\_MSMF, HAVE\_VFW, HAVE\_LIBV4L, etc...).

To enable/disable APIs, you have to:

1. re-configure OpenCV using appropriates CMake switches (e.g. -DWITH\_MSMF=ON -DWITH\_VFW=ON ...) or checking related switch in cmake-gui
2. rebuild OpenCV itself

**Use 3rd party drivers or cameras**

Many industrial cameras or some video I/O devices don't provide standard driver interfaces for the operating system. Thus you can't use VideoCapture or VideoWriter with these devices.

To get access to their devices, manufactures provide their own C++ API and library that you have to include and link with your OpenCV application.

It is a common case that these libraries read/write images from/to a memory buffer. If it so, it is possible to make a Mat header for memory buffer (user-allocated data) and process it in-place using OpenCV functions. See **cv::Mat::Mat()**for more details.

**The FFmpeg library**

OpenCV can use the FFmpeg library (<http://ffmpeg.org/>) as backend to record, convert and stream audio and video. FFmpeg is a complete, cross-reference solution. If you enable FFmpeg while configuring OpenCV than CMake will download and install the binaries in OPENCV\_SOURCE\_CODE/3rdparty/ffmpeg/. To use FFmpeg at runtime, you must deploy the FFMepg binaries with your application.