Video Processing (VPP) Sample

Overview

VPP Sample works with **Intel® Media Server Studio 2017 for Windows* Server** (hereinafter referred to as "**SDK**").

It demonstrates how to use the **SDK** API to create a simple console application that performs video processing of raw video sequences.

Features

VPP Sample supports the following video formats:

Format type	
input (uncompressed)	NV12, YV12, YUY2, RGB3, RGB4, IMC3, YUV400, YUV411, YUV422h, YUV422v, YUV444, UYVY, AYUV, Y210, Y410
output (uncompressed)	NV12, YUY2, RGB4, YV12, AYUV, Y210, Y410

Hardware Requirements

See `<install-folder>\Media Samples Guide.pdf`.

Software Requirements

See `<install-folder>\Media Samples Guide.pdf`.

How to Build the Application

See `<install-folder>\Media Samples Guide.pdf`.

Running the Software

See `<install-folder>\Media Samples Guide.pdf`.

The executable file requires the following command-line switches to function properly:

-lib type	type of used library: sw, hw (default: sw)
-d3d	use d3d9 surfaces
-d3d11	use d3d11 surfaces
-plugin_guid GUID	use VPP plug-in with specified GUID
-p GUID	use VPP plug-in with specified GUID
-extapi	use RunFrameVPPAsyncEx instead of RunFrameVPPAsync. Need for PTIR.

-gpu_copy	Specify GPU copy mode. This option triggers using of InitEX instead of Init.
-sw width	width of src video (def: 352)
-sh height	height of src video (def: 288)
-scrX x	cropX of src video (def: 0)
-scrY y	cropY of src video (def: 0)
-scrW w	cropW of src video (def: width)
-scrH h	cropH of src video (def: height)
-sf frameRate	frame rate of src video (def: 30.0)
-scc format	format (FourCC) of src video (def: nv12. support nv12|yv12|yuy2|rgb3|rgb4|imc3|yuv400|yuv411|yuv422h|yuv422v|yuv444|uyvy|ayuv|y210|y410)
-sbitshift 0|1	shift data to right or keep it the same way as in Microsoft's P010
-sbitdepthluma value	shift luma channel to right to "16 - value" bytes
-sbitdepthchroma value	shift chroma channel to right to "16 - value" bytes
-spic value	picture structure of src video: • -1 - unknown • 0 - interlaced top field first • 1 - progressive (default) • 2 - interlaced bottom field first • 3 - single field
-dw width	width of dst video (def: 352)
-dh height	height of dst video (def: 288)
-dcrX x	cropX of dst video (def: 0)
-dcrY y	cropY of dst video (def: 0)
-dcrW w	cropW of dst video (def: width)
-dcrH h	cropH of dst video (def: height)
-df frameRate	frame rate of dst video (def: 30.0)
-dcc format	format (FourCC) of dst video (def: nv12. support nv12|yuy2|rgb4|yv12|ayuv|y210|y410)
-dbitshift 0|1	shift data to right or keep it the same way as in Microsoft's P010
-dbitdepthluma value	shift luma channel to left to "16 - value" bytes
-dbitdepthchroma value	shift chroma channel to left to "16 - value" bytes
-dpic value	picture structure of dst video: • -1 - unknown • 0 - interlaced top field first • 1 - progressive (default) • 2 - interlaced bottom field first • 3 - single field

-composite <ParametersFile>	Composition of several input files in one output. The location of substreams on the primary stream is described in the parameter file. The syntax of the parameters file is:
	`primarystream=<video file name>`
	`width=<input video width>`
	`height=<input video height>`
	`cropx=<input cropX (def: 0)>`
	`cropy=<input cropY (def: 0)>`
	`cropw=<input cropW (def: width)>`
	`croph=<input cropH (def: height)>`
	`framerate=<input frame rate (def: 30.0)>`
	`fourcc=<format (FourCC) of input video (def: nv12. support nv12|yuy2)>`
	`picstruct=<picture structure of input video,`
	`0 = interlaced top field first`
	`2 = interlaced bottom field first`
	`3 = single field`
	`1 = progressive (default)>`
	`dstx=<X coordinate of input video located in the output (def: 0)>`
	`dsty=<Y coordinate of input video located in the output (def: 0)>`
	`dstw=<width of input video located in the output (def: width)>`
	`dsth=<height of input video located in the output (def: height)>`
	`stream=<video file name> width=<input video width>`
	`...`
	`The parameters file may contain one primary stream (which goes first) and up to 64 substreams.`

-di_mode (mode)	set type of deinterlace algorithm 8 - reverse telecine for a selected telecine pattern (use -tc_pattern). For PTIR plug-in 2 - advanced or motion adaptive (default) 1 - simple or BOB
-deinterlace (type)	enable deinterlace algorithm (alternative way: -spic 0 -dpic 1) type is tff (default) or bff
-rotate (angle)	enable rotation. Supported angles: 0, 90, 180, 270.
-scaling_mode (mode)	specify type of scaling to be used for resize.
-denoise (level)	enable denoise algorithm. Level is optional range of noise level is [0, 100]
-detail (level)	enable detail enhancement algorithm. Level is optional range of detail level is [0, 100]
-pa_hue hue	procamp hue property. range [-180.0, 180.0] (def: 0.0)
-pa_sat saturation	procamp satursation property. range [0.0, 10.0] (def: 1.0)
-pa_con contrast	procamp contrast property. range [0.0, 10.0] (def: 1.0)
-pa_bri brightness	procamp brightness property. range [-100.0, 100.0] (def: 0.0)
-gamut:compression	enable gamut compression algorithm (xvYCC->sRGB)
-gamut:bt709	enable BT.709 matrix transform (RGB->YUV conversion)(def: BT.601)
-frc:advanced	enable advanced FRC algorithm (based on PTS)
-frc:interp	enable FRC based on frame interpolation algorithm
-tcc:red	enable color saturation algorithm (R component)
-tcc:green	enable color saturation algorithm (G component)

-tcc:blue	enable color saturation algorithm (B component)
-tcc:cyan	enable color saturation algorithm (C component)
-tcc:magenta	enable color saturation algorithm (M component)
-tcc:yellow	enable color saturation algorithm (Y component)
-ace	enable auto contrast enhancement algorithm
-ste (level)	enable Skin Tone Enhancement algorithm. Level is optional range of ste level is [0, 9] (def: 4)
-istab (mode)	enable Image Stabilization algorithm. Mode is optional mode of istab can be [1, 2] (def: 2) where: 1 means upscale mode, 2 means croppping mode
-view:count value	enable Multi View preprocessing. range of views [1, 1024] (def: 1)
-svc id width height	enable Scalable Video Processing mode id-layerId, width/height-resolution
-ssitm (id)	specify YUV<->RGB transfer matrix for input surface.
-dsitm (id)	specify YUV<->RGB transfer matrix for output surface.
-ssinr (id)	specify YUV nominal range for input surface.
-dsinr (id)	specify YUV nominal range for output surface.
-mirror (mode)	mirror image using specified mode.
-n frames	number of frames to VPP process
-iopattern IN/OUT surface type	IN/OUT surface type: sys_to_sys, sys_to_d3d, d3d_to_sys, d3d_to_d3d (def: sys_to_sys)
-async n	maximum number of asynchronious tasks. def: -async 1
-perf_opt n m	n: number of prefetech frames. m : number of passes. In performance mode app preallocates bufer and load first n frames, def: no performace 1
-pts_check	checking of time stampls. Default is OFF
-pts_jump	checking of time stamps jumps. Jump for random value since 13-th frame. Also, you can change input frame rate (via pts). Default frame_rate = sf
-pts_fr	input frame rate which used for pts. Default frame_rate = sf
-pts_advanced	enable FRC checking mode based on PTS
-pf file for performance data	file to save performance data. Default is off
-roi_check mode seed1 seed2	checking of ROI processing. Default is OFF mode - usage model of cropping var_to_fix - variable input ROI and fixed output ROI fix_to_var - fixed input ROI and variable output ROI var_to_var - variable input ROI and variable output ROI seed1 - seed for init of rand generator for src seed2 - seed for init of rand generator for dst range of seed [1, 65535]. 0 reserved for random init
-tc_pattern (pattern)	set telecine pattern 4 - provide a position inside a sequence of 5 frames where the artifacts starts. Use to -tc_pos to provide position 3 - 4:1 pattern 2 - frame repeat pattern 1 - 2:3:3:2 pattern 0 - 3:2 pattern
-tc_pos (position)	Position inside a telecine sequence of 5 frames where the artifacts starts - Value [0 - 4]
-reset_start (frame number)	after reaching this frame, encoder will be reset with new parameters, followed after this command and before -reset_end

-reset_end	specifies end of reset related options

Below are examples of a command-line to execute **VPP Sample**:

```
sample_vpp -lib sw -sw 352 -sh 144 -scc rgb4 -dw 320 -dh 240 -dcc nv12
-denoise 32 -istab -i input.rgb -o output.nv12
```

```
$ sample_vpp -lib hw -scc nv12 -dcc nv12 -composite
parameters.par -o out.yuv

The example of parameters.par:
primarystream=input_720x480.yuv
width=720
height=480
cropx=0
cropy=0
cropw=720
croph=480
dstx=0
dsty=0
dstw=720
dsth=480
stream=input_480x320.yuv
width=480
height=320
cropx=0
cropy=0
cropw=480
croph=320
dstx=100
dsty=100
dstw=320
dsth=240
```

Known Limitations

- Scene change detection is not supported (-vanalysis option not effective) with platform specific **SDK** libraries for Intel® HD Graphics 3000/2000 and later, also unsupported in software **SDK** libraries starting with API version 1.6.
- RGB3 (RGB 24-bit) input format is unsupported despite the fact that sample code and sample binary expose it as supported.

Legal Information

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting *Intel's Web Site*.

MPEG is an international standard for video compression/decompression promoted by ISO. Implementations of MPEG CODECs, or MPEG enabled platforms may require licenses from various entities, including Intel Corporation.

Intel, the Intel logo, Intel Core are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel.

Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804
