AGENDA

NVIDIA Video Technologies
New SDK Release
Major Focus Areas
Video SDK Features
Software Flow
FFmpeg Performance and Benchmarking Tips
Benchmarks
NVIDIA VIDEO TECHNOLOGIES
VIDEO CODEC SDK
A comprehensive set of APIs for GPU-accelerated Video Encode and Decode

The SDK consists of two hardware acceleration interfaces:

**NVENCODE** API for video encode acceleration

**NVDECODE** API for video decode acceleration (formerly called NVCUVID API)

Independent of CUDA/3D cores on GPU
NVIDIA VIDEO TECHNOLOGIES

SOFTWARE

FFMPEG & LIBAV
Easy access to NVIDIA GPU hardware acceleration

VIDEO CODEC SDK
A comprehensive set of APIs for GPU-accelerated Video Encode and Decode for Windows and Linux CUDA, DirectX, OpenGL interoperability

HARDWARE

NVENC
Independent Hardware Encoder Function

H.264
MPEG-4/AVC
H.265
HEVC

NVDEC
Independent Hardware Decoder Function

H.264
MPEG-4/AVC
H.265
HEVC
MPEG2
VP8
VP9
NVIDIA VIDEO TECHNOLOGIES

Decode HW*
- Formats:
  - MPEG-2
  - VC1
  - VP8
  - VP9
  - H.264
  - H.265
  - Lossless
- Bit depth:
  - 8 bit
  - 10 bit
- Color**
  - YUV 4:2:0
- Resolution
  - Up to 8K***

* See support diagram for previous NVIDIA HW generations
** 4:2:2 is not natively supported on HW
*** Support is codec dependent

Encode HW*
- Formats:
  - H.264
  - H.265
  - Lossless
- Bit depth:
  - 8 bit
  - 10 bit
- Color**
  - YUV 4:4:4
  - YUV 4:2:0
- Resolution
  - Up to 8K***
VIDEO SDK EVOLUTION

Video SDK 8.0

SDK 4.0
Maxwell 1
H.264
4:4:4, lossless
2014

SDK 5.0
Maxwell 2
HEVC
Perf++
2015

SDK 6.0
ARGB
Quality+
Dec+Enc
ME-only
2015

SDK 7.x
Pascal
10-bit encode
FFmpeg
ME-only for VR
Quality++
2016

SDK 8.0
10-bit transcode
10/12-bit decode
OpenGL
Dec. optimizations
WP, AQ, Enc.
Quality
2017
MAJOR FOCUS AREAS
VIDEO TRANSCODING

Performance/Watt

- Content variety
- Codecs, resolutions, quality, bitrate
- Live, VOD, ultra-low-latency, broadcast, archives
- Pre-encoded or encoded-on-demand
- Performance/Watt
GAME/APP STREAMING

Ultra-low-latency

Stream

➢ Interactive, single frame latency
➢ Capture: NvFBC, Encode: NvENC, Decode: NvDEC
➢ 4K, HDR

Record, Broadcast

➢ Quality
GPU VIRTUALIZATION

Quality & reliability

- Capture + encode
- Low-latency
- H.264, HEVC
- 4:2:0, 4:4:4, lossless
- Multiple-displays
MOTION-ESTIMATION ONLY MODE

Accuracy

- Video frame interpolation
- Camera stitching (mono to stereo)
- Camera stabilization
- Computer vision

Frame #($N$) is interpolated based on motion vectors between frame #$N$ and frame #$N+1$.
VIDEO SDK FEATURES
# ENCODE FEATURES (1/2)

<table>
<thead>
<tr>
<th>H.264</th>
<th>HEVC</th>
<th>Use-case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base, Main, High</td>
<td>Main, Main10</td>
<td>Baseline standards</td>
</tr>
<tr>
<td>8-bit</td>
<td>8-bit, 10-bit</td>
<td>10-bit for HDR</td>
</tr>
<tr>
<td>B-frames</td>
<td>No B-frames</td>
<td>Higher compression &amp; quality</td>
</tr>
<tr>
<td>Up to 4096 × 4096</td>
<td>Up to 8192 × 8192</td>
<td>High-res</td>
</tr>
<tr>
<td>YUV 4:2:0, 4:4:4</td>
<td></td>
<td>Subsampled or full-res chroma (e.g. wireframes)</td>
</tr>
<tr>
<td>Lossless</td>
<td></td>
<td>High-quality archiving</td>
</tr>
<tr>
<td>Error resiliency: Intra refresh, LTR, ref-pic invalidation</td>
<td></td>
<td>Handle streaming bit errors</td>
</tr>
</tbody>
</table>
## ENCODE FEATURES (2/2)

<table>
<thead>
<tr>
<th>H.264</th>
<th>HEVC</th>
<th>Use-case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate control modes: 1-pass, 2-pass</td>
<td>Quality vs performance</td>
<td></td>
</tr>
<tr>
<td>Look-ahead</td>
<td>Efficient bit distribution across GOP; higher quality</td>
<td></td>
</tr>
<tr>
<td>Adaptive quantization, ΔQP</td>
<td>Finer quality control</td>
<td></td>
</tr>
<tr>
<td><strong>Weighted prediction (SDK 8.0)</strong></td>
<td>Fade-in/fade-out, explosion</td>
<td></td>
</tr>
<tr>
<td>RGB inputs</td>
<td>Direct NVFBC interoperability</td>
<td></td>
</tr>
<tr>
<td>ME-only mode, <strong>MV-hints (SDK 8.0)</strong></td>
<td>Motion stabilization, Optical flow for VR stereo stitching, Frame interpolation</td>
<td></td>
</tr>
<tr>
<td>1-3 NVENCs per chip</td>
<td>High throughput</td>
<td></td>
</tr>
<tr>
<td>CUDA, DX, <strong>OGL (Linux) (SDK 8.0)</strong></td>
<td>Easy integration</td>
<td></td>
</tr>
</tbody>
</table>
## DECODE FEATURES

<table>
<thead>
<tr>
<th>Feature</th>
<th>Use-case</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG2, VC-1, MPEG-4, H.264, HEVC, VP8, VP9</td>
<td>Baseline standards</td>
</tr>
<tr>
<td>8-bit (all codecs), <strong>10/12 bit (HEVC, VP9)</strong> (SDK 8.0)</td>
<td>HDR decoding</td>
</tr>
<tr>
<td>Up to 8192 × 8192 for HEVC, 4096 × 4096 for H.264</td>
<td>High-res</td>
</tr>
<tr>
<td>Error resiliency and concealment</td>
<td>Internet streaming</td>
</tr>
</tbody>
</table>
VIDEO SDK - CONTENTS (1/2)

- Header, documentation, sample applications
- Binaries (.dll, .so) in NVIDIA display driver
- Unified API for **Windows** & **Linux**
- NVIDIA developer zone
- Encode limitations
  - Unconstrained: Tesla, GRID, Quadro $\geq$ X2000 ($X = K, M, P$)
  - 2 sessions/system: GeForce, Quadro $<$ X2000
- No decode limitations
VIDEO SDK - CONTENTS (2/2)

Sample Applications

- **Decode**: DX9, DX11, CUDA, OpenGL
- **Encode**: Basic functionality, features (NvEncoder)
- **Encode**: Performance (NvEnodePerf)
- **Encode**: CUDA interop, D3D interop, OGL interop,
- **Encode**: Low-latency (NVEncoderLowLatency)
- **Transcode** (NvTranscoder)
- Coming soon: Reusable classes
FFMPEG/LIBAV

- Major SW focus area for past 6 months
- Feature parity with Video SDK 7.1, SDK 8.0 post GTC
- End-to-end FFmpeg transcoding @ best possible quality & perf
SOFTWARE FLOW
ENCODE APP FLOW

Client application

Initialize, Configure, Encode

NVENC API

Configure HW

DirectX

NVENC Driver

HW Encode

CUDA

OpenGL

Encrypted bitstream

NVENC firmware + hardware

OpenGL-CUDA interop

NVENC-CUDA interop
ENCODE APP FLOW

API Functions

Defined in nvEncodeAPI.h
DECODE APP FLOW

- Source
- Parser
- NVDEC Driver
- NVDEC

Data flow:
- Bitstream
- Demux
- Video frames

Decode API calls:
- YUV
- RGB
- DX
- CUDA

Callbacks:
- Call backs

- DECODE APP FLOW
- Parser
- Client application
- NVDEC Driver
- NVDEC

Data flow:
- Bitstream
- Demux
- Video frames

Decode API calls:
- YUV
- RGB
- DX
- CUDA

Callbacks:
- Call backs
DECODE APP FLOW

API functions

Structures

Defined in dynlink_nvcuvid.h, dynlink_cuviddec.h

cuvidGetDecoderCaps()
CUVIDDECODECAPS

cuvidCreateDecoder()
CUVIDDECODECREATEINFO

cuvidDecodePicture()
CUVIDFICPARAMS

CUDA kernels

cuvidDestroyDecoder()

APIs

Query capabilities
- Codecs, resolutions supported

Create decoder
- W/H, scaling, bit-depth

Decode picture
- Picture parameters from bitstream parser

Post-processing
- scaling, CSC Etc.

Clean-up
FFMPEG APP FLOW

`ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -c:a copy -vf scale_npp=1280:720 -c:v h264_nvenc -b:v 5M output.mp4`

- **Chain of filters**

- **-hwaccel cuvid:** Use end-to-end NVIDIA hardware acceleration
- **h264_cuvid:** Use NVCUVID/NVDECODE
- **h264_nvenc:** Use NVENCODE
- **scale_npp:** high-perf CUDA scaling
HARDWARE ACCELERATED TRANSCODE USING FFmpeg
PERFORMANCE CONSIDERATIONS - FFmpeg

- Minimize memory (PCIe) transfers
- Saturate on-chip encoder/decoder
- Efficient M:N command line
- Minimize I/O
- Encode settings
- GPU Clocks
SW TRANSCODE

```bash
ffmpeg -c:v h264 -i input.mp4 -c:a copy -c:v h264 -b:v 5M output.mp4
```

System Memory

32 fps*

*1:2 transcode, fps per session
4 GHz Intel i7-6700K
SW TRANSCODE + SCALE

ffmpeg -c:v h264 -i input.mp4 -vf scale=1280:720 -c:a copy -c:v h264 -b:v 5M output.mp4

System Memory

Bitstream → SW Decode → YUV

Bitstream → Preprocess (e.g. scaling) → YUV

YUV → SW Encode → Bitstream

29 fps*

*1:2 transcode, fps per session
4 GHz Intel i7-6700K
```
ffmpeg -y -vsync 0 -c:v h264_cuvid -i input.mp4 -c:a copy -c:v h264_nvenc -b:v 5M output.mp4
```
**GPU UNOPTIMIZED TRANSCODE + CPU SCALE**

```
ffmpeg -y -vsync 0 -c:v h264_cuvid -i input.mp4 -c:a copy -vf scale=1280:720 -c:v h264_nvenc -b:v 5M output.mp4
```
HIGH-PERF GPU OPTIMIZED TRANSCODE

```
ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -c:a copy -vf scale_npp=1280:720 -c:v h264_nvenc -b:v 5M output.mp4
```
PERFORMANCE CONSIDERATIONS

Saturating encoder/decoder

- Pipelining
- Input/output buffers
- Tools: nvidia-smi, Microsoft GPUView
ANALYZING PERFORMANCE BOTTLENECKS
Microsoft GPUView (Windows only)
# ANALYZING PERFORMANCE BOTTLENECKS

**nvidia-smi (Windows & Linux)**

<table>
<thead>
<tr>
<th>#</th>
<th>gpu</th>
<th>pwr</th>
<th>temp</th>
<th>sm</th>
<th>mem</th>
<th>enc</th>
<th>dec</th>
<th>mclk</th>
<th>pclk</th>
</tr>
</thead>
<tbody>
<tr>
<td># idx</td>
<td>W</td>
<td>C</td>
<td>%</td>
<td>%</td>
<td>%</td>
<td>%</td>
<td>MHz</td>
<td>MHz</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>44</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>405</td>
<td>139</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>44</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>405</td>
<td>139</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>44</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>405</td>
<td>139</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>24</td>
<td>45</td>
<td>5</td>
<td>3</td>
<td>14</td>
<td>1</td>
<td>3802</td>
<td>1506</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>33</td>
<td>47</td>
<td>14</td>
<td>9</td>
<td>44</td>
<td>36</td>
<td>3802</td>
<td>1885</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>32</td>
<td>47</td>
<td>13</td>
<td>9</td>
<td>42</td>
<td>38</td>
<td>3802</td>
<td>1885</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>33</td>
<td>48</td>
<td>15</td>
<td>10</td>
<td>54</td>
<td>41</td>
<td>3802</td>
<td>1885</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>32</td>
<td>46</td>
<td>13</td>
<td>9</td>
<td>44</td>
<td>38</td>
<td>3802</td>
<td>1885</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>33</td>
<td>48</td>
<td>15</td>
<td>9</td>
<td>44</td>
<td>36</td>
<td>3802</td>
<td>1885</td>
<td></td>
</tr>
</tbody>
</table>
PARALLEL TRANSCODES (1:N)

Single command line

```
ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4
```
PARALLEL TRANSCODES (1:N)

Multiple command lines


ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -vf scale_npp=1280:720 -c:a copy -c:v h264_nvenc -b:v 5M output2.mp4


...
PARALLEL TRANSCODES (1:N)

Single command line

**PROS**

- Low init time per transcode (amortized)
- Minimize memory transfers
- Leverage high encoder perf
- Low memory overhead

**CONS**

- Complex command line
- 1:N use-case only
- Unsuitable for 1:1 VOD
- Typically encoder-limited
PARALLEL TRANSCODES (1:N)

Multiple command lines

PROS

Simple command line
Easy scripting
Use-case: 1:1 VOD

CONS

High init time per transcode
High memory overhead
Process-level scheduling optimizations
Typically decoder-limited
Multiple disk I/O for input
PARALLEL TRANSCODERES (M:N)

Hybrid approach

Most flexible approach
Balance memory utilization/complexity/perf
Maximum utilization of encode/decode capacity
## ENCODE SETTINGS

<table>
<thead>
<tr>
<th></th>
<th>Highest Quality</th>
<th>Minimum Delay</th>
<th>Highest Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use-case</strong></td>
<td>Transcoding, Archiving, Broadcast streaming (w/ latency), surveillance</td>
<td>Game &amp; app streaming, surveillance</td>
<td>All, w/ high performance requirement</td>
</tr>
<tr>
<td><strong>NVENC API preset to use</strong></td>
<td>High quality (HQ) presets</td>
<td>Low-latency (Low delay) presets</td>
<td>High performance (HP) presets</td>
</tr>
<tr>
<td><strong>Latency (set by the application via VBV buffer size)</strong></td>
<td>Depends on what application sets; Typically &gt; 8-10 frames</td>
<td>Depends on what application sets; Typically 1 frame</td>
<td>Depends on what application sets</td>
</tr>
<tr>
<td><strong>PSNR delta (0 = High quality)</strong></td>
<td>0 dB (reference)</td>
<td>Approx. -0.5 dB</td>
<td>Approx. -0.5-2 dB</td>
</tr>
<tr>
<td><strong>Advanced features typically used</strong></td>
<td>Look-ahead, B-frames (H.264 only), adaptive B-frames (H.264 only), AQ (Adaptive Quantization)</td>
<td>Strict frame-size compliance low VBV (Video Buffering Verifier), AQ (Adaptive Quantization)</td>
<td></td>
</tr>
<tr>
<td><strong>Motion search and mode</strong></td>
<td>High, 2-pass</td>
<td>Medium, 1-pass/2-pass</td>
<td>Low, 1-pass</td>
</tr>
<tr>
<td><strong>Modes</strong></td>
<td>All high quality modes</td>
<td>Most modes</td>
<td>Most modes</td>
</tr>
<tr>
<td><strong>Entropy coding</strong></td>
<td>CABAC (H.264)</td>
<td>CABAC (H.264)</td>
<td>CAVLC (H.264)</td>
</tr>
</tbody>
</table>
BENCHMARKS
ENCODE PERFORMANCE

H.264 1080p (1920x1080) 4:2:0 8bit 30fps (SINGLE NVENC)

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration.

### Number of Streams / NVENC

<table>
<thead>
<tr>
<th>Series</th>
<th>#NVENC</th>
<th>GPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kepler</td>
<td>x1</td>
<td>Kepler Quadro K2000/K2000D/K4000/K4200/K5000/K5200/K6000</td>
</tr>
<tr>
<td>Maxell 1st Gen</td>
<td>x2</td>
<td>Kepler Tesla K20X/K40, Maxwell Quadro K2200 (1st Gen)/M2000 (2nd Gen)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maxwell (2nd Gen) Tesla M4, Pascal Quadro P2000/P4000</td>
</tr>
<tr>
<td>Maxell 2nd Gen</td>
<td>x3</td>
<td>Kepler Tesla K10/K80, Kepler GRID K2/K520</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maxwell (2nd Gen) Quadro M4000/M5000/M6000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maxwell (2nd Gen) Tesla M6/M40, Pascal Quadro P5000/P6000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pascal Tesla P4/P40</td>
</tr>
<tr>
<td>Pascal</td>
<td>x4</td>
<td>Pascal Quadro GP100, Pascal Tesla P100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Kepler GRID K1/K340, Maxwell (2nd Gen) Tesla M60</td>
</tr>
</tbody>
</table>

Note: All GPUs not featured above are limited to 2 simultaneous sessions.
ENCODE PERFORMANCE

HEVC 4K (3840x2160) 4:2:0 8bit 30fps (SINGLE NVENC)

Number of Streams / NVENC

- MAXWELL (2ND GEN)
  - Highest Quality: 3
  - Highest Performance: 7

- PASCAL
  - Highest Quality: 5
  - Highest Performance: 13

Note: All GPUs not featured above are limited to 2 simultaneous sessions

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration.
# ENCODE PERFORMANCE

H.264 1080p (1920x1080) 4:4:4 8bit 30fps (SINGLE NVENC)

**Number of Streams / NVENC**

<table>
<thead>
<tr>
<th></th>
<th>Highest Quality</th>
<th>Highest Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXWELL 1ST GEN</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>MAXWELL 2ND GEN</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>PASCAL</td>
<td>7</td>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#NVENC</th>
<th>GPUs</th>
</tr>
</thead>
</table>
| 1      | Maxwell Quadro K2200 (1st Gen)/M2000 (2nd Gen)  
Maxwell (2nd Gen) Tesla M4  
Pascal Quadro P2000/P4000 |
| 2      | Maxwell (2nd Gen) Quadro M4000/M5000/M6000  
Maxwell (2nd Gen) Tesla M6/M40  
Pascal Quadro P5000/P6000  
Pascal Tesla P4/P40 |
| 3      | Pascal Quadro GP100  
Pascal Tesla P100 |
| 4      | Maxwell (2nd Gen) Tesla M60 |

**Note:** All GPUs not featured above are limited to 2 simultaneous sessions

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration.
Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration.

Note: All GPUs not featured above are limited to 2 simultaneous sessions.
DECODE PERFORMANCE
NVDEC H.264 YUV 4:2:0

Number of 30fps Streams / NVDEC

<table>
<thead>
<tr>
<th></th>
<th>4096 x 4096</th>
<th>3840 x 2160</th>
<th>2560 x 1440</th>
</tr>
</thead>
<tbody>
<tr>
<td>TESLA M60</td>
<td>4</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>TESLA P40</td>
<td>5</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration.
ENCODE PERF/QUALITY

Encode quality latest results (slow/med: ±0.4 dB within x264)
MOTION VECTOR QUALITY

➢ KITTI Vision Benchmark Suite for Optical Flow
➢ Measures distortion of motion vectors compared to “true” motion
➢ Average distortion ≈ 7%, improves 1-2% by motion hints
ME-ONLY MODE

Frame 0

Source: http://www.cvlibs.net/datasets/kitti/, under Creative Commons License
ME-ONLY MODE

Frame 1

Source: http://www.cvlibs.net/datasets/kitti/, under Creative Commons License
ME-ONLY MODE
Motion Vector Distortion

“True” motion —> NVENC estimated motion

Distortion score = 2%
RESOURCES


FFmpeg GIT: https://git.ffmpeg.org/ffmpeg.git

Libav GIT: https://git.Libav.org/libav.git

FFmpeg builds with hardware acceleration: http://ffmpeg.zeranoe.com/builds/

Video SDK support: video-devtech-support@nvidia.com

Video SDK forums: https://devtalk.nvidia.com/default/board/175/video-technologies/