OpenVX 1.1
May 2016

Neil Trevett | Khronos President
NVIDIA Vice President Developer Ecosystem
Khronos is an Industry Consortium of over 100 companies creating royalty-free, open standard APIs to enable software to access hardware acceleration for graphics, parallel compute and vision.
Vision Pipeline Challenges and Opportunities

Growing Camera Diversity

Flexible sensor and camera control to GENERATE an image stream

Diverse Vision Processors

Use efficient acceleration to PROCESS the image stream

Sensor Proliferation

Combine vision output with other sensor data on device

© Copyright Khronos Group 2016 - Page 3
OpenVX - Low Power Vision Acceleration

- Higher level abstraction API
  - Targeted at real-time mobile and embedded platforms

- Performance portability across diverse architectures
  - Multi-core CPUs, GPUs, DSPs and DSP arrays, ISPs, Dedicated hardware...

- Extends portable vision acceleration to very low power domains
  - Doesn’t require high-power CPU/GPU Complex
  - Lower precision requirements than OpenCL
  - Low-power host can setup and manage frame-rate graph
OpenVX Graphs

- OpenVX developers express a graph of image operations (‘Nodes’)
  - Nodes can be on any hardware or processor coded in any language
  - E.g. on GPU nodes may be implemented in OpenCL or CUDA
- Minimizes host interaction during frame-rate graph execution
  - Host processor can setup graph which can then execute almost autonomously
OpenVX Framework Efficiency...

Graph Scheduling
- Split the graph execution across the whole system: CPU / GPU / dedicated HW
  - Faster execution or lower power consumption

Memory Management
- Reuse pre-allocated memory for multiple intermediate data
  - Less allocation overhead, more memory for other applications

Kernel Merge
- Replace a sub-graph with a single faster node
  - Better memory locality, less kernel launch overhead

Data Tiling
- Execute a sub-graph at tile granularity instead of image granularity
  - Better use of data cache and local memory
OpenVX and OpenCV are Complementary

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Community driven open source library</th>
<th>Open standard API designed to be implemented by hardware vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conformance</td>
<td>Extensive OpenCV Test Suite but no formal Adopters program</td>
<td>Implementations must pass defined conformance test suite to use trademark</td>
</tr>
<tr>
<td>Consistency</td>
<td>Available functions can vary depending on implementation / platform</td>
<td>All core functions must be available in all conformant implementations</td>
</tr>
<tr>
<td>Scope</td>
<td>Very wide 1000s of imaging and vision functions Multiple camera APIs/interfaces</td>
<td>Tight focus on core hardware accelerated functions for mobile vision - but extensible Uses external/native camera API</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Memory-based architecture Each operation reads and writes to memory</td>
<td>Graph-based execution Optimizable computation and data transfer</td>
</tr>
<tr>
<td>Typical Use Case</td>
<td>Rapid experimentation and prototyping - especially on desktop</td>
<td>Production development &amp; deployment on mobile and embedded devices</td>
</tr>
<tr>
<td>Embedded Deployment</td>
<td>Re-usable code</td>
<td>Callable library</td>
</tr>
</tbody>
</table>
Example Relative Performance

- Arithmetic: 1.1
- Analysis: 2.9
- Filter: 8.7
- Geometric: 1.5
- Overall: 2.5

OpenCV (GPU accelerated)
OpenVX (GPU accelerated)

NVIDIA implementation experience. Geometric mean of >2200 primitives, grouped into each category, running at different image sizes and parameter settings.
Layered Vision Processing Ecosystem

Implementers may use OpenCL or Compute Shaders to implement OpenVX nodes on programmable processors.

And then developers can use OpenVX to enable a developer to easily connect those nodes into a graph.

The OpenVX graph enables implementers to optimize execution across diverse hardware architectures.

OpenVX enables the graph to be extended to include hardware architectures that don’t support programmable APIs.
OpenVX Status

- OpenVX 1.1 Specification released 2\textsuperscript{nd} May 2016 at Embedded Vision Summit
  - 18 months after OpenVX 1.0 in October 2014
  - Expands node functionality AND enhances graph framework

- OpenVX 1.0 open source sample implementation and conformance tests
  - Will be updated to OpenVX 1.1 in 1H16

- Roadmap discussions
  - Significantly broaden node functionality - including in-graph neural nets
  - OpenVX SC - refining OpenVX for markets requiring API safety certification

= shipping conformant implementation
What’s New in OpenVX 1.1?

• Expanded node functionality AND enhanced graph framework
  - Plus many minor improvements and clarifications

• Laplacian pyramids
  - Computational photography use cases

• Targets - for execution flexibility on heterogeneous devices
  - Application can control on which accelerator to run nodes

• Median, erode and dilate image filters
  - Including custom patterns

• Improved read and write data to and from OpenVX objects
  - Easier to use and less error prone

• Improved API for extending OpenVX with user kernels
  - More convenience and flexibility
Safety Critical Working Group

New Generation APIs for safety certifiable vision, graphics, compute

New Khronos Safety Critical Advisory Panel
Defining guidelines for creating specifications for ISO 26262 and DO-178B/C certification

Many future safety critical use cases involve vision and compute acceleration (e.g. neural nets)
Thank You!

• OpenVX information
  - www.khronos.org/openvx

• OpenVX Discussion Forum - feedback welcome!

• Any company or organization is welcome to join Khronos for a voice and a vote in any of these standards
  - www.khronos.org

• Neil Trevett
  - ntrevett@nvidia.com
  - @neilt3d