The OpenVX Standard API: Computer Vision for the Masses

Kiriti Nagesh Gowda
OpenVX Chair | The Khronos Group
Sr. Staff Engineer | AMD

AMD KHRONOS GROUP

Agenda

- OpenVX - Open, Royalty-free Standard for Computer Vision
- OpenVX 1.3.1 - An Overview
- Conformance
- Vendor Applications - Case Study
- Open-Source Cross-Platform Application - Case Study
- OpenVX - Future Work
- Summary

OpenVX - Open, Royalty-free Standard for Computer Vision

KHRONOS GROUP

Open, Royalty-free Standard for Computer Vision

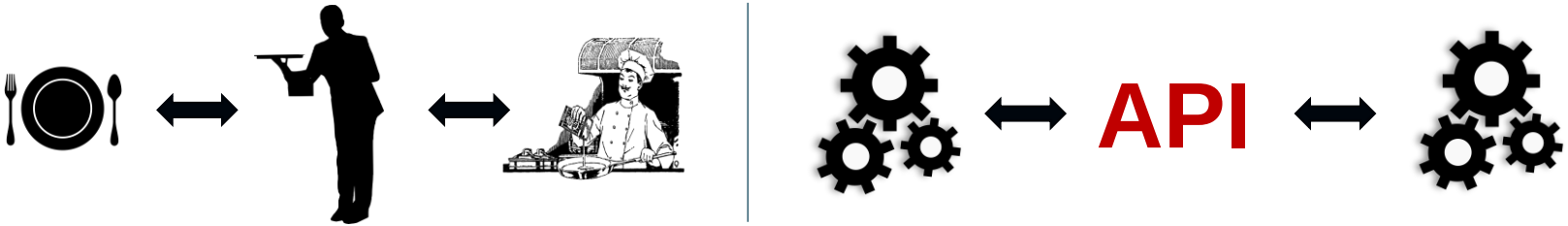
OpenVX™ is an open, royalty-free API standard for a cross-platform acceleration of computer vision applications

High-level graph-based abstraction for portable, efficient vision processing

Open, Royalty-free Standard for Computer Vision

What is an API?

Application Programming Interface



Why APIs are important

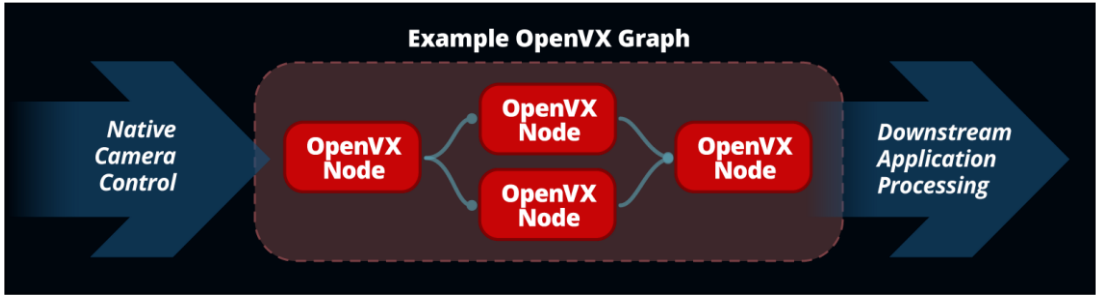
- Building Blocks
- Speeds up development
- Portability
- Innovation

OpenVX enables portable, performance, and power-optimized computer vision processing, especially important in embedded and real-time use cases

Open, Royalty-free Standard for Computer Vision

Open, Royalty-free Standard for Computer Vision

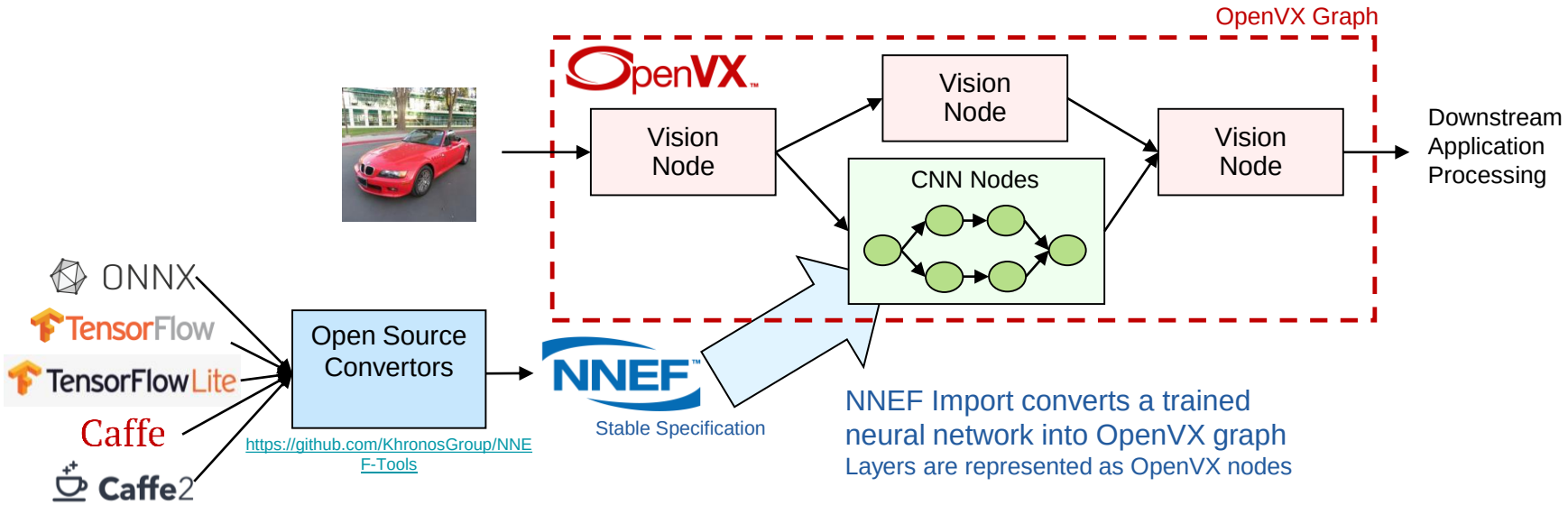
OpenVX™ Graph Framework



- Fuse nodes when possible to achieve better overall performance
- Auto graph-level memory optimizations to achieve a low memory footprint
- Deployed on a wide range of computer hardware, including small embedded CPUs, ASICs, APUs, discrete GPUs, and heterogeneous servers

Open, Royalty-free Standard for Computer Vision

OS Support: Linux, Windows, Android, iOS, Raspbian, Embedded OS, no OS



OpenVX 1.3.1 - An Overview

KHRONOS GROUP

OpenVX 1.3.1 - An Overview

The defined OpenVX 1.3.1 feature sets include:

- **Graph Infrastructure** - baseline for other feature sets
- **Vision** - core vision functionality
- **Enhanced Vision** - functions introduced in OpenVX 1.2
- **Neural Network Inferencing** - including tensor objects
- **NNEF Kernel Import** - including tensor objects
- **Binary Images** – one bit image processing
- **Safety Critical** - reduced features to enable easier safety certification

OpenVX 1.3.1 - Highlights

- Three Conformance Feature Sets:
  - **Vision** - OpenVX 1.1 equivalent vision functions
  - **Neural Network** - OpenVX 1.2 equivalent neural-network functions, plus the neural network extension, and the tensor object
  - **NNEF Import** - kernel import plus the tensor object
- Two Optional Feature Sets:
  - **U1** - binary image support
  - **Enhanced Vision** - vision functions introduced in OpenVX 1.2
- One Organizational Feature Set:
  - **Base Feature Set** - basic graph infrastructure
- One Informational Feature Set:
  - **Deployment Feature Set** - for safety critical usage

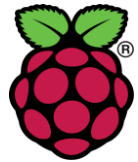
OpenVX - Conformance

OpenVX - Conformance

- OpenVX Work Group provides comprehensive conformance test suite
- Implementations must pass exhaustive conformance test suite to be conformant
- Hardware vendors provide optimized Conformant Implementation of OpenVX drivers
  - Architected to get the best performance from their silicon architecture
  - Ready for developers to use

OpenVX - Conformance

Conformant Implementations of OpenVX from the following vendors



OpenVX - Newest Contributing Member

Khronos Group Welcomes Robert Bosch GmbH as Contributor Member to OpenVX Working Group

- BOSCH working with the OpenVX Group to add multiple extensions to OpenVX

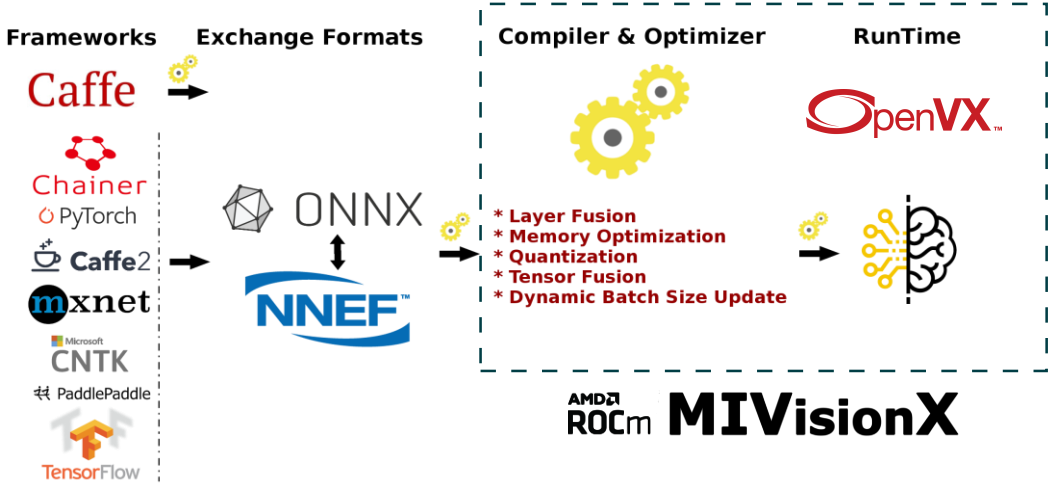
OpenVX - Vendor Applications: A Case Study

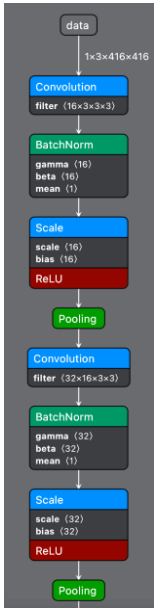
KHRONOS GROUP

OpenVX - Vendor Applications Case Study

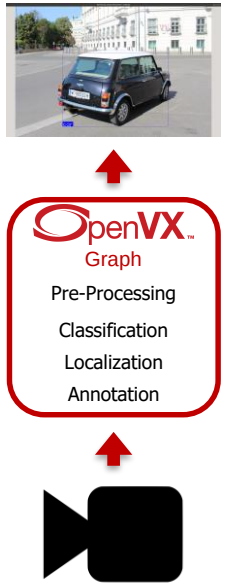
AMD

Application: Object detection with neural networks





AMD ROCm MIVisionX



OpenVX - Vendor Applications Case Study

AXIS COMMUNICATIONS

Application: Reliable motion detection

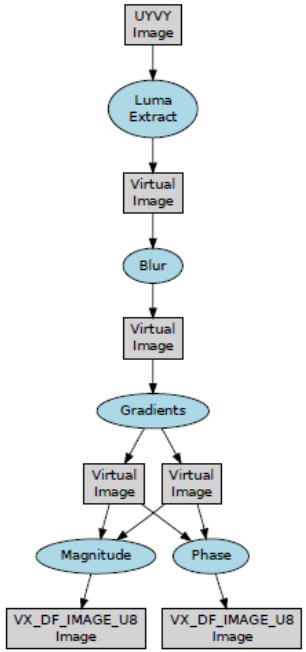
- Uses OpenVX API internally for accelerating algorithm on custom HW blocks
- Compute heavy algorithm for reliable motion detection

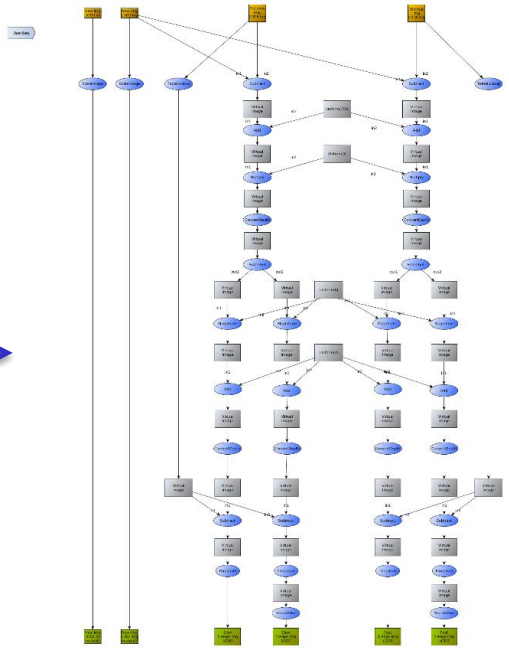
Before OpenVX:

- Hand optimized custom assembler by algorithm developers

After OpenVX:

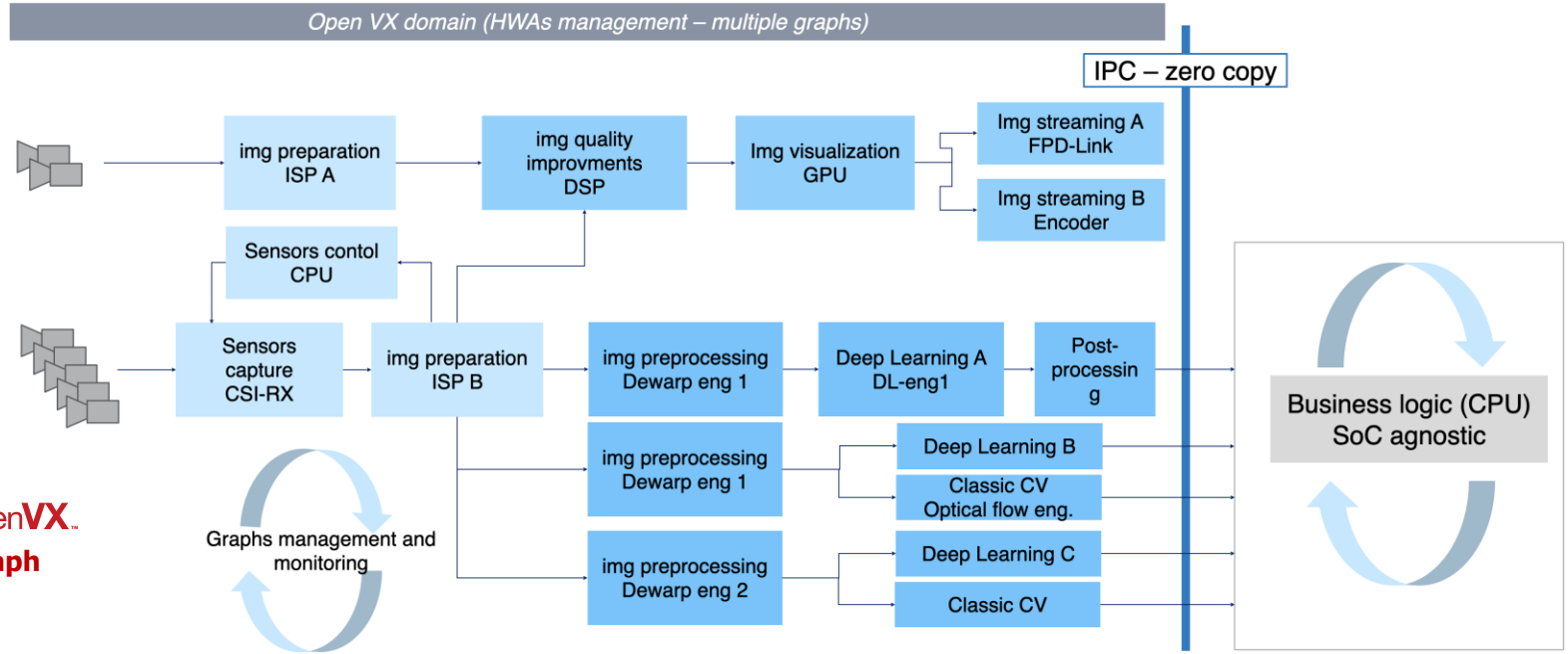
- Algorithm developers "draw" algorithms as graphs
- Driver developers implement the needed graph API





OpenVX - Vendor Applications Case Study

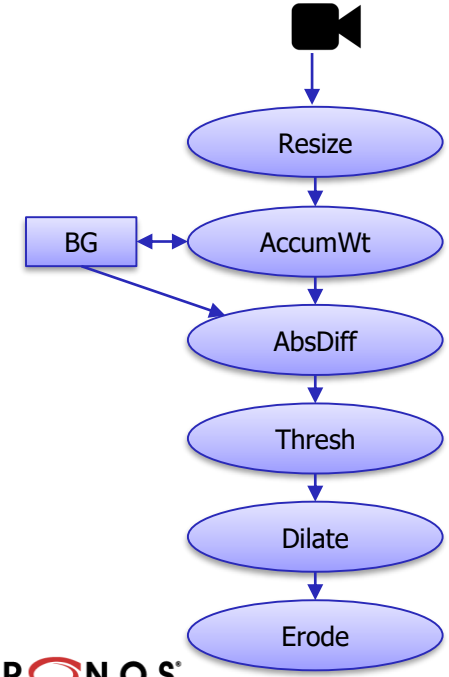
BOSCH

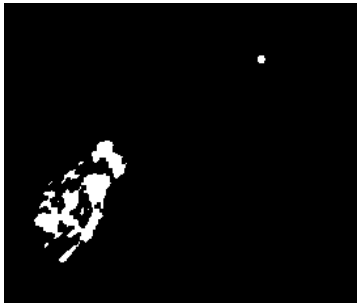


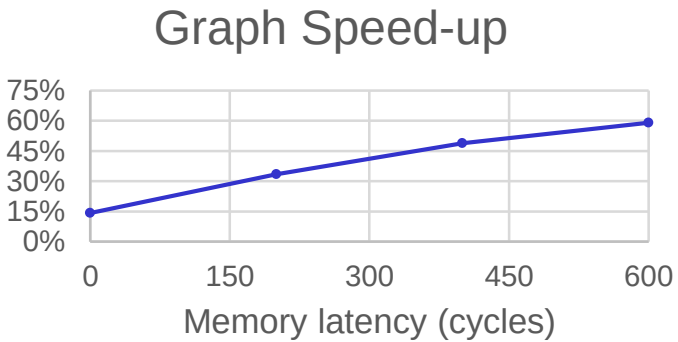
OpenVX - Vendor Applications Case Study

cadence

Application: Background subtraction for video security







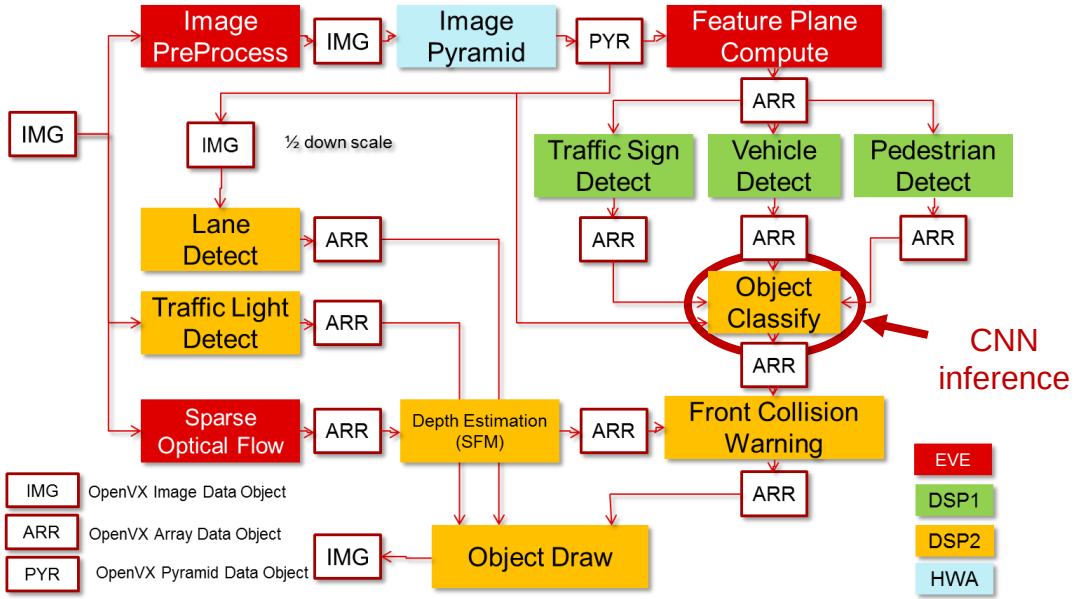
KHRONOS GROUP

OpenVX - Vendor Applications Case Study

Texas Instruments

Application: Front camera ADAS

OpenVX Graph



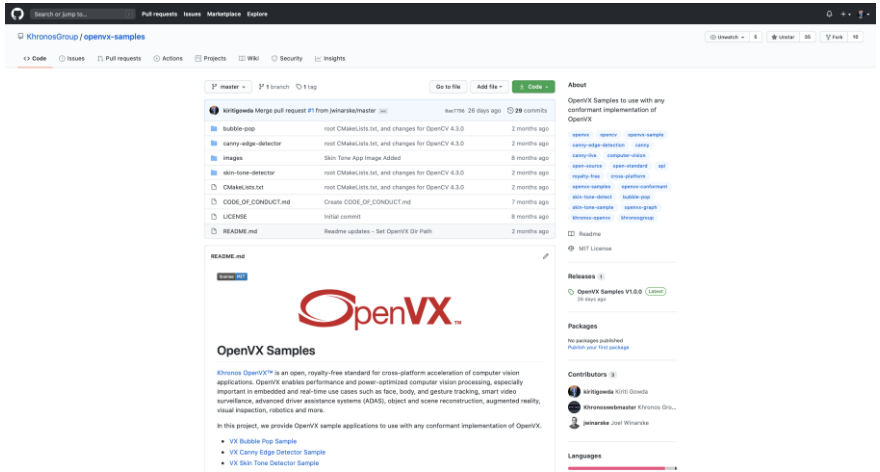
An OpenVX Cross-Platform Application: A Case Study

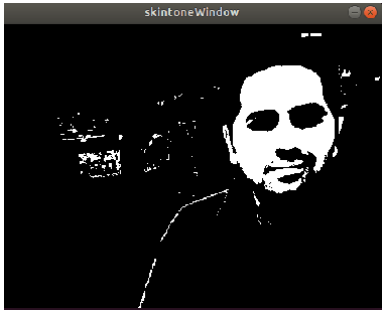
KHRONOS GROUP

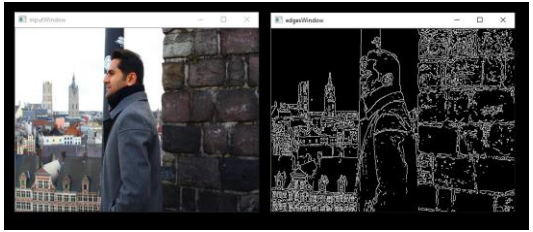
OpenVX Cross-Platform Application: A Case Study

Open-Source OpenVX Samples

Open-Source OpenVX sample applications, to use with **any conformant implementation** of OpenVX available on GitHub



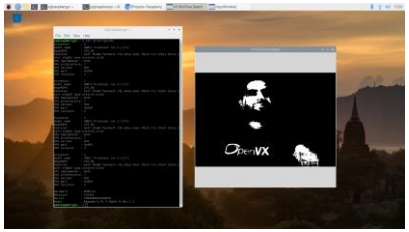




OpenVX Cross-Platform Application: A Case Study

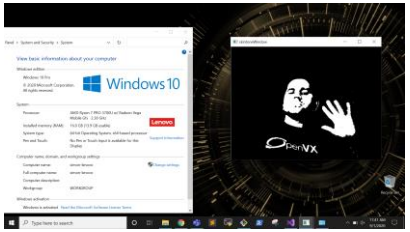
SkinTone Detector Sample

On Raspberry Pi 4 Model B Rev 1.2

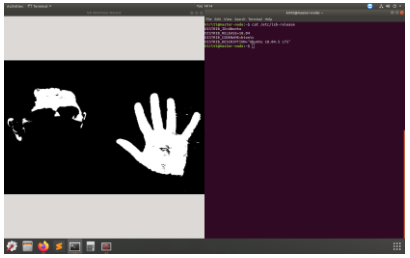


* using open-source OpenVX Raspberry Pi Implementation for OpenVX Libraries

On X86 Processor Windows



On X86 Processor Linux

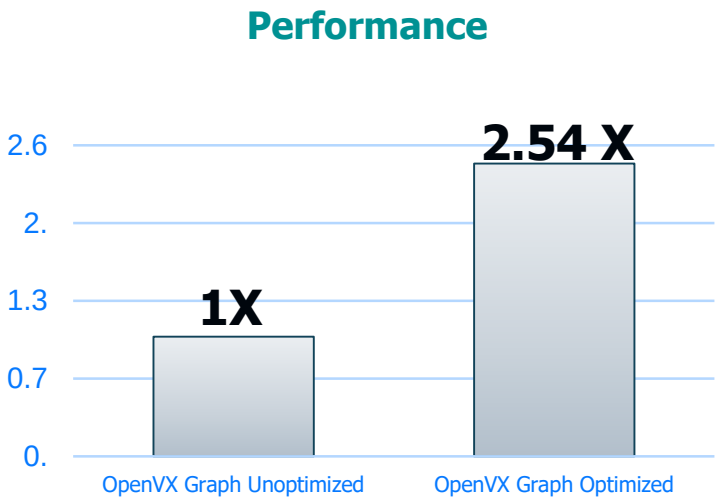


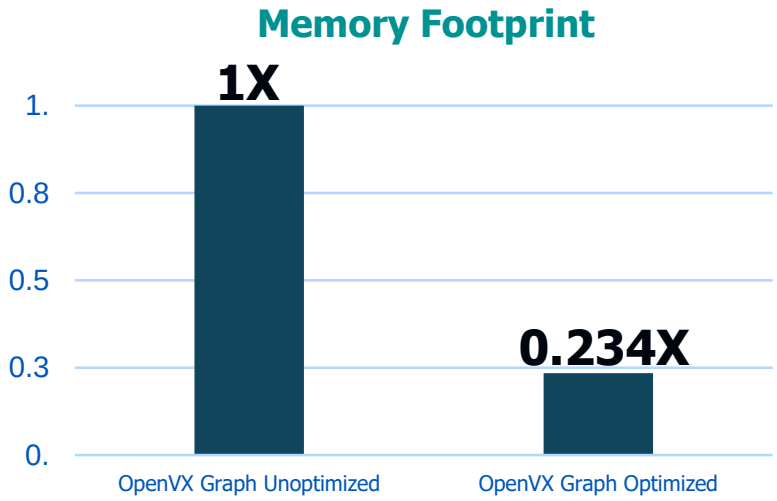
On MacOS

* using AMDs open-sourced MIVisionX for OpenVX Libraries

OpenVX Cross-Platform Application: A Case Study

SkinTone Detector Sample





* using AMDs open-sourced MIVisionX for OpenVX Libraries

OpenVX - Future Work

OpenVX Camera Capture Extension

- OpenVX relies on external libraries for input
  - may not be supported across required operating systems
  - may be too large for embedded systems
  - have different functionality gaps
- OpenVX can use a simple camera capture API library that works everywhere
  - Linux, Windows, Android, iOS, Raspbian, Embedded OS, no OS
  - initialize()/reinitialize(), capture(), release(), getAttributes(): size, format, etc.
- Once the image is captured, OpenVX will wrap it (not copy it) into an vx_image object

OpenVX Camera Capture Extension

OpenVX - Summary

KHRONOS GROUP

Summary

- OpenVX is unique in being the only vision API shipped as an optimized driver
- OpenVX delivers performance comparable to hand-optimized, non-portable code
- Acceleration on a wide range of vision hardware architectures
- OpenVX provides a high-level graph-based abstraction
  - Enables graph-level optimizations
  - Can be implemented on almost any hardware or processor
- Portable, Efficient Vision Processing!

Acknowledgement

Thanks To

- **Mike Schmit – Director of Software Engineering, AMD**
- **Neil Trevett – President, The Khronos Group**
- **AMD's MIVisionX Team**
- **OpenVX Working Group**
- **Embedded Vision Summit 2023 Organizers**

Resource Slide

OpenVX Resources - Khronos

Sample Implementation:

https://github.com/KhronosGroup/OpenVX-sample-impl

Sample Applications:

https://github.com/KhronosGroup/openvx-samples

Tutorial Material:

https://github.com/rgiduthuri/openvx_tutorial

Conformant Implementations

https://www.khronos.org/conformance/adopters/conformant-products/openvx

Khronos OpenVX API Registry

https://www.khronos.org/registry/OpenVX/

OpenVX for Raspberry Pi

https://www.raspberrypi.org/blog/openvx-api-for-raspberry-pi/

AMD ROCm MIVisionX - OpenVX

https://gpuopen-professionalcompute-libraries.github.io/MIVisionX/

Disclaimers and attributions

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

AMD, the AMD Arrow logo, EPYC, Radeon, ROCm and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

BACKUP SLIDES

OpenVX - Open-Source Tools

KHRONOS GROUP

OpenVX - Open-Source Tools

Scripting Tool - RunVX

- RunVX is a command-line tool to execute OpenVX graphs
- It encapsulates most of the routine OpenVX calls, thus speeding up development and enabling rapid prototyping.
- As input, RunVX takes a GDF (Graph Description Format) file, a simple and intuitive syntax to describe the various data, nodes, and dependencies.
- The tool has other useful features, such as, file read/write, data compares, image and keypoint data visualization, etc.

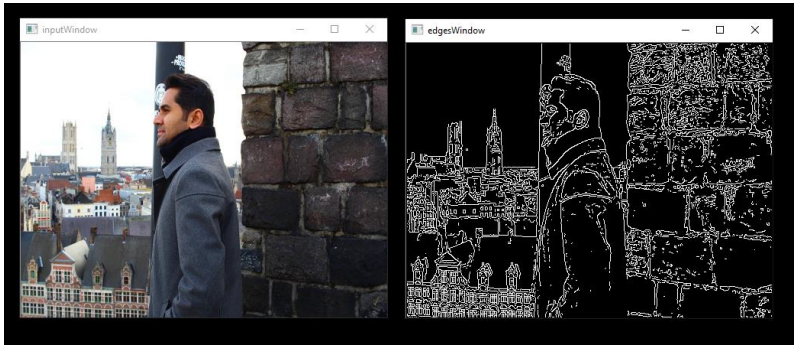
Open-Sourced on GitHub - https://github.com/GPUOpen-ProfessionalCompute-Libraries/MIVisionX

```
# create input and output images
data input  = image:480,360,RGB2
data output = image:480,360,U008

# specify input source for input image and request for displaying input and output images
read input  ../images/face.jpg
view input  inputWindow
view output edgesWindow

# compute luma image channel from input RGB image
data yuv  = image-virtual:0,0,IYUV
data luma = image-virtual:0,0,U008
node org.khronos.openvx.color_convert input yuv
node org.khronos.openvx.channel_extract yuv !CHANNEL_Y luma

# compute edges in luma image using Canny edge detector
data hyst = threshold:RANGE,U008,U008:INIT,80,100
data gradient_size = scalar:INT32,3
node org.khronos.openvx.canny_edge_detector luma hyst gradient_size !NORM_L1 output
```



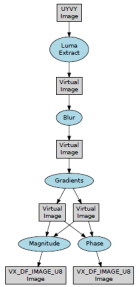
OpenVX - Open-Source Tools

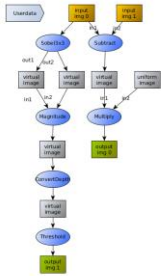
AXIS COMMUNICATIONS

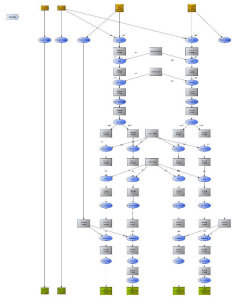
Graph Tool - openvx-graph-parser

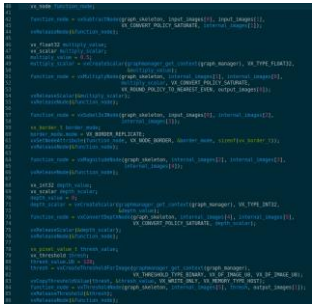
Tool for OpenVX C code generation from graphical calculation graph definitions. Includes a Python based parser which takes graphs defined in graphml format and generates valid OpenVX C code.

Open-Sourced on GitHub - https://github.com/AxisCommunications/openvx-graph-parser









OpenVX - Open-Source Tools

Python Binding - PyVX

pycbindgen is a python based frontend tool for generating python module for a given C library. It uses pycaparser and CFFI python modules underneath.

OpenVX functionality in Python

Will be **Open-Sourced on GitHub** - https://github.com/KhronosGroup