Enabling Augmented Reality Camera Processing, Vision Acceleration and Sensor Fusion

Neil Trevett
Vice President NVIDIA, President Khronos
Khronos Standards

3D Asset Handling
- Advanced Authoring pipelines
- 3D Asset Transmission Format with streaming and compression

Visual Computing
- Object and Terrain Visualization
- Advanced scene construction

3D Asset Handling
- Advanced Authoring pipelines
- 3D Asset Transmission Format with streaming and compression

Camera Control API

OpenVX 1.0 Provisional Released!

Sensor Processing
- Mobile Vision Acceleration
- On-device Sensor Fusion

Over 100 companies defining royalty-free APIs to connect software to silicon

Acceleration in the Browser
- WebGL for 3D in browsers
- WebCL - Heterogeneous Computing for the web

© Copyright Khronos Group 2013 - Page 2
Mobile Compute Driving Imaging Use Cases

- Requires significant computing over large data sets

<table>
<thead>
<tr>
<th>Computational Photography</th>
<th>Face, Body and Gesture Tracking</th>
<th>3D Scene/Object Reconstruction</th>
<th>Augmented Reality</th>
</tr>
</thead>
</table>

Time
Accelerating AR to Meet User Expectations

- Mobile is an enabling platform for Augmented Reality
  - Mobile SOC and sensor capabilities are expanding quickly

- But we need mobile AR to be 60Hz buttery smooth AND low power
  - Power is now the main challenge to increasing quality of the AR user experience

- What are the silicon acceleration APIs on today’s mobile SOCs and OS
  - And how they can be used to optimize AR performance AND power

**SOC** = ‘System On Chip’
Complete compute system minus memory and some peripherals
Why are AR Standards Needed?

State-of-the-art Augmented Reality on mobile today before acceleration

http://www.youtube.com/watch?v=xw3M-TNOo44&feature=related
Where AR Standards Can Take Us

Ray-tracing and light-field calculations running today on CUDA laptop PC - 50+ Watts

Ongoing research to use depth cameras to reconstruct global illumination model in real-time

Need on mobile devices at 100x less power = 0.5W

High-Quality Reflections, Refractions, and Caustics in Augmented Reality and their Contribution to Visual Coherence
P. Kán, H. Kaufmann, Institute of Software Technology and Interactive Systems, Vienna University of Technology, Vienna, Austria
Mobile SOC Performance Increases

- Tegra 2: Dual A9
- Tegra 3: Quad A9, Power saver 5th core
- Tegra 4: Quad A15
- Tegra 4 (2014)
- Full Kepler GPU
- CUDA 5.0
- OpenGL 4.3
- Denver 64-bit CPU
- Maxwell GPU
- HTC One X+
- Google Nexus 7
- Parker

Device Shipping Dates:
- 2011
- 2012
- 2013
- 2014
- 2015

100x perf increase in four years
Power is the New Design Limit

- The Process Fairy keeps bringing more transistors..
- ..but the ‘End of Voltage Scaling’ means power is much more of an issue than in the past

In the Good Old Days
Leakage was not important, and voltage scaled with feature size

\[
\begin{align*}
L' &= L/2 \\
D' &= 1/L^2 = 4D \\
f' &= 2f \\
V' &= V/2 \\
E' &= CV^2 = E/8 \\
P' &= P
\end{align*}
\]

Halve L and get 4x the transistors and 8x the capability for the same power

The New Reality
Leakage has limited threshold voltage, largely ending voltage scaling

\[
\begin{align*}
L' &= L/2 \\
D' &= 1/L^2 = 4D \\
f' &\approx 2f \\
V' &\approx V \\
E' &= CV^2 = E/2 \\
P' &= 4P
\end{align*}
\]

Halve L and get 4x the transistors and 8x the capability for 4x the power!!
Mobile Thermal Design Point

Typical max system power levels before thermal failure
Even as battery technology improves - these thermal limits remain

4-5" Screen takes 250-500mW

7" Screen takes 1W

6-10W

10" Screen takes 1-2W
Resolution makes a difference - the iPad3 screen takes up to 8W!
How to Save Power?

• Much more expensive to MOVE data than COMPUTE data

• Process improvements WIDEN the gap
  - 10nm process will increase ratio another 4X

• Energy efficiency must be key metric during silicon AND app design
  - Awareness of where data lives, where computation happens, how is it scheduled
Hardware Save Power e.g. Camera Sensor ISP

- **CPU**
  - Single processor or Neon SIMD - running fast
  - Makes heavy use of general memory
  - Non-optimal performance and power

- **GPU**
  - Programmable and flexible
  - Many way parallelism - run at lower frequency
  - Efficient image caching close to processors
  - *BUT* cycles frames in and out of memory

- **Camera ISP (Image Signal Processor)**
  - Little or no programmability
  - Data flows thru compact hardware pipe
  - Scan-line-based - no global memory
  - Best perf/watt
Power is the New Performance Limit

- Lots of space for transistors on SOC - but can’t turn them all on at same time!
  - Would exceed Thermal Design Point of mobile devices
- GPUs are much more power efficient than CPUs
  - When exploiting data parallelism can be x10 as efficient - but can go further...
- Dedicated units can increase locality and parallelism of computation
  - Dark Silicon - specialized hardware - only turned on when needed

Enabling new mobile AR experiences requires pushing computation onto GPUs and dedicated hardware
OpenVX - Power Efficient Vision Processing

- Acceleration API for real-time vision
  - Focus on mobile and embedded systems
- Diversity of efficient implementations
  - From programmable processors, through GPUs to dedicated hardware pipelines
- Tightly specified API with conformance
  - Portable, production-grade vision functions
- Complementary to OpenCV
  - Which is great for prototyping
OpenVX Graphs

- Vision processing directed graphs for power and performance efficiency
  - Each Node can be implemented in software or accelerated hardware
  - Nodes may be fused by the implementation to eliminate memory transfers
  - Tiling extension enables user nodes (extensions) to also run in local memory

- VXU Utility Library for access to single nodes
  - Easy way to start using OpenVX

- EGLStreams can provide data and event interop with other APIs
  - BUT use of other Khronos APIs are not mandated
OpenVX 1.0 Function Overview

• Core data structures
  - Images and Image Pyramids
  - Processing Graphs, Kernels, Parameters

• Image Processing
  - Arithmetic, Logical, and statistical operations
  - Multichannel Color and BitDepth Extraction and Conversion
  - 2D Filtering and Morphological operations
  - Image Resizing and Warping

• Core Computer Vision
  - Pyramid computation
  - Integral Image computation

• Feature Extraction and Tracking
  - Histogram Computation and Equalization
  - Canny Edge Detection
  - Harris and FAST Corner detection
  - Sparse Optical Flow
OpenVX Participants and Timeline

- Aiming for specification finalization by mid-2014
- Itseez is working group chair
- Qualcomm and TI are specification editors
## OpenVX and OpenCV are Complementary

<table>
<thead>
<tr>
<th>Governance</th>
<th>Open Source Community Driven No formal specification</th>
<th>Formal specification and conformance tests Implemented by hardware vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Very wide 1000s of functions of imaging and vision Multiple camera APIs/interfaces</td>
<td>Tight focus on hardware accelerated functions for mobile vision Use external camera API</td>
</tr>
<tr>
<td>Conformance</td>
<td>No Conformance testing Every vendor implements different subset</td>
<td>Full conformance test suite / process Reliable acceleration platform</td>
</tr>
<tr>
<td>Use Case</td>
<td>Rapid prototyping</td>
<td>Production deployment</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Memory-based architecture Each operation reads and writes memory</td>
<td>Graph-based execution Optimizable computation, data transfer</td>
</tr>
<tr>
<td>Portability</td>
<td>APIs can vary depending on processor</td>
<td>Hardware abstracted for portability</td>
</tr>
</tbody>
</table>
OpenVX and OpenCL are Complementary

<table>
<thead>
<tr>
<th>Use Case</th>
<th>General Heterogeneous programming</th>
<th>Domain targeted - vision processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>Language-based – needs online compilation</td>
<td>Library-based - no online compiler required</td>
</tr>
<tr>
<td>Target Hardware</td>
<td>‘Exposed’ architected memory model – can impact performance portability</td>
<td>Abstracted node and memory model - diverse implementations can be optimized for power and performance</td>
</tr>
<tr>
<td>Precision</td>
<td>Full IEEE floating point mandated</td>
<td>Minimal floating point requirements – optimized for vision operators</td>
</tr>
<tr>
<td>Ease of Use</td>
<td>Focus on general-purpose math libraries with no built-in vision functions</td>
<td>Fully implemented vision operators and framework ‘out of the box’</td>
</tr>
</tbody>
</table>
Stereo Machine Vision

OpenVX Graph

- Camera 1
  - Stereo Rectify with Remap
  - Compute Depth Map (User Node)
  - Detect and track objects (User Node)
- Camera 2
  - Stereo Rectify with Remap
  - Image Pyramid
  - Compute Optical Flow
  - Frame Delay
  - Object coordinates
Typical Imaging Pipeline

- Pre- and Post-processing can be done on CPU, GPU, DSP...
- ISP controls camera via 3A algorithms
  - Auto Exposure (AE), Auto White Balance (AWB), Auto Focus (AF)
- ISP may be a separate chip or within Application Processor

Need for advanced camera control API:
- to drive more flexible app camera control
- over more types of camera sensors
- with tighter integration with the rest of the system
Advanced Camera Control Use Cases

- High-dynamic range (HDR) and computational flash photography
  - High-speed burst with individual frame control over exposure and flash

- Rolling shutter elimination
  - High-precision intra-frame synchronization between camera and motion sensor

- HDR Panorama, photo-spheres
  - Continuous frame capture with constant exposure and white balance

- Subject isolation and depth detection
  - High-speed burst with individual frame control over focus

- Time-of-flight or structured light depth camera processing
  - Aligned stacking of data from multiple sensors

- Augmented Reality
  - 60Hz, low-latency capture with motion sensor synchronization
  - Multiple Region of Interest (ROI) capture
  - Multiple sensors for scene scaling
  - Detailed feedback on camera operation per frame
Camera API Architecture (FCAM based)

- **No global state**
  - State travels with image requests
  - Every stage in the pipeline may have different state
    - \(\rightarrow\) allows fast, deterministic state changes

- **Synchronize devices**
  - Lens, flash, sound capture, gyro...
  - Devices can schedule Actions
    - E.g. to be triggered on exposure change
    - Enables device synchronization
Android Camera HAL v3

• Camera HAL v1 focused on simplifying basic camera apps
  - Difficult or impossible to do much else
  - New features require proprietary driver extensions
  - Extensions not portable - restricted growth of third party app ecosystem

• Camera HAL v3 is a fundamentally different API
  - Apps can have more control, and more responsibility
    - Enables more sophisticated camera applications
  - Interface is clean and easily extensible
    - Faster time to market and higher quality
  - Flexible primitives for building sophisticated use-cases
    - Adopts some FCAM ideas
Android Camera HAL v3

- Java control API
  - (not released yet)
  - FCam-like direct control per frame
  - E.g. WB and exposure

- Metadata returned
  - Histograms
  - Results of 3A

- Multiple streams
  - Gralloc buffers
  - Output streams can be processed by Java or native APIs
  - Input stream can be re-injected for ‘reprocessing’
Example: Best Shot / Beautification flow

- App requests YUV images
- App picks “best” YUV image and applies beautification guided by face detection
- Image recirculated for ANR and JPEG encoding
Example: Funhouse Video

Advanced Camera Application

Capture Request Queue

Intermediate Results

GPU warping logic

Interface

Camera HAL Device v3

Image capture

Configuration

Meta data

YUV

YUV

Meta data

Media Recorder

Example: Funhouse Video

Advanced Camera Application

Capture Request Queue

Intermediate Results

GPU warping logic

Interface

Camera HAL Device v3

Image capture

Configuration

Meta data

YUV

YUV
Visual Sensor Revolution

• Single sensor RGB cameras are just the start of the mobile visual revolution
  - IR sensors - LEAP Motion, eye-trackers

• Multi-sensors: Stereo pairs -> Plenoptic array -> Depth cameras
  - Stereo pair can enable object scaling and enhanced depth extraction
  - Plenoptic Field processing needs FFTs and ray-casting

• Hybrid visual sensing solutions
  - Different sensors mixed for different distances and lighting conditions

• GPUs today - more dedicated ISPs tomorrow?

Dual Camera
LG Electronics

Plenoptic Array
Pelican imaging

Capri Structured Light 3D Camera
PrimeSense
Khronos Camera API

- Catalyze camera functionality not available on any current platform
  - Open API that aligns with future platform directions for easy adoption
  - E.g. could be used to implement future versions of Android Camera HAL

- Control multiple sensors with synch and alignment
  - E.g. Stereo pairs, Plenoptic arrays, TOF or structured light depth cameras

- More detailed control per frame
  - Format flexibility, Region of Interest (ROI) selection

- Global Timing & Synchronization
  - E.g. Between cameras and MEMS sensors

- Application control over ISP processing (including 3A)
  - Including multiple, re-entrant ISPs

- Flexible processing/streaming
  - Multiple output streams and streaming rows (not just frames)
  - RAW, Bayer and YUV Processing
Camera API Design Milestones and Philosophy

- C-language API starting from proven designs
  - e.g. FCAM, Android camera HAL V3
- Design alignment with widely used hardware standards
  - e.g. MIPI CSI
- Focus on mobile, power-limited devices
  - But do not preclude other use cases such as automotive, surveillance, DSLR...
- Minimize overlap and maximize interoperability with other Khronos APIs
  - But other Khronos APIs are not required
- Provide support for vendor-specific extensions

- Apr13 Group charter approved
- Jul13 1Q14 Provisional specification
- 4Q13 First draft specification
- 1Q14 2Q14 Sample implementation and tests
- 2Q14 3Q14 Specification ratification
How Many Sensors are in a Smartphone?

- Light
- Proximity
- 2 cameras
- 3 microphones
- Touch
- Position
  - GPS
  - WiFi (fingerprint)
  - Cellular (tri-lateration)
  - NFC, Bluetooth (beacons)
- Accelerometer
- Magnetometer
- Gyroscope
- Pressure
- Temperature
- Humidity
Sensor Industry Fragmentation ...
‘Always On’ Camera and Sensor Processing

- Visual sensor revolution - driving need for significant vision acceleration
  - Multi-sensors: Stereo pairs -> Plenoptic arrays -> Active depth cameras
- Devices should be always environmentally-aware - e.g. ‘wave to wake’
  - BUT many sensor use cases consume too much power to actually run 24/7
- Smart use of sensors to trigger levels of processing capability
  - ‘Scanners’ - very low power, always on, detect events in the environment

---

ARM 7
1 MIPS and accelerometers can detect someone in the vicinity

DSP / Hardware
Low power activation of camera to detect someone in field of view

GPU / Hardware
Maximum acceleration for processing full depth sensor capability

© Copyright Khronos Group 2013 - Page 32
Portable Access to Sensor Fusion

Apps request semantic sensor information
StreamInput defines possible requests, e.g.
“Provide Skeleton Position” “Am I in an elevator?”

Apps Need Sophisticated Access to Sensor Data
Without coding to specific sensor hardware

Processing graph provides sensor data stream
Utilizes optimized, smart, sensor middleware
Apps can gain ‘magical’ situational awareness

Advanced Sensors Everywhere
RGB and depth cameras, multi-axis motion/position, touch and gestures, microphones, wireless controllers, haptics keyboards, mice, track pads
StreamInput Concepts

• Application-defined filtering and conversion
  - Set up a graph of nodes to generate required semantics
  - Standardized node intercommunication
  - Filter nodes modify data from inputs
  - Can create virtual input devices

• Node types enable specific data abstractions
  - Unicode text node for keyboards, speech recognition, stylus, etc.
  - 3D skeleton node for depth-sensing camera or a motion-tracking suit ...

• Extensibility to any sensor type
  - Can define new node data types, state and methods

• Sensor Synchronization
  - Universal time stamp on every sample

• Provision of device meta-data
  - Including pictorial representations
StreamInput - Sensor Fusion

- Defines access to high-quality fused sensor stream and context changes
  - Implementers can optimize and innovate generation of the sensor stream

Platforms can provide increased access to improved sensor data stream - driving faster, deeper sensor usage by applications

OS Sensor OS APIs
(E.g. Android SensorManager or iOS CoreMotion)

Middleware
(E.g. Augmented Reality engines, gaming engines)

Low-level native API defines access to fused sensor data stream and context-awareness

StreamInput implementations compete on sensor stream quality, reduced power consumption, environment triggering and context detection - enabling sensor subsystem vendors to increased ADDED VALUE

Middleware engines need platform-portable access to native, low-level sensor data stream

Mobile or embedded platforms without sensor fusion APIs can provide direct application access to StreamInput

Sensor

Sensor Hub

Applications

...
AR needs not just advanced sensor processing, vision acceleration, computation and rendering - *but also for all these subsystems to work efficiently together*
Advanced Mobile Vision App Categories

Mobile Visual Computing Device

3D Reconstruction (constructs 3D geometry)
- User Facing
  - 3D Reconstruction
  - Facial Modeling
- Scene Facing
  - Object Reconstruction
  - Scene Reconstruction

Tracking (constructs positions and motions)
- User Facing
  - Face and gesture tracking
  - Body Tracking
- Scene Facing
  - Environmental Feature Tracking
  - Indoor/Outdoor Positional Tracking
Use Case: IKEA Furniture Catalog

- Select catalog item, and display in own home
  - 8.5m downloads and 600k model placements - mainly Europe
- When have depth sensors -> auto-scaling, no marker needed
Use Case: Environmental WebGL Model Viewer

- Take any WebGL Model and display it in environment
- Auto scaling
- Many 3D model stores coming online
  - TurboSquid
  - Sketchfab etc. etc.

http://sketchfab.com/show/tX9lMamJrkJFziNh2o6mw137D3f
Use Case: Interior Space Capture

• Drive wide adoption through use of standard mobile devices
  - Real Estate, Training, Historical Preservation

• Needs accurate sensor fusion
  - Kinect doesn't work with large spaces, 5 degree drift after 360 degrees
  - Need depth accuracy 1/2 to 3 inches
Object Capture, Search, Share

- 3D’s ‘YouTube Moment’

Capture, Search and Share Loop
Photos -> Facebook
Videos -> YouTube
3D -> Facebook?

Inspire and Motivate

Object and 6D Panorama Capture

3D Printing (e.g. shapeways.com)

Upload, Share and Comment
WebGL FB Plug-in

3D Descriptor Database

Manufacturers provide 3D Object Descriptors - much more information than Cortexica-style 2D search
Summary

• Advanced mobile GPUs and new sensors are combining to make vision processing the next wave of mobile visual computing
• Power is the new performance limiter - developers need vision acceleration to preserve battery life AND provide high performance
• Khronos is building a family of interoperating APIs to enable advanced mobile vision applications

• Join Khronos to influence to direction of mobile vision processing!

• ntrevett@nvidia.com