Cameras are Everywhere

Interactive
Systems that respond to user actions (PC, Gaming, Mobile)
- Motion/Gesture tracking and recognition
- Body tracking

Ubiquitous
“Cameras Everywhere” Distributed Systems (Mobile, Camera, DIY-SOHO)
- Point and shoot
- HDR
- Surveillance

Environmental
Imaging systems that are situationally aware (Camera, Mobile, PC)
- Face Detection/Track
- Gesture tracking
- Object tracking
Mobile Compute Driving Imaging Use Cases

- Requires significant computing over large data sets
How Imaging Sensors Work

Bayer GRBG pattern
- 50% green
- 25% red and blue

Bayer CFA is one type of pattern

http://www.photoaxe.com
Typical Imaging Pipeline

- Pre-processing is non-existent in basic use-cases
- 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)
Typical Image Signal Processor (ISP)

- Simplified Pipeline for illustration purpose
- Non programmable
- Some (re-entrant) pipeline may accept YUV input too
- May be part of a SoC or a separate processor

Statistics
- For AF, AE, AWB
- For adaptation over time of pipeline’s filters
Pipelined Sensor Model

- Traditional one-shot sensor model
  - Need to know which parameters were used
  - \(\rightarrow\) reset pipeline between shots \(\rightarrow\) slow
- Viewfinding / video mode:
  - Pipelined, high frame rate
  - Settings changes take effect later
- Need new model for Computational Photography
  - Need parameterized SEQUENCE of images to feed advanced algorithms
- Real image sensors are pipelined
  - While one frame exposing
  - Next one is being prepared
  - Previous one is being read out

![Diagram of Image Sensor Pipeline](image.png)
High Dynamic Range (HDR)

- HDR works by combining differing exposures into the same image
- A variety of methods for HDR, based on application
  - Multiple frame HDR (requires frame memory)
  - Interlace HDR
  - Multiple Zone HDR

- HDR requires
  - Precise control over camera parameters (exposure)
  - Fast capture and processing of multiple images
  - Note: with interlace HDR, only 1 image is needed
Image stitching, panoramic images

- Made with PTgui

Create high quality panoramic images
Image stitching, panoramic images

- Requires processing of multiple images
- Requires position / geometry information
- Requires control of camera (e.g. AE lock)

• Made with

© Copyright Khronos Group 2013 - Page 11
Typical Burst Sequence Applications

HDR

Low-light imaging

Creative use of flash

All-in-Focus Imaging
Some Existing APIs

• FCAM (2010) Stanford/Nokia, open source
  - Capture of stream of camera images with precision control
    - A pipeline that converts requests into image stream
    - All parameters packed into the requests - no visible state
    - Programmer has full control over sensor settings for each frame in stream
  - Control over focus and flash
    - No hidden daemon running
  - Control ISP
    - Can access supplemental statistics from ISP if available

• Android New Camera HAL (2013)
  - Extends these concepts
Camera, Sensor Fusion and Vision Acceleration
Increasing Use of Imaging Sensors

Perceptual Imaging
1. Uses the full array of mobile sensors
2. to extract information in real-time
3. about the user and environment
4. to generate enhanced user interactions

Photography
Input = 2D Camera
Processors = ISP + CPU
Product = Static Images

Computational Photography
Input = MEMS + 2D Camera
Processors = ISP + CPU + GPU
Product = Real Time Images

Perceptual Imaging
Input = MEMS + Depth Camera
Processors = ISP + CPU + GPU + DSP
Product = Real Time Extracted Information

1. Uses the full array of mobile sensors
2. to extract information in real-time
3. about the user and environment
4. to generate enhanced user interactions
Low Power Environment Scanning

- Many sensor use cases would consume too much power to be running 24/7
  - Environment aware use cases have to be very low power
- ‘Scanners’ - very low power, always on, detect things in the environment
  - Trigger the next level of processing capability

1 MIP and accelerometers can detect someone in the vicinity

Low power activation of camera to detect someone in field of view

GPU acceleration for precision gesture processing
StreamInput Sensor Fusion Stack

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

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

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

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

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

Hardware transport interfaces are defined by each system, e.g. IIO or HID sensor

Sensor

Sensor Hub

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

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

Hardware transport interfaces are defined by each system, e.g. IIO or HID sensor
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
  - Even 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?
OpenVX

- **Vision Hardware Acceleration Layer**
  - Enables hardware vendors to implement accelerated imaging and vision algorithms
  - For use by high-level libraries or apps
- **Focus on enabling real-time vision**
  - On mobile and embedded systems
- **Diversity of efficient implementations**
  - From programmable processors, through GPUs to dedicated hardware pipelines

Dedicated hardware can help make vision processing performant and low-power enough for pervasive ‘always-on’ use
OpenVX Power Efficiency

- OpenVX Graph for power and performance efficiency
  - Each Node can be implemented in software or accelerated hardware
  - Nodes may be fused by the implementation
  - Eliminates transferring the image to and from memory
- EGLStreams can provide data and event interop with other APIs
  - BUT use of other Khronos APIs are not mandated
- VXU Utility Library provides efficient access to single nodes
  - Open source implementation - easy way to start using OpenVX
## OpenVX and OpenCV are Complementary

<table>
<thead>
<tr>
<th>Governance</th>
<th>Open Source</th>
<th>Community Driven</th>
<th>No formal specification</th>
<th>Formal specification and full conformance tests</th>
<th>Implemented by hardware vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Very wide</td>
<td>1000s of functions of imaging and vision</td>
<td>Multiple camera APIs/interfaces</td>
<td>Tight focus on hardware accelerated functions for mobile vision</td>
<td>Use external camera API</td>
</tr>
<tr>
<td>Conformance</td>
<td>No Conformance testing</td>
<td>Every vendor implements different subset</td>
<td></td>
<td>Full conformance test suite / process Reliable acceleration platform</td>
<td></td>
</tr>
<tr>
<td>Use Case</td>
<td>Rapid prototyping</td>
<td></td>
<td></td>
<td>Production deployment</td>
<td></td>
</tr>
<tr>
<td>Efficiency</td>
<td>Memory-based architecture</td>
<td>Each operation reads and writes memory</td>
<td>Sub-optimal power / performance</td>
<td>Graph-based execution Optimized nodes and data transfer</td>
<td>Highly efficient</td>
</tr>
</tbody>
</table>
Khronos APIs for Augmented Reality

MEMS Sensors

Advanced Camera Control and stream generation

Camera Control API

MEMS Sensor Fusion

Precision timestamps on all sensor samples

EGLStream Stream frames between APIs

Feature Tracking and SLAM

Application on CPUs and GPUs

Audio Rendering

3D Rendering and Video Composition On GPU

MEMS Sensors

EGLStream

Precision timestamps on all sensor samples

EGL

Feature Tracking and SLAM

OpenVX

Audio Rendering

Application on CPUs and GPUs

OpenSL ES

3D Rendering and Video Composition On GPU

OpenCL

© Copyright Khronos Group 2013 - Page 22
Advanced Camera Control Requirements

- Generate image bursts with parameterized control
  - For processing by flexible combination of CPU, GPU, ISP and DSP

- Control of new wave sensors - multi-sensor synch and alignment
  - Stereo pairs, Plenoptic arrays
  - Depth Cameras: Time-of-flight and Structured light

- Enable system-wide sensor time-stamping
  - Synchronize MEMS and image sensor samples

- Enable data routing with and to Khronos ecosystem APIs
  - EGLStreams for efficient routing
  - OpenVX, OpenCL, OpenGL etc. for processing
Camera Control API
Camera Control API Goals

- Provide functional portability for advanced camera applications
  - Reduce extreme fragmentation for ISVs wanting more than point and shoot

- Generate image bursts with parameterized camera control and ISP control
  - For downstream processing by flexible combination of CPU, GPU and DSP

- Control multiple sensors with multi-sensor synch and alignment
  - Stereo pairs, Plenoptic arrays, Depth Cameras

- Enable system-wide sensor time-stamping
  - Synchronize MEMS and image sensor samples

- This functionality is not available on any current platform APIs
  - Make this API align with future platform direction for easy adoption
Camera Control API

- Controls flash, lens and sensor
- Controls ISP

Compute/Image/Vision Acceleration APIs

Applications
Camera API Design Philosophy

• C-language API starting from proven designs  
  - e.g. FCAM, Android camera platform
• 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, DSLR...
• Minimize overlap and maximize interoperability with other Khronos APIs  
  - But other Khronos APIs are not required
• Provide support for vendor-specific extensions

Apr13 | Jul13 | 4Q13 | 1Q14 | 2Q14 | 3Q14
---|---|---|---|---|---
Group charter approved | | First draft specification | | Sample implementation and tests | Specification ratification
Participating Companies

True Multi-vendor initiative including sensor, processor and platform vendors
Khronos Camera Control Working Group

• Royalty free API for portable access to advanced mobile camera functionality
  - Reduce fragmentation and encourage more advanced camera applications

• Control for the new wave of sensors to enable advanced imaging and vision
  - Multiple sensors, depth cameras, synchronized sensors

• Provide sophisticated camera functionality not available on today’s platforms
  - But work to enable easy adoption by platform vendors

• Eager to contribute? Join Khronos Camera WG!
  - Detailed design is just starting - great time for your input and requirements!
  - http://www.khronos.org/camera

• Mikaël Bourges-Sévenier
  - msevenier@aptina.com