NVIDIA Mobile Computer Vision

- Focus on Mobile Vision Applications
- Optimize core algorithms for Tegra hardware
- Enable Developers
Graphics

Render Images From Scenes

Inverse Problems

Massively Parallel

Computer Vision

Understand Scenes From Images
Perception

- Where is the device?
- What’s nearby?
- Who’s nearby?
- What is the user doing?

Interaction

NVIDIA Confidential
Computer Vision = Smart Photography
Get the right shot, automatically

Face Detection

Scene Classification

Stabilization
Computer Vision = New Applications

Augmented Reality

- Augmented Reality Ghost Hunter (Argh)
- Wordlens

Gesture interfaces

- Google Goggles
Computer Vision = Vehicle Safety

New limit: 80 km/h

[country: EU; mode: video]
Traditional cameras vs. camera phones

<table>
<thead>
<tr>
<th>Year</th>
<th>Film cameras</th>
<th>Digital cameras</th>
<th>Camera phones</th>
</tr>
</thead>
<tbody>
<tr>
<td>1999</td>
<td>10</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>2000</td>
<td>10</td>
<td>10</td>
<td>50</td>
</tr>
<tr>
<td>2001</td>
<td>10</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>2002</td>
<td>10</td>
<td>30</td>
<td>200</td>
</tr>
<tr>
<td>2003</td>
<td>20</td>
<td>40</td>
<td>300</td>
</tr>
<tr>
<td>2004</td>
<td>30</td>
<td>50</td>
<td>400</td>
</tr>
<tr>
<td>2005</td>
<td>40</td>
<td>60</td>
<td>500</td>
</tr>
<tr>
<td>2006</td>
<td>50</td>
<td>70</td>
<td>600</td>
</tr>
<tr>
<td>2007</td>
<td>60</td>
<td>80</td>
<td>700</td>
</tr>
<tr>
<td>2008</td>
<td>70</td>
<td>90</td>
<td>800</td>
</tr>
<tr>
<td>2009</td>
<td>80</td>
<td>100</td>
<td>900</td>
</tr>
</tbody>
</table>
## Trends in camera phone sales

**Sales keep growing**

<table>
<thead>
<tr>
<th>Year</th>
<th>Sales (Million)</th>
<th>Percentage of Phones</th>
</tr>
</thead>
<tbody>
<tr>
<td>2003</td>
<td>85</td>
<td>16%</td>
</tr>
<tr>
<td>2010</td>
<td>805</td>
<td>65%</td>
</tr>
<tr>
<td>2014</td>
<td>1.3</td>
<td>85%</td>
</tr>
</tbody>
</table>

**Average resolution grows too**

<table>
<thead>
<tr>
<th>Year</th>
<th>Resolution</th>
<th>2010 5+MP:</th>
<th>2014 5+MP:</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008</td>
<td>1 MP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2009</td>
<td>2 MP</td>
<td>~ 13%, &gt;100 million</td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td>5+MP: ~ 13%, &gt;100 million</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>5 MP</td>
<td>~ 42%, &gt;550 million</td>
<td></td>
</tr>
</tbody>
</table>
Mobile Computing on a Tear

Cumulative Shipments

- iOS & Android
- MacOS & Windows

Source: Gartner, Apple, NVIDIA
TEGRA
The Mobile Super Chip
Tegra 2 – Heterogeneous Multi-core

**CPU**
- Dual Cortex-A9, up to 1GHz

**GRAPHICS**
- 8 Core ULP GeForce

**VIDEO**
- 1080P H.264

**MEMORY**
- LPDDR2 – 600, DDR2 - 667

**IMAGING**
- Ultra High Performance Image Processor

**AUDIO**
- HW Audio

**STORAGE**
- eMMC, NAND, USB
ARM is Pervasive and Open

Annual Shipments

- ARM
- x86

Units in Billions

Source: ARM, Mercury Research, NVIDIA
PROJECT KAL-EL

- World’s first mobile quad-core CPU
- New 12-Core NVIDIA GPU with support for 3D stereo
- Extreme HD – 2560x1600
- 5X Tegra 2
TEGRA ROADMAP

- Tegra 2 (2010)
- Kal-El (2011)
- Wayne (2012) 5x performance
- Logan (2013)
- Stark (2014)

Y-axis: Performance
X-axis: Years (2010-2014)
Software Solutions for Mobile Computer Vision
OpenCV

Thousands of Developers, Cross Platform API

- Open standard for Compute Vision
- Analogous to OpenGL for Graphics
- 12 years old, professionally developed
- Optimized for x86 SSE, CUDA GPU
- Over 3 Million Downloads!
- > 500 Algorithms

Common API for Server, Workstation, Desktop and now Mobile Platforms!
OpenCV Functionality Overview

Image processing
- General Image Processing
- Segmentation
- Machine Learning, Detection
- Image Pyramids
- Transforms
- Fitting

Video, Stereo, and 3D
- Camera Calibration
- Features
- Depth Maps
- Optical Flow
- Inpainting
- Tracking
OpenCV for CUDA

- Open Source
- Supported by NVIDIA
- Beta with OpenCV 2.2
- Release with OpenCV 2.3
Current OpenCV Cuda Functionality

- Initialization, Information, Data allocation & movement
- Per-element operations (add, subtract, logical)
- Color Conversion
- Geometrical transforms (rotate, warp, scale, remap)
- Mean-Shift transforms (filtering, segmentation)
- Corner Detectors
- Reductions with/without mask (norm, sqrSum, integral)
- Template Matching
- Filter Engine (convolutions, blur, morphology)
- Histograms
- Stereo Correspondence
- Histogram of Gradient (HOG) descriptors (pedestrian detection)
- Haar features, face detection
- Feature detection for object recognition (Speeded Up Robust Features, SURF)
- Other Image transforms (flip, LUT, split, phase, dFFT Canny)
- Panorama stitching
- Point Cloud Library support (in progress)
Android

- Fastest Growing OS Ever
- 39% of mobile market (vs. iOS 28%)*

*Source: http://blog.nielsen.com/nielsenwire/online_mobile/in-u-s-smartphone-market-android-is-top-operating-system-apple-is-top-manufacturer/
NVIDIA Confidential
OpenCV for Android

OpenCV 2.3.1 for Android:
- Native Android Camera Support
- Multithreading
- Java API
- Tegra HW Optimizations (soon)

Wiki with the latest information:
http://opencv.willowgarage.com/wiki/Android

Support/discussion group:
https://groups.google.com/group/android-opencv

Released Today!
OpenCV on Tegra

Optimized for ARM, Tegra & Android

Bringing the most popular Computer Vision Library to the worlds most popular Processor Architecture on the Fastest Growing OS
Tegra Android™ Development Pack

“All-in-one” installer for Android NDK development under Eclipse

Includes:
- Tegra OS Images (Gingerbread / HoneyComb)
- Android SDK
- Android NDK
- Java Development Kit
- Cygwin
- Apache Ant
- PerfHUD ES
- OProfile
- Eclipse (including ADT and CDT plug-ins)
- Nvidia Debug Manager for Android (NVDM)
- Tegra SDK Samples
- Flash Android OS Images to the Tegra DevKit
- Import + compile Tegra SDK Samples in Eclipse

http://developer.nvidia.com/tadp
Android Apps

Java Applications
- Java Code
  - Java Compiler
- Java Bytecode
- Dalvic Bytecode
- Dalvic Compiler
- Hardware Native Assembly

Native Applications
- Java Code
  - JNI Wrapper
- C/C++
  - gcc
- Hardware Native Assembly

Optimized for mobile, low power
OpenCV supports Java and Native

- Most Android apps written in Java (SDK)
- Use Native Applications (NDK) for high performance or hardware optimization (e.g. Neon)

Simply using OpenCV functions: **JAVA**

Writing your own computer vision algorithms: **NATIVE**

Advanced camera control (FCAM): **NATIVE**
OpenCV Java API Example

- Import the needed OpenCV modules into your Java source files

```java
import org.opencv.core.Mat;
import org.opencv.Size;
import org.opencv.core;
import org.opencv.utils;
import org.opencv.improc;
```

```java
public class SimpleJavaOpenCVActivity extends Activity {
    private static final String TAG = "SimpleJavaOpenCV\:Activity";
    private MenuItem item_preview, item_canny, item_sobel, item_blur;
    public static final int view_mode_preview = 0,
                    view_mode_canny = 1,
                    view_mode_sobel = 2,
                    view_mode_blur = 3;

    public int view_mode;
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
```
OpenCV Java API

- Java OpenCV code is just like C++ OpenCV code!

**Create an OpenCV Mat from Android Camera data**

**Use common OpenCV functions**

**Make an Android Bitmap From an OpenCV Mat**

```java
Canvas canvas = holder.lockCanvas();
Mat yuv = new Mat(h, h/2, w, MatType.CV_8UC1);
yuv.put(0, 0, data);
Mat rgba = new Mat(h, w, MatType.CV_8UC4);

SimpleJavaOpenCVActivity a = (SimpleJavaOpenCVActivity)getContext();
int view_mode = a.getView_mode();
if(view_mode == SimpleJavaOpenCVActivity.view_mode_preview) {
    improc.cvColor(yuv, rgba, improc.CV_YUV420I2RGB, 4);
} else if (view_mode == SimpleJavaOpenCVActivity.view_mode_canny) {
    Mat gray = yuv.submat(0, h, 0, w);
    Mat edges = new Mat(h, w, MatType.CV_8UC1);
    improc.canny(gray, edges, 80, 100);
    improc.cvColor(edges, rgba, improc.CV_GRAY2BGRA, 4);
} else if (view_mode == SimpleJavaOpenCVActivity.view_mode_sobel) {
    Mat gray = yuv.submat(0, h, 0, w);
    improc.sobel(gray, gray, core.CV_8U, 1, 1);
    improc.cvColor(gray, rgba, improc.CV_GRAY2BGRA, 4);
} else if (view_mode == SimpleJavaOpenCVActivity.view_mode_blur) {
    improc.cvColor(yuv, rgba, improc.CV_YUV420I2RGB, 4);
    improc.blur(rgba, rgba, new Size(15, 15));
}

Bitmap bmp = Bitmap.createBitmap(w, h, Bitmap.Config.ARGB_8888);
utils_MAT_TO_BITMAP(rgba, bmp);
canvas.drawBitmap(bmp, (canvas.getWidth()-w)/2, (canvas.getHeight()-h)/2, null);
```
Thank You to

For continuous support and innovation in OpenCV
**VISION:** Computer vision — machines that see and understand their environments — can bring compelling capabilities to many applications

**EMBEDDED VISION:** Embedded vision brings computer vision into the mainstream, by incorporating it into a wide variety of electronic products cost-effectively

**EMBEDDED VISION ALLIANCE:** The Embedded Vision Alliance is an industry partnership seeking to transform the electronics industry by inspiring and empowering engineers to design systems that see and understand

The Alliance’s first initiative is [www.Embedded-Vision.com](http://www.Embedded-Vision.com) — the engineer’s go-to resource for embedded vision technology information and know-how
Visit **www.Embedded-Vision.com** – the engineer’s go-to resource for embedded vision technology information and know-how
THANK YOU