OpenGL ES in the Mobile Graphics Ecosystem

Tom Olson

OpenGL ES working group chair

Director, Graphics Research, ARM Ltd

Outline

- Why Mobile Graphics?
- OpenGL ES Overview
- Getting Started with OpenGL ES
- Conclusion

Why Mobile Graphics?




- Mobile graphics is growing quickly
  - Embedded platforms will eventually outnumber desktops
- Factors driving the trend
  - Demand: Consumers want it
  - Technology: Displays, GPUs, CPUs, batteries, memory
  - Standards: Operating systems, APIs
  - … and the commercial infrastructure is emerging - the rise of open distribution channels is creating a market for applications

What is OpenGL ES?

- OpenGL ES is the dominant C/C++ API for Mobile 3D
  - Products from every major mobile phone manufacturer
  - Moving rapidly into other mobile and embedded devices

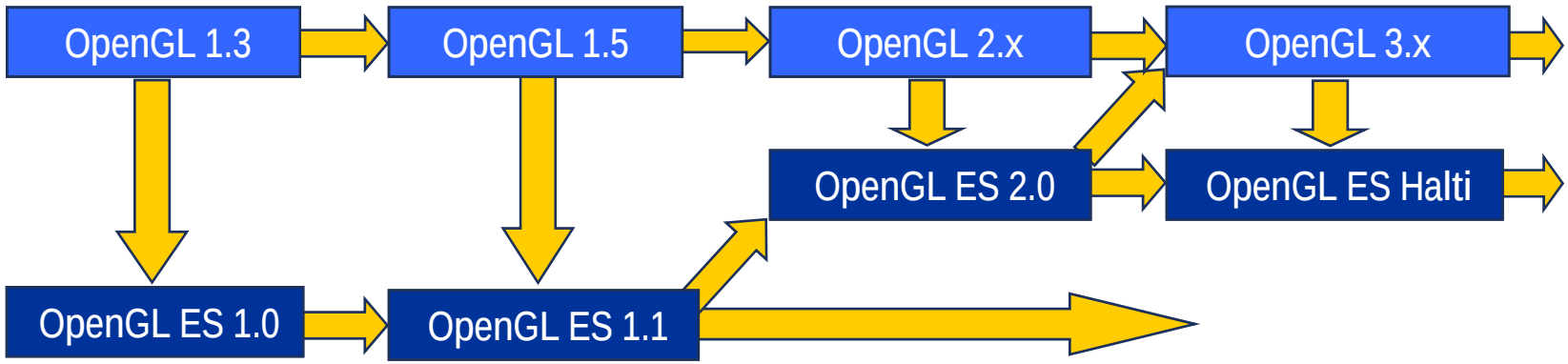
- Available for (or a standard part of) most mobile OSes
  - Symbian
  - Mac OS X on iPhone
  - Linux: Maemo, Android, etc
  - BREW
  - Windows Mobile / WinCE
  - Palm OS

- Over 54 million hardware-accelerated OpenGL ES platforms have shipped to date. Many more are coming.

OpenGL ES Features

- Based on desktop OpenGL
  - Leverages the OpenGL ecosystem
  - Extensible to allow innovation / evolution
- Optimized for mobile devices
  - Gets rid of redundancy & rarely-used features
  - Adds mobile-friendly data types
- Full-Featured
  - All the most used desktop features are available
- *OpenGL ES gives you the power of OpenGL in a much smaller package*

OpenGL ES Versions

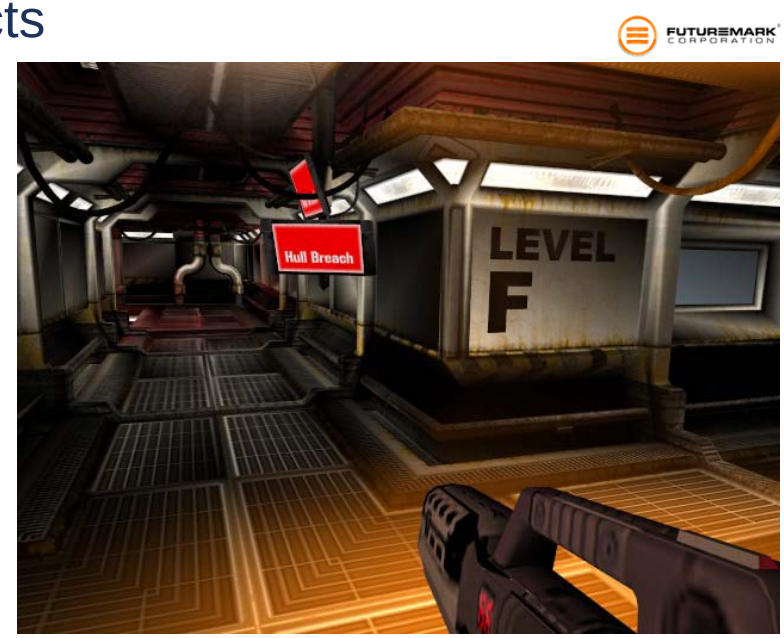


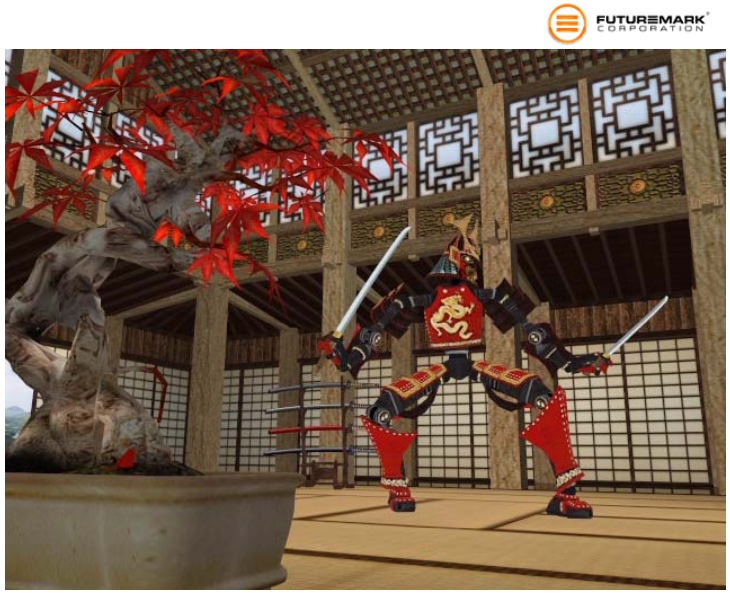
- OpenGL ES Versioning Model
  - OpenGL ES design philosophy is to minimize redundancy
  - When new versions add better methods, old methods are dropped
  - Note the difference from desktop GL!

- OpenGL ES Versions
  - OpenGL ES 1.x: fixed function graphics
  - OpenGL ES 2.0: shader based graphics

OpenGL ES 1.1

- Key Features
  - Vertex Arrays / Vertex Buffer Objects
  - Full Transform & (almost) Lighting
  - Multitexturing (min 2 units)
  - DOT3 bump mapping
  - Fixed Point & Float profiles





Features Removed

- Begin / End
- Color Index Mode
- Imaging Subset
- Quads/Polygons

OpenGL ES 2.0

- Key Features
  - Vertex / fragment shaders
  - Removes fixed function pipeline
  - High level language (GLSL ES 1.0)
  - On-line or off-line compilation
  - Super-compact, efficient API

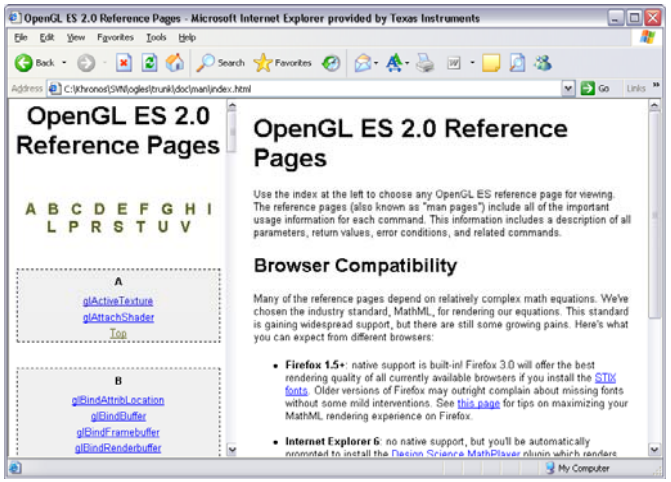
Advanced Feature Set
- Eight attribs / varyings / textures
- Cube Maps
- Dependent texture reads
- Framebuffer Objects

What comes next?

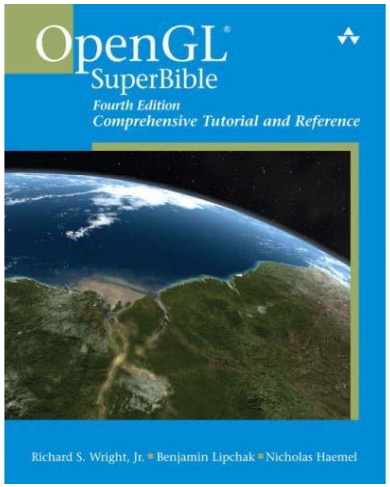
- Working group is committed to supporting OpenGL ES 2.0
  - Working actively on conformance testing program
  - Ongoing projects to improve documentation, tools, ecosystem
- But, work has begun on OpenGL ES ‘Halti’
- Tentative Goals
  - Maintain compatibility with OpenGL ES 2.0
  - Minimize / reduce differences from desktop OpenGL
    - Working closely with OpenGL ARB
    - Note similarity of OpenGL ES 2.0 to non-deprecated subset of OpenGL 3.0
  - Improve driver efficiency
  - Adopt the most modern / advanced features from OpenGL

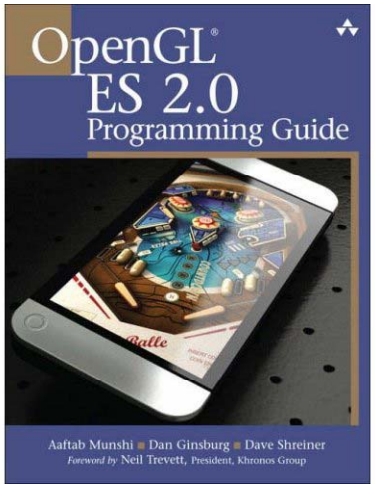
Getting Started: Information

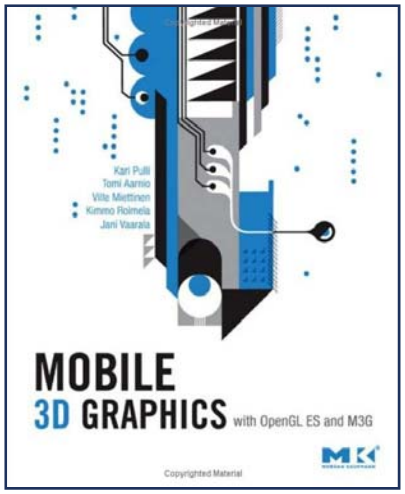
- Khronos.org
  - News: http://www.khronos.org
  - Khronos OpenGL ES API registry : ~/registry/gles
  - OpenGL ES 2.0 man pages: ~/opengles/sdk/docs/man
- Developer Sites
  - ZeusCMD, etc (tutorials)
  - Beware of platform dependencies



Getting Started: Information







- Books
  - Most new OpenGL books have some ES material
  - Some books focus exclusively on OpenGL ES
  - Check dates and API versions covered

Getting Started: Development Tools

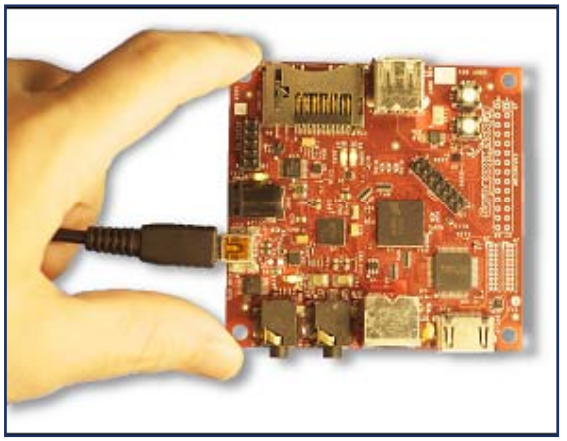
- ES 1.x Desktop Implementations
  - Vincent (open source)
  - Imagination PowerVR Insider OpenGL ES 1.1 SDK
  - *Note: Gerbera is no longer available*
- ES 1.x Desktop+embedded SDKs
  - Series 60 SDK from Nokia
  - Symbian SDK from Sony-Ericsson
  - iPhone SDK from Apple
- ES 2.0 Desktop Implementations / SDKs / Tools
  - AMD OpenGL ES 2.0 Emulator
  - PowerVR OpenGL ES 2.0 SDK (from Imagination)
  - AMD Rendermonkey 1.8 (or higher - GLSL ES only)
  - ARM Desktop Emulators (available 2H 2009)

About Open GL ES 2.0 Emulators

- What they do
  - Emulate OpenGL ES 2.0 on desktop hardware
  - Map ES function calls to a desktop OpenGL 2+ implementation
- Desktop Emulators: Pro
  - Easy way to get started
  - Low cost if you have the right 3D card
  - Great C/C++ debug environment
- Desktop Emulators: Con
  - Only work with certain graphics cards
  - Emulation isn't perfect
  - Performance, precision, language differences not handled well
  - Demoing on desktop lacks 'cool factor'

Embedded ES 2.0 Development Tools

- Beagle Board
  - Open Source board + SW
  - Low cost ($150 US)
  - TI OMAP: ARM Cortex A8 + SGX GPU
  - Linux (various)
  - Currently has a steep learning curve



- ARM mass-market development board
  - ARM CPU + Mali™ GPU
  - Available 2H 2009

Practical Considerations

- What to expect from mobile platforms:
- CPU Speed
  - 300 to 600 MHz
  - High-end platforms will go multicore soon
- GPU Speed
  - Peak fill rates of 200-500MHz in next generation
- GPU Architecture
  - Most devices use deferred rasterization (tiling or chunking)
  - Few have DXT*; other texture compression (ETC1) is often available
- Other Considerations
  - No virtual memory
  - Limited file storage
  - Slow networks

Conclusion

- Mobile Graphics is here now!
  - The market is exploding
- The technical possibilities are exciting
  - The HW / SW platforms are increasingly sophisticated
  - Performance is good and getting better
- OpenGL ES is a great low-level graphics API
  - Mobile friendly
  - Powerful, modern feature set
  - Simpler and easier to learn than desktop APIs
- You can get started with it today!