GPUs in a Flash

Lee Thomason | Flash Player Architect
Mapping the Software Vector Renderer to the GPU

- Brief Overview of Flash Rendering
- A Journey of Technology and API
- Deployment
The Problem

Today’s web, but faster, mobile, and using less power.

- Existing web?
- New content for mobile?
- Video?
The Flash Rendering Model

http://www.krazydad.com/bestiary/bestiary_yinyang.html
Anti-Aliasing

- **Software:** clever 16x box filter (4x mobile)
  - Distinguishes coverage vs. transparency
- High DPI good!
- ...but no AA bad

MSAA
Edge Abutment

- One write per pixel. Not a painters algorithm.
- Edges abut exactly
- No light leaks
- Renderer correctly distinguishes transparency and coverage
(Almost) Everything is Transparent

- In practice so wildly common it’s hard to optimize for opaque cases.
- Still one write per pixel.
Quadratic Beziers

- Odd choice from tooling perspective.
- Handy for the runtime.
Key renderer traits

- Anti-aliased
- Edge abutment
- Transparency
- Quadratic Beziers
Please Note

- Only sharing experiments with public tech.
- Only showing prototype.
mudbubble.com
Approach

Started with technology.
Tessellation

- **Sweep**
- **Seems** straightforward.
- Ugly edge cases
  - Bezier intersections
  - Floating point issues
- Detail on one side impacts other side.
- “Stripey”
Stencil
Stencil

- Works for any closed path!
- Simple
- Fun optimization problems
- Object or world space
Stencil
Mobile first!

Stencil was optional. Never available in *any* HW we tested.
Logic Ops

- Clever abuse of the logic ops. XOR + DST Blend looks a lot like stencil.
- Lots of edge cases (masking) hard to get right
- Fill bound
Next implementation: OpenVG

- Coolest thing ever.
- Prototype in a week!
- We contributed changes to VG 1.1 to support Flash
Flash Changes

- Flash 10
  - Supports 2D shaders (PixelBender)
  - Future support for 3D
- FlashLite
  - VG support shipping
OpenGL ES 2.0 Arrives!

- Simple.
- Consistent across our platforms.
- Can be emulated on other APIs.

Basis for flat acceleration and 3D for foreseeable future.
Implementation: OpenGL ES 2.0

- The return of stencil
- Shaders
Fill rate

- Always need more fill.
- Content notorious for unseen objects.
Tests

- Depth is used for strokes.
- Stencil is used for masking.
Back but better!

- 2 stage approach
  - “Convex” pieces
  - Triangulation is adaptive

- Spent time on edge cases
  - Integer coordinates
Tessellation

- **Pros**
  - Maps well to 3D cards, how games work
  - Lots of optimization room
  - Frees stencil
  - Uses less fill

- **Cons**
  - Unusual (but real) content fail
  - Integer object space
  - Still want more fill

**Working!**
Practicalities

- Drivers. Drivers. Drivers.
- OS support for GPUs.
- Video integration. (Go Tegra!)
Current Status

- NVIDIA has been working closely with Adobe to optimize performance and battery consumption
- Tegra 2 is being built into multiple Android phones and tablets with a fully accelerated Flash Player