Flash Lite and OpenVG

Angus Davis
Mobile Client Engineering
Adobe Systems Incorporated

August, 2008
Overview

- Flash Lite
- OpenVG 1.0, OpenGL ES 1.1, OpenGL ES 2.0
- Performance
- Challenges
- Lessons Learned
- Conclusion
Flash Lite

- Current version: Flash Lite 3
- Used to run Flash content/UI on devices
- Subset of Flash 8 (w/o filters/blend modes)
- Solid, bitmap, linear/radial/focal gradient fills
- Stroking with SVG style joins and caps
- Color transforms (scale & bias for each RGBA color channel)
- Multi-level vector shape masking
- Video
- Software renderer can’t keep up with growing screen sizes
Limitations of OpenVG 1.0

- **Masking memory usage**
  - Need to render mask shape to VGimage
  - VGImage is 32 bit/pixel since can’t create EGL surface with A8 image

- **No color transforms on images**
  - Need to apply CT to image copy and then update VGImage
  - Increases memory usage and data traffic
Limitations of OpenGL ES 1.1 and OpenGL ES 2.0

- Tessellation of vector shapes by CPU – most shapes are complex
- Generation of stroke outlines by CPU
- Non existent stencil buffer – useful for tessellation
- Texture must be generated for radial gradients
- Masking done using depth buffer (limit 2 layers)
- Caching tessellation is required for performance but increases memory usage
- Tessellation of curved edges depends on mapping to screen
- More data movement (vertices vs. paths)
- OpenGL ES 2.0 is much better than OpenGL ES 1.1
  - Color transforms & radial gradients can be done using shaders
  - Stencil buffer can be used for tessellation
  - Tessellation and data traffic are still an issue
Performance: Software vs. OpenGL ES 1.1

- Tests on Nokia N93, 240x234, RGB565
- Best result is 33% improvement with OpenGL ES
Performance: Software vs. OpenVG 1.1

- Tests on AMD A250, WVGA
- 300MHz ARM11 with 150MHz Z160
- 4-5X performance improvement with OpenVG
Implementation challenges with OpenVG 1.1

- Converting Flash shapes to VG paths
  - Flash shape is a collection of edges which divide space into regions
  - Need to assemble an unordered set of edges into an ordered set of edges

- Video performance
  - Video decoders video frame to RGB buffer
  - VGImage updated
  - VGImage drawn to surface

- Device text performance
  - Host renders text to frame buffer
  - VGImage created from frame buffer
  - VGImage drawn to screen
  - OpenVG 1.1 font API requires application to provide layout information
Limitations of OpenVG 1.1 in supporting Flash 10 and beyond

- Filter performance (not for real-time)
- Missing blend modes
- Programmability via shading language
- Better than 4x MSAA
- Integration with other Khronos APIs (OpenGL ES)
Lessons learned from Flash Rendering with OpenVG 1.1

- Use dirty regions, avoid updating entire screen
- Use scissor rects for rectangular masks, update regions
- Masking is slow
- Stroking is slow
- Examine VG call log
- Cull as much as possible
Conclusion

- OpenVG 1.1 near perfect match for Flash Lite 3
- Larger screens on mobile devices require HW acceleration of Flash
- Better performance with OpenVG than OpenGL ES
- Enhancements to OpenVG 1.1 are required to render Flash 9 and beyond