WebGL, WebCL and Beyond!

Neil Trevett
VP Mobile Content, NVIDIA
President, Khronos Group
Two WebGL-focused Sessions Today

- Industry ecosystem and standards for 3D and compute
  - What is 3D anyway – jungle survival primer
  - Khronos and open standard acceleration APIs for native apps
  - The evolution of pervasive 3D on mobile platforms
  - WebGL and WebCL as part of HTML5
  - Web apps and use cases beyond games – augmented reality

- Hands-On with WebGL
  - Steve Baker - Intific

WebGL Reference Cards at end of session!
What is Real-time 3D Graphics?

Computer graphics is the science and art of using computers to create and enjoy beautiful, interactive experiences. The processor that makes these amazing experiences possible is the GPU.
3D Pipeline Basics

- The art of “faking” realistic looking scenes or objects using heuristic techniques learned over the years
- The objects making up a scene are held in a database
- Surfaces of objects are broken down into a grid of polygons
- The vertices of the polygons are located in 3D coordinate space - x, y, z
- Each vertex has a “material” – color and reflective properties
- Vertices are positioned in 3D space – matrix math zooms and rotates
3D Pipeline Basics – Pixel Shading

• Project each polygon onto the screen
  - Determine which pixels are affected

• Smooth Shading
  - Run lighting equation at each vertex
  - Compute vertex color depending on how lights interact with surface angles and properties
  - Interpolate colors between the vertices

• Texture Mapping
  - “Wallpaper” each polygon with an image
  - For each pixel compute image coordinates in image to paste

• Environment Mapping
  - Paste reflection of image of environment at each pixel
Fundamental 3D Processing Stages

Operations on Vertices
- Geometry
- Rasterization

Operations on Pixels
- Color
- Clip
- Write

Traversal
What objects are in current scene?

Transforms
Where are the polygons?

Lighting
What color are the polygons?

Rasterize
What shape are they on the screen?

What color is each pixel?

Which pixels are visible?

Write
Write the pixels to the framebuffer
Actual 3D Pipelines

OpenGL ES 1.x Fixed Function Pipeline

- API
  - Primitive Processing
    - Vertex Buffer Objects
  - Transform and Lighting
  - Primitive Assembly
  - Rasterizer
  - Texture Environment
  - Colour Sum
  - Fog
  - Alpha Test
  - Depth Stencil
  - Colour Buffer Blend
  - Dither
  - Frame Buffer

OpenGL ES 2.0 Programmable Pipeline

- API
  - Primitive Processing
  - Vertex Shader
  - Primitive Assembly
  - Rasterizer
  - Vertex Buffer Objects
  - Fragment Shader
  - Depth Stencil
  - Colour Buffer Blend
  - Dither
  - Frame Buffer
3D has evolved over more than 30 years

‘Doom’ on a PC — 1993
id Software

‘Samaritan’ Real-time Demo on a PC — 2011
Epic Unreal Engine

http://www.youtube.com/watch?v=RSXyztq_0uM
Khronos and Hardware APIs

- Khronos defines open, royalty-free standards to access graphics, media, compute and input hardware
- Khronos APIs are low-level – just above raw silicon – to create the “foundation” functionality needed on every platform
- Safe forum for industry cooperation
  ‘By the industry for the industry’
  - Open to any company to join
  - IP framework to protect members and industry

APIs enable software developers to turn silicon functionality into rich end user experiences
Over 100 members – any company worldwide is welcome to join Board of Promoters
Khronos Family of Standards

Khronos creates royalty-free specifications to meet real market needs and helps drive industry adoption across multiple platforms.
A New Era in Personal Computing

- **PC**: 1990’s
- **Internet**: 2000’s
- **Mobile Computing**: 2010’s
20 Years Faster to 100M Per Year

Cumulative Shipments

- iOS & Android
- MacOS & Windows

Source: Gartner, Apple, NVIDIA
Mobile Silicon Experiential Processing
Mobile Roadmap Acceleration

- TEGRA 2 (2011)
- KAL-EL (2012)
- WAYNE (2013)
- LOGAN (2014)
- STARK (2014)

Production Devices:
- Core 2 Duo - Macbook Air

Performance:
- 1x TEGRA 2
- 5x KAL-EL
- 10x WAYNE
- 50x LOGAN
- 75x STARK

© Copyright Khronos Group, 2011 - Page 15
Mobile - Android Becoming Dominant OS

Source: Gartner, NVIDIA
OpenGL Ecosystem – 3D Everywhere

Leading-edge functionality developed first on desktop

OpenGL ES 2.0 on desktop as subset of OpenGL 4.2 for mobile content flexibility – including native support for WebGL

WebGL driving new-generation security features into OpenGL family

Mobile functionality subset that is deployed on billions of devices

Pervasive OpenGL ES 2.0 availability enables Browser vendors to build 3D directly into HTML5
OpenGL ES Pervasiveness

- **OpenGL ES 1.1** – fixed-function pipeline
  - Based on OpenGL 1.5
  - Vertex Arrays / Buffer Objects
  - Transform & Lighting
  - Multi-texturing (min 2 units)
  - Fixed-point & Floating-point profiles

- **OpenGL ES 2.0** – programmable pipeline
  - Based on OpenGL 2.0
  - Adds vertex and fragment shader programming
  - Removes fixed function pipeline
  - Super-compact, efficient API
  - High level language (GLSL ES)
  - On-line or off-line compilation
WebGL – 3D on the Web – No Plug-in!

- Historic opportunity to bring accelerated 3D graphics to the Web
  - WebGL defines JavaScript binding to OpenGL ES 2.0

- Leveraging HTML5 and uses `<canvas>` element
  - Enables a 3D context for the canvas

- JavaScript is easily fast enough now for visual computing
  - Plus OpenGL ES 2.0 enables local geometry caching and GPGPU computation

Being defined by major browsers and GPU vendors working together

Availability of OpenGL and OpenGL ES on almost every web-capable device

JavaScript binding to OpenGL ES 2.0

HTML5 Canvas Tag and increasing JavaScript performance
WebGL Implementation Anatomy

Content downloaded from the Web. Middleware can make WebGL accessible to non-expert 3D programmers

Browser provides WebGL functionality alongside other HTML5 specs - no plug-in required

OS Provided Drivers. WebGL on Windows can use Google Angle to create conformant OpenGL ES 2.0 over DX9

Content
JavaScript, HTML, CSS, ...

JavaScript Middleware

WebGL

HTML5

JavaScript

CSS

OpenGL ES 2.0
OpenGL
DX9/Angle
HTML5 Content Architecture

HTML content generated by layout engine ‘on page’

<video> tag

All content passes through CSS layout

CSS Layout and Transforms

Composition of off-screen buffers

Composition needs to be GPU accelerated

JavaScript drives interactivity for 2D and 3D graphics

Video, Vector Graphics and 3D created off-screen buffers

 creado off-screen buffers

Lorem ipsum
WebGL and HTML Interaction

• **3D is not trapped in a rectangular window**
  - 3D can overlay and underlay HTML content
  - Easy to make HUDs or user interfaces

• **Strong ties with other advanced HTML5**
  - WebGL can use HTML5 `<video>`
    or canvas as a texture

• **Can use 3D for core Web UI – as well as content**
  - Advanced transforms and special effects

• **Render HTML DOM sub-tree as texture**
  - Support user interaction when in 3D
  - Mozilla and Google prototyping as extension

• **WebGL is democratization of 3D**
  - Accessible, pervasive, enabling
  - Spawning amazing innovation
WebGL Deployment

• WebGL 1.0 Released at GDC March 2011
  - Mozilla, Apple, Google and Opera working closely with GPU vendors

• Typed array 1.0 spec ratified by Khronos in May
  - Supporting bulk data transfer between threads (workers)
  - Many use cases - background mesh loading, generation, deformation, physics ...

• 1.0.1 release of WebGL spec and conformance suite imminent
  - 100% robust stance on security
  - Fixing bugs in 1.0.0 conformance suite
  - Implementations will report getContext("webgl") (not experimental)

<table>
<thead>
<tr>
<th>Show all versions</th>
<th>IE</th>
<th>Firefox</th>
<th>Safari</th>
<th>Chrome</th>
<th>Opera</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 versions back</td>
<td>6.0</td>
<td>3.6</td>
<td>3.2</td>
<td>10.0</td>
<td>10.6</td>
</tr>
<tr>
<td>2 versions back</td>
<td>7.0</td>
<td>4.0</td>
<td>4.0</td>
<td>11.0</td>
<td>11.0</td>
</tr>
<tr>
<td>Previous version</td>
<td>8.0</td>
<td>5.0</td>
<td>5.0</td>
<td>12.0</td>
<td>11.1</td>
</tr>
<tr>
<td>Current</td>
<td>9.0</td>
<td>6.0</td>
<td>5.1</td>
<td>13.0</td>
<td>11.5</td>
</tr>
<tr>
<td>Near future</td>
<td>9.0</td>
<td>7.0</td>
<td>6.0</td>
<td>14.0</td>
<td>12.0</td>
</tr>
<tr>
<td>Farther future</td>
<td>10.0</td>
<td>8.0</td>
<td>6.0</td>
<td>15.0</td>
<td>12.1</td>
</tr>
</tbody>
</table>

WebGL is not enabled by default in Safari

http://caniuse.com/#search=webgl
Aquarium Demo

• On PC and Android

Frameworks and Tools

• WebGL is deliberately low level to enable the full power and flexibility of OpenGL ES 2.0
• If you are not an expert 3D programmer – don’t panic!
• WebGL is perfect foundational layer for JavaScript middleware frameworks
• Lots of utilities and tools already appearing
Declarative 3D for the Web

- Need to enable ‘non-expert’ web programmers with layers over WebGL
  - 10,000s of 3D programmers worldwide versus millions of web developers
  - Middleware and layered architectures play a vital role

- W3C Incubator for Declarative 3D
  - “easy way to add interactive high-level declarative 3D objects to the HTML-DOM”
  - X3DOM (www.x3dom.org/) and XML3D (www.xml3d.org/)

- Bind 3D even closer into the browser stack
  - Use as much HTML5 machinery as possible – DOM, JavaScript, CSS
  - Focus on driving optimized WebGL/OpenGL ES 2.0 back-end
  - Use Typed Arrays and drive for optimal performance
WebGL Security

• Any new functionality in the browser increases exposure to attack
  - True since the beginning of the web – the new functionality becomes hardened

• ANY graphics in the browser need the GPU drivers to be hardened
  - HTML, Canvas, WebGL, Adobe Molehill, Silverlight 5 …

• WebGL is designed with security as the highest priority
  - Hardening is being strongly promoted and enabled

• Short term – browser vendors will maintain white and black lists
  - Compromised system can have WebGL disabled until mitigation developed

• Longer term – GPUs provide increasingly robust security and multi-tasking
  - GPU becoming a first-class computing platform alongside CPU
WebGL Security in the Press!

- Confusion in the industry as we start this hardening process
  - Shader programs *cannot* access general system resources or perform out of range memory access!

- Issues in the Press
  - Cross domain image access – timed loop attack – SOLVED!
    - WebGL and HTML spec updates - mandating CORS for video, images and audio
    - Servers have to grant cross-domain access to media resources
  - DOS Attacks and general hardening
    - ARB_robustness extensions that provide additional protection being mandated
    - New robustness spec limits the side-effects of a GPU reset after a DOS attack
    - ANGLE shader validator improved; more improvements coming
Flash Stage 3D aka ‘Molehill’

- **GPU-friendly 3D ‘stage’ behind classic Flash graphics**
  - No interaction with classic Flash except classic 2D overlays the new 3D

- **Different design approach to WebGL**
  - Defines an OpenGL ES 2.0 assembler

- **Portability at the cost of functionality**
  - No loops
  - Lowest common denominator

- **Competition will ensure WebGL keeps it’s eyes on the ball for security and portability**
Possible Flash Stage3D vs WebGL Adoption

WebGL has already started to ship

Adobe invest significant QA and porting resources for application portability on any Flash-supported platform

WebGL is an open standard that will ship on every platform – and be supported by MANY vendors

Competition is Healthy!
Mobile Web versus Apps

- Mobile Apps have functional and aesthetic appeal
  - Beautiful, responsive, focused
- HTML5 with WebGL can provide the same level of “App Appeal”
  - Highly interactive, rich visual design
- Using HTML5 to create ‘Web Apps’ has many advantages
  - Portable to any browser enabled system
  - Same code can run as app or as web page
  - Web page is discoverable through the web – not a closed app store
- Need to evolve tools to package a web page as an app - with no chrome
  - As Adobe has done with Air for Flash applications
  - E.g. Blackberry WebWorks:
    http://us.blackberry.com/developers/browserdev/opensource.jsp
Processor Parallelism

**CPUs**
Multiple cores driving performance increases

**GPUs**
Increasingly general purpose data-parallel computing

**Emerging Intersection**

**Heterogeneous Computing**

Multi-processor programming – e.g. OpenMP

Graphics APIs and Shading Languages

OpenCL is a programming framework for heterogeneous compute resources
The BIG Idea behind OpenCL

• OpenCL execution model ...
  - Define N-dimensional computation domain
  - Execute a kernel at each point in computation domain

• C Derivative to write kernels – based on ISO C99
  - APIs to discover devices in a system and distribute work to them

• Targeting many types of device
  - GPUs, CPUs, DSPs, embedded systems, mobile phones.. Even FPGAs

```c
void trad_mul(int n,
              const float *a,
              const float *b,
              float *c)
{
    int i;
    for (i=0; i<n; i++)
        c[i] = a[i] * b[i];
}
```

```c
kernel void dp_mul(global const float *a,
                   global const float *b,
                   global float *c)
{
    int id = get_global_id(0);
    c[id] = a[id] * b[id];
} // execute over “n” work-items
```

Traditional loops

Data Parallel OpenCL
WebCL – Parallel Computing for the Web

• Khronos launching new WebCL initiative
  - First announced in March 2011
  - API definition already underway

• JavaScript binding to OpenCL
  - Security is top priority

• Many use cases
  - Physics engines to complement WebGL
  - Image and video editing in browser

• Stay close to the OpenCL standard
  - Maximum flexibility
  - Foundation for higher-level middleware
Visual Computing Ecosystem

- OpenCL
- Compute and mobile APIs interoperate through EGL
- High performance compute and graphics interop – buffer and events
- JavaScript bindings to OpenCL
- Parallel computation in HTML5
- WebGL
- WebCL
WebCL Open Process and Resources

- **Khronos open process to engage Web community**
  - Public specification drafts, mailing lists, forums
  - [http://www.khronos.org/webcl/](http://www.khronos.org/webcl/)
  - webcl_public@khronos.org

- **Khronos welcomes new members to define and drive WebCL**
  - info@khronos.org

- **Nokia open sourced prototype for Firefox in May 2011 (LGPL)**
  - [http://webcl.nokiaresearch.com](http://webcl.nokiaresearch.com)

- **Samsung open sourced prototype for WebKit in July 2011 (BSD)**

**Deformation Demo:**

- Calculates and renders transparent and reflective deformed spheres on top of photo background

- Performance comparison on Mac
  - JS: ~1 FPS
  - WebCL: 87-116 FPS

- [http://www.youtube.com/user/SamsungSISA#p/a/u/1/9Ttux1A-Nuc](http://www.youtube.com/user/SamsungSISA#p/a/u/1/9Ttux1A-Nuc)
Expanding HTML5 Capability

- HTML5 evolving into cross-platform programming platform
  - Gradually exposing complete system capabilities

- WebGL will enable visually rich, dynamic HTML5 ‘apps’
  - Portable to any Web-capable system, Web discoverable
  - Run in browser or with no ‘Chrome’ – like Flash/Air, Rim WebWorks

- Opportunity to synergize Web and native APIs to expand HTML5
  - Leverage APIs investments, reduce developer learning cycles
Visual-based Augmented Reality

Camera video stream sent to the compositor

Camera images used to track the camera’s location and orientation

Camera-to-scene transform locks the 3D rendering to the real world

3D augmentations composited with video stream

3D Augmentation Rendering
Compute Power Driving Sensor Innovation

Diverse Devices and Platforms
Need Application Portability
Touch screens, controllers, microphones etc.
Mobile phones, tablets, desktop systems

Positional Sensor Fusion
Combined Sensor Processing
Gyro, accelerometer, compass
Application control and situational awareness

No cross-platform API for accessing and enabling these innovative input devices

Cameras as Sensors
Gesture and Motion Detection
Depth cameras – a la Kinect
Standard cameras inc. stereo

StreamInput
Cross-platform Sensor API

© Copyright Khronos Group, 2011 - Page 39
Core StreamInput Concepts

• Application discovers what sensors can provide required semantic input
  - “Am I in an elevator, in a moving car, or being carried in a pocket”
  - “Need full body tracking and gestures”

• Handles almost any sensor
  - Multi-axis motion/position sensors, capacitive multi-touch surfaces
  - RGB and Depth sensing cameras – including mono and stereo
  - Microphone arrays for speech recognition etc.
  - Haptic devices, Biometric sensors etc. etc

• Graph-based API
  - Application sets up a graph of input device nodes
  - Graph generates high or low level data for app

• Multi-sensor synchronization
  - Timestamp EVERYTHING in a system
Khronos APIs in Example AR Flow

**StreamInput**

- Positional and GPS Sensor Data
- Imaging/Tracking (as StreamInput Filter)

**Camera Object**
- Preprocess and generate video streams

**Media Object**
- Video stream to CPU

**EGLStream Player**
- Video stream to GPU

**Application on CPU**
- Filter Graph
- High-level Position and Tracking Semantics

**Audio Rendering**
- 3D Rendering and Video Composition

**Controls**
- Camera
- Control Camera
Get Involved!

• Engage with the WebGL working group on Khronos forums and mailing lists
• Let us know if you have news or links that Khronos can help highlight
  - info@khronos.org or edit the Wiki
• Join Khronos to have a voice in how the specs evolve!
  - Any company is very welcome

http://www.khronos.org/webgl/wiki/Main_Page
In Summary

- WebGL brings another vital piece of system capability into the HTML5 browser for web apps – 3D graphics
- WebGL is being deployed right now on PC – soon on mobile – and is being strongly supported by browser and GPU vendors
- WebGL is a low-level, secure technology – that can be used directly and will support a rich ecosystem of tools and frameworks
- WebGL and WebCL show how to take well proven native APIs and bring them to the web – with more to come!
Questions?

Come get a Reference Card!