Real-Time Graphics for Feature Film Production
GPU Team

David Yu
Matthew Williams
Pol Jeremias-Vila
Takahito Tejima
George Elkoura
Dirk Van Gelder
Overview

- Challenges
- Pixar Tech for Real-Time Workflows
- Effects
- Future Work
Challenges

- Production assets
- Optimized for beauty, not real-time rendering
- Average hardware configuration:
  - GPU: 12 GB NVIDIA Quadro M6000
  - CPU: 16-core Intel Haswell @ 2.3GHz
  - RAM: 64 GB
6,000 Meshes
81,000 Instances
52 Million Faces
100+ Billion Triangles
This is a small shot
Heavy Assets

- Fx: millions of instances
- Chars: hundreds of meshes
- Crowds: tens of thousands of chars
- Large sets: billions of polys in frustum
- Subdivision: geometric growth in complexity
Presto Animation System

- Primary tool for:
  - Previs
  - Layout
  - Simulation
  - Rigging
  - Animation
  - Crowd Animation
Universal Scene Description

- Deployed on Finding Dory
- Scene described in terms of layers and references
- Designed for multi-user workflows
- DCC Interchange
- Supported by Apple’s SceneKit
- Open source release on Github July 26, 2016
#usda 1.0

def Cube "HelloCube" {
    double size = 1.0
}

#usda 1.0

def Cube "HelloCube" {
    double size = 1.0
}

#usda 1.0

def Cube "Cube" {
    double size = 1.0
}

def "World" {
    over "CubeInstance1" (  
        references = </Cube>
        instanceable = true) {}
    over "CubeInstance2" (  
        references = </Cube>
        instanceable = true) {}
}
#usda 1.0

def Cube "Cube" {
    double size = 1.0
}

def "World" {
    over "CubeInstance1" {
        references = </Cube>
        instanceable = true
    }
    over "CubeInstance2" {
        references = </Cube>
        instanceable = true
    }
}
USD Composition Arcs

- Layers
- Inheritance
- Variants
- References
- Payloads
USD Composition Arcs

- Layers
- Inheritance
- Variants
- References
- Payloads
( demo )
Universal Scene Description (USD) is an efficient, scalable system for authoring, reading, and streaming time-sampled scene description for interchange between graphics applications.

- **Download USD Preview**
- **Download SketchUp Plugin**

**Open Source**
- Announcement
- SIGGRAPH 2013 Presentation
- USD Presentation Slides
  - Download: 10MB

**Demonstration Videos**
- Integration:
  - Part 1: Scripting API
  - Part 2: ASCII File Format
  - Part 3: Asset Structure

**USD in Action**
- Pipeline Workflows
- Alembic Integration
- Katana Integration
Subdivision Surfaces
What is a subdivision surface?
What is a subdivision surface?
What is a subdivision surface?
What is a subdivision surface?
What is a subdivision surface?

Poly Count: 48, 384, 1,572,864
What is a subdivision surface?

Creased edges with varying sharpness and more!
OpenSubdiv

- RenderMan’s subdivision algorithm
- Fast evaluation for rigging
- Fast rendering for preview
- Broad API for many clients
OpenSubdiv

- Suitable for off-line rendering
- Suitable for real-time applications
- Major improvements in version 3.0
  - Simpler GPU shader configuration
  - Faster CPU refinement
  - Evaluation API
  - Simpler API
- Coming this Summer version 3.1
  - Smooth UV evaluation on patches
  - Separate refinement levels
  - 2nd derivative evaluation
OpenSubdiv Evaluators

- **Osd**: parallel evaluation
  - OpenGL Compute
  - OpenGL Transform Feedback
  - CUDA
  - TBB
  - OpenCL
  - ISPC

- **Far**: single-threaded evaluation
  - Ideal for off-line rendering
Feature Adaptive Subdivision

Uniform: 48 1,572,864
Feature Adaptive Subdivision

Uniform: 48  1,572,864
Adaptive: 48  1,100
Integrated Tools & Render Engines
Available on Github

github.com/PixarAnimationStudios/OpenSubdiv
HYDRA
Real-Time Render Engine
Hydra: Real-Time Render Engine

- Render engine for content creation
- Built for modern graphics APIs
  - Fastest w/ OpenGL 4.5 + Bindless Textures + Bindless Buffers + Direct State Access
  - Graceful fallback to OpenGL 4.0
- Designed for scalability

See the GTC 2015 talk for technical details.
Hydra Engine Features

- Highlighting for faces, points, etc
- Just-in-time data consumption
- Consumes RenderMan inputs
- Hardware tessellated curves
- Configurable representations
- Render-based object picking
- Heavily multi-threaded
- Wireframe on surface
- OpenSubdiv tessellation

See the GTC 2015 talk for technical details
Hydra Engine Features

- Highlighting for faces, points, etc
- Just-in-time data consumption
- Consumes RenderMan inputs
- Hardware tessellated curves
- Configurable representations
- Render-based object picking
- Heavily multi-threaded
- Wireframe on surface
- OpenSubdiv 3.0

▶ Designed for feature film production

See the GTC 2015 talk for technical details
def Cube "Cube" {
    double size = 1.0
}

def "World" {
    over "CubeInstance1" {
        references = </Cube>
        instanceable = true
    }
    over "CubeInstance2" {
        references = </Cube>
        instanceable = true
    }
}
Hardware Instancing

<table>
<thead>
<tr>
<th>Instancer</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>Prototype-1</td>
<td></td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td></td>
<td>Prototype-2</td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Hardware Instancing

Instancer
- [Translate Rotate Scale]
- [Translate Rotate Scale]
- [Translate Rotate Scale]
- [Translate Rotate Scale]
# Hardware Instancing

<table>
<thead>
<tr>
<th>Instancer</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
</tr>
</tbody>
</table>
Hardware Instancing

<table>
<thead>
<tr>
<th>Instancer</th>
<th>Offset</th>
<th>Prototype-1</th>
<th>Prototype-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[ Translate Rotate Scale ]</td>
<td>[ Instance Index ]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
But what about instancers
But what about instancers that instance
But what about instancers that instance more instancers?
But what about instancers that instance more instancers?!
Recursive Instancing

Instance all the things
( demo )
Hydra Integration

- **Push-Pull Data Model**
  - Push: Scenegraph declare primitives
  - Pull: Hydra requests data from primitives on demand
  - Avoids pulling all data up front
Hydra Integration

Diagram:
- Scenegraph
- Adapter
- Hydra RenderIndex
Hydra Integration

Scenegraph  Adapter  Hydra RenderIndex
Scenegraph  Adapter
Hydra Integration

- Presto Scenegraph
- Presto Adapter
- USD Scenegraph
- USD Adapter
- Hydra RenderIndex
Integrated with ...
Presto
Hydra for Virtual Production

INDUSTRIAL
LIGHT & MAGIC
Millennium Falcon

- 14,500 meshes
- 140 textures @ 8k each
- Hybrid renderer:
  - Native ILM scenegraph
  - Hydra populates g-buffer
  - ILM’s engine does deferred shading & lighting
Hydra open sourced with USD!
Effects
Requirements

- Easy to setup and maintain
- Reliable approximation of the final render
  - Don’t mislead the artist
- Keep the common case fast
- Minimal asset build requirements
Building Effects: The Render Graph
Real-Time Effects in Presto

- Depth of Field
- Soft Shadows
- Ambient Occlusion
- Portal Reflections
- Reflection & Refraction
- Motion Blur
- Barrel Distortion

- Scene & Camera Lights
- Murk Light
- Post-Effect Lights
- Camera Masking
- Curvature
- Fur / Hair / Feathers
- Refractive Eyes
( demo )
Physically-Based Shading

- RenderMan
- Real-time OGL approximation

More info:
blog.selfshadow.com/publications/[s2012,s2013,s2014,s2015]-shading-course
Physically-Based Shading

- RenderMan
- Real-time OGL approximation

More info:
blog.selfshadow.com/publications/[s2012,s2013,s2014,s2015]-shading-course
Physically-Based Shading

- RenderMan
- Real-time OGL approximation

More info: blog.selfshadow.com/publications/[s2012,s2013,s2014,s2015]-shading-course
Physically-Based Shading

- Complex material networks
  - Baked signals from RenderMan

- diffuse color
- specular color
- specular roughness
- displacement
- clearcoat color
- clearcoat roughness
- roughspecular color
- roughspecular roughness
Physically-Based Shading

- Complex material networks
  - Baked signals from RenderMan
Physically-Based Shading

- Complex material networks
  - Baked signals from RenderMan
We need compression, lots of compression!

- Hello BPTC!

- BC7 & BC6H

16 pixels
* 3 channels
* 1 byte
= 48 bytes

16 bytes
Lots of Textures

Uncompressed

Compressed
UV and Ptex

- Support for both!
- **UV**
  - Characters
- **Ptex**
  - Sets
  - Backlot
Shader Baking

- Sample RenderMan shaders used for final image into OpenGL textures
- Us Bxdf inputs, combined when necessary due to real time constraints
- Use Katana livegroups wired through command line python scripts
- Built-in support new in RenderMan 21
Future Work

- Ground truth effects (accumulation buffer, motion blur, etc)
- Volumetric effects
- Subsurface Scattering
- Hydra & OpenSubdiv support for Vulkan/DX12/Metal
Thank You!

USD Team

Pixar R&D Department

All Pixar Artists and TDs
Special Thanks

Contributors

○ Hydra and USD: Jeremy Cowles
○ OpenSubdiv: Barry Fowler
We’re Hiring!

www.pixar.com/careers