Managing 3D Assets with Model I/O

Session 602

Nick Porcino Apple Inc.
Remi Palandri Apple Inc.
Claudia Roberts Apple Inc.
Model I/O
Framework for handling 3D assets and data

Import and export 3D asset files
• Describe lighting, materials, environments
• Process and generate asset data
• Bake lights, subdivide and voxelize meshes

For Physically Based Rendering
• Designed for PBR tools and pipelines

Integrated with Xcode 7 and GameKit APIs
Available for iOS 9 and OS X 10.11
Introduction
Model I/O Workflow

Content Creation Tool → Import → Model I/O

- SceneKit
- Metal
- OpenGL
Introduction

Model I/O Workflow

Content Creation Tool → Import → Model I/O → Bake → SceneKit, Metal, OpenGL
Introduction

Model I/O Workflow

Content Creation Tool → Import → Model I/O → Bake → Export → SceneKit → Metal → OpenGL
Agenda

The Model I/O Framework

Features overview
Data types and physical motivation
Geometry and voxels
Advanced lighting
Baking
Model I/O Overview
File Formats

Import formats

• Alembic .abc
• Polygon .ply
• Triangles .stl
• Wavefront .obj

Export formats

• Triangles .stl
• Wavefront .obj
Import and Export

Import

MDLAsset *asset = [[MDLAsset alloc] initWithURL:myURL];

Export

[asset exportAssetToURL:myURL];
Physical Realism

Realistic lights
• IES profile, temperature, image based

Realistic materials
• Lambert / Blinn-Phong, physical BRDF

Realistic cameras
• From lens to sensor

Realistic environments
• Panoramic photographs
• Procedural skies
Modify and Bake Assets

Ambient occlusion
[mesh generateAmbientOcclusionTextureWithQuality: ... ];

Light and shadow baking
[mesh generateLightMapTextureWithQuality: ... ];

Normals calculation
[mesh addNormalsWithAttributeName: ... ];

Tangent basis
[mesh addTangentBasisForTextureCoordinateAttributeName: ... ];
Voxels
Create voxels from an asset

```
[[MDLVoxelArray alloc] initWithAsset: ... ];
```

Find voxels in a region

```
NSData *indices = [voxels voxelsWithinExtent:extent];
```

Constructive solid geometry

```
[voxels unionWithVoxels:sphereVoxels];
```

Create a mesh

```
MDLMesh *mesh = [voxels meshUsingAllocator:allocator];
```
System Integration

SceneKit, Metal, and OpenGL

Preview in Finder and QuickLook

Edit in Xcode

Playgrounds and Swift support
Data Types
MDLAsset

Overview

Indexed container for 3D objects and materials
Transform hierarchies
Meshes
Cameras, lights
Created procedurally or from a URL
MDLAsset
MDLObject

- name
- components
- transform
- bounding box
- parent
- children

MDLTransformComponent

User defined

- User defined
- User defined

MDLObject

MDLObject

MDLObject

MDLObject

MDLObject

MDLObject

...
MDLMesh

Contains one or more vertex buffers
Vertex buffers contain information such as positions, and normals
Contains one or more submeshes
Submeshes contain triangle or polygon indices
Submeshes share data in vertex buffers
MDLMesh

MDLObject
- name
- components
- transform
- bounding box
- parent
- children

MDLMesh
- Generators
  - Boxes, ellipsoids, planes, etc.
  - Subdivided model
- Modifiers
- Bakers
  - buffers
  - descriptors
  - submeshes
MDLMesh

MDLObject
- name
- components
- transform
- bounding box
- parent
- children

MDLMesh
- Generators
- Modifiers
- Bakers
- buffers
- descriptors
- submeshes

Compute normals, tangents, make vertices unique
MDLMesh

- MDLObject
  - name
  - components
  - transform
  - bounding box
  - parent
  - children

- MDLMesh
  - Generators
  - Modifiers
  - Bakers
  - buffers
  - descriptors
  - submeshes

Ambient occlusion, light maps
MDLMesh

MDLObject
- name
- components
- transform
- bounding box
- parent
- children

MDLMesh
- buffers
- descriptors
- submeshes
- Generators
- Modifiers
- Bakers

MDLMeshBuffer
- data
- length
- allocator
- Map
Materials

Intuitive parameters

A dielectric, like clay
Materials

Intuitive parameters

A pure metal
Materials

Intuitive parameters

A combination

Metallic with an acrylic clear coat
Materials

Intuitive parameters

A satin finish
Materials

Intuitive parameters

A variety of finishes
MDLMaterial

MDLSubmesh
- name
- index buffer
- geometry type
- material

MDLMaterial
- name
- properties
- scattering function
- base material

Single inheritance
MDLMaterial

MDLSubmesh
- name
- index buffer
- geometry type
- material

MDLMaterial
- name
- properties
- scattering function

MDLMaterialProperty
- name
- semantic
- type
- value
Lights

Physical realism

Classic CG lights

Lights with physical parameters

• Geometry

• Lumens, color temperature

Baked light maps

IES standard light files
MDLLight

MDLObject → MDLLight → MDLPhysicallyPlausibleLight

- name
- components
- transform
- bounding box
- parent
- children

- type
- color
- lumens
- angles
- attenuation
MDLLight

MDLObject
- name
- components
- transform
- bounding box
- parent
- children

MDLLight
- type

MDLPhysicallyPlausibleLight
- color
- lumens
- angles
- attenuation

MDLPhotometricLight
- light web
MDLLight

- MDLObject
  - name
  - components
  - transform
  - bounding box
  - parent
  - children

- MDLLight
  - type

- MDLPhysicallyPlausibleLight
  - color
  - lumens
  - angles
  - attenuation

- MDLLightProbe
  - reflective map
  - irradiance map
  - spherical harmonics
MDLCamera

A physical description of a camera
Lens characteristics
Shutter properties
Sensor properties
What the camera can see
How the image will be exposed
MDLCamera

MDLObject
- name
- components
- transform
- bounding box
- parent
- children

MDLCamera
- visibility
- lens glass
- lens geometry
- exit aperture
- sensor
- exposure
Default exposure settings
Underexposed and flashed for shadow detail
Skies

Physical realism

Create a sky through physics

• Time of day
• Atmospheric condition
Skies

Physical realism

Create a sky through photography

• Take a spherical panorama with your phone or DSLR
Skies
Physical realism

Create a sky through photography
• Take a spherical panorama with your phone or DSLR
Skies

Physical realism

Prepare it for rendering

MDLTexture *sky = [[MDLURLTexture alloc] initWithURL:picURL name:"skypano"];

Create a cube map for reflection and irradiance

MDLTexture *cube = [MDLTexture irradianceTextureCubeWithTexture:sky ...];
Advanced Lighting

Irradiance

Incoming light from all directions
Efficient low-frequency representation
Important for physically based rendering
Skies

Physical realism

And perfectly match the lighting
Integration with SceneKit

- **MDLAsset** → SceneKit root node
- **MDLMesh** → SCNNode with SCNGeometry
- **MDLLight** → SCNLight
- **MDLCamera** → SCNCamera
- **MDLMaterial** → SCNMaterial
Integration with MetalKit

MetalKit will prepare buffers for rendering
• Traverse asset to find lights, cameras
• Use to drive custom Metal renderer
See “What’s New in Metal, Part 2”
Integration with OpenGL and GLKit is similar
Models and Voxels
Claudia Roberts
Agenda
Geometry and modeling in Model I/O

Normal smoothing
Subdivision surfaces
Voxels
Demo
Normal Smoothing

Shared vertex normals

Approximates curvature of more complicated geometry

Flat vs smooth shading

Add smoothed out normals to the spaceship

spaceship addNormalsWithPathAttributeNamed:@“normals” creaseThreshold:0.5;
Subdivision Surfaces
Vary surface detail

Generate subdivided mesh from source mesh
Increase level of detail only when and where necessary

MDLMesh *mesh = [MDLMesh newSubdividedMesh:spaceship
    submeshIndex:0
    subdivisionLevels:2];
Voxels

Physical realism

Volumetric representation consistent with the real world

Procedural modeling/generation

Easily explore dataset via neighborhood, child traversal

Facilitates real-world operations like slicing and cutting

Constructive Solid Geometry operations
Voxels

Physical realism

Volumetric representation consistent with the real world

Procedural modeling/generation

Easily explore dataset via neighborhood, child traversal

Facilitates real-world operations like slicing and cutting

Constructive Solid Geometry operations
Voxels

Physical realism

Volumetric representation consistent with the real world

Procedural modeling/generation

Easily explore dataset via neighborhood, child traversal

Facilitates real-world operations like slicing and cutting

Constructive Solid Geometry operations
Voxels

Physical realism

Volumetric representation consistent with the real world

Procedural modeling/generation

Easily explore dataset via neighborhood, child traversal

Facilitates real-world operations like slicing and cutting

Constructive Solid Geometry operations
Voxels

MDLVoxelArray

Sparse volume grid accessed by a spatial index
Voxels

MDLVoxelArray

Sparse volume grid accessed by a spatial index
Quick neighbor finding
Voxels

MDLVoxelArray

Sparse volume grid accessed by a spatial index
Quick neighbor finding
Interior, exterior, surface shell levels
Voxels

MDLVoxelArray

Sparse volume grid accessed by a spatial index
Quick neighbor finding
Interior, exterior, surface shell levels
Model healing and clean-up
Create polygonal mesh from voxels
Voxels

MDLVoxelArray operations

// Set voxels corresponding to mesh
[grid setVoxelsForMesh:m divisions:25 interiorShells:0.f exteriorShells:0.f];
// Set voxels corresponding to mesh
[grid setVoxelsForMesh:m divisions:25 interiorShells:0.f exteriorShells:0.f];

// Given a second MDLVoxelArray, perform boolean operations
[grid intersectWithVoxels:voxels];
[grid unionWithVoxels:voxels];
[grid differenceWithVoxels:voxels];
Voxels

MDLVoxelArray operations

// Set voxels corresponding to mesh
[grid setVoxelsForMesh:m divisions:25 interiorShells:0.f exteriorShells:0.f];

// Given a second MDLVoxelArray, perform boolean operations
[grid intersectWithVoxels:voxels];
[grid unionWithVoxels:voxels];
[grid differenceWithVoxels:voxels];

// Retrieve voxel data
NSData *voxelData = [grid getVoxelIndices];
Voxels

MDLVoxelArray operations

// Set voxels corresponding to mesh
[grid setVoxelsForMesh:m divisions:25 interiorShells:0.f exteriorShells:0.f];

// Given a second MDLVoxelArray, perform boolean operations
[grid intersectWithVoxels:voxels];
[grid unionWithVoxels:voxels];
[grid differenceWithVoxels:voxels];

// Retrieve voxel data
NSData *voxelData = [grid getVoxelIndices];

// Create mesh from voxel grid
MDLMesh *mesh = [grid meshUsingAllocator:allocator];
Demo
Voxels
Advanced Lighting and Baking
Remi Palandri
Advanced Lighting

Global illumination

Global illumination looks great
But very expensive
We want to approximate GI
Balance performance/quality
Advanced Lighting

Ambient occlusion
Advanced Lighting

Ambient occlusion

Measure of geometry occlusion
Advanced Lighting

Ambient occlusion

Measure of geometry occlusion
Advanced Lighting

Ambient occlusion

Measure of geometry occlusion

Uses offline raytracing
Advanced Lighting

Ambient occlusion

Measure of geometry occlusion
Uses offline raytracing
Input—a mesh and a set of occlusion meshes
Advanced Lighting

Ambient occlusion

Measure of geometry occlusion
Uses offline raytracing
Input—a mesh and a set of occlusion meshes
Output—a set of occlusion values
Advanced Lighting
Ambient occlusion

Measure of geometry occlusion
Uses offline raytracing
Input—a mesh and a set of occlusion meshes
Output—a set of occlusion values
Stored in vertices or a texture
Advanced Lighting

Ambient occlusion

Measure of geometry occlusion
Uses offline raytracing
Input—a mesh and a set of occlusion meshes
Output—a set of occlusion values
Stored in vertices or a texture
// Bake the spaceship with itself
[shipMesh generateAmbientOcclusionVertexColorsWithQuality:0.6
 attenuationFactor:0.8
 objectsToConsider:@[shipMesh]
 vertexAttributeNamed:@“aoTextureCoord” ];

// Bake the ground floor with its surrounding objects
[groundMesh generateAmbientOcclusionTextureWithQuality:0.6
 attenuationFactor:0.8
 objectsToConsider:@[boxMesh, groundMesh]
 vertexAttributeNamed:@“aoTextureCoord”
 materialPropertyNamed:@“aoTextureProperty” ];
Demo

Xcode Integration
Advanced Lighting
Light maps
Advanced Lighting

Light maps

Computes the effect of lights
Advanced Lighting

Light maps

Computes the effect of lights
Advanced Lighting

Light maps

Computes the effect of lights
Advanced Lighting

Light maps

Computes the effect of lights
Supports lots of lights
Advanced Lighting
Light maps

Computes the effect of lights
Supports lots of lights
Calculated offline
Advanced Lighting

Light maps

Computes the effect of lights
Supports lots of lights
Calculated offline
Complex lights supported
Summary

Import and export 3D asset files
Physical basis for rendering
Models, lights, cameras, materials, skies
Integration with the system and frameworks
Tools in Xcode
More Information

Documentation and Videos
http://developer.apple.com

Apple Developer Forums
http://developer.apple.com/forums

Developer Technical Support
http://developer.apple.com/support/technical

General Inquiries
Allan Schaffer, Game Technologies Evangelist
aschaffer@apple.com
## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enhancements to SceneKit</td>
<td>Mission</td>
<td>Wednesday 2:30PM</td>
</tr>
<tr>
<td>What's New in Metal, Part 2</td>
<td>Mission</td>
<td>Thursday 9:00AM</td>
</tr>
<tr>
<td>Labs</td>
<td>Graphics, Games, and Media Lab D</td>
<td>Time</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Model I/O Lab</td>
<td>Tuesday 3:30PM</td>
<td></td>
</tr>
<tr>
<td>Model I/O Lab</td>
<td>Wednesday 9:00AM</td>
<td></td>
</tr>
</tbody>
</table>