MOANA ISLAND SCENE

Cesar Velazquez & Rasmus Tamstorf

Introduction

This document describes the result of the conversion by Walt Disney Animation Studios (WDAS) of the Moana Island Scene to use Pixar's Universal Scene Description (USD) format. Additional information about the Moana Island Scene based on the original release in JSON/OBJ format is available in the original README document.

The main focus of this version of the USD data set is renderability, leaving the optimizations needed for improved interactivity for future work. The goal has been to convert the original data using a combination of

files from the JSON/OBJ data set along with some data from the original production shot to produce USD files capable of rendering final quality, single-pass images. As such it is not 100% equivalent to the JSON/OBJ version, and this is by no means the only way to create a USD version of the Moana Island Scene. However, we hope that it will satisfy many of the requests we have received for a USD version.

Even within our focus, it should be noted that this release does not represent a “perfect” USD file. The known limitations are outlined at the end of this document.

License

Attribution

As with the JSON/OBJ version of the data set, attribution is not required, but it is appreciated. There is no required format, but any attribution must not otherwise violate the terms of the license (e.g., no implied endorsement). As an example, a caption accompanying an image generated from the data set may read:

"The USD version of the Moana Island Scene. Publicly available dataset courtesy of Walt Disney Animation Studios."

Citations in academic papers should follow the style of the journal. For journals following the APA citation style, the suggested format is:

"Walt Disney Animation Studios (2022). Moana Island Scene USD (v2.0) [Data set].
Retrieved from https://www.disneyanimation.com/resources/moana-island-scene/"

Release History

v2.1 June 2023

Updates to apply MaterialBindingAPI where needed by USD 23.08 and higher.

v2.0 February, 2022

First USD version. Original data unchanged.

v1.1 August 9th, 2018

Various data updates, including:

- Better curve extrapolation for PBRT and update to PBRT sample code.
- Corrections to original curve degree data for all curve-based primitives.
- Fixed bad color translation for IronwoodA1 in PBRT data.
- Corrected a mistake in isPalmRig xgFrond curve definition.
- Removed isPandanusA xgTwigs primitive description after confirming it wasn't present in production shot.
- Added missing transformMatrix data to isIronwoodB json file.
- Fixed bad texture references in isGardeniaA.
- Generated better color for isPalmRig xgFrondsA from original material.
- Fixed duplicate meshname in isDunesA xgPalmDebris_archivePalmdead0004_mod.obj.
- Replaced duplicated json primitive files for isBayCedar_bonsai[ABC]_xgBonsai.json with actual primitive files.
- Minor updates to documentation text to reflect above changes. Not all the changes are reflected in the images in this document yet.

v1.0 July 4th, 2018

Initial Release.

Contents

The data for the USD version is provided in a single tar-file. This version has been authored with USD v21.08 and RenderMan v24.0. The content of the tar file is organized as follows :

island/ - Base directory. All paths are relative to this directory.
island/README-USD.pdf - This file.
island/License.txt - The license for this data set.

island/usd/island.usda - Base data set containing all assets and their associated material and geometry information along with cameras and lights.

island/usd/islandPrman.usda - Base data set with modifications to the lights. These modifications are specific to Pixar's RenderMan and include :

- Environment maps use PRMan .tex formatted images.
    - islandsunVIS.png is replaced with islandsunCam.tex
    - islandsun.exr is replaced with islandsunEnv.tex
- The exposure values of the lights have been adjusted from the original values to better match Hyperion renders.
- Visibility to the camera has been disabled for quad lights.

materials/material.usda - Contains the materials bound to geometry in the scene.
ref/ - Reference images rendered using USD+Renderman and Disney's Hyperion renderer.
textures/ - All the textures used by the elements and lights. These are shared with the JSON/OBJ version of the dataset.

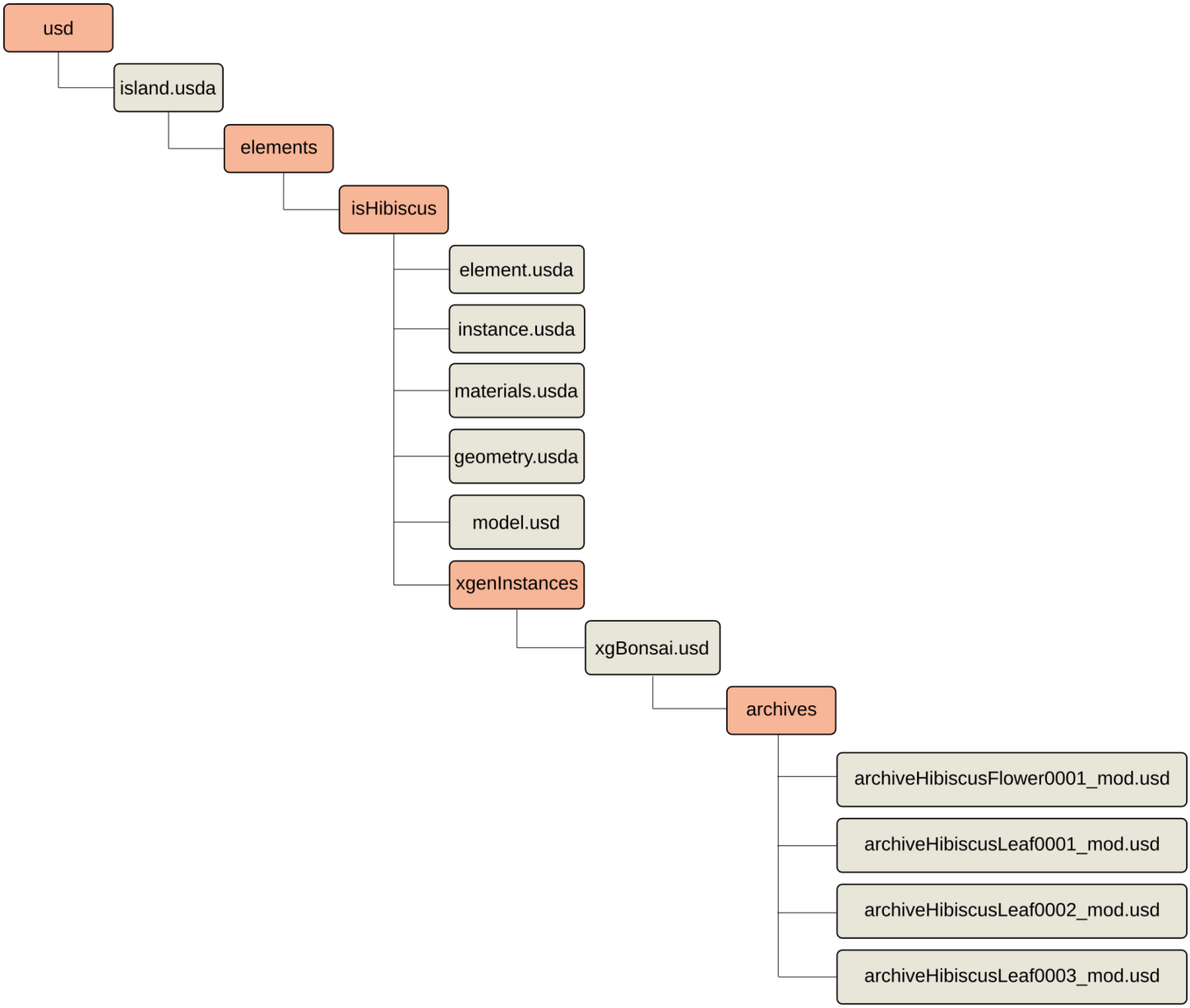
Elements

Each element is organized into its own subdirectory containing all the material and geometric information for it. All element and material descriptions are stored in ascii .usda files while mesh and XGen point instancer data are stored in binary .usd files.

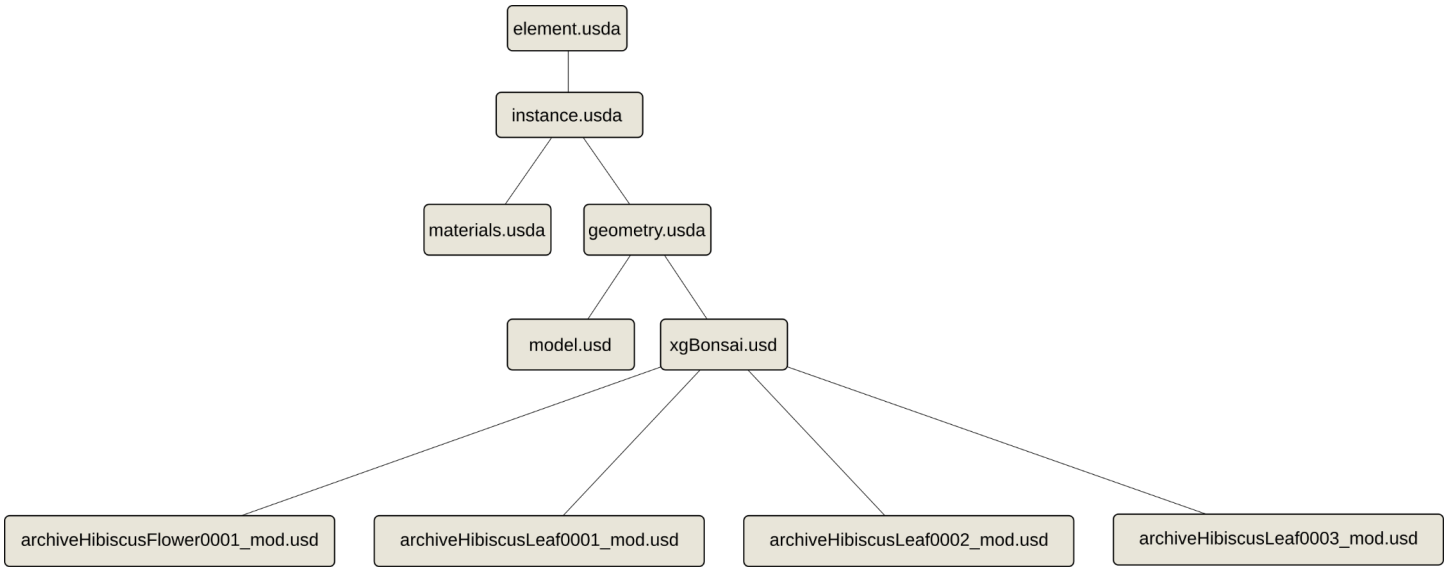
This is the common structure used to store elements on disk:

- **element.usda**
    - This file is referenced by island.usda and serves as the main entry point for the element. It will typically contain instanced copies of the element which are placed around the island.
- **instance.usda**
    - This file is referenced by element.usda and serves as the main entry point for a single copy of element. It references the element's material and geometry usd files.
- **materials.usda**
    - Contains material definitions for all meshes within the element. There are currently two materials definitions per element. One for RenderMan and another for USD's Storm renderer.
- **geometry.usda**

    - Main entry point for the element's geometry information. It references both standard geometry and XGen based instance geometry.
- **model.usd**
    - This will typically contain non XGen based geometry for the element. Examples include tree trunks and ground planes.
- **xgenInstances**
    - This directory contains both XGen point instance data as well as instance prototype files as well.
- **archives**
    - This subdirectory under xgenInstances typically holds prototype geometry used in instancing.



Common file directory structure for an element

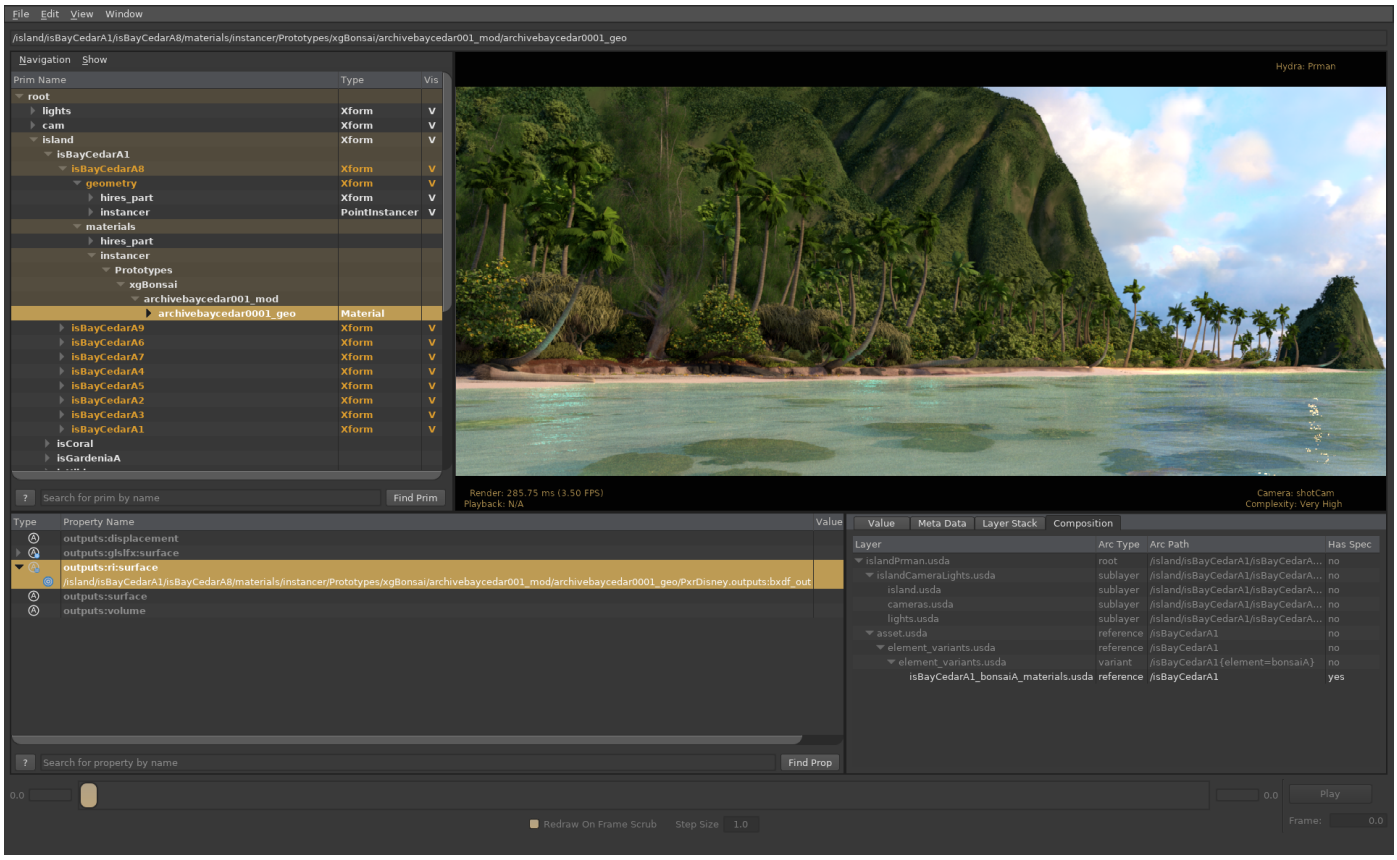


Common USD file reference diagram for an element

Rendering

The original JSON/OBJ data set focused on PBRT as a test renderer. When looking at renderers to test this USD data set with, we have been focusing on Pixar's RenderMan which is freely available for non-commercial purposes. It has native support for ptex texture files as well as shader support for Disney's Principled Material. Since it has a Hydra delegate, it also allows for interactive preview using Usdview which is convenient for testing. In particular, this makes iterating through different light and material parameter values much easier.

Support for other renderers can be achieved by translating material definitions to the target renderer. Each element has a materials.usda file which contains material definitions and binding information. Material parameter values can be read by either traversing the PRMan material definition for each object, or by reading the material attribute values located on each mesh (See the Materials Section).



Usdview with RenderMan delegate for Hydra

Cameras

The seven cameras from the original release are included in the USD version and can be found in:

/island/usd/cameras.usda

shotcam - Renderman

rootsCam - Renderman

palmsCam - Renderman

grassCam - Renderman

birdseyeCam - Renderman

dunesCam - Renderman

beachCam - Renderman

Geometry

All mesh data has been converted to USD as UsdGeomMesh primitives with subdivision surfaces enabled. Curves have been converted to USD as UsdGeomBasisCurves primitives. The curve “width” attribute has been taken from the JSON files and adjusted along the length of the curve to visually match profile data from the original shot.

Matching width profiles along curves

Hyperion Curves

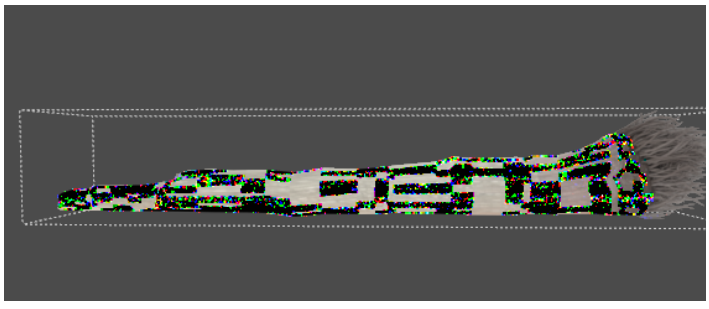
RenderMan Curves

XGen conversion

The Disney XGen data has been converted to USD as UsdPointInstancer primitives. Instance prototypes have been created and put into the “Archives” subdirectory. For most instance prototypes, colors from ptex maps have been baked into the displayColor attribute.

Geometry Fixes

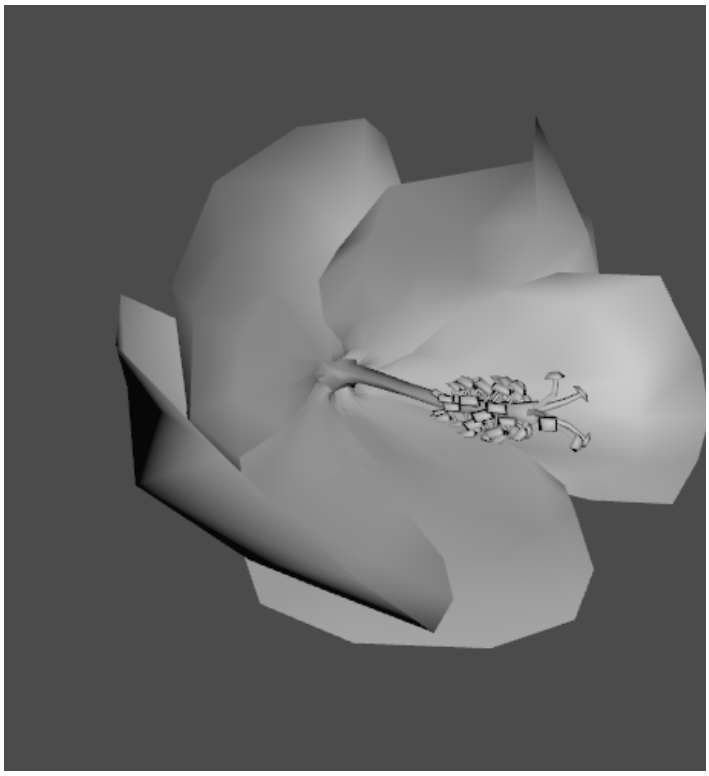
Several of the meshes in the data set have been modified to avoid a render artifact we noticed in our tests. For these meshes, we were seeing noisy “firefly” type render artifacts. Inspecting the meshes, we found the source of the artifact to be overlapping faces causing a mismatch in the ptex faceId count between the mesh and the ptex file. We corrected this by removing overlapping faces and vertices so the faceId count between the mesh and ptex file matched each other. We looked at a few of the OBJ files from the original data set and found the same error there. How this artifact shows up in renders depends on how the renderer handles the situation of having too many geometry faces and not enough ptex faces.



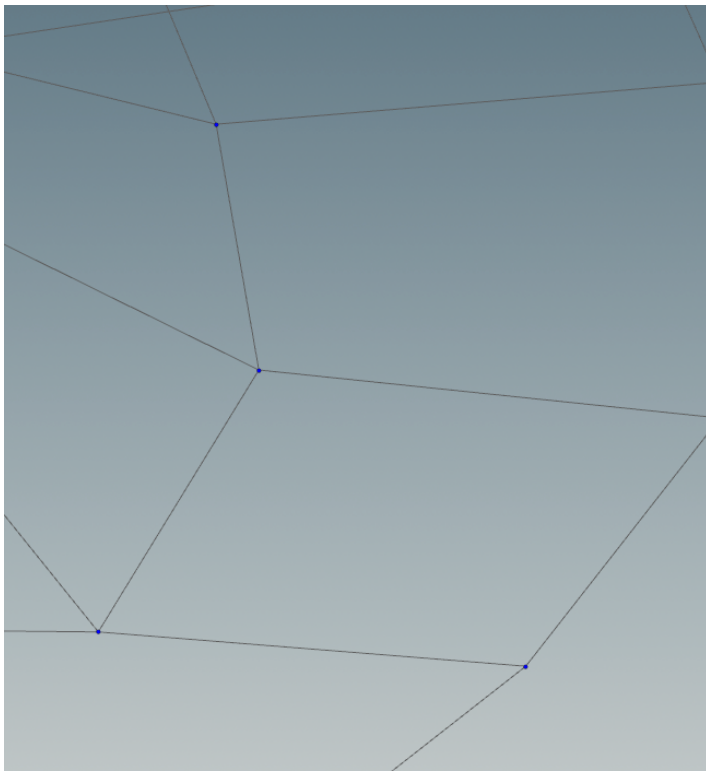
Render artifact due to differing faceId counts

Fixed Geometry

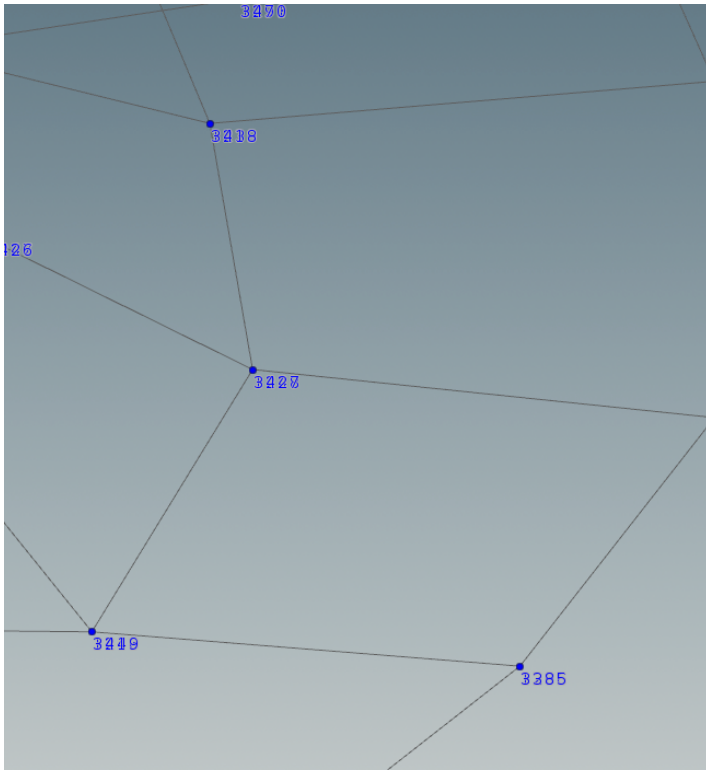
Duplicate points and overlapping geometry



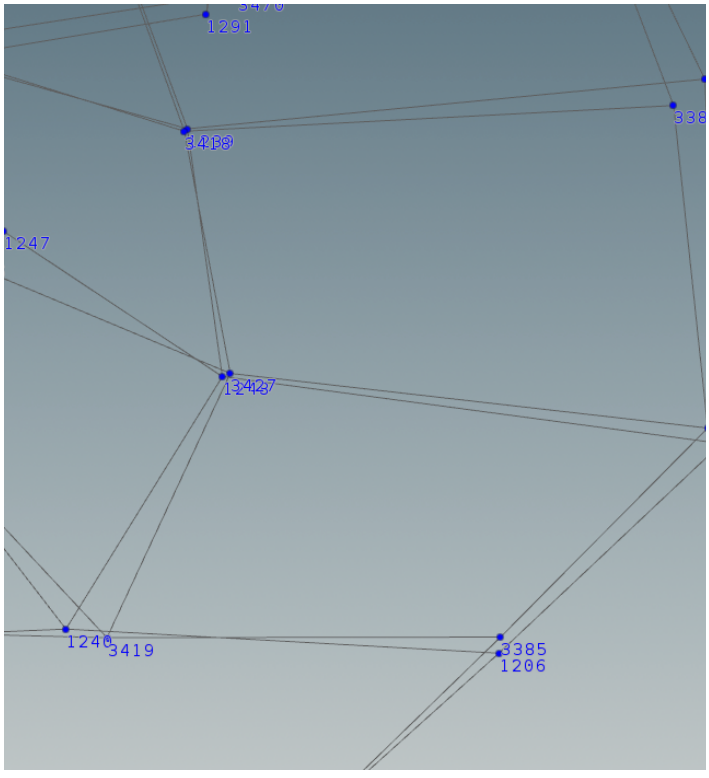
Original geometry



Close-up of leaf section



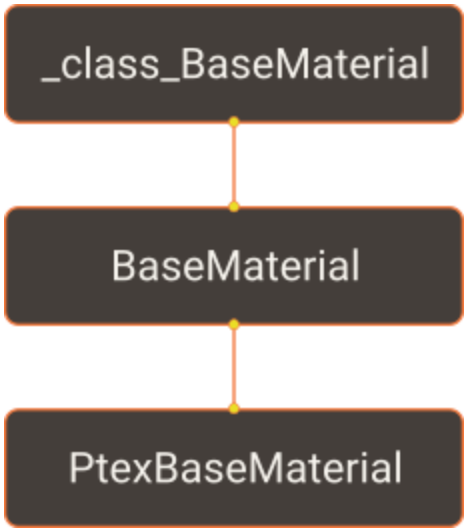
Overlapping point numbers visible



Separating overlapping points

Materials

Materials used for the original production version of this dataset were built using Disney Animation's proprietary in-house toolset. To facilitate the transfer to USD, we have simplified these materials to a single, common material called "**BaseMaterial"**, which is built around the Renderman shader **PxrDisneyBsdf**, an implementation of Disney Animation's **Principled Shader**.



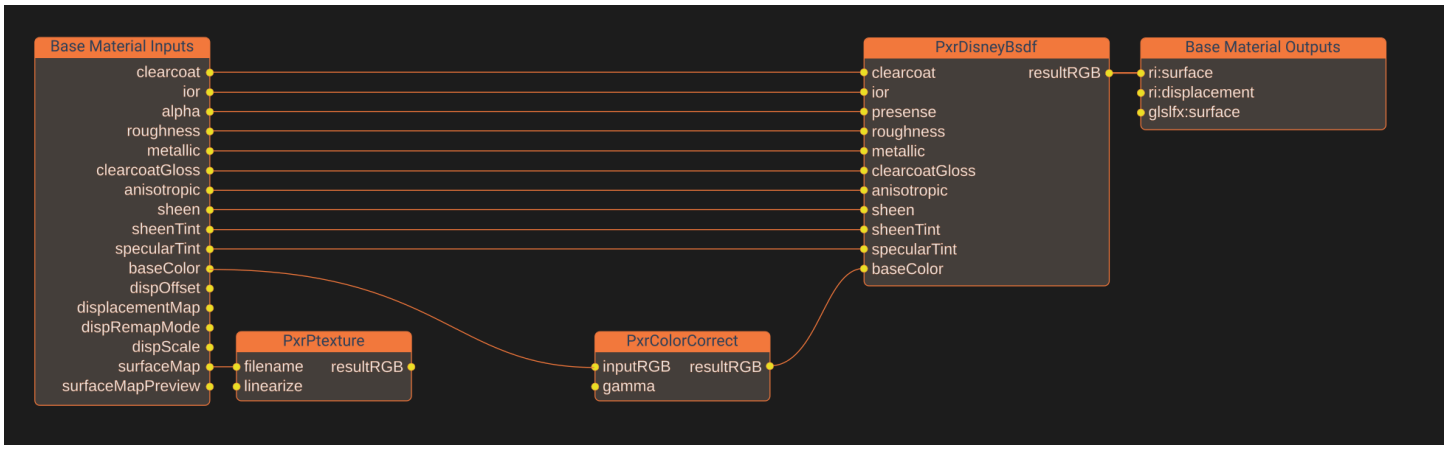
BaseMaterial

BaseMaterial provides three render targets:

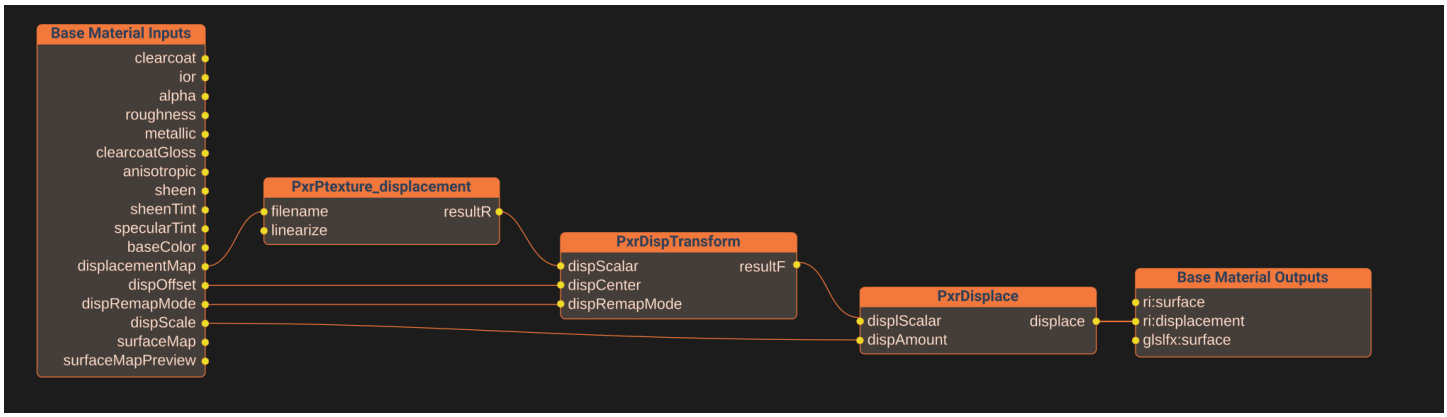
1. PxrDisneyBsdf - Renderman surface shader
2. PxrDisplacement - Renderman displacement shader
3. UsdPreviewSurface - Storm surface shader

Material input connections, such as **baseColor**, **clearcoat**, **displacementMap**, are connected to their respective counterparts in the enclosed **UsdShadeShader** nodes. Input connection values are overridden by each bound primitive's values from the original JSON/OBJ data set and from the material palettes in the original production shot.

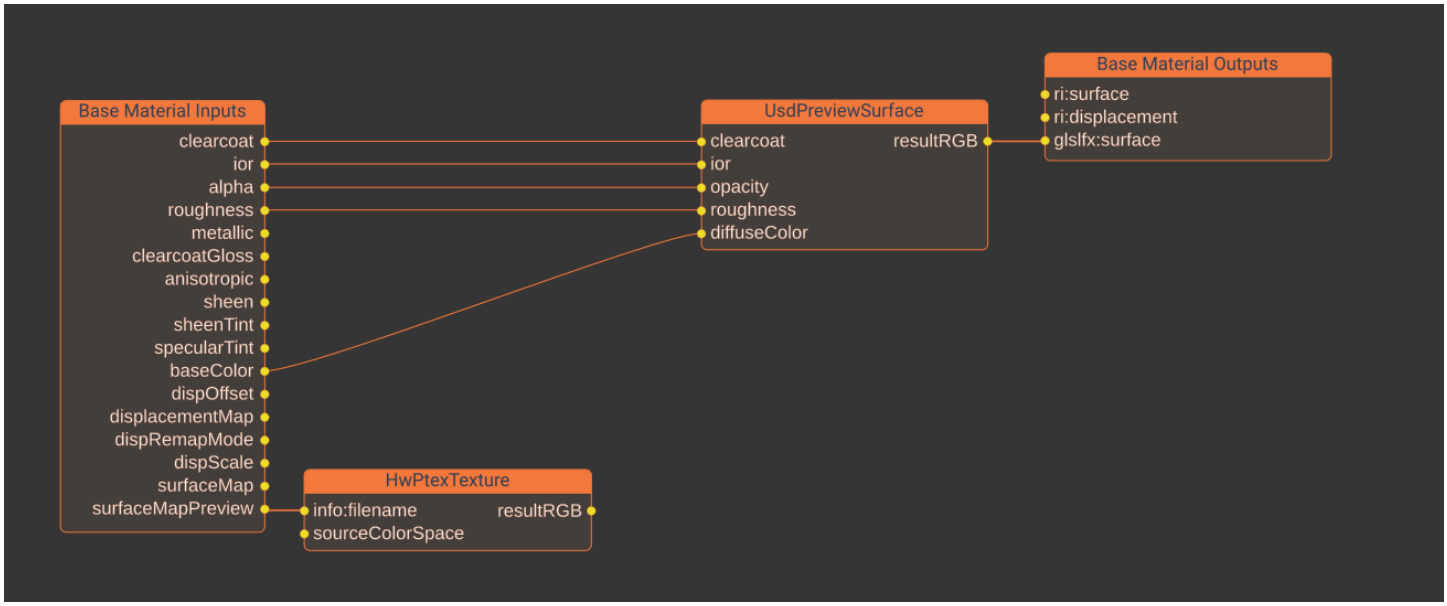
Colorspace conversion from sRGB to Linear is approximated through the use of the PxrColorCorrect node for Renderman shaders and through the use of the **sourceColorSpace** parameter on the **HwPtexTexture** node for Storm. For **BaseMaterial**, colors are set through the **Color3f** input, **"baseColor".**



BaseMaterial Renderman surface shading network



BaseMaterial Renderman displacement shading network



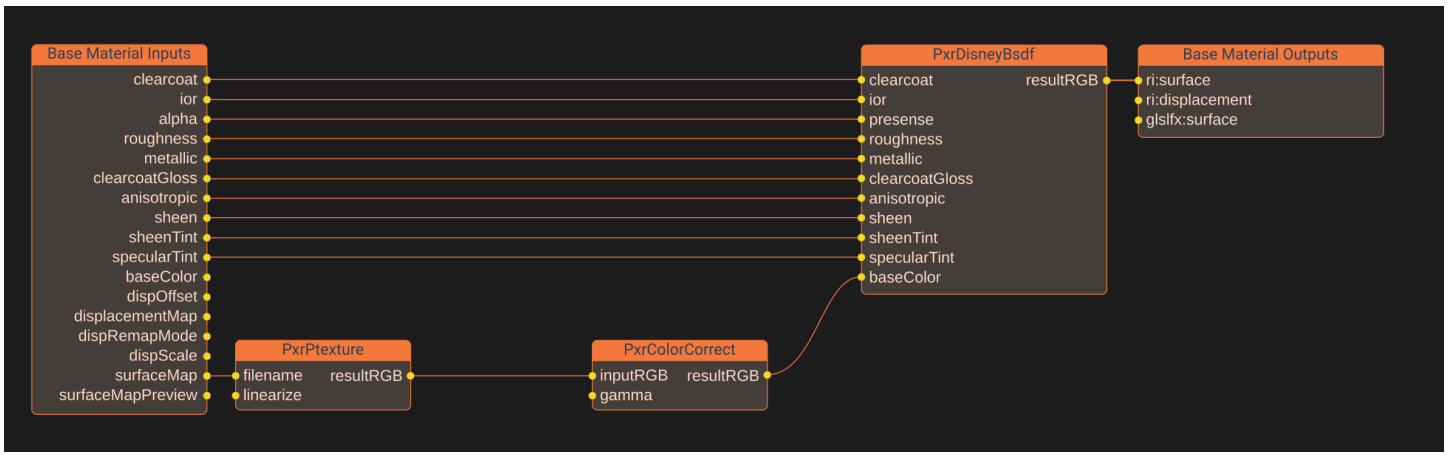
BaseMaterial Storm surface shading network

_class_BaseMaterial

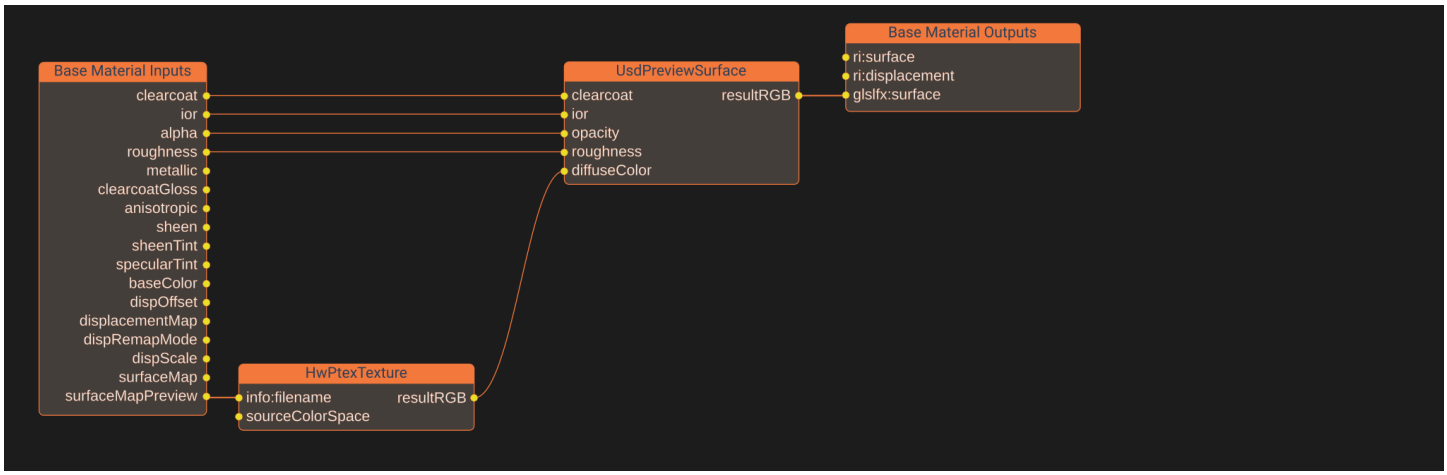
BaseMaterial inherits from <_class_BaseMaterial> so that referencing contexts (e.g. shots, or just super-layers of the island.usda) can add to or modify the definition inside a local </_class_BaseMaterial> prim, without needing to edit this layer.

PtexBaseMaterial

The material **PtexBaseMaterial**, which is a descendant of **BaseMaterial**, is used in cases where a texture map is provided. It overrides the shader color inputs to use a texture map instead of a constant color.



PtexBaseMaterial Renderman surface shading network



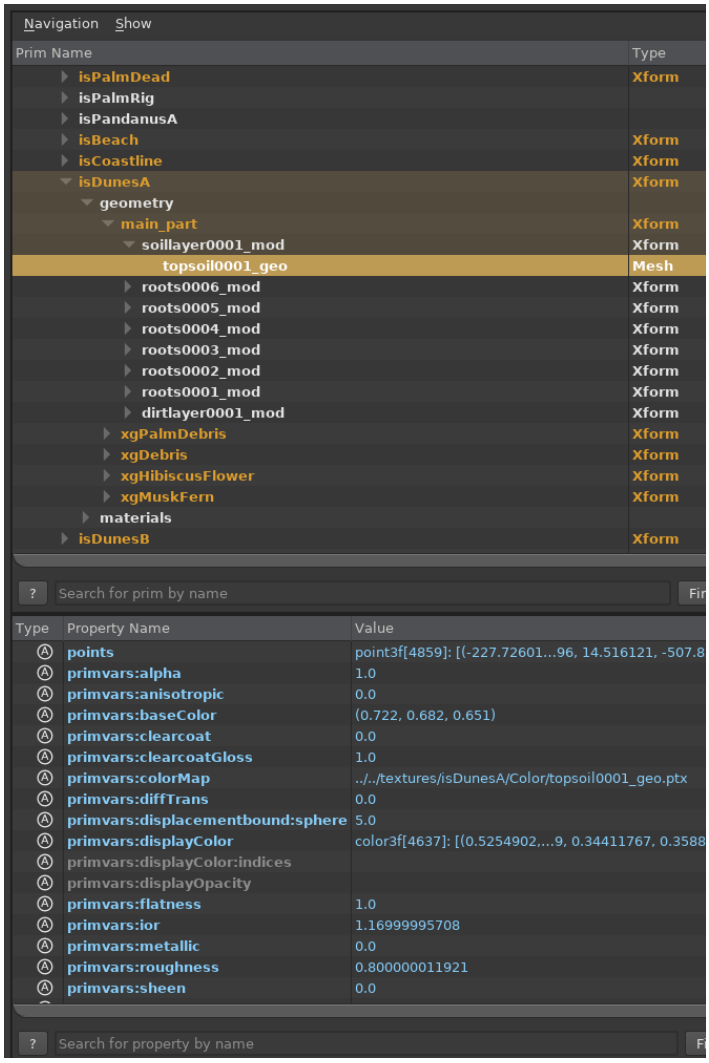
PtexBaseMaterial Storm surface shading network

Storing material attributes on the mesh

Primvars

To help support development on other renderers, a primvar attribute has been created on the mesh for each attribute in the json file for that element.

materials.usda



materials.json


```
    "soil": {
        "diffTrans": 0.0,
        "baseColor": [
            0.722,
            0.682,
            0.651,
            1.0
        ],
        "specTrans": 0.0,
        "colorMap": "textures/isDunesA/Color/",
        "clearcoatGloss": 1.0,
        "assignment": [
            "topsoil0001_geo"
        ],
        "clearcoat": 0.0,
        "specularTint": 0.0,
        "ior": 1.17,
        "metallic": 0.0,
        "flatness": 1.0,
        "sheen": 0.0,
        "sheenTint": 0.5,
        "anisotropic": 0.0,
        "alpha": 1,
        "roughness": 0.8,
        "type": "thin",
        "displacementMap": "textures/isDunesA/Displacemen
    },
```


Ptex Usage

Ptex maps have also been baked into the “displayColor” attribute on each mesh. This should allow renderers that don’t support ptex to still have some access to color information.

Ptex in RenderMan

Ptex in UsdPreviewSurface

Ptex baked into displayColor attribute

Displacement Shaders

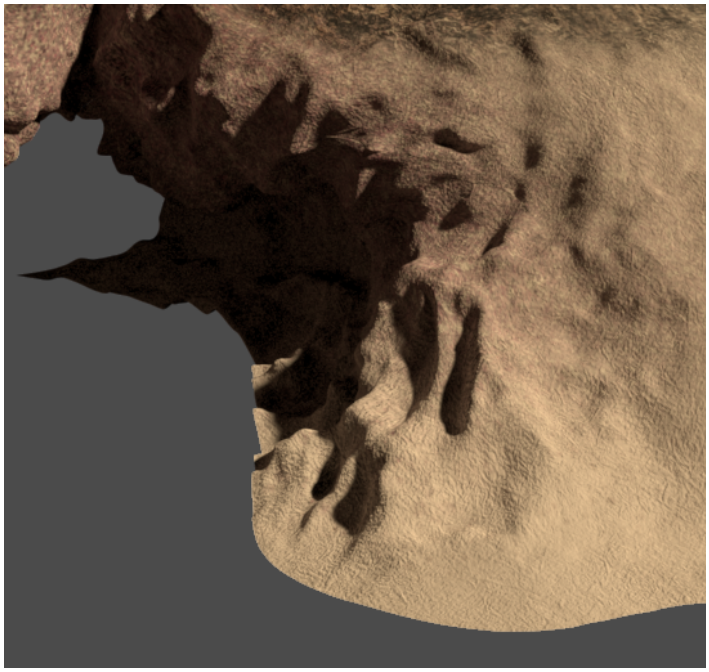
Displacement shaders have been implemented for RenderMan materials. Displacement bounds have been set as attributes on the mesh. All the displacement maps have been normalized between 0 and 1.

displacement mesh attributes:

- primvars:dispScale - displacement multiplier.
- primvars:dispOffset - shift displacement value to support negative displacement.
- primvars:displacementMin - minimum displacement value.
- primvars:displacementMax - maximum displacement value.
- primvars:displacementMap - path to displacement texture.

Displacement Shader Examples

isHibiscus displacement shaders on trunk bark



isDunesA topsoil displacement

Textures

This USD data set still uses the ptex texture files from the original release with the following additions copied over from the original shot:

- isHibiscus
    - Detail map for trunk bark.
- isKava
    - Ptex flood filled maps for branches, tallroot and smallroot meshes.
- isDunesA
    - Restored the original layers of displacement maps for topsoil0001.
- Environment maps
    - PRMan lat-long environment maps for skydome and camera.

element	path
isHibiscus	island/textures/isHibiscus/Color/trunk_base_geo_features.pts
isKava	island/textures/isKava/Color/tallroot*.ptx island/textures/isKava/Color/smallroot*.ptx island/textures/isKava/Color/isKava_base_hBranch*.ptx
isDunesA	island/textures/isDunesA/Displacement/topsoil0001_geo_duneSide.ptx island/textures/isDunesA/Displacement/topsoil0001_geo_smushy.ptx
environment maps	/island/textures/islandsunEnv.tex /island/textures/islandsunCam.tex

Lights

Reproducing the lighting from the Hyperion render was particularly challenging. Beyond the placement of lights in the scene, there were other lighting features which contributed heavily to the final look of the Hyperion render. We were also limited in what lighting parameters we could adjust through the PRMan Hydra delegate.

We first translated lighting information from lights.json into lights.usda. Quad lights were converted into UsdLuxRectLights. Color values have had a 2.2 gamma correction applied to them.

sky_dome_llc
The Hyperion dome light allowed for two environment maps to be used. One for lighting the scene and another to be visible to the camera. In order to duplicate this in USD, the dome light, “sky_dome_llc”, was split into two dome lights for the USD conversion:

- sky_dome_env_llc
    - This light is used to light the scene.
    - Visibility to the camera is disabled.
- sky_dome_cam_llc

- This light is used as a background image for the camera.
- Visibility to the camera is enabled.
- This light excludes everything under /island

sky_dome_env_llc environment map

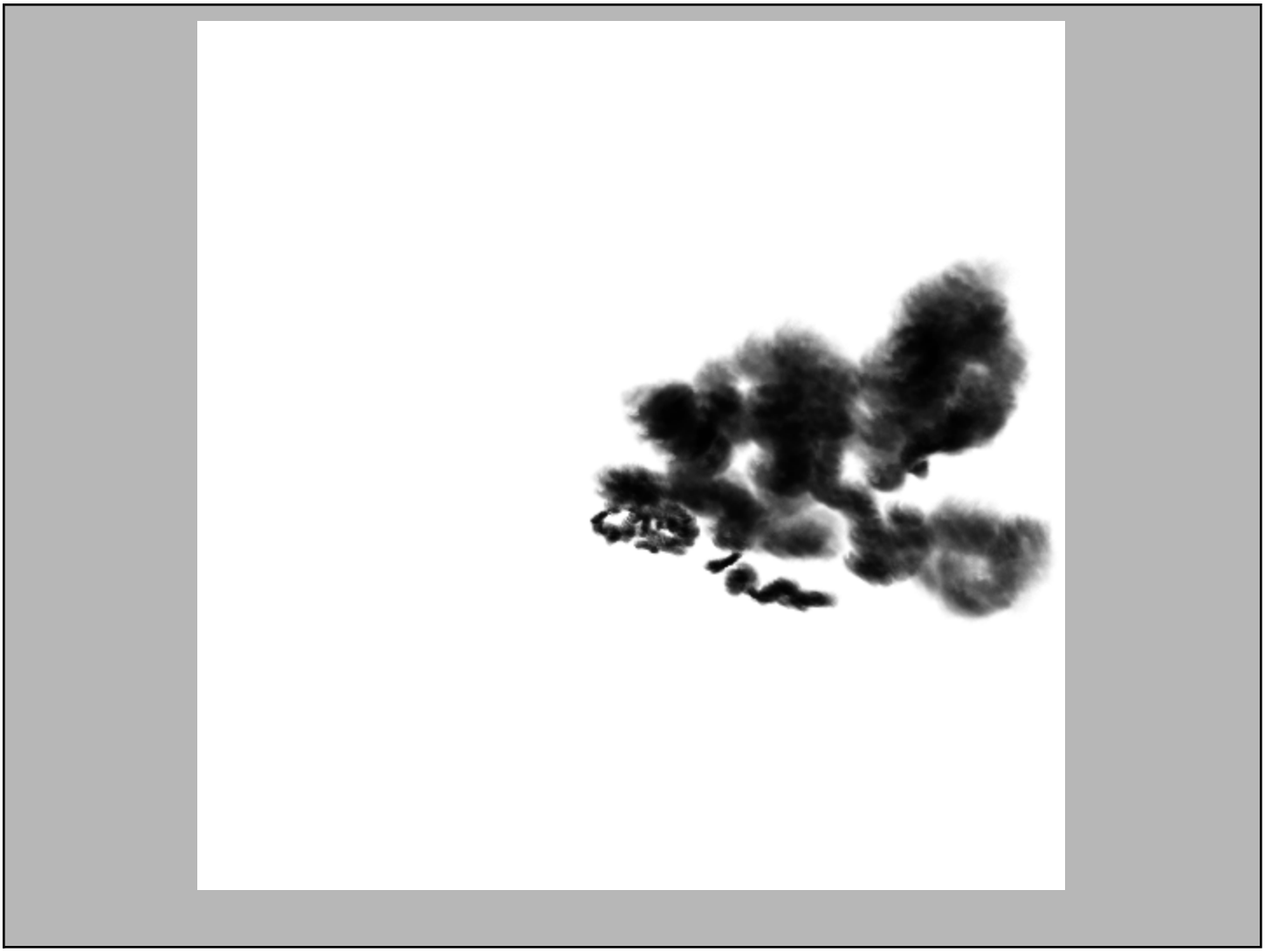
sky_dome_cam_llc camera map

Light filters

The quad light “sun_quad_llc” contributed to the overall illumination and warmth of the lighting in Hyperion. To add more interesting variations, Hyperion used the texture map “islandsun_cloudmap.ptx” as a cookie filter. This allowed artists to “paint” the mountains in the background to receive less light and the palm trees in the foreground to receive more light. While the USD API supported the creation of shader based light filters, the HdPrman delegate currently does not support the translation of them to PRMan.

Hyperion - with cookie filter

Hyperion - without cookie filter



“islandsun_cloudmap.ptx” image used for cookie filter.

en_SUN_refract

Hyperion also used photon mapping to illuminate and add caustics to the regions below the ocean surface. While we were able to allow lights to illuminate the ocean floor using path-tracing, we were not able to enable photon mapping to generate caustics. This was due to current limitations in enabling photon mapping in PRMan through the HdPrman delegate. The light, “en_SUN_refract” was converted from the original production shot. It was used for generating caustics underneath the ocean. While we were not able to get caustics working, we have included this light and disabled it for future testing.

Element Notes

In this section we provide an overview of all the elements along with reference renders from both RenderMan and the GL viewer in Usdview. Where relevant we also provide notes about changes that have been made specifically for the USD release.

IsBayCedarA1

Model Variants

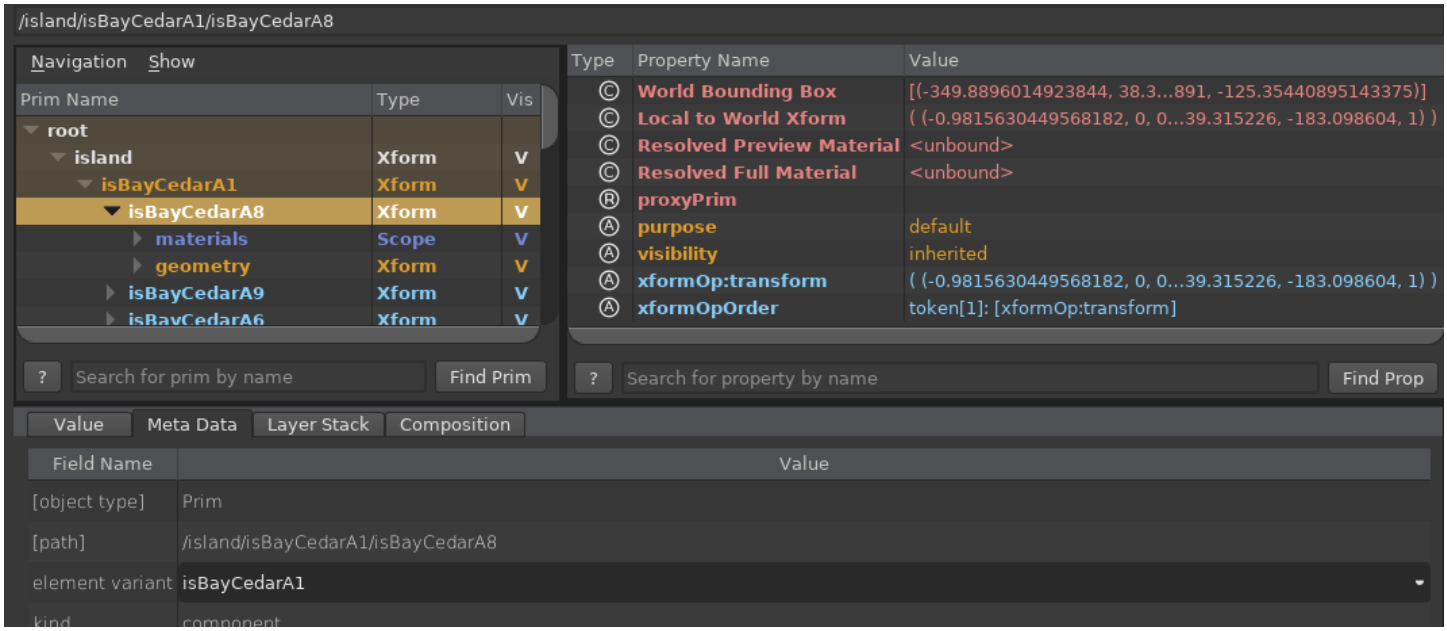
bonsaiA

bonsaiB

bonsaiC

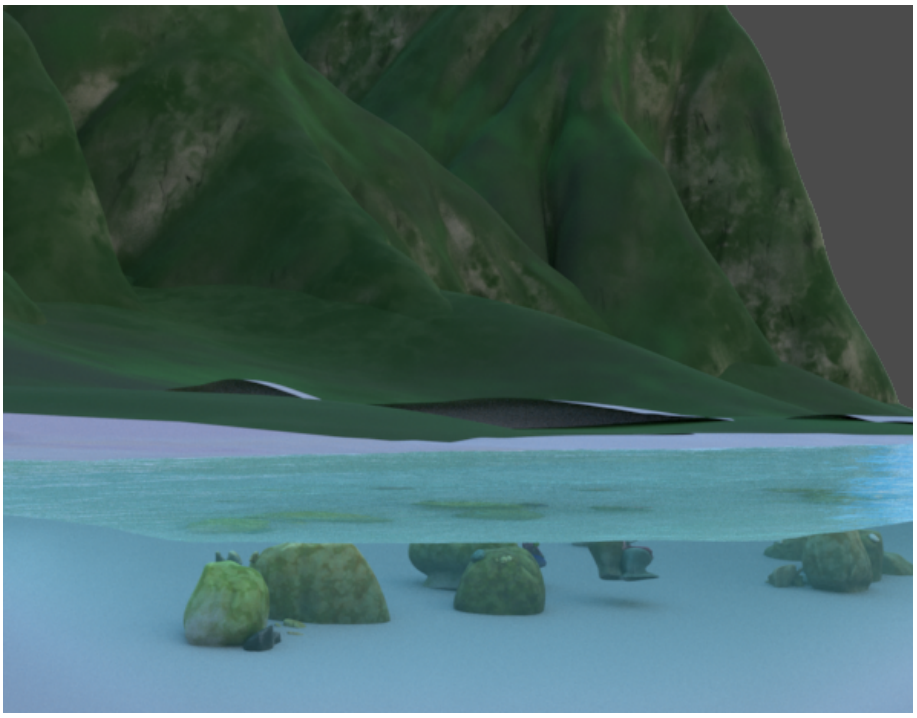
isBayCedarA1

Element variants can be set in the instance primitive.



OsOcean

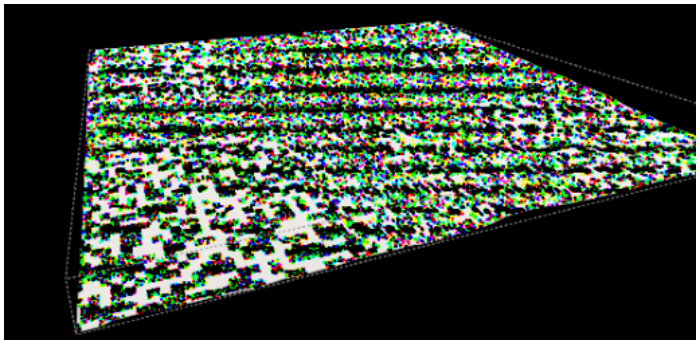
The ocean element from the previous pbrt/obj version of this dataset has been updated for this USD release. In the original production shot, the ocean surface mesh and underwater volumetrics were procedurally generated at render time. A cached version of that procedurally generated ocean surface mesh was included in the previous pbrt/obj version of this data set and is included in this version as well. New to the USD version is the addition of a volume element under the ocean surface. It is a hazy volume that extends from the water surface to the ocean floor (see image below). It has been constructed to have a similar appearance as the original volume element. The PxrDisneyBsdf shader node has been replaced with PxrSurface to take advantage of some of the built-in diffuse transmission parameters.



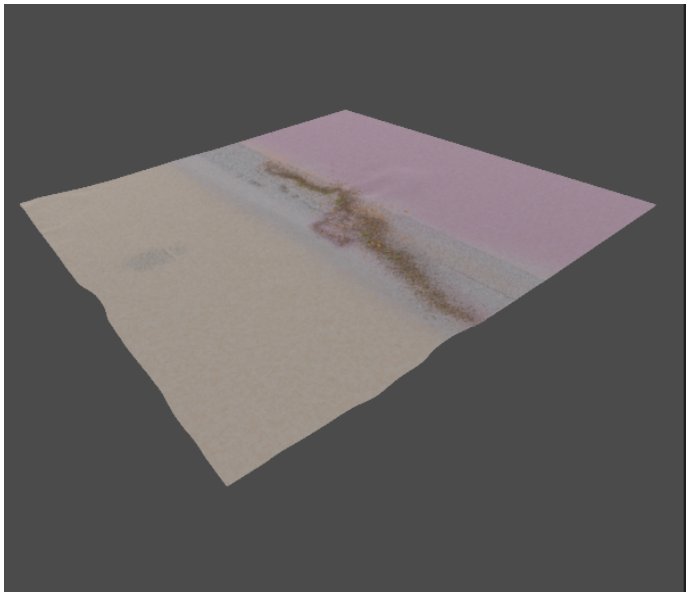
osOcean volume

isBeach

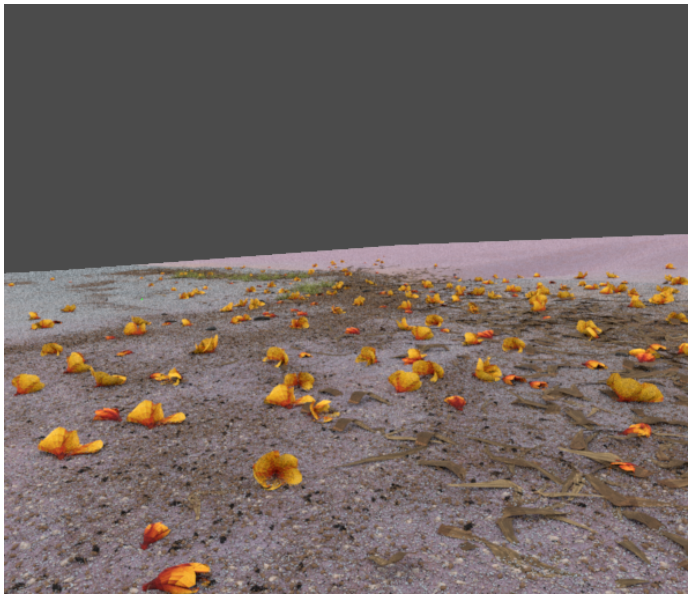
Several meshes from this element suffered from render artifacts (see Geometry Fixes)



fireflies and other artifacts



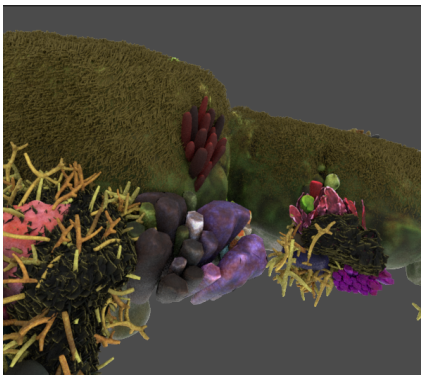
Distant view



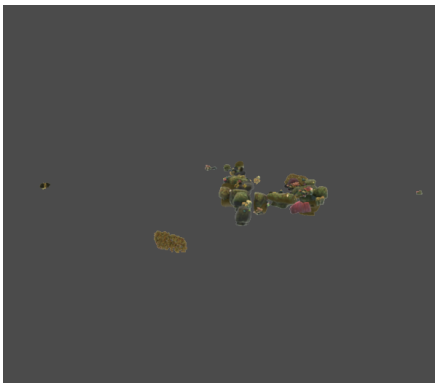
Close-up view

isCoral

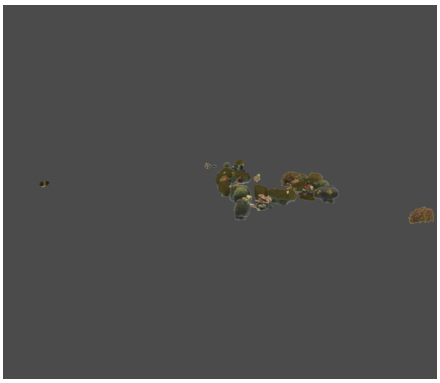
Model Variants



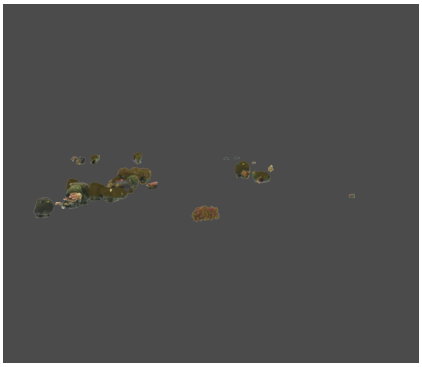
isCoral



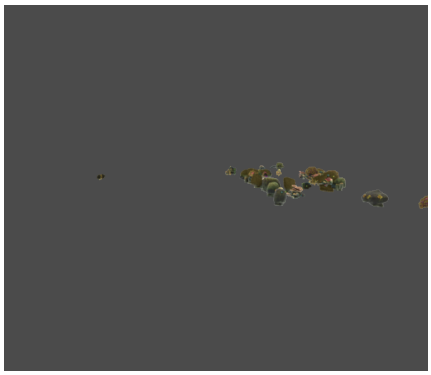
isCoral1



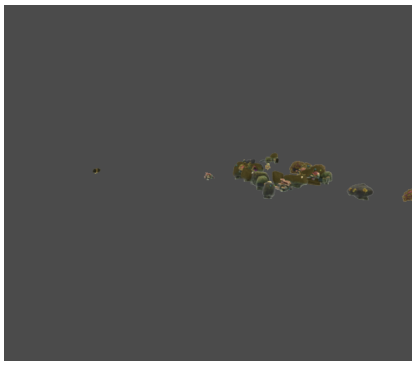
isCoral2



isCoral3

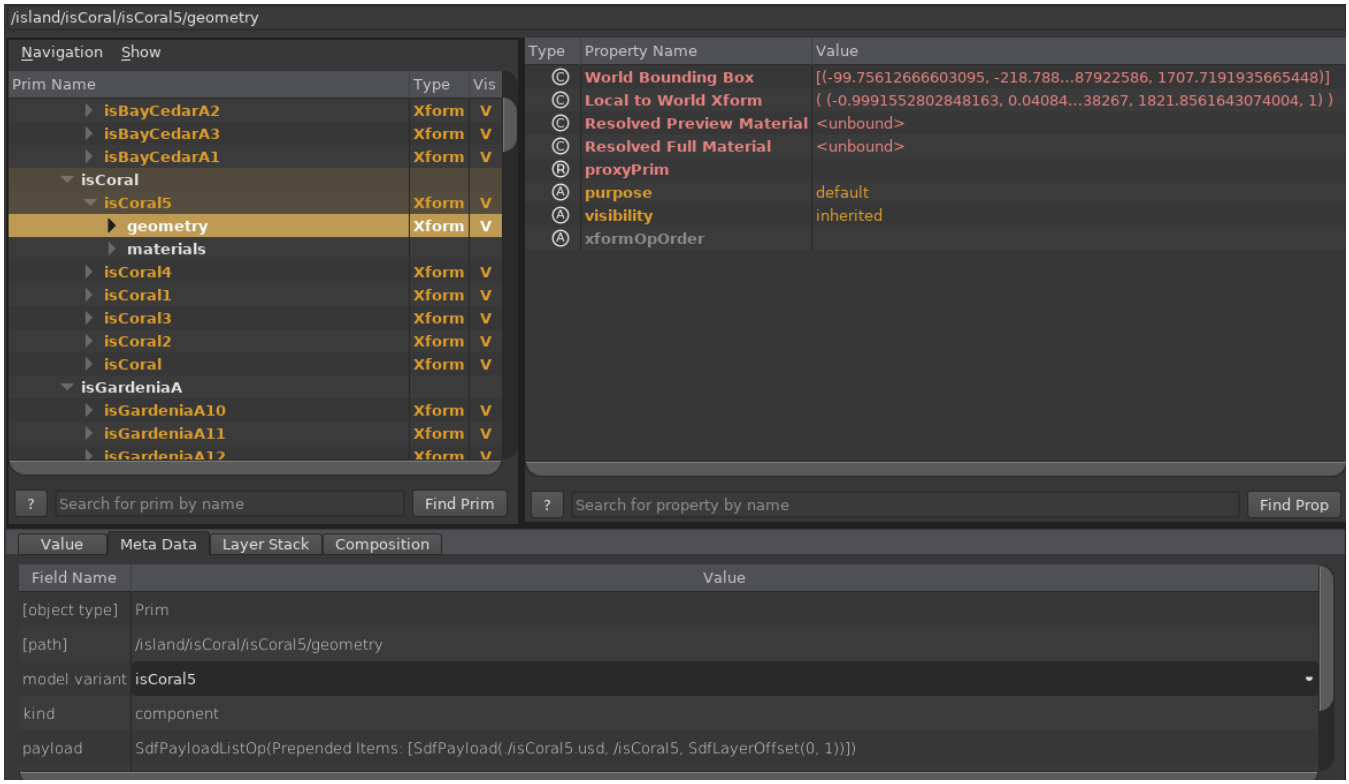


isCoral4



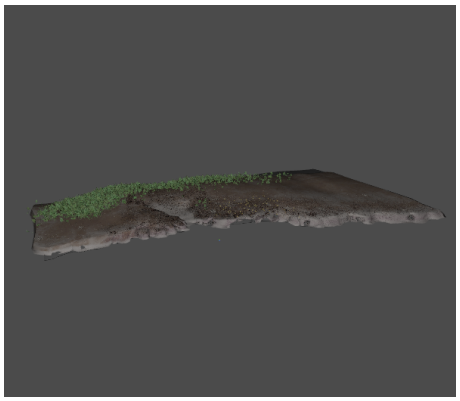
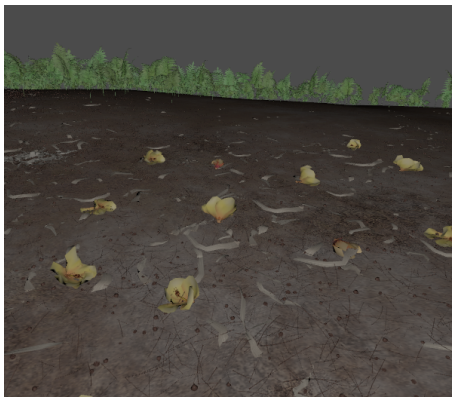
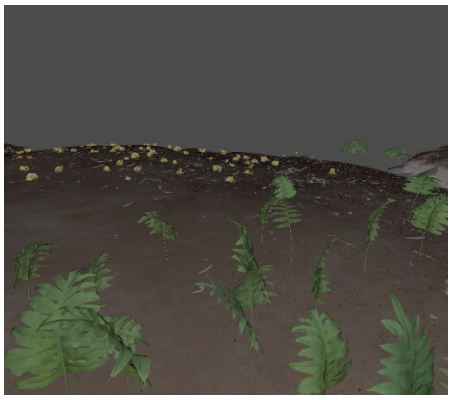
isCoral5

Model variants can be found in each element's "geometry" primitive.

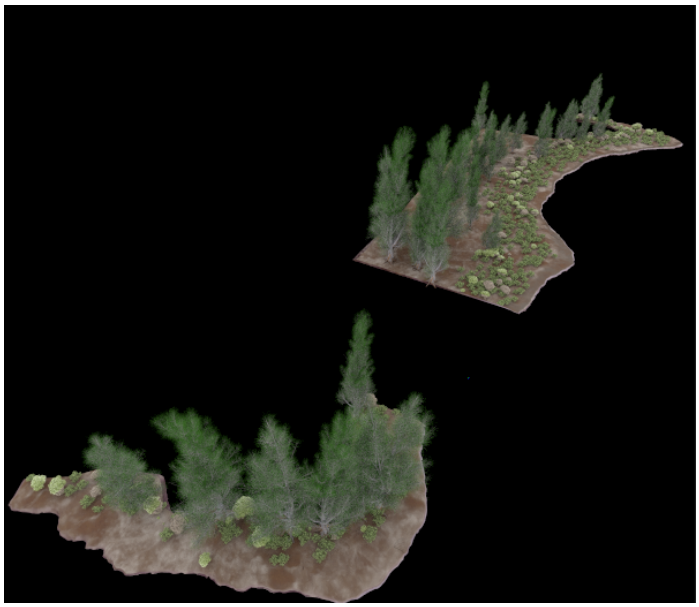
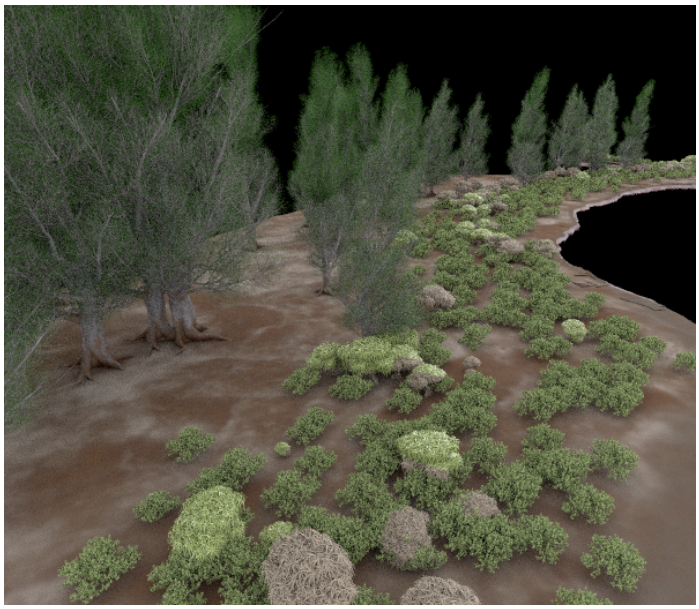


isDunesA

Topsoil0001 has a custom displacement shader that takes advantage of some additional displacement maps.

isDunesB

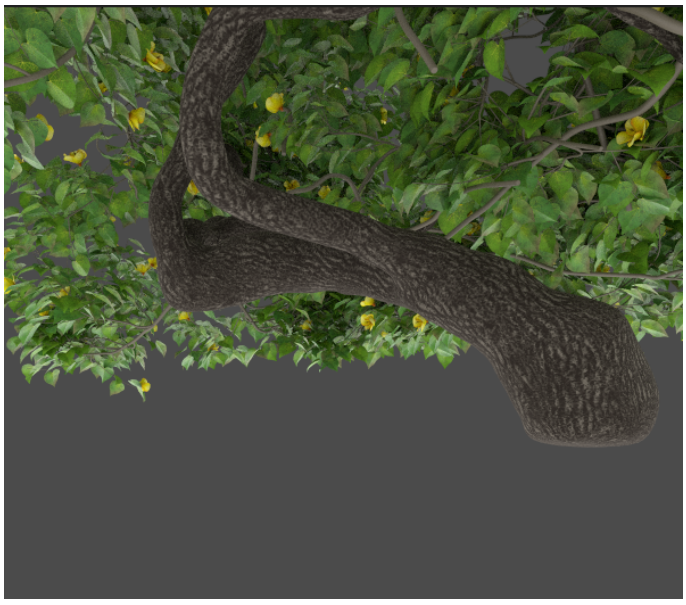
isGardeniaA

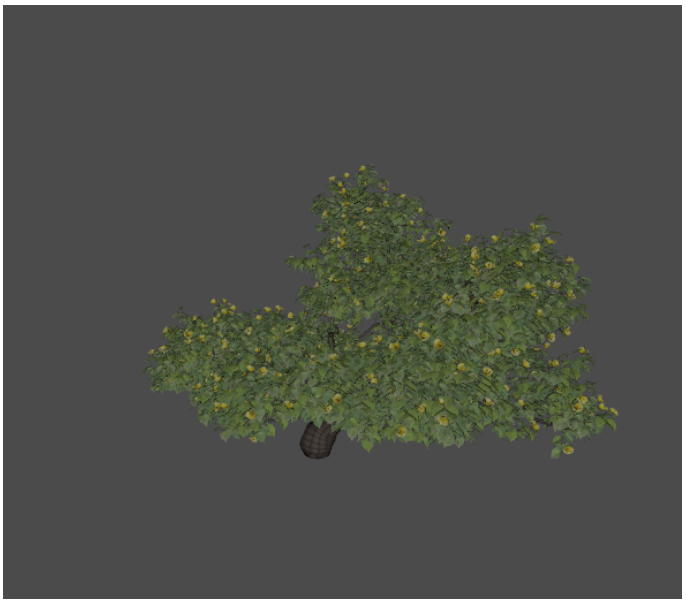
GL Viewer

RenderMan

isHibiscus

A detailed surface map and bump map has been added to the trunk geometry.





GL Viewer

RenderMan

isHibiscusYoung

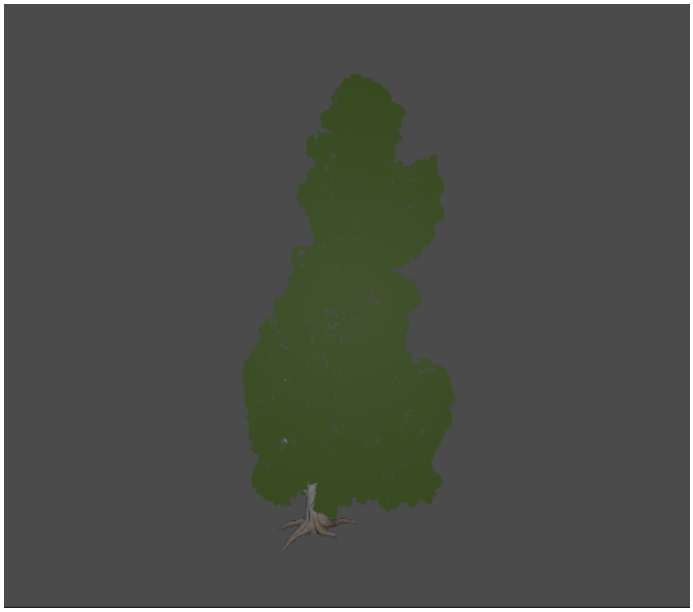
The previous PBRT/OBJ release did not include the trunk geometry for this element. The USD version has been updated to include the trunk geometry.

GL Viewer

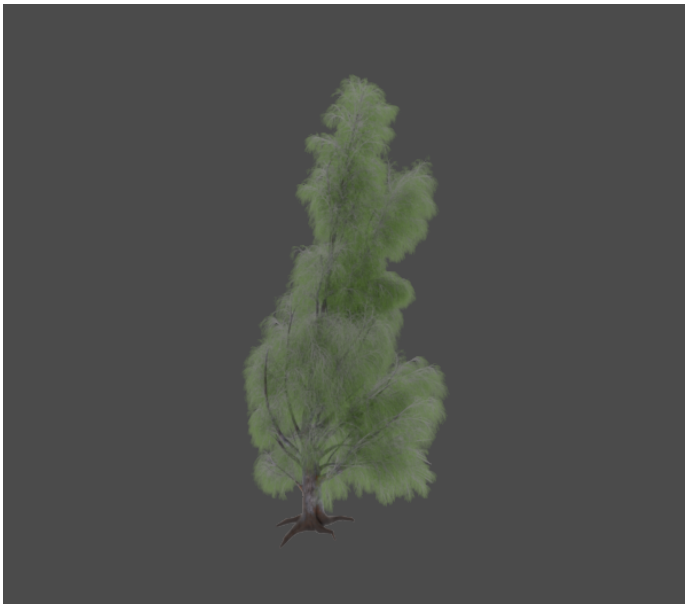
RenderMan

isIronwoodA1

Re-exported curves from original production shot. Converted curves from polylines to basis curves. The needle baseColor was re-extracted from the original production shot.



GL Viewer



RenderMan

isIronwoodB

Re-exported curves from original production shot. Converted curves from polylines to basis curves.

GL Viewer

RenderMan

isKava

GL Viewer

RenderMan

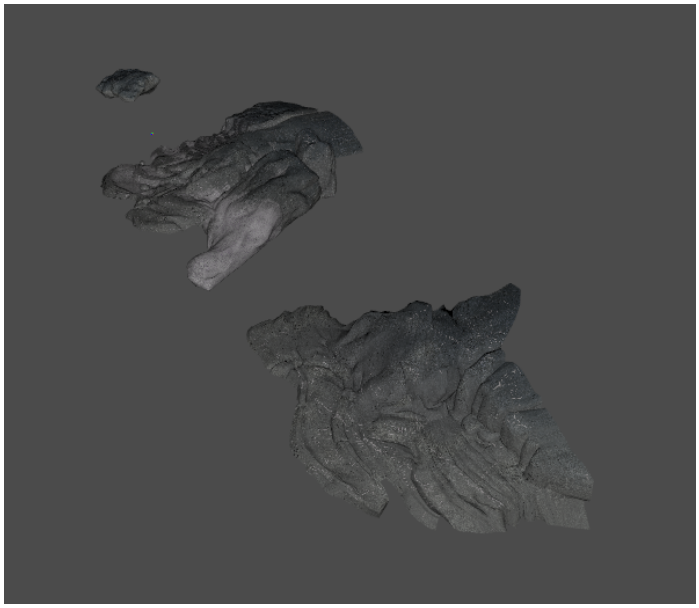
isLavaRocks

We noticed that the OBJ version of this mesh differed from the Hyperion render. So we rebuilt it from the original shot.

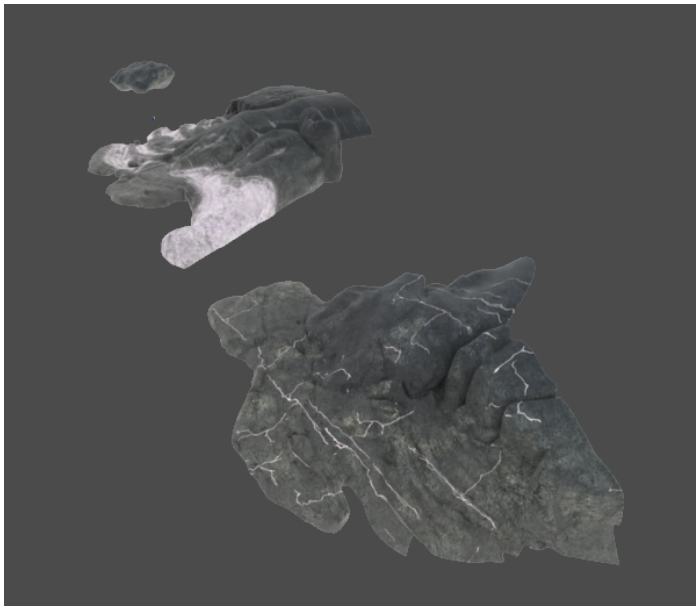
RenderMan - Initial conversion to USD (missing geometry circled)

Hyperion render

RenderMan - Fixed mesh

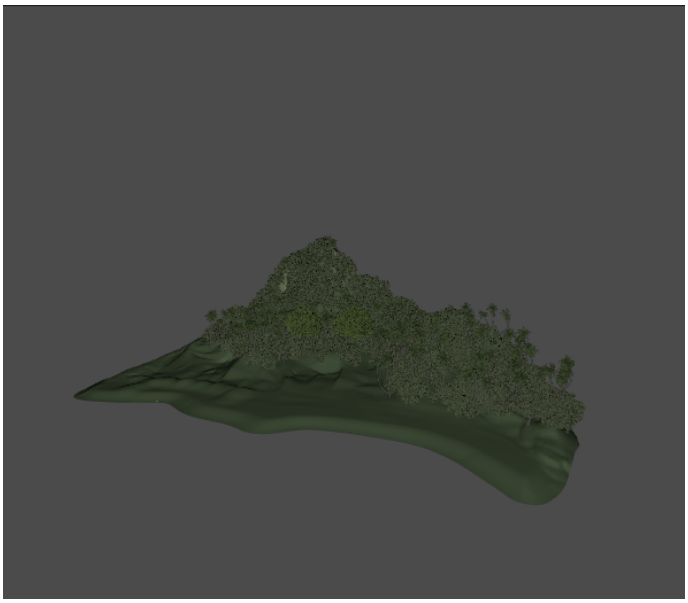


GL Viewer

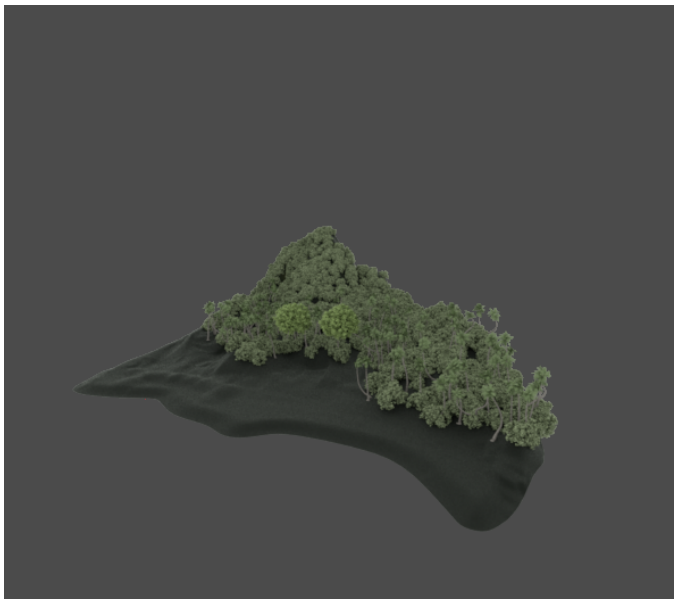


RenderMan

isMountainA

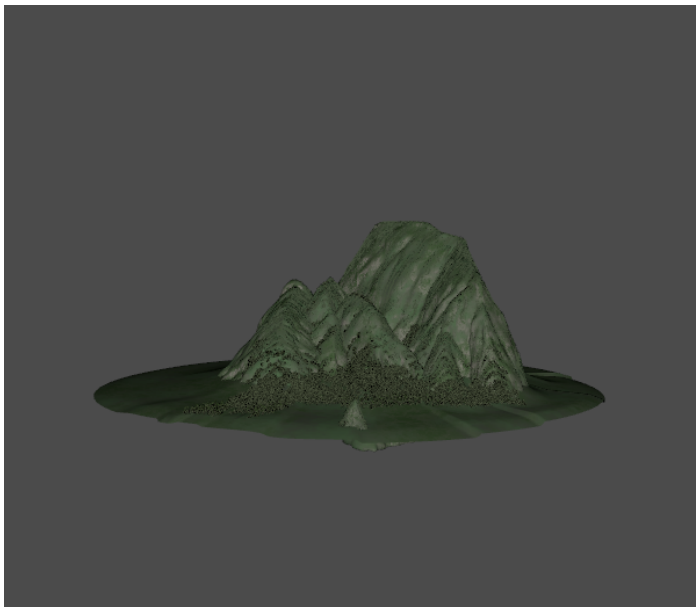


GL Viewer

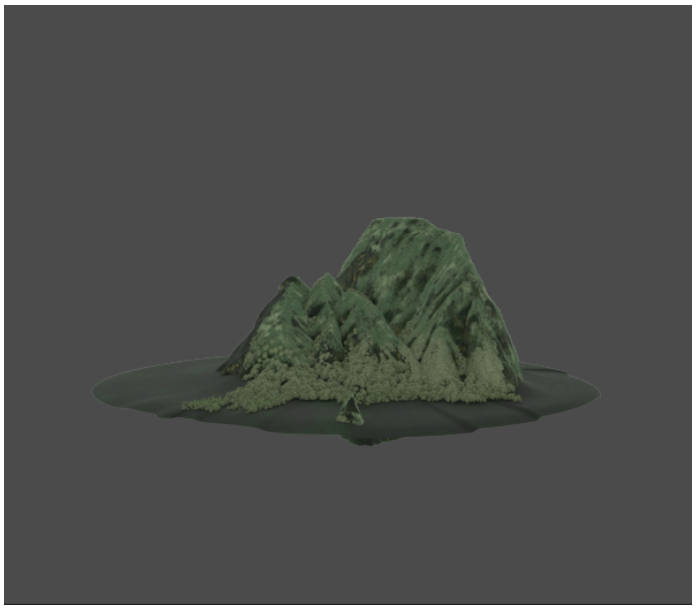


RenderMan

isMountainB

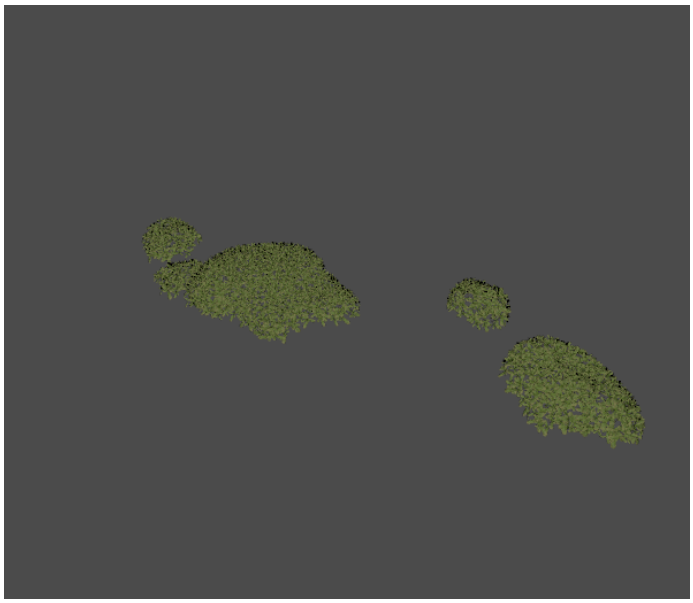


GL Viewer

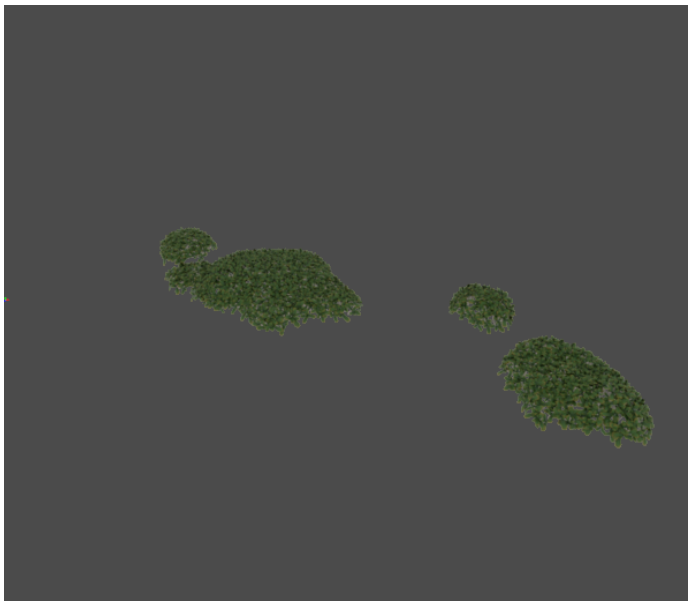


RenderMan

isNaupakaA



GL Viewer



RenderMan

isPalmDead

GL Viewer

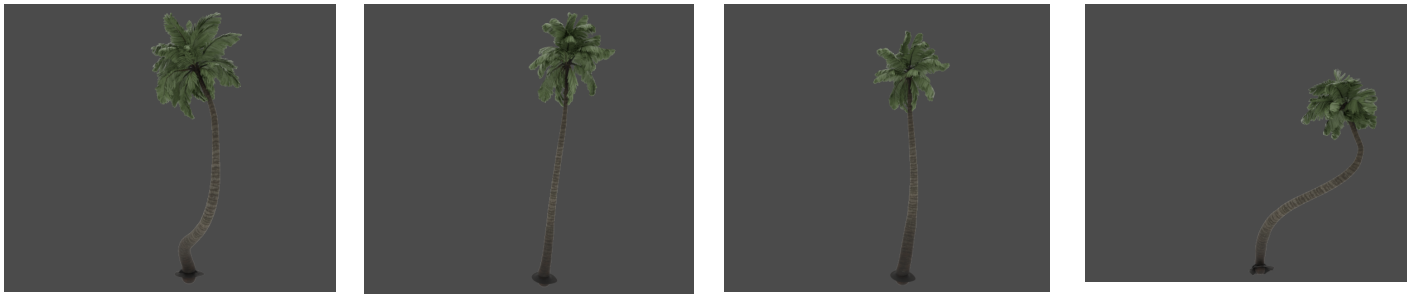
RenderMan

isPalmRig

The PxrDisneyBsdf shader node bound to the frond leaves has been replaced with PxrSurface to take advantage of the diffuse transmission parameters.

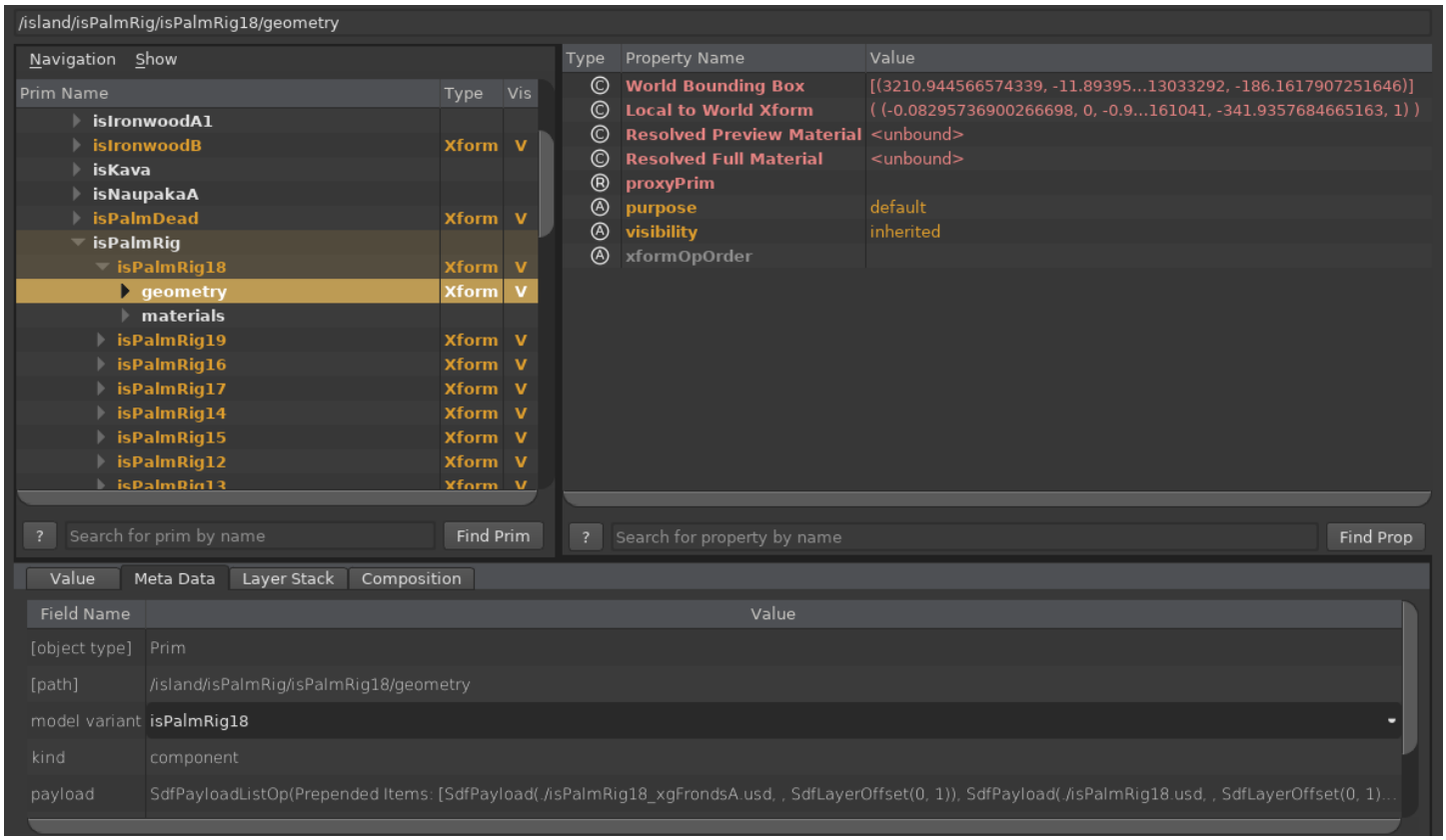
GL Viewer

RenderMan



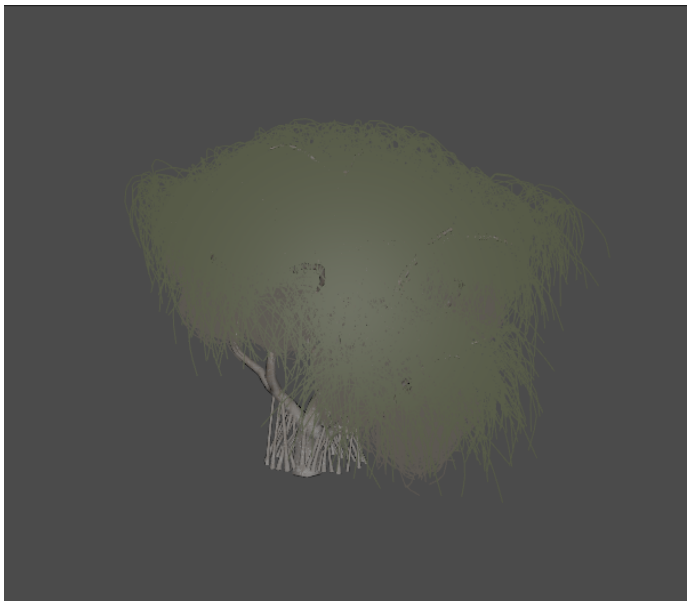
isPalmRig model variants (4/34)

Variants can be set in each element's "geometry" primitive.



isPandanusaA

We went back to the original XGen definition for the leaves to better match the tapered width for each curve.



GL Viewer

RenderMan

Limitations and Future Work

The conversion of the Moana Island Scene to USD has been a learning process for everyone involved. While we feel we've met our initial goal of creating a data set capable of rendering a final quality image, there is still room for improvement in the future.

The current data set is focused on final rendering. Future versions could take advantage of the USD "Purpose" attribute to offer a data set that is also optimized for interactive viewing. This would most likely mean doing another pass on each element to further optimize it for reduced complexity, memory usage and load times. Included in this optimization would be restructuring the data to allow wider usage of the "instanceable" attribute to improve memory performance. The re-introduction of animated geometry is another potential upgrade for this data set. As mentioned in **Lights** section, support in HdPrman for caustics and light filters would benefit the overall look and feel of the renders.

Finally, we have focused on using RenderMan as the target renderer for this release, but we hope that it provides a template for how to experiment with other renderers.

Acknowledgements

We wish to thank all those who lent their valuable time and expertise to this project.

Ray Haleblian
David Aguilar
Shaila Haque
Heather Pritchett
Ricky Rieckenberg
Dan Teece
Mark Lee
Matthew Willams
Joe Longson
F. Sebastian Grassia
Pixar's USD team
Pixar's RenderMan team
Blue Sky Studios
SideFX Software