USD and MaterialX on the Web

Implementing WebGPU for the Hydra Storm Renderer

Henrik Edstrom, Autodesk
Aura Munoz, Autodesk

July 11, 2023
Open Standards and Open Source @ Autodesk
What is MaterialX?

• Expressive, flexible graph-based material system
• Describes intent and not implementation
  • Authoring tool agnostic
  • Renderer agnostic - can target both high-end off-line (e.g., Arnold) and various real time engines
• Node-based pattern generation / procedural textures
• Can express different PBR models, including Standard Surface, USD Preview Surface, and glTF PBR
• Extensible shader generation
  • GLSL / GLSL ES
  • OSL
  • MDL
  • ...
MaterialX on the Web - Autodesk Contributions

- Web Assembly Build & JavaScript Bindings
- WebGL2 compatible Shader Generation (framework agnostic)
- MaterialX Viewer for the Web
- Was released in April 2022 as part of the official version 1.38.4
- Supported Browsers: Chrome, Firefox, Edge, Safari
- Slack: #materialx channel on academysoftwarefdn.slack.com

Version 1.38.4

- Added a sample Web Viewer, built and deployed through GitHub Actions.
- Added a MaterialX graph for the glTF PBR shading model.
- Added new ‘worleynoise2d’ and ‘worleynoise3d’ nodes, with implementations in GLSL, OSL, and MDL.
- Added new ‘surface_unlit’ node, with implementations in GLSL, OSL, and MDL.
- Added support for the glTF geometry format in MaterialXRender and MaterialXView.

AMD MaterialX library: https://matlib.gpuopen.com/main/materials/all
Universal Scene Description - OpenUSD

- Extensible framework for collaboration on complex scenes
- Robust scene interchange between tools from different vendors
- Hydra rendering abstraction layer and Storm rasterizer
USD and Hydra on the Web - Autodesk Contributions

- Support WebAssembly
  - Additional compile target
  - Add support for 32bit Emscripten architecture
  - Wasm TBB open-source project
    - Fix by us that prevents crashes
- JavaScript bindings
  - USD and Hydra
- hdJavaScript
  - THREE.js RenderDelegate
- Early draft PRs Dec 2021 / Feb 2022:
  - https://github.com/autodesk-forks/USD/pull/1
  - https://github.com/autodesk-forks/USD/pull/2
- Slack: #wg-usd-web channel
  - academysoftwarefdn.slack.com

Demos: https://autodesk-forks.github.io/USD/

Current/active branch:
https://github.com/autodesk-forks/usd/tree/adsk/feature/webgpu

USD Proposal:
USD and WebGPU

• Why WebGPU?
  • Cross platform, including desktop applications
  • Unlocks compute capabilities
  • Modern graphics API with possibilities to grow
  • Write storage buffers
Hydra

- Scene Delegate: UsdImaging
- Render Delegate: HdStorm, HdPrman, HdEmbree, HdAurora

Application

Hydra
Current HdStorm Render Pipeline

Stage Scene Data → UsdImagingDelegate → HdRenderIndex → HdStRenderDelegate → Viewport

Create, Destroy, Commit, Draw

HGI (Hydra Graphics Interface)

HgiGL → GLSL
HgiMetal → MSL
HgiVulkan → SPIR-V

Driver compiler → GPU

Resources:
- Command
- Texture
- Sampler
- Buffer
- Shader Function
- ... ...
Extended HdStorm Render Pipeline

Stage Scene Data → UsdImagingDelegate → HdRenderIndex → HdStRenderDelegate → Viewport

HGI (Hydra Graphics Interface)

Resources:
- Command
- Texture
- Sampler
- Buffer
- Shader Function
- ...

HGI DX → HGI GL → HGI Metal → HGI Vulkan

HLSL → GLSL → MSL → SPIR-V → WGSL

Driver compiler → GPU

Knife
- color
- blade
- .color
- body
3-step Technical Path of POC

HgiWebGPU Only

Implement the initial WebGPU Hydra Graphics Interface - HgiWebGPU

Dawn-based Solution

Developed and completed HdStorm pipeline based on Google Dawn using USDSample as test application

Emscripten-based Solution

Compiled USD to WASM with emscripten to enable Hydra Storm pipeline
Demos
Capabilities

• General functionality
  - USD file loader, including composite capability
  - Hydra rendering framework
  - Flexible plugin system
  - Scene Delegate with UsdImaging

• HdStorm functionality
  - Efficient mesh batching
  - Highlighting for faces, points, edges
  - GLSL materials
  - Compute kernels
  - UsdPreviewSurface support
  - Color Correction
  - Cross platform support for Windows / Mac / Linux / Browsers
  - And more!
Limitations

• Missing built-ins:
  - gl_primitiveID
  - barycentric coords
  - atomicAdd
  - missing support for float32 filtering
  - Tessellation and Geometric stages
  - gl_Pointsize is readonly

• Reached limits:
  - Max bytes per sample
  - Max interstage attributes
  - Max write storage per stage
Links


• USD Web Working Group (ASWF) - [https://wiki.aswf.io/display/WGUSD/USD+Web+Visualization](https://wiki.aswf.io/display/WGUSD/USD+Web+Visualization)
  • Slack: #wg-usd-web channel [https://academysoftwarefdn.slack.com](https://academysoftwarefdn.slack.com)

• MaterialX (ASWF) – [https://materialx.org](https://materialx.org)
  • Slack: #materialx channel [https://academysoftwarefdn.slack.com](https://academysoftwarefdn.slack.com)
A recording of this presentation will be available at
https://www.khronos.org/events/webglwebgpu-meetup-july-2023

For more information on WebGL, please visit
https://www.khronos.org/webgl

Email: public_webgl@khronos.org