MaterialX Interoperability in USD/Hydra

Overview:

MaterialX is an open standard for representing rich material and look-development content in computer graphics, enabling its platform-independent description and exchange across applications and renderers. MaterialX content can be encoded in other containers such as Usd, gltf and other formats. Similar to how a file can be transported in a zip package and ensure it's integrity, it is essential that when MaterialX data flows in and out of these containers it's integrity is maintained.

The current MaterialX import in handled by USD Scene Delegate using the UsdMtlx library. MaterialX document can be imported into Usd by:

- Attaching a MaterialX file e.g. usd_mtlx_example.usda.txt
- Run MaterialX file through Usdcat to flatten a mtlx reference (e.g. run usdcat *-f)* usd_mtlx_example_flatten.usda.txt

(see MaterialX in Hydra slides https://www.materialx.org/assets/ASWF_OSD2021_MaterialX_slides_final.pdf)

1. MaterialX as UsdShade
(transport, composition, file format plugin, interop, shader definition)

The import workflow of MaterialX to UsdShade is done by the UsdMtlx plugin. The UsdMtlx plugin translates MaterialX data to UsdShade.

The UsdMtlx documentation outlines many of the unsupported features. These limitations cause data loss and prevent use of MaterialX with Usd/Hydra for Production use. Adding support for some of these features will require more in-depth discussion with the Usd Team and more engagement from the community. However, some low hanging items can be addressed as Pull requests to Usd by the community.

Goal: Implement missing features in UsdMtlx Plugin, such that MaterialX content is maintained when imported into Usd.

These includes improvements to:

- Shader Definition: Treat MaterialX nodedef attributes (units, color space, fallbacks value, limits) as Shader data attributes instead of metadata.
- Shader Discovery: Standardize on nodedef/nodegraph name mapping for better discovery of node variants. Improve versioning and nodedef namespace.
- User Shader libraries: Simplify transport of Custom nodedef (for modeling and viewport workflow)
- Interop Integrity: For lossless synchronization, ensure validation mechanism / tests for Mtlx UsdShade Mtlx

2. MaterialX in Hydra
(visualization, viewport, runtime, rendering)

For HdStorm, UsdShade Network is translated into HdMaterial Network. A MaterialX document is reconstructed from HdMaterial Network to create glslfx using MaterialX CodeGen.

The missing features in the UsdMtlx plugin makes HdStorm hard to use and extend for other Viewport work-flows.

2.a HdMaterialNetwork from USD Scene Delegate (UsdShade)

There are (at least) two possible rendering workflows.

1. The translation process to UsdShade and back to MaterialX is undesired. UsdShade allows for resolving of inputs only. That is optimally rendering can occur from
    a. the source MaterialX reference or
    b. shader code can be independently (pre)-generated from MaterialX (or other source) but is associated with USDShade nodes (e.g. code references per node in OSL).
2. A translation process is required for UsdShade.

If MaterialX is required then CodeGen (or ShaderGen) functionality is available. Ideally this would be outside of Hydra (which is not the current case)

Therefore an application may be required to maintain MaterialX for rich material authoring workflows and translate to UsdShade networks for rich runtime visualization workflows. Often, an application might need to revert back to MaterialX for rich transport of material assets.
This dual mode operation requires unnecessary Shader Code regen such as during:

- MaterialX topology updates require complete UsdShade regen.
- MaterialX input updates require complete UsdShade regen.

For USD Scene Delegate we want to continue to use USDShade - HdMaterialNetwork as the native runtime in Hydra but we want MaterialX as the ground truth and not have static snapshots in Usd form as they are duplicates that can get out of sync.

Goals:

Improve MaterialX - UsdShade updates (both topology and inputs)

Improve HdMaterialNetwork to MaterialX update.

Improve Translation / ShaderGen independent from Hydra.

This includes improving performance issues due to shader recompilation, handling updates to materials, material topology changes. We may also need to introduce Hydra and/or MaterialX APIs to report material input and network updates.

2.b HdMaterialNetwork from Custom Scene Delegate

A custom Scene delegate that uses MaterialX needs to build a custom plugin to convert MaterialX Network to HdMaterial Network, By refactoring the UsdMtlx code, we will be able to reuse this for Custom Scene Delegates. We will also benefit from improvements suggested earlier.

Miscellaneous items (for further discussion)

- For multi-backend support for Vulkan, MDL, Metal, etc applications need to maintain multiple Shader Gen paths that can easily get out of sync and are hard to maintain.
- MaterialX nodes as ground truth definitions of UsdPreviewSurface.
- UsdLux v.s MaterialX Light nodes
- Customize shaders for selection, highlighting.
- For Hardware renderers customize shader stages, render state management

3. Knowledge Sharing

Goal: To facilitate sharing of knowledge / examples around consumption and rendering use cases.
This includes

- ensuring that the documentation is up-to-date with a given version of USD and MaterialX.
- a robust set of examples of types / configurations of shaders / materials, (in USD format, in MTLX format).
- examples of custom shader generation integration for different renderers.

Areas for Review

What is the desired process to keep USD up to date with a given version of MaterialX.

Property	Description	USD level	USD /MTLX level	MTLX level	Status	Proposal(s)
API change handling	Some versions break API compatibility.		Y	Y	Need to discuss this. Jonathan Stone noted future versions need to be more careful about possible compatibility breaking changes (API, definition etc).	• Individual fixes as they come (e.g. PR-1633)
Compile time version	Allow compile time version exposure.			Y		• PR to know version of MTLX at compile time PR-704

Property	Description	USD level	USD /MTLX level	MTLX level	Status	Proposal(s)
namespace	Helpful to qualify definitions with a given scope to avoid name clashes. e.g. there may be a Adobe vs ILM namespace.				Issue 1614 logged	• Encode namespace as part of definition identifier (no consensus) (PR 1631)
colorspace	Color management tagging for inputs as well as color management system specification.	Y			• Agreement to support as formal property. Needs scoping and definition. • Issue 1532 logged for MTLX export. Internal issue: USD-6703 (old - pre-agreement)	• "Follow what is going on with MaterialX". (vague)
ui value properties	Hints for UI. e.g. ui min, max, step etc.	Y			Separate proposal for this.	

unit / dimension support	Support for a real world unit or dimension for an given shader input. The type of the unit may be dynamic / data driven.				Issue 1632 logged. Internal issue: USD-6928.	
tokens	Import and resolving of token names used for geometric and file identifiers. Can include consistency of token delimiters		Y			
fileprefx					Issue 974 logged. Admin: not closed	• Resolved in PR 977.
version	Definition versioning				tbd	

Property	Description	USD level	USD /MTLX level	MTLX level	Status	Proposal(s)
Support nodes with multiple outputs					issue 1581 logged. Internal issue: USD-6820	
definition information association	Import does not capture the correct association between a definition and a nodegraph sometimes.				Issue 1629 logged	• Specific code fix proposed for conversion to MaterialX
definition variations	Definitions may be specified in a variety of different ways. Not all appear to be consumable properly. Some deal with definition discovery.				issue 1636 logged	• Refactor import discovery logic. This includes being more aligned with MaterialX Version 1.38.x (PR 1641)
definition search path	Related to definition discovery (definition not found in path)				Issue 1586 logged. Internal issue USD-6941	Two different (independent?) PRs. Suggestion: Should generalize the proposal here. • consider USD search path env variable on read for mtlx (partial, no concensus) (PR 1628) • runtime vs compile time constants for pathing: (PR-1610)
load / reload	Robustness / repeatability of loading documents				• Issue 1504 logged. Internal issue: USD-6670 (data loss) • Issue 1502 logged. Internal issue: USD-6669 (node missing)	

Information Sharing

Issues Filtering

The "Areas for Review" is currently *hand-culled* from the current list of all issues with the keyword MaterialX. Not all are applicable.
Q: Iis there a better way to manage this such as by adding labels to allow for quick filtering of relevant issues.

https://github.com/PixarAnimationStudios/USD/issues?q=is%3Aissue+is%3Aopen+Materialx

https://github.com/PixarAnimationStudios/USD/pulls?q=is%3Apr+is%3Aopen+materialx

Documentation for Unsupported Features

Unsupported features documentation. There is already the intent to update this. (Current notes appear to be valid for MaterialX version 1.37)
https://graphics.pixar.com/usd/docs/api/usd_mtlx_page_front.html#usdMtlx_unsupported

Content Examples

- *WIP : Leaving a call out for USD example contributions.*

Characteristic	USD	MaterialX
Namespaces and Versioning		https://github.com/autodesk-forks/MaterialX/blob/adsk_contrib/dev/resources/Materials/TestSuite/adsklib/archviz/adsk_metal.mtlx
Tokens		https://github.com/autodesk-forks/MaterialX/blob/adsk_contrib/dev/resources/Materials/TestSuite/stdlib/texture/tokenGraph.mtlx
Polymorphic signature variations:		https://github.com/autodesk-forks/MaterialX/blob/adsk_contrib/dev/resources/Materials/TestSuite/pbrlib/surfaceshader/unlit_surfaceshader.mtlx
...		

Shading / Rendering Integration

- WIP: To decide what to add here. Leaving a call out for USD example contributions.

- want to include role of shaders and backends
- help to aid those that want to figure out how their renderer fits in
- custom shader generator integration examples would be useful here.

Issues

Issues found related to workflow of USDShade to MaterialX conversion. For code generation this includes the topic of how to get performant updates. The underlying assumption appears to be requirement to always perform conversion and to make this process performant, but alternative approaches such as keeping correspoding MTLX in memory might be a consideration.

Property	Description	USD level	USD /MTLX level	MTLX level	Status	Proposal(s)
Unique nodegraph creation	Duplicate graphs being created during conversion		Y		Issue 1659 logged	• Append a nodegraph suffix (PR-1658). Remark left: suggest to create a unique child name using MTLX utilities. (makeUniqueChildName).

Property	Topic	Description	USD level	USD /MTLX level	MTLX level	Status	Proposal(s)
selection hilighting	Display feature	HdStorm has no selection highlighting for geomteric primitives with MaterialX materials applied.		Y		Issue 1614 logged	• Modify GLSLFX shadergen to apply additional color overrides ? (PR-1647)
parameter update	Performance	Prevent compiles on parameter changes by changing GLSFX code generation to use templated code and parameter buffer updates.		Y	Y	USD Issue: MTX Issue: 710	• MaterialX input parameters are no longer hardcoded in the shader but set from the parameter buffer • Support float and integer types with sizes between 1 and 4. Make sure that all other types are still hardcoded, so that there are no gaps • USD: PR-1664 • ADSK MTLX PR: 1320.
Texture binding code injection	Binding code injection		Y				• PR-1634 to address non-bindless texture issue • Q: This addresses GLSLFX, but as MTLX code gen changes this may not be robust or is this a one-off ?
usdview slow on idel	Performance	Root cause not known yet.		Y		USD Issue: 1638	

Reference:

Metadata discussions

- Sdr can not differentiate between vector4 and color4 MaterialX inputs

https://groups.google.com/g/usd-interest/c/G8BSo8jzfyY/m/eCYFEWhrBQAJ

- Associate unit information on value types

https://groups.google.com/g/usd-interest/c/uqCk8_TQ5uI/m/AWQArwGmAwAJ

Hydra discussions

- Transport MaterialX using custom scene delegate

https://groups.google.com/g/usd-interest/c/tlID9vK210g/m/Kft8l3CeAgAJ

- Dirty bits for HdMaterial

https://groups.google.com/g/usd-interest/c/xytT2azlJec/m/22Tnw4yXAAAJ