Adopting Open Standards: Blender's Journey on Supporting OpenPBR

/* ACADEMY SOFTWARE FOUNDATION

open source days '26

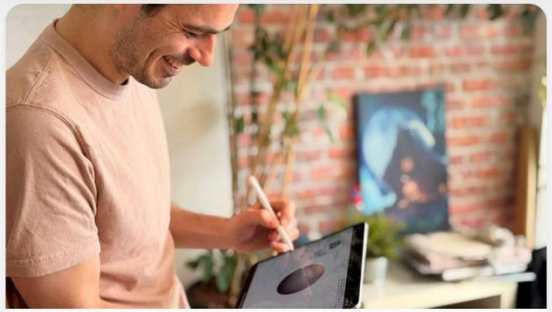
Sebastian Herholz,
Francesco Siddi


PUSH START
©2026 BLENDER

Quick updates

- Blender 5.2 LTS is out
- Blender Lab announced
- New Roadmap page


Blender Lab
Open Source Innovation

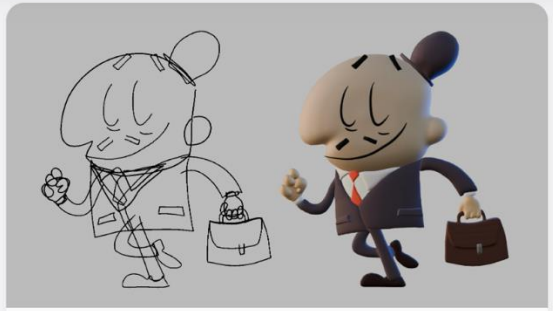


Beyond mouse and keyboard

Touch and Pen

Support and improve the user experience when using Blender on iPad, Android, and regular graphic tablets.

In Progress



2D + 3D Animation

Clay Pencil

Generate 3D animated meshes with 2D animation techniques.

In Progress

Rendering

Volume Rendering

Make unbiased volume rendering research practical for GPUs and production rendering.

In Progress

Rendering

Light Transport

Integrate and improve advanced light transport methods for production rendering.

Planned

Core

USD Authoring

Expand authoring capabilities and enable Blender to read and write USD files more natively and seamlessly.

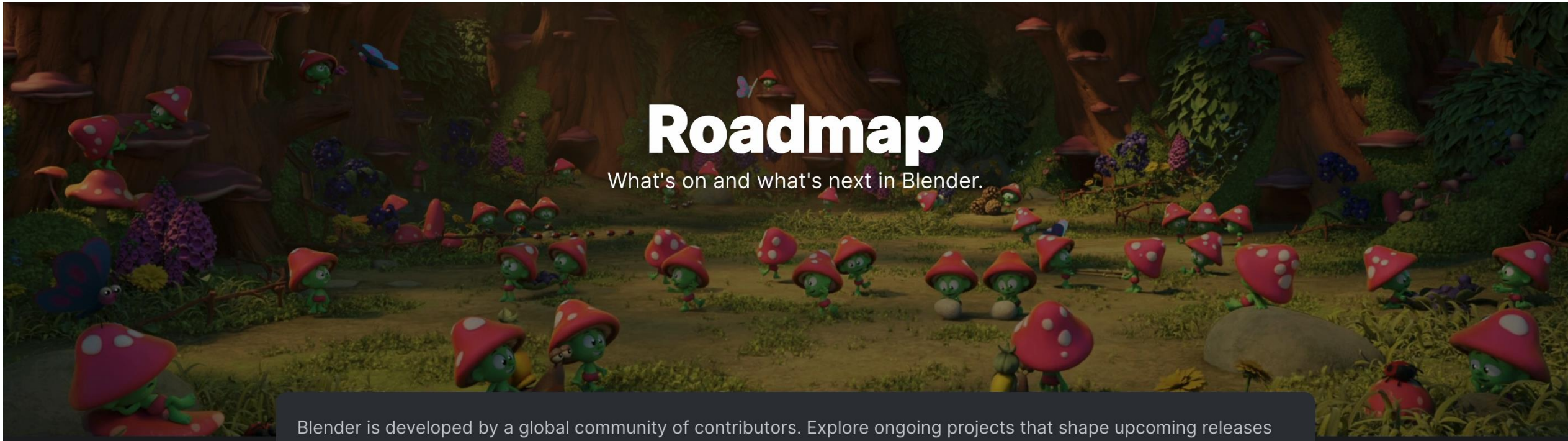
Requires Funding and Stakeholders

Beyond mouse and keyboard

VR & XR

Improve the user experience in VR and AR, adding features such as Location Scouting.

To be released in Blender 5.2 LTS



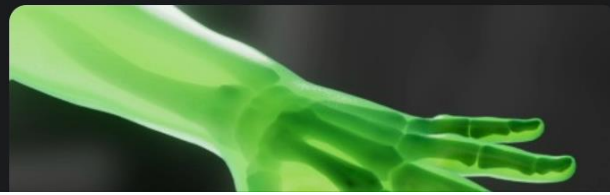
Roadmap

What's on and what's next in Blender.

Blender is developed by a global community of contributors. Explore ongoing projects that shape upcoming releases and get involved in work happening beyond the Roadmap at developer.blender.org.

Ongoing Projects



Policies & Guidelines

- Improved the Contributor (License) Agreement process
  - Clearer for individuals
  - Better structured for Companies
- Upcoming AI contribution policies
  - Public, community-led discussion, followed by core contributors input
- AI-enabled features design guidelines
  - tl;dr: Assistive, sovereign, IP concious


OVERGROWN
A feature film project by Blender Studio

Industry Adoption

- 3rd presentation at the Open Source Days
- Blender Conference showcases (HTTYD, IF, Sonic)

Bringing concepts to life - Framestore's Visual Development — Blender Conference 2025

59K views • 10 months ago

Blender

"Bringing concepts to life - **Framestore's** Visual Development" by Stefan Mayr, Juan Francisco Hernández Mendoza at **Blender** ...

VFX workflows for Sonic the Hedgehog 3 — Blender Conference 2025

35K views • 9 months ago

Blender

"VFX workflows for **Sonic** the Hedgehog 3" by Joel Prager at **Blender** Conference 2025 A walkth of how **Blender** was used ...

Industry Adoption

- 3rd presentation at the Open Source Days
- Blender Conference showcases (HTTYD, IF, Sonic)
- Gathered feedback while working on Blender License compliance guidelines
- Earlier this year, Netflix Animation Studios joined the Blender Development Fund!

Netflix Animation Studios

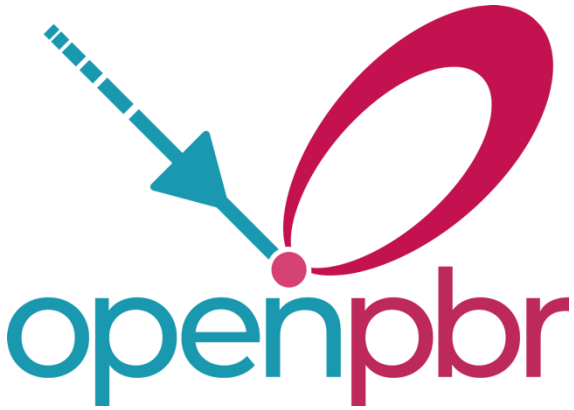
The Blender Development Fund Patron membership has enabled

- Conversations and feedback on studio-specific pipeline integrations (Blender Rez packaging, USD I/O)
- Story and previz artists to engage with the Blender project to improve story tools
- Work on industry standards integrations, such as OpenPBR, which will be presented by Sebastian shortly
- Setting up a corporate friendly framework to engage with a GPL-based software project


open
Source
days '26

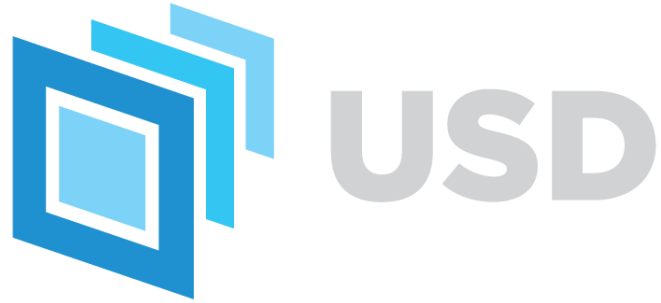
Open Standards

Open Standards for Assets



- Unified material and appearance model

- Material interexchange system



- Scene/asset representation
- Workflow

Blender + OpenPBR

OpenPBR: What it is and what is it for?

- Community driven material specification:
  - Initialized by Autodesk and Adobe
- PBR-based Über shader:
  - Goal: consistent look across different applications and renderers
- Ongoing project:
  - Current version: v1.1.1
- Supported by:
  - Arnold (Autodesk), Karma (SideFX), VRay (Chaos), Octane (OTOY), …
- Sources:
  - Current specification: https://academysoftwarefoundation.github.io/OpenPBR/
  - GitHub: https://github.com/AcademySoftwareFoundation/OpenPBR

openpbr

Specification v1.1, 2024-06-28. ASWF

This document is a specification of a surface shading model intended as a standard for computer graphics: the OpenPBR Surface model. Designed as an über-shader, it aims to be capable of accurately modeling the vast majority of CG materials used in practical visual effects and feature animation productions. The model has been developed as a synthesis of the Autodesk Standard Surface and the Adobe Standard Material models.

Shader Playground, rendered in Arnold for Maya, using OpenPBR Surface.

1 Historical background and objectives

Interchange of computer graphics scene assets remains a significant problem, especially with regard to surface appearance. Different renderers and 3D engines use different shading systems, shading languages, and fixed-function pipelines which are not easily interoperable. Furthermore, accurately modeling surface appearance remains a complex and pertinent problem and is a topic of active scientific research. Despite this, there is a strong need *today* for industries engaged in physically based rendering (both offline and real-time) to find a standardized material model that covers the most common use cases in typical workflows.

Over the years, certain de facto standard appearance definition frameworks have emerged as different vendors have come to the consensus of separating the definition of materials from the light transport simulation in the scene. Such general frameworks include MaterialX [Smythe2016], Open Shading Language (OSL) [Gritz2010], and Material Definition Language (MDL) [Kettner2015], which allow specifying the material as a combination of primitive surface reflectance models. These frameworks alone however are not sufficient for look development by end users, who should not be expected to build complex surface shaders from the ground up for routine tasks. Instead it is much more convenient in

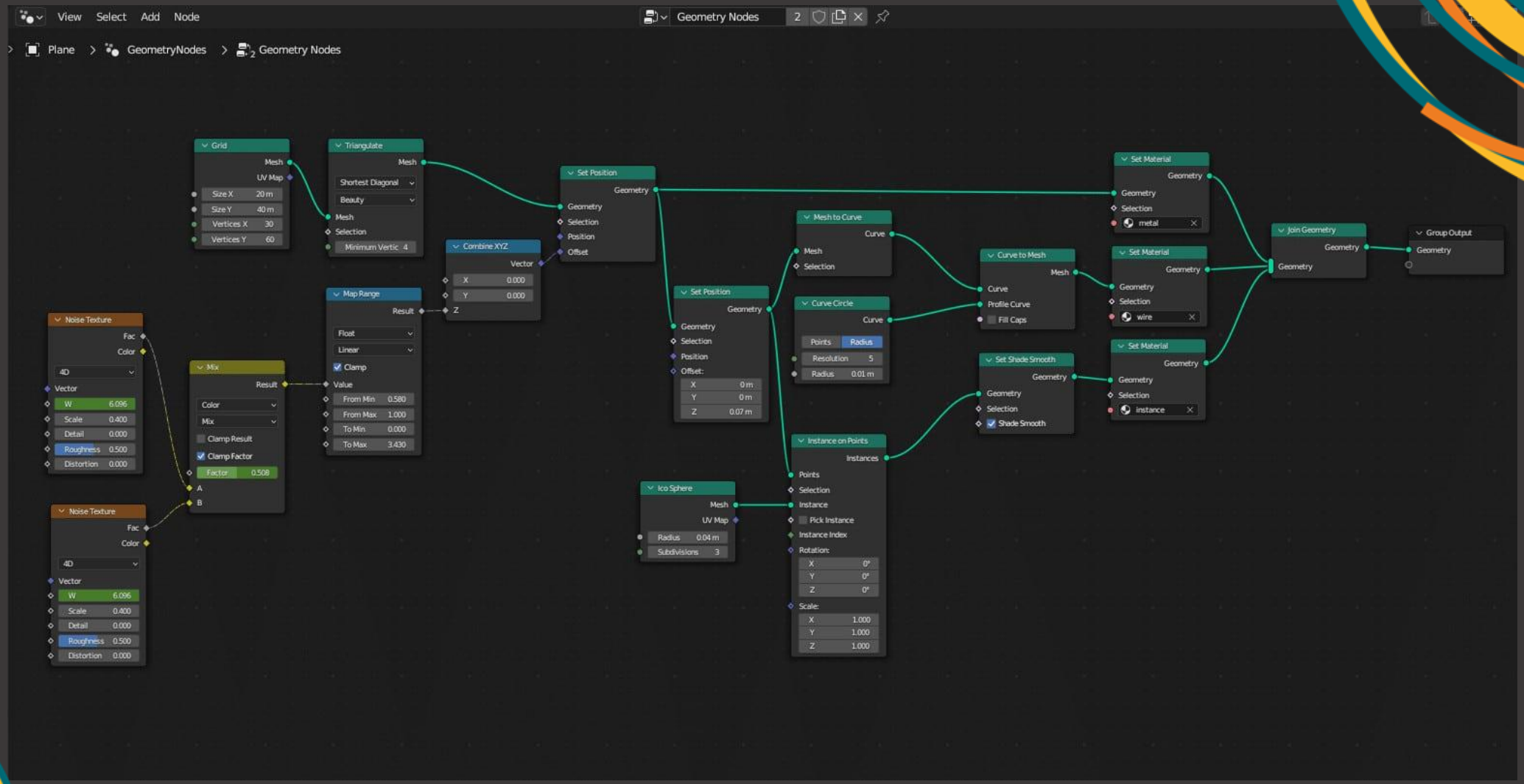
Cycles

Cycles

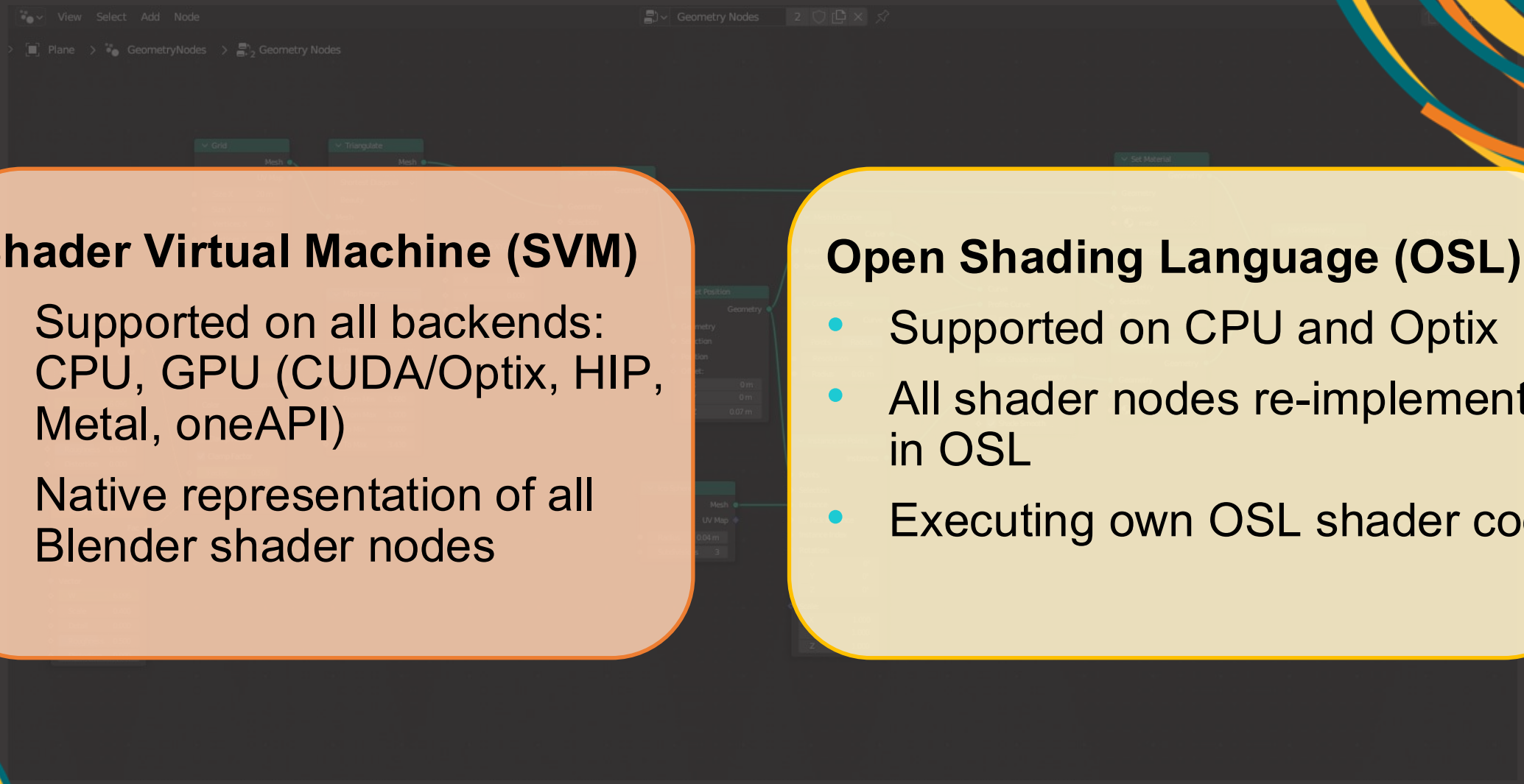



- Blender's in-house renderer
- Fully path tracing-based:
  - PBR and NPR rendering
- CPU and GPU support:
  - CPU: X86, ARM
  - GPU: Nvidia, AMD, Intel
- Wide user base and use case:
  - From hobbyist to professionals
  - Movies, arch-viz, animation, product-viz, …

Cycles: Material/Shading Backends



Cycles: Material/Shading Backends



Shader Virtual Machine (SVM)

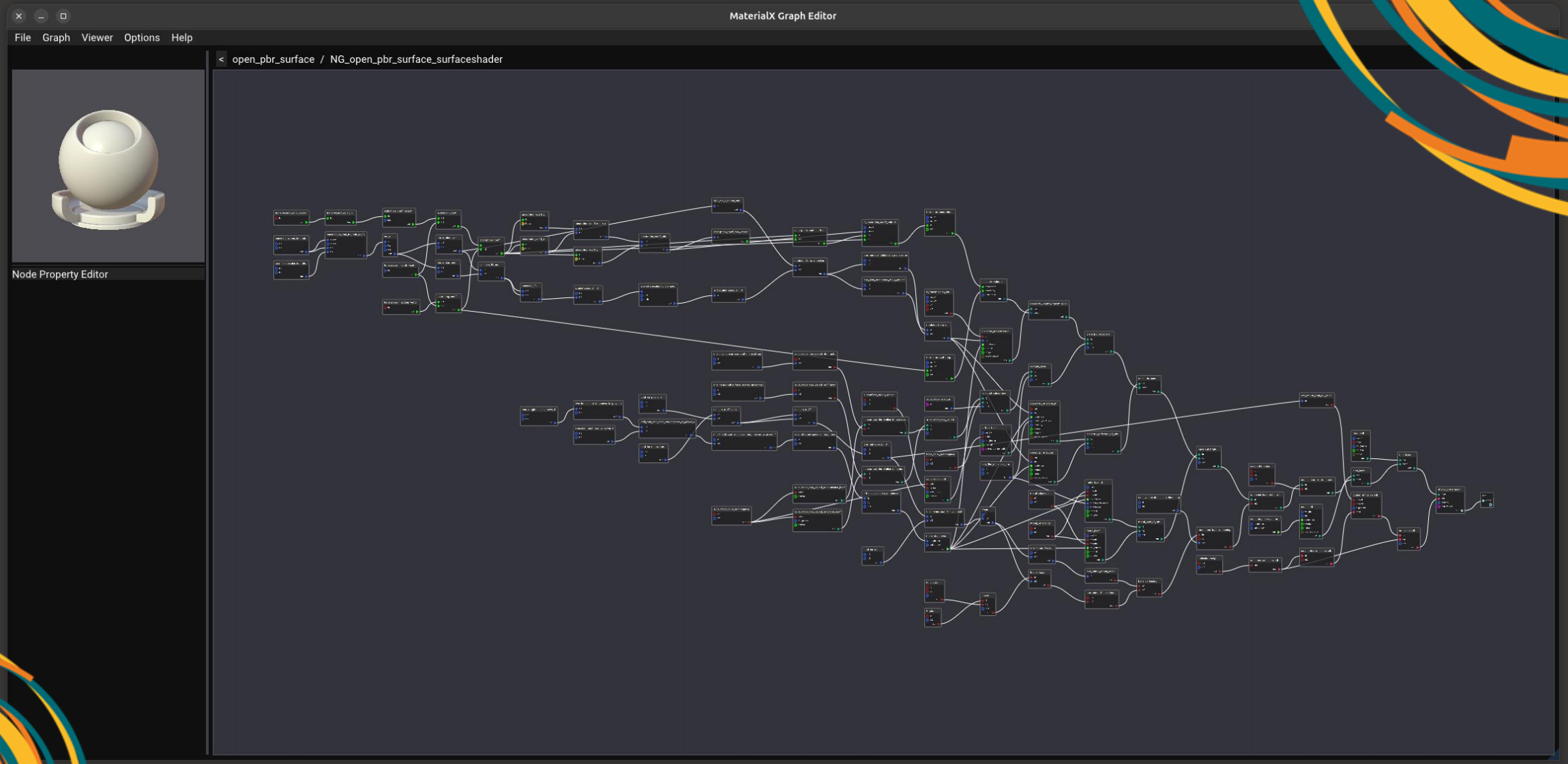
- Supported on all backends: CPU, GPU (CUDA/Optix, HIP, Metal, oneAPI)
- Native representation of all Blender shader nodes

Open Shading Language (OSL)

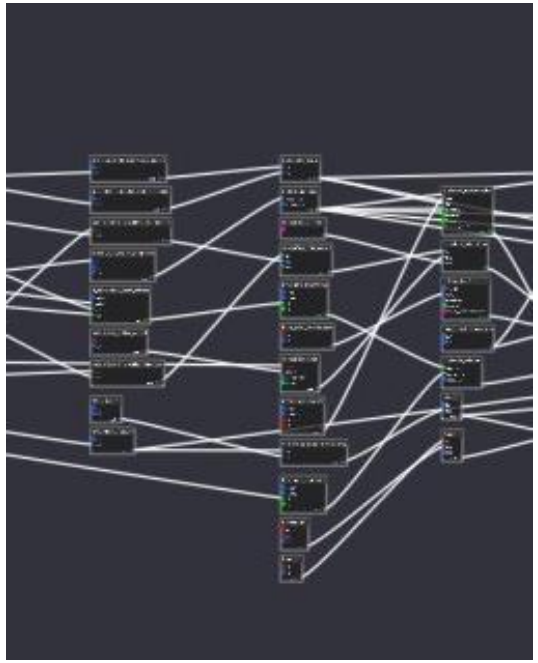
- Supported on CPU and Optix
- All shader nodes re-implement in OSL
- Executing own OSL shader code

Part 1: OpenPBR - OSL

MaterialX OpenPBR Shader Graph



MaterialX OSLCodeGen



MATERIALX

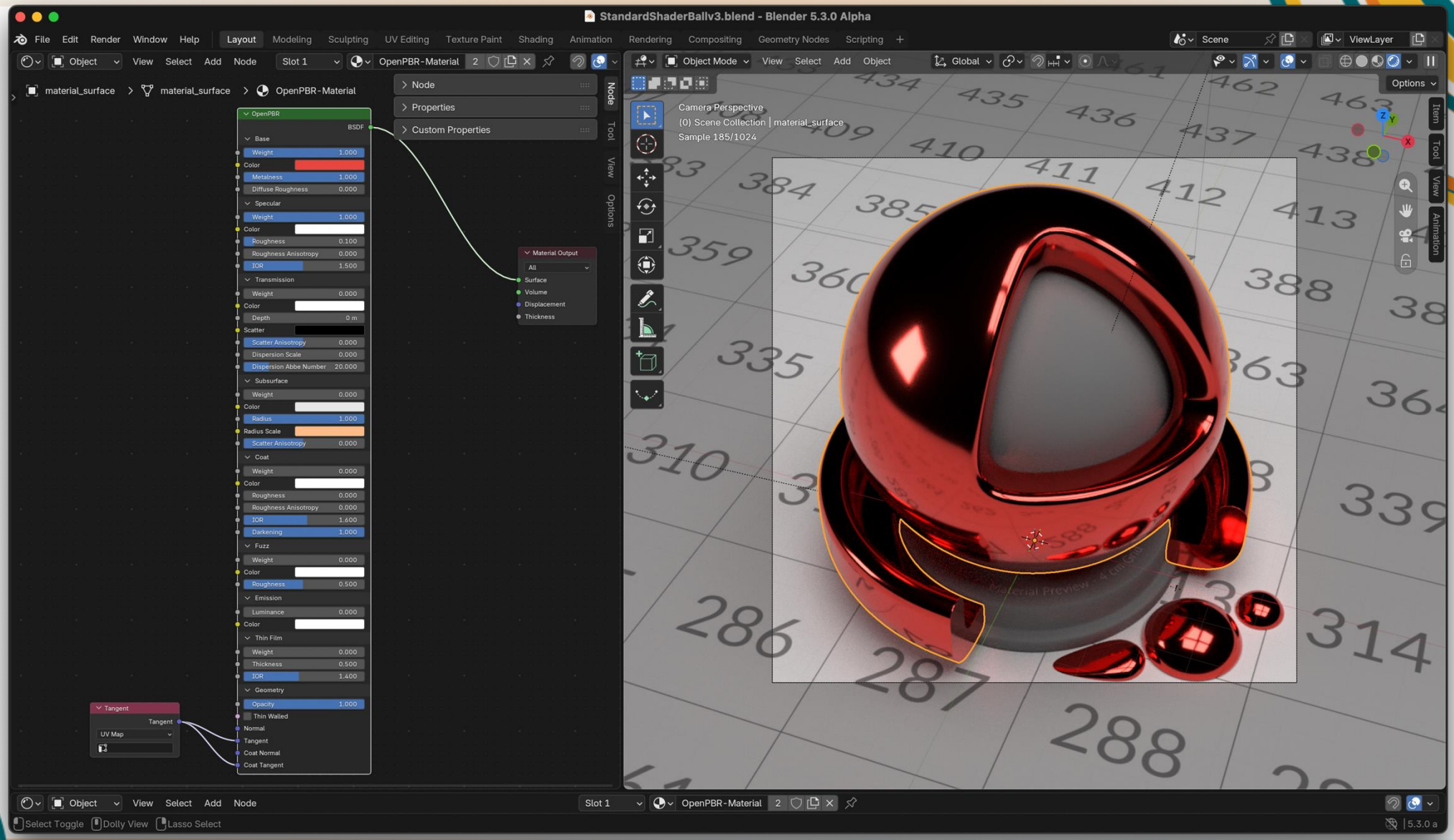
OSLCodeGen

```
586
587  BSDF_T fuzz_bsdf_out = fuzz_weight * sheen_bsdf(geometry_normal, fuzz_color, fuzz_roughness);
588  BSDF_T coat_bsdf_out = null_closure();
589  mx_dielectric_bsdf(coat_weight,
590                     color(1.000000, 1.000000, 1.000000),
591                     coat_ior,
592                     coat_roughness_vector_out,
593                     0.000000,
594                     1.500000,
595                     geometry_coat_normal,
596                     geometry_coat_tangent,
597                     "multi_ggx",
598                     "R",
599                     coat_bsdf_out);
600  BSDF_T metal_bsdf_tf_out = null_closure();
601  mx_generalized_schlick_bsdf(specular_weight,
602                              metal_reflectivity_out,
603                              metal_edgecolor_out,
604                              color(1.000000, 1.000000, 1.000000),
605                              5.000000,
606                              main_roughness_out,
607                              thin_film_thickness_nm_out,
608                              thin_film_ior,
609                              geometry_normal,
610                              geometry_tangent,
611                              "multi_ggx",
612                              "R",
613                              metal_bsdf_tf_out);
614
615  BSDF_T metal_bsdf_out = null_closure();
616  mx_generalized_schlick_bsdf(specular_weight,
617                              metal_reflectivity_out,
618                              metal_edgecolor_out,
619                              color(1.000000, 1.000000, 1.000000),
620                              5.000000,
621                              main_roughness_out,
622                              0.000000,
623                              1.500000,
624                              geometry_normal,
```

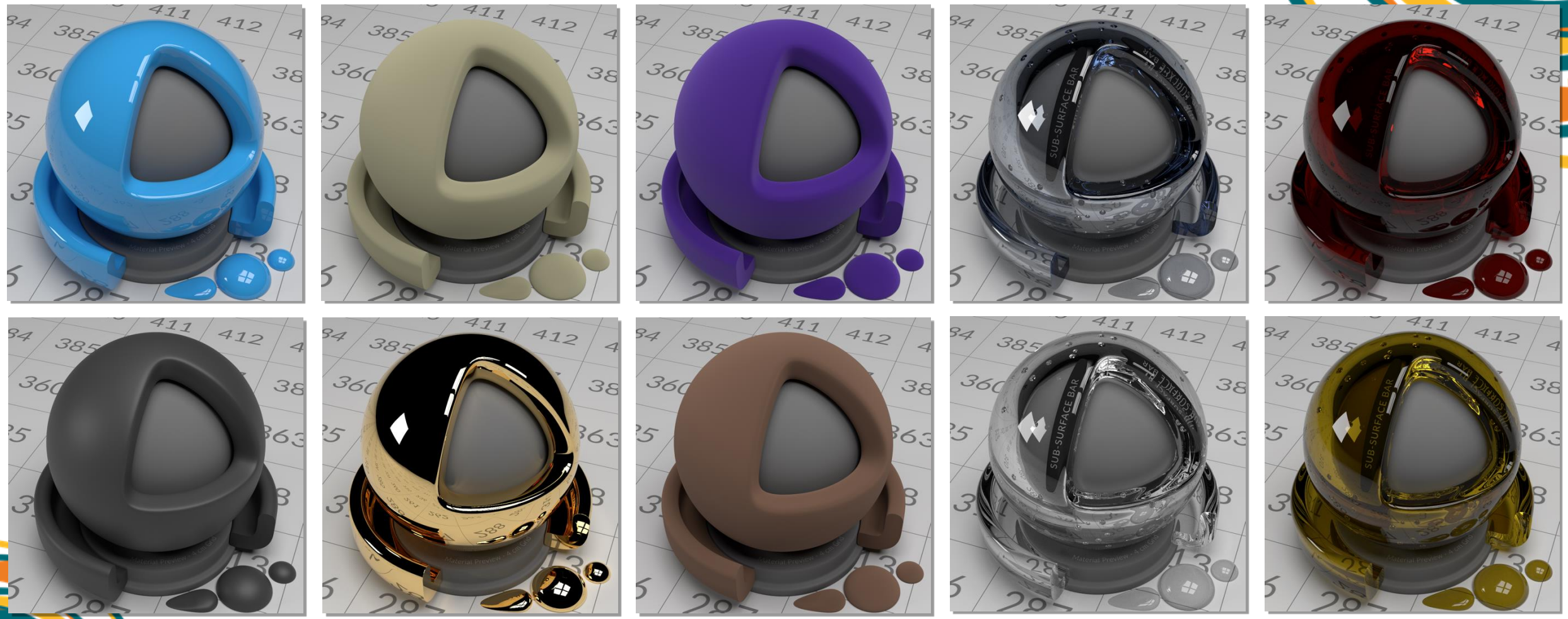
OSL

- Converting of the OpenPBR MaterialX graph to OSL using OSLCodeGen

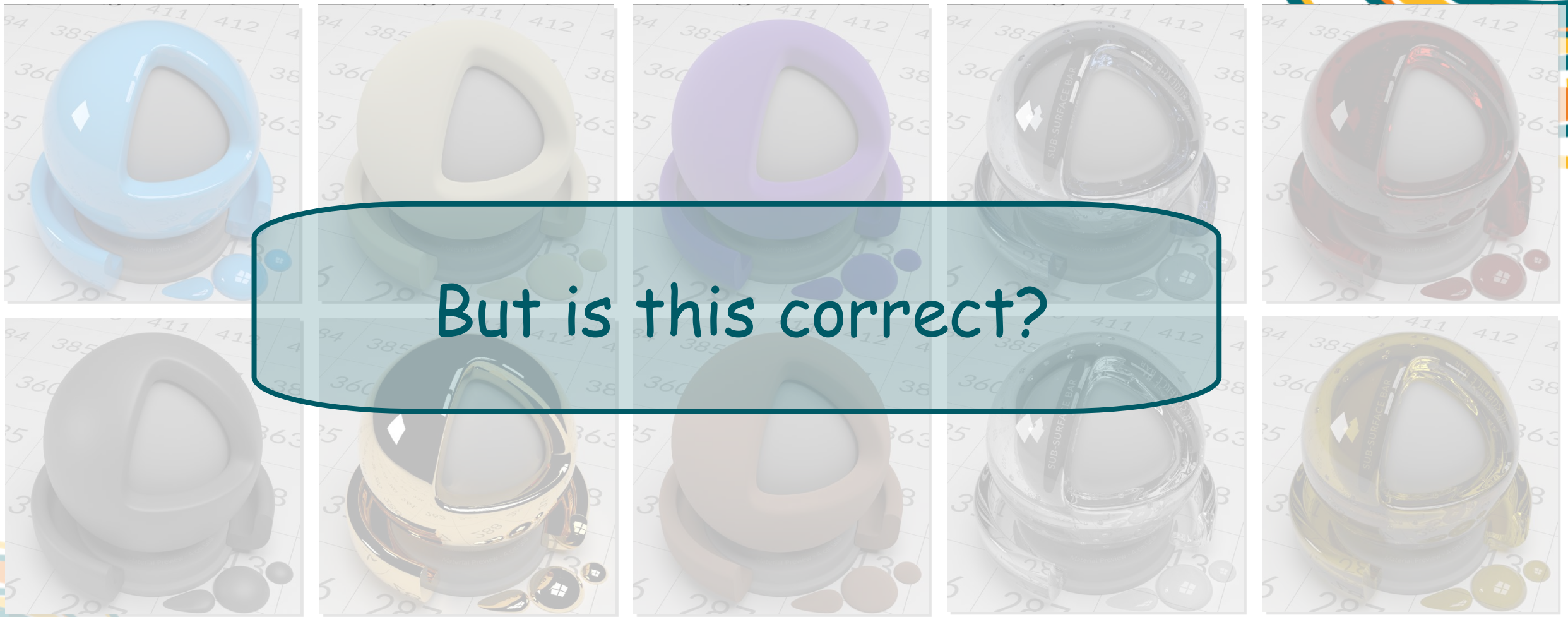
Initial OSL OpenPBR support



Initial OSL OpenPBR support



Initial OSL OpenPBR support



Part 2: OpenPBR - SVM

OpenPBR SVM integration

Clean-Room implementation:

- Re-implementing OpenPBR word-by-word according to the v1.1.1 spec

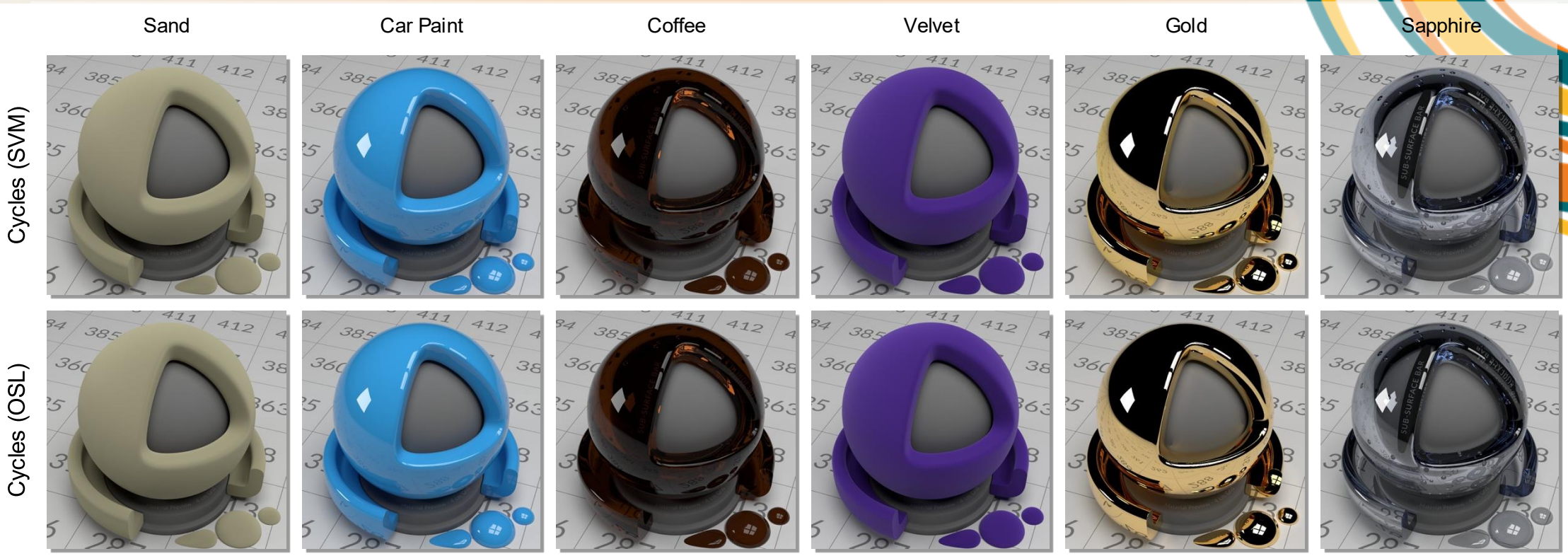
OpenPBR SVM integration

Weizhen Huang
(Blender)

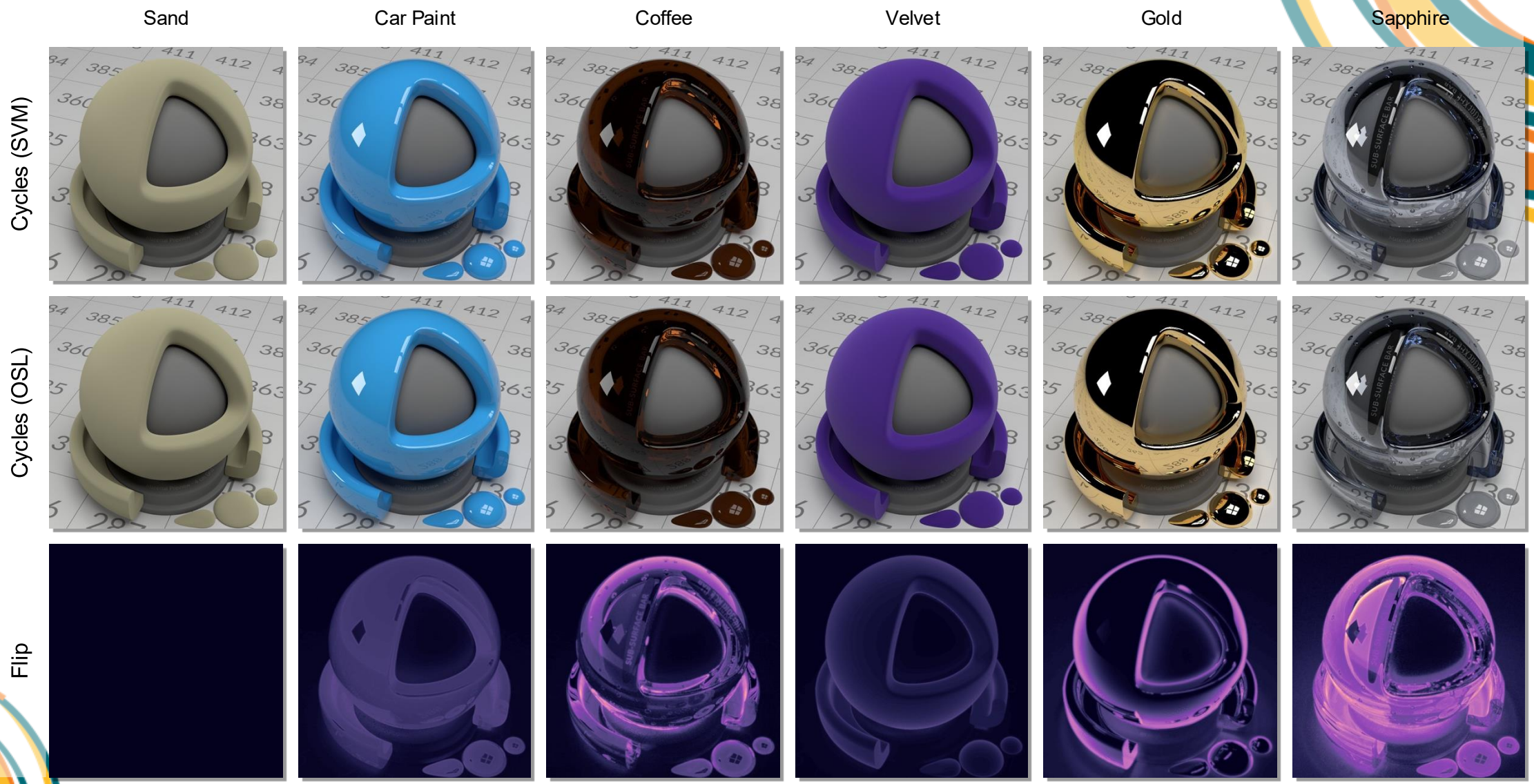
Clean-Room implementation:

- Re-implementing OpenPBR word-by-word according to the v1.1.1 spec

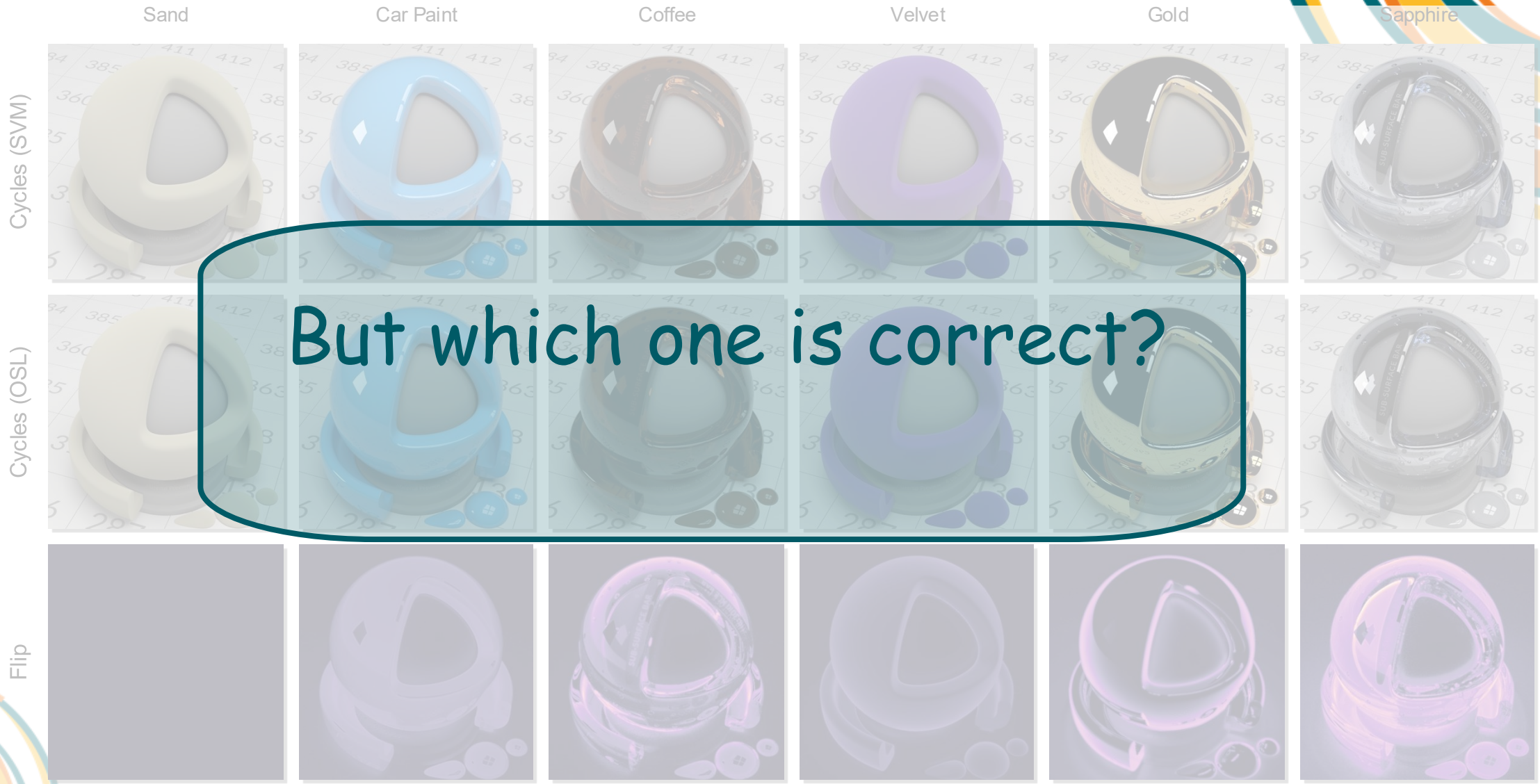
OpenPBR: SVM vs OSL



OpenPBR: SVM vs OSL

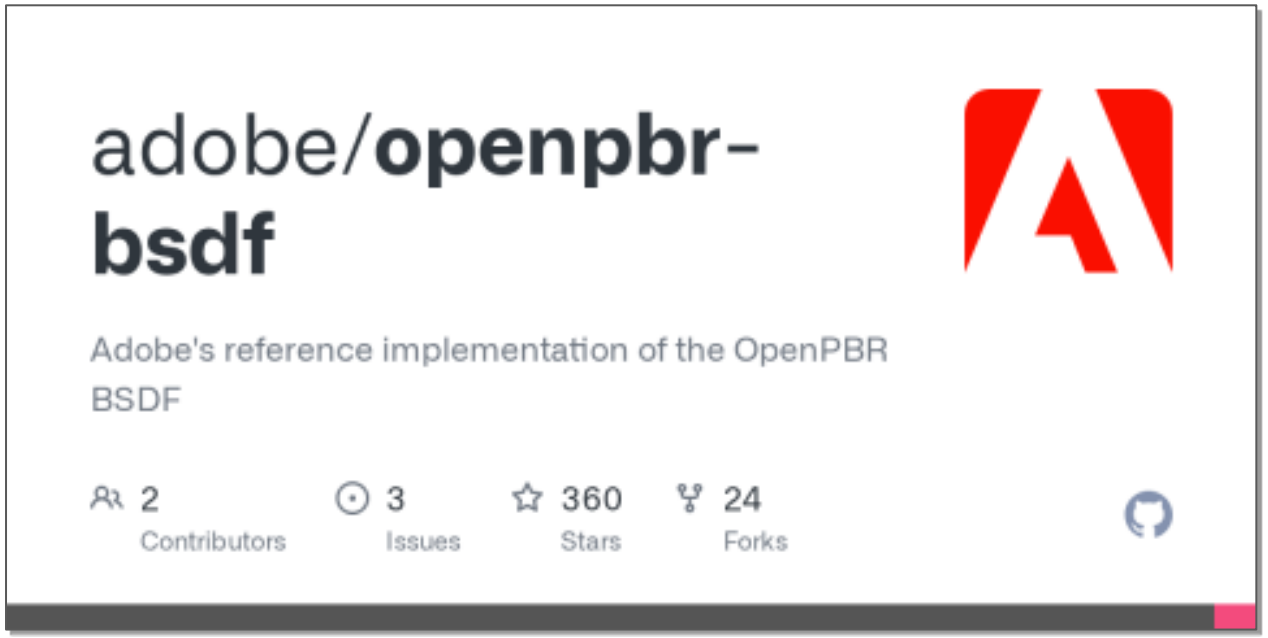


OpenPBR: SVM vs OSL



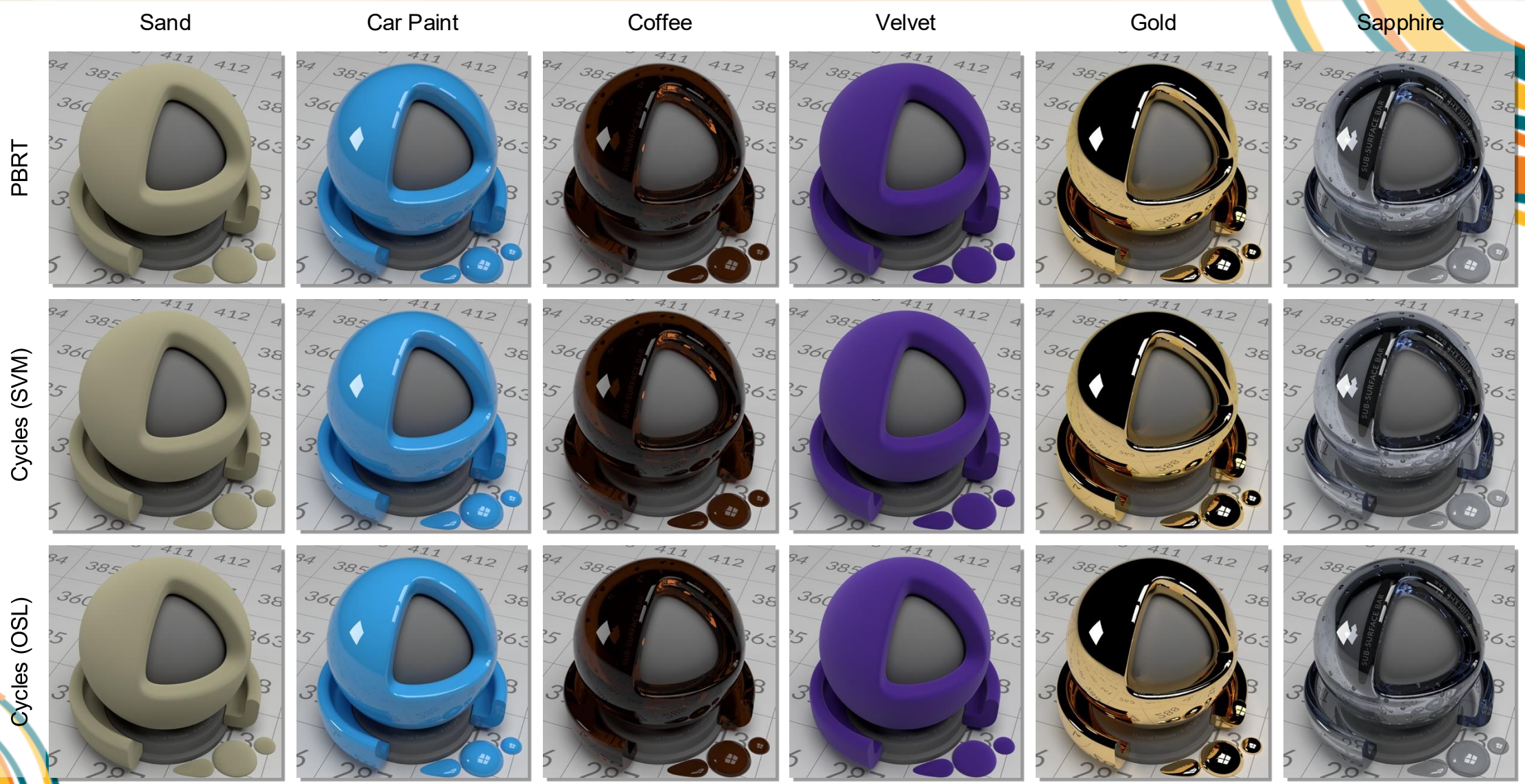
Part 3: OpenPBR - ???

Adobe’s OpenPBR library

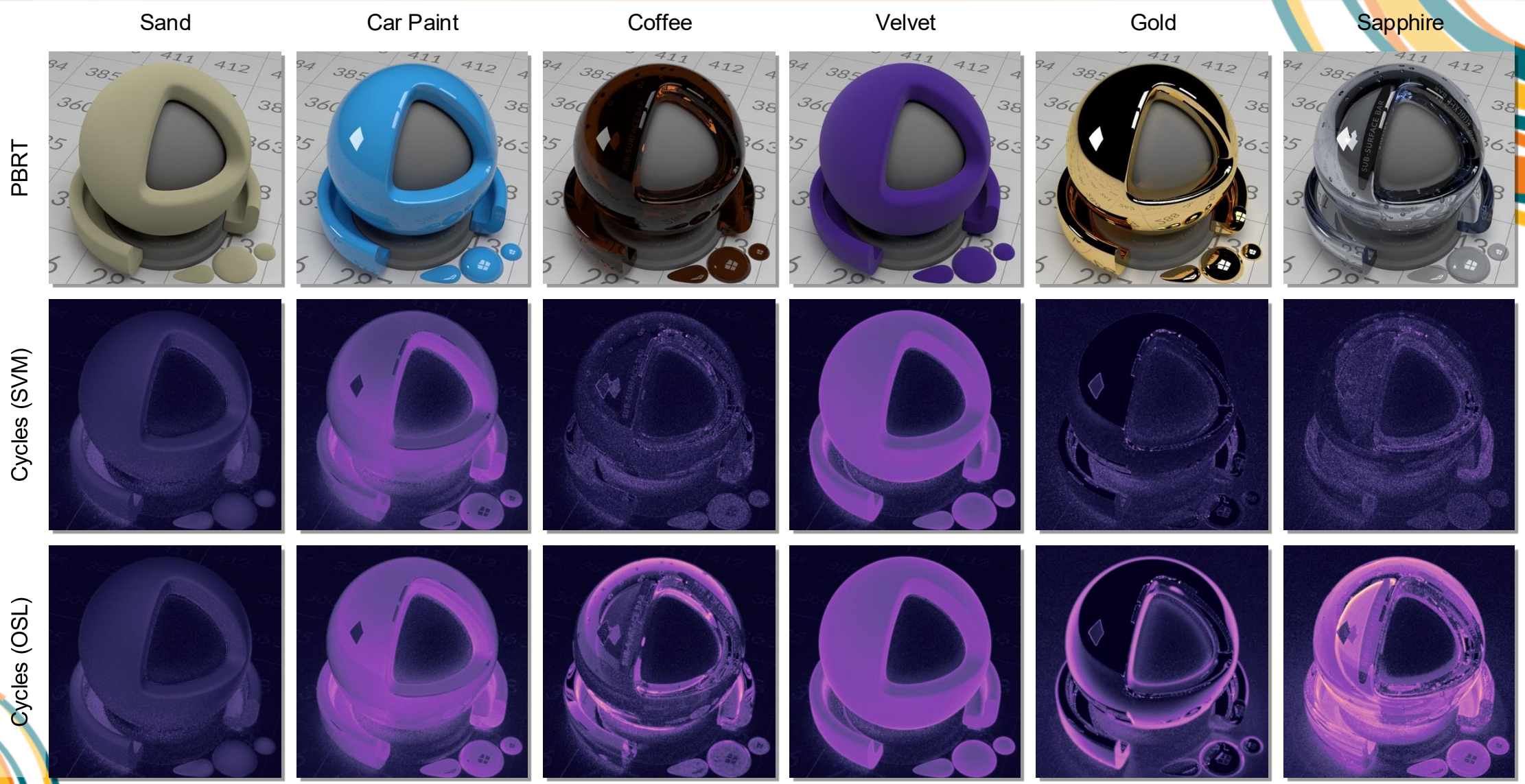


- Adobe’s OpenPBR reference implementation
- GitHub: https://github.com/adobe/openpbr-bsdf

OpenPBR: PBRT vs SVM vs OSL

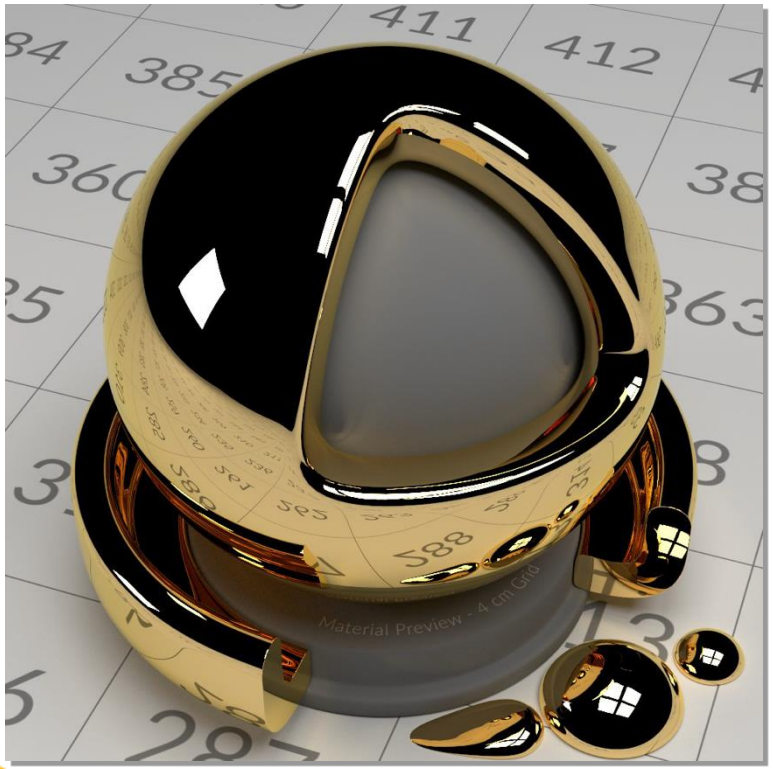


OpenPBR: PBRT vs SVM vs OSL

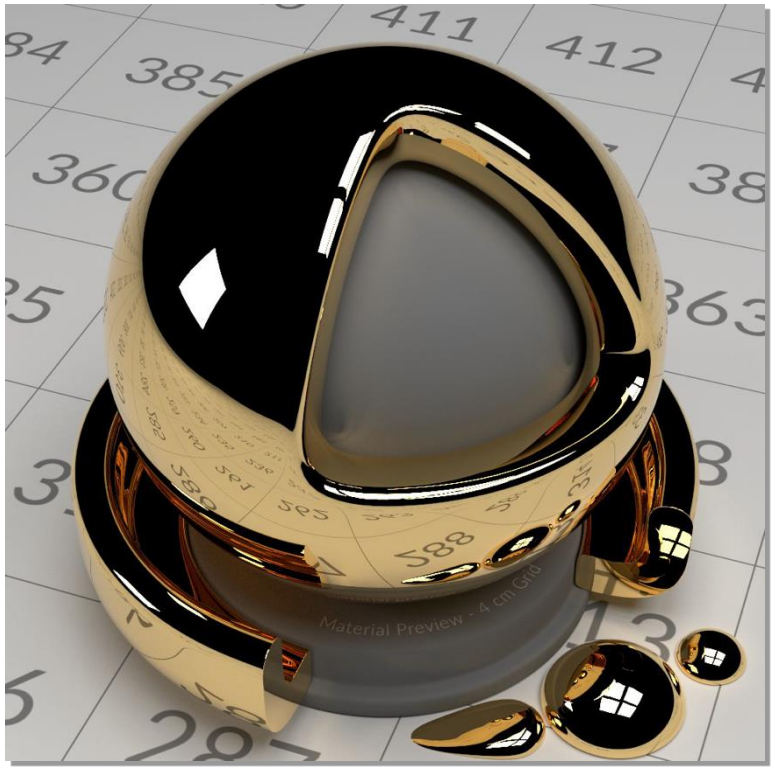


Examples: OSL (tint-82)

Cycles (SVM)



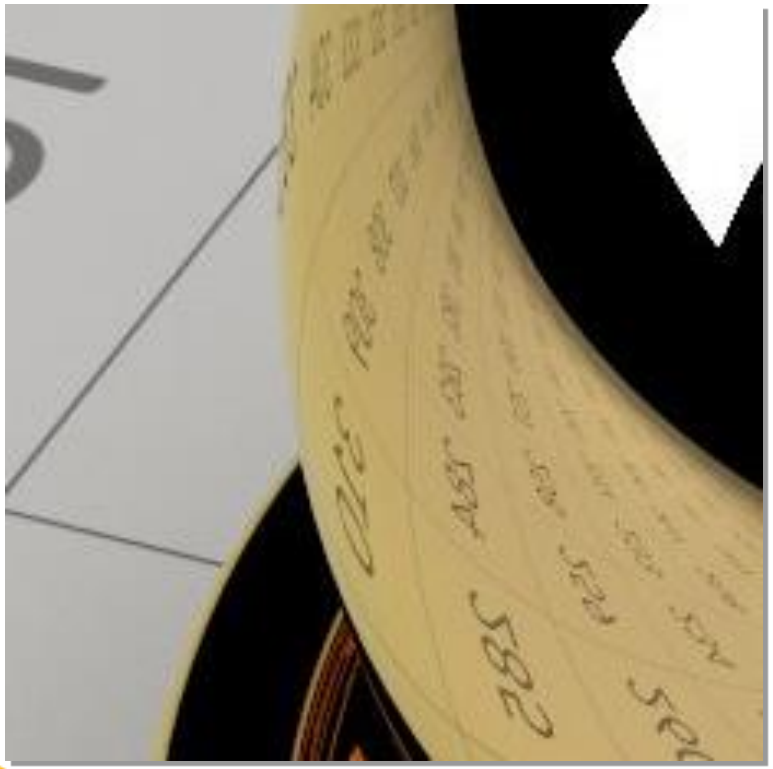
Cycles (OSL)



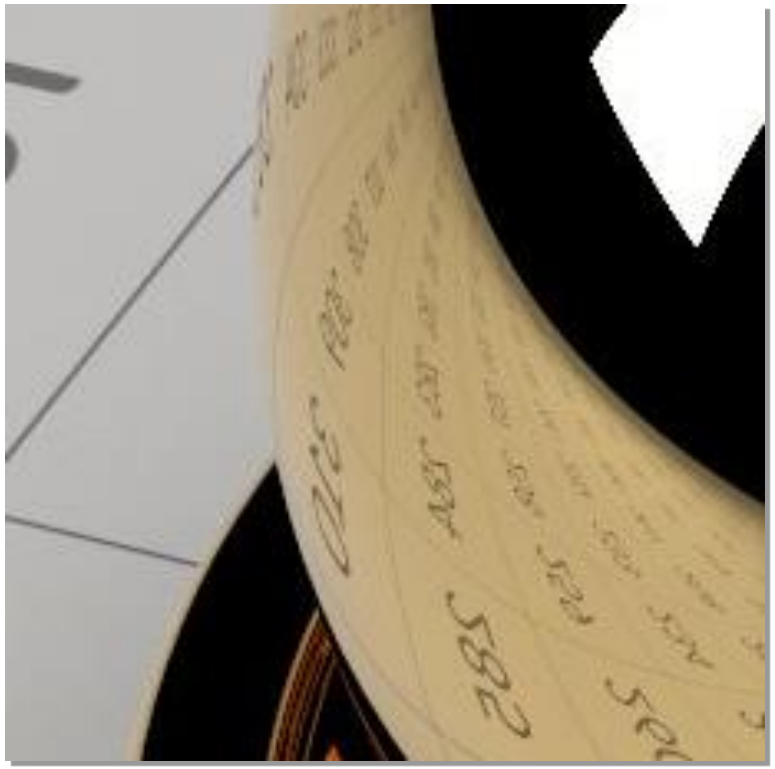
FLIP

Examples: OSL (tint-82)

Cycles (SVM)



Cycles (OSL)



FLIP

Examples: Layering

Cycles (SVM)

Cycles (OSL)

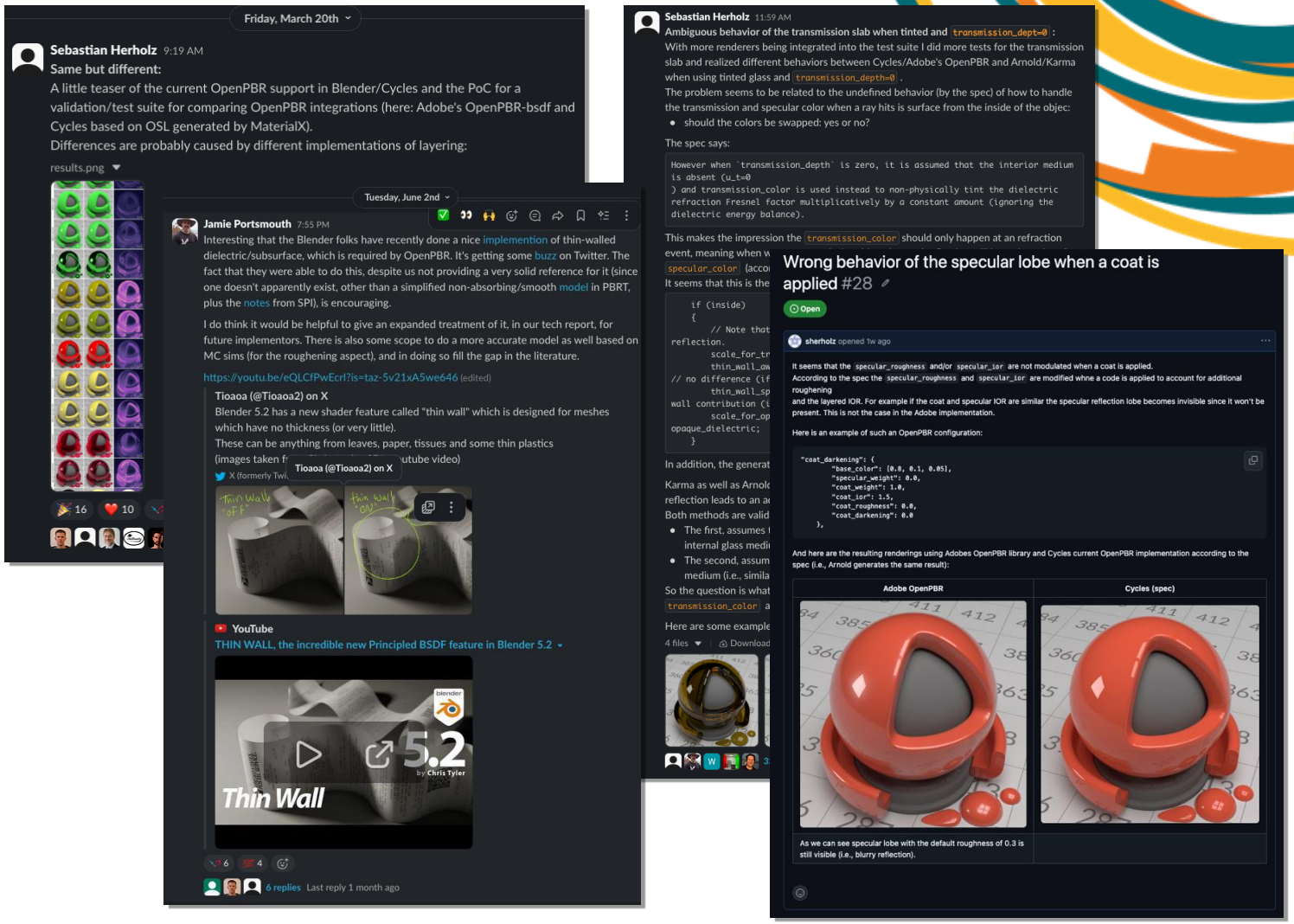
FLIP

Why do the results differ?

- Initial Findings:
  - OpenPBR specification:
    - Wording in the spec is sometimes ambiguous
    - Proposes multiple option of how to implement an effect:
      - E.g., layering, iridescence, coat-darkening
  - MaterialX - OSL:
    - OpenPBR shader graph does not fully cover the spec
    - OSLCodeGen cannot fully translate MaterialX to OSL closures (e.g., missing features or different definitions)

Interacting with the Community

- Community Discussions:
  - Slack: **#openpbr**
  - Meetings: bi-weekly (Tue. 19:00 CEST)



Interacting with the Community

- Community Discussions:
  - Slack: #openpbr
  - Meetings: bi-weekly (Tue. 19:00 CEST)

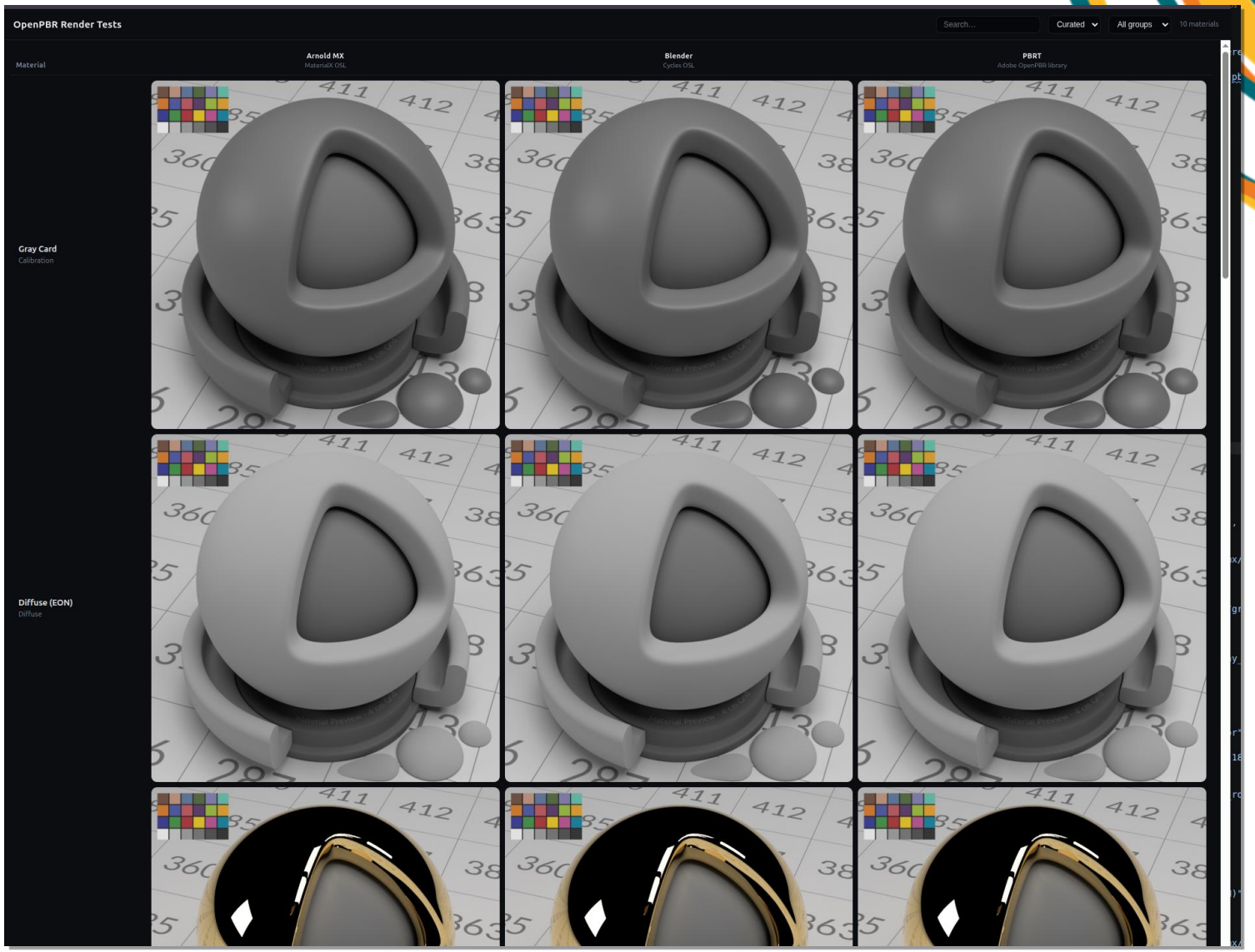
Agreement to work on fixing these problems

Validation pipeline (WIP)

- Collection of OpenPBR parameter sets
- Unified scene and light setup
- Carefully designed test and calibration setup
- Curated rendering scripts to replicate results
  - Easy to expand for new renderers
- Supported renderers:
  - Cycles, PBRT (Adobe's OpenPBR), Arnold, Karma, VRay, MoonRay (WIP), Glimpse (WIP)

Interactive Viewer/Database

Jamie Portsmouth
Autodesk (Arnold)



Take home message

- OpenPBR has the potential in ensuring consistent asset appearances across DCC applications and renderers
- Current implementations are already quite good but not perfect matches **yet**:
  - MaterialX-based and internal implementations
- Specification and MaterialX graph still require some improvements:
  - Community is willing to tackle open problems
- Rigorous validation and testing frameworks important
  - Ideally covering all parts of the stack (OSL, MaterialX, OpenPBR)

Take home message

- OpenPBR has the potential in ensuring consistent asset appearances across DCC applications and renderers
- Current implementations are already quite good but not perfect matches **yet**:
  - MaterialX-based and internal implementations
- Specification and MaterialX graph still require some improvements:
  - Community is willing to tackle open problems
- Rigorous validation and testing frameworks import
  - Ideally covering all parts of the stack (OSL, MaterialX, OpenPBR)

Join the community:

Together we can do it

Outlook: Blender and Open Standards

- OpenPBR is the first step:
  - MaterialX and better USD support are future plans
- Test package release:
  - Planned for September (2026)
- Active participation in upcoming OpenPBR versions:
  - Feature proposals
  - User community feedback

Barbershop 2016 (custom shaders)

Barbershop 2026 (OpenPBR WIP)

er By: Andy Goralczyk (Blender Studio)

/* ACADEMY SOFTWARE FOUNDATION

open source days '26