Custom OCIO Config

Using your own OCIO config

OCIO Roles

RenderMan relies on a few OpenColorIO roles that need to be defined in your **config.ocio** file.

- **rendering**: MANDATORY
  - This is the space in which all computations are done by the renderer.
- **data**: STANDARD
  - An input that doesn't need to be converted because it provides numerical data (normal map, bump map, roughness map, etc).
- **srgb_linear**: MANDATORY
  - A linear sRGB/Rec.709 image, like an HDRI environment map.
- **srgb_texture**:
  - A non-linear sRGB image, like a JPEG, PNG, etc.
    - This role has been added to our basic linear configuration and to the Filmic-Blender configuration.
    - In ACES-1.2, this is already an alias to **Utility - sRGB - Texture**.

In practice, these 4 roles will be what most people need to effectively use colour-management. Still it is possible to add more role through the JSON configuration file.

JSON Configuration file

Many configurations contains a large number of color spaces, for special cases that are not related to 3D rendering. The only things that matter to 3D artists boil down to a handful of controls.

In order to provide a central configuration point, RenderMan requires a JSON configuration file to provide:

- A list of Roles/ColorSpaces available in UI controls.
- A list of aliases (shortened forms) that will be inserted in texture names to disambiguate them.
- A list of help strings that can be displayed in the UI controls.
- A set of rules for the Texture Manager to assign initial txmake settings.

File name

The JSON file's name is based on the directory containing the `config.ocio` file:

```
$RMANTREE/lib/ocio/basic/config.ocio    rman_color_config_basic.json
$RMANTREE/lib/ocio/ACES-1.2/config.ocio    rman_color_config_ACES-1.2.json
```

File location

At startup, the files in `$RMANTREE/etc` will always be read.

If the `$RMAN_COLOR_CONFIG_DIR` environment variable is defined, all files matching the `rman_color_config_*.json` pattern in that directory will be loaded. This mechanism can be used to override stock configurations or add support for a custom OpenColorIO configuration.

Sample file

rman_color_config_ACES-1.2.json

```
{
    "help": [
        "This file helps RenderMan and its DCC plugins leverage arbitrary OCIO ",
        "configurations by providing:",
        "  - A subset of relevant Roles/ColorSpaces to be displayed in user ",
        "    interfaces.",
        "  - Their corresponding short forms (aliases) which will be inserted in",
        "    the texture names to disambiguate their contents.",
        "  - Conversion rules to allow the texture manager to assign reasonnable",
        "    defaults to new files.",
        "",
        "WARNING: the srgb_linear role must ALWAYS be defined",
        "",
        "This file MUST be named in a specific way:",
        "   rman_color_manager_CONFIGNAME.json",
```

```
        "   where CONFIGNAME is the name of the directory containing the config.ocio",
        "   file.",
        "This file can be located in:",
        "   $RMAN_COLOR_CONFIG_DIR",
        "   $RMANTREE/etc",
        "Rules are simple python strings that must eval to True or False.",
        "They can use the following substitution tokens:",
        "        'node_type':  PxrTexture, PxrDomeLight, etc",
        "   'classification':  bxdf, pattern, light, lightfilter, etc",
        "         'img_name':  The image name without its extension",
        "          'img_ext':  The image's extension",
        "        'img_atlas':  True if a texture atlas",
        "         'img_type':  'int' or 'float'",
        "        'img_depth':  8, 16 or 32",
        "        'img_nchan':  the number of channels in the image",
        "       'ocioconfig':  full path of OCIO config",
        "  'ocioconfig_name':  name of OCIO config: 'basic', 'ACES-1.2', etc"
    ],
    "ocio_aliases": {
        "rendering": "acescg",
        "srgb_texture": "srgbtex",
        "srgb_linear": "srgblin",
        "data": "data"
    },
    "ocio_aliases_help": {
        "rendering": "The image is already in the correct color space for rendering (no conversion).",
        "srgb_texture": "The image is a sRGB texture and needs to be converted to rendering space.",
        "srgb_linear": "The image is a sRGB/Rec.709 linear image (like an HDRI) and needs to be converted to
rendering space.",
        "data": "The image contains data that should be used as-is (no conversion)."
    },
    "conversion_rules": {
        "pattern": {
            "args": {
                "texture_type": "regular",
                "smode": "periodic",
                "tmode": "periodic",
                "texture_format": "openexr",
                "texture_filter": "catmull-rom",
                "resize": "round-",
                "data_type": "half",
                "compression": "pxr24",
                "compression_level": null,
                "ocioconvert": "srgb_texture"
            },
            "rules": {
                "'%(img_ext)s' in ('.exr', '.hdr')": {
                    "args": {
                        "texture_format": "openexr",
                        "data_type": null,
                        "ocioconvert": "srgb_linear",
                        "compression": null
                    }
                },
                "'%(node_type)s' in ('PxrBump', 'PxrNormalMap')": {
                    "args": {
                        "texture_format": "pixar",
                        "data_type": null,
                        "compression": "lossless",
                        "ocioconvert": "data"
                    }
                },
                "'%(node_type)s' == 'PxrBumpRoughness'": {
                    "args": {
                        "texture_format": "openexr",
                        "data_type": "half",
                        "texture_filter": "box",
                        "compression": "pxr24",
                        "mipfilter": "box",
                        "resize": "round-",
                        "ocioconvert": "data",
```

```
                        "bumprough": {
                            "factor": 2.0,
                            "normalmap": 0,
                            "invert": 0,
                            "invertU": 0,
                            "invertV": 0,
                            "refit": 1
                        }
                    }
                },
                "re.search(r'(roughness|rough|anisotropy|aniso|metallic|metalness|normal|bump|height|mask)', '%
(img_name)s', flags=re.IGNORECASE) is not None": {
                    "args": {
                        "texture_format": "pixar",
                        "data_type": null,
                        "compression": "lossless",
                        "ocioconvert": "data"
                    }
                },
                "'%(img_ext)s' is '.exr' and '%(img_name)s'.lower().endswith('_acescg')": {
                    "args": {
                        "ocioconvert": "",
                        "texture_format": "openexr",
                        "data_type": null,
                        "compression": null
                    }
                },
                "%(img_atlas)d == True": {
                    "args": {
                        "smode": "clamp",
                        "tmode": "clamp"
                    }
                }
            }
        },
        "light": {
            "args": {
                "texture_type": "regular",
                "smode": "black",
                "tmode": "black",
                "texture_format": "openexr",
                "texture_filter": "gaussian",
                "resize": "round-",
                "data_type": null,
                "compression": "pxr24",
                "compression_level": null,
                "ocioconvert": "srgb_linear"
            },
            "rules": {
                "'%(node_type)s' == 'PxrDomeLight'": {
                    "args": {
                        "texture_type": "envlatl",
                        "smode": "periodic",
                        "tmode": "clamp"
                    }
                }
            }
        },
        "lightfilter": {
            "args": {
                "texture_type": "regular",
                "smode": "black",
                "tmode": "black",
                "texture_format": "openexr",
                "texture_filter": "gaussian",
                "resize": "round-",
                "data_type": "half",
                "compression": "pxr24",
                "compression_level": null
            },
            "rules": {
```

```
                "'%(img_type)s' == 'int'":
                {
                    "arg": {
                        "ocioconvert": "srgb_texture"
                    }
                },
                "'%(img_type)s' == 'float'":
                {
                    "args": {
                        "ocioconvert": "srgb_linear"
                    }
                }
            }
        },
        "imageplane": {
            "args": {
                "texture_type": "regular",
                "smode": "black",
                "tmode": "black",
                "texture_format": "openexr",
                "texture_filter": "catmull-rom",
                "resize": "up",
                "data_type": null,
                "compression": "pxr24",
                "compression_level": null
            },
            "rules": {
                "'%(img_type)s' == 'int'":
                {
                    "arg": {
                        "ocioconvert": "srgb_texture"
                    }
                },
                "'%(img_type)s' == 'float'":
                {
                    "args": {
                        "ocioconvert": "srgb_linear"
                    }
                }
            }
        }
    }
}
```

Rules

Rules allow the Texture Manager to set reasonnable conversion defaults to newly referenced images. If done well, it fully automates texture conversion.

Rules are defined by node types: pattern, light, lightfilter and imageplane. This allows you to specialize conversion without relying on a complex set of rules.

args

This is a dictionnary of txmake flags with their value to define a baseline that can be modified by rules.

Rules with a null value are disabled.

texture_type	"regular" for generic textures or "envlatl" for environment light textures
texture_format	The file format of the texture: "pixar", "tiff", "openexr"
resize	Resize image to a power of 2 using given mode: "none", "up", "up-", "down", "down-", "round", "round-".
texture_filter	Filter used for resize: "point", "box", "triangle", "sinc", "gaussian", "gaussian-soft", "catmull-rom", "mitchell", "cubic", "lanczos", "bessel", "blackman-harris"
smode	Repeat mode in s direction: "black", "clamp", "periodic"
tmode	Repeat mode in t direction: "black", "clamp", "periodic"

data_type	"float" (float 32), "half" (float 16), "byte" (int 8), "short" (int 16)
compression	Compression used on texture tiles to save disk space. Texture fomat dependend: see txmake help.
compression_level	Compression level for compressors supporting that parameter: see txmake help.
ociocolorspace	The destination colorspace, that is the colorspace of the texture
ocioconvert	The source colorspace, that is the colorspace of the image
ociodither	Optionaly dither pixel values when outputing to a non-float texture format. Not recommended.
bumprough	A dict containing the bump to roughness conversion values: "factor", "normalmap", "invert", "invertU", "invertV", "refit"
mipfilter	The resizing filter used to create MIP levels. Same as texture_filter.

rules

Rules are Python strings that should evaluate to True or False.

All rules are evaluated in the file's order to modify the args dict. You can add a **`"break": true`** entry in your rule if you want to skip the following rules when the rule is True.

They can the following contain substitution tokens:

node_type	PxrTexture, PxrDomeLight, etc
classification	bxdf, pattern, light, lightfilter, etc
img_name	The image name without its extension
img_ext	The image's extension
img_atlas	True if a texture atlas
img_type	'int' or 'float'
img_depth	8, 16 or 32
img_nchan	the number of channels in the image
ocioconfig	full path of OCIO config
ocioconfig_name	name of OCIO config: 'basic', 'ACES-1.2', etc