https://doi.org/10.2352/ISSN.2470-1173.2020.3.MOBMU-205

Using ACES Look Modification Transforms (LMTs) in a Visual Effects Production Environment – Part 1: Rec.709 Inputs and Outputs

Eberhard Hasche, Dominik Benning, Reiner Creutzburg
Technische Hochschule Brandenburg, Department of Informatics and Media, Magdeburger Str. 50, D-14770 Brandenburg, Germany

Email: hasche@th-brandenburg.de, benning@th-brandenburg.de, creutzburg@th-brandenburg.de

Abstract

Look Modification Transforms (LMTs) are a very powerful component of the Academy Color Encoding System (ACES) and offer extraordinary flexibility in ACES-based workflows. In ACES, the look of a project can be defined by a Look Transform, previously referred to as a Look Modification Transform, or LMT, applied before the Output Transform. The advantage of this approach is that technically the Look Transform should not have to change regardless of whether the Output Transform chosen is for [1].

The Academy Technical Bulletin TB-2014-10 describes the application of a Look Transform using the Academy Common LUT Format (CLF). Since CLF is not yet widely supported, any look applied using a LUT or grading operation, which spans an entire scene or show, and is used in series with per-shot adjustments can be considered to be a Look Transform. Therefore the import and export to other programs and color pipelines like 8 bit image processing programs, as well as video and film cameras, are of particular importance [1].

This paper aims to discuss the possibilities of importing and exporting Rec.709-coded imagery into the Foundry's Nuke 12 ACEScg workspace using the OCIO ACES1.1 configuration. Presented are the implemented Rec.709 transforms and possible alternatives. We also discuss the behavior of transforms concerning processing the transfer function (gamma) and color conversion of primaries and their complementary colors.

.

1. Aim and overview of tools and software

Developed by the Academy of Motion Picture Arts and Sciences, the Academy Color Encoding System (ACES) [2] addresses many changes in the traditional color workflow. The new challenges are a linear scene-referred workflow, wide color gamuts, and the usage of High Dynamic Imagery.

The system targets a dark viewing environment with a peak white luminance of only 48 cd/m^2 (nits). This dark environment requires an adjustment in the form of a transfer function (gamma), and a modification of the saturation, which can be very different from that of bright office environments and also broadcast mastering, which takes place in a *dim* environment.

Since the ACES system successfully addresses modern color workflows, it is also used by other industries, such as streaming services like Netflix [3]. But the game industry is also in the process of integrating this system into their applications:

> *Adoption of ACES has widened, reaching a broader range of applications such as video games, being integrated directly into realtime engines [4].*

These areas of application, which are broader than initially intended, require extended possibilities of import and export between programs with different color systems. A consistent workflow is an indispensable prerequisite for successfully communicating Look Modifications (LMTs). Technologies, which use LMT in- and export include matte painting and texturing with still images. Another widespread scope of application is the video production of any kind, be it the support of film shooting with action cameras, the use of witness cameras, and standalone video productions using modern film editing tools like The Foundry Nuke.

In this paper, we investigate the capabilities of the Rec.709 transforms available in the OpenColor IO System concerning the transfer functions (EOTFs) and the behavior of the Rec.709 primaries. For this purpose, we use *test image 1*, which consists of a greyscale gradient, the color bars Red, Green, Blue, and their complementary colors Yellow, Cyan, Magenta. The image is encoded with a Rec.709 transfer function and exported as an uncompressed 8 bit TARGA image. The export behavior of HDR content is also tested.

Rec.709 is a common abbreviation in the moving image sector for *Recommendation ITU-R BT.709-6 (06/2015) (Parameter values for the HDTV standards for production and international programme exchange) [5]*. In this specification, among other parameters, the transfer function, the white point, and the primaries are specified.

The Color Transform Language (CTL) [6] is used to calculate the transforms used in the ACES system. The scripts can be executed on the macOS platform using the terminal. The CTL serves here for the analysis of processes and as a reference.

The Foundry NukeX12.0v3, the central image processing program in the film industry, is used as the host program [7]. To apply the transforms available in ACES/CTL, OpenColorIO ACES1.1 was used, which is part of the resources of NukeX12.0v3

> *(OCIO) is a complete color management solution geared towards motion picture production with an emphasis on visual effects and computer animation. ... OCIO is compatible with the*

Academy Color Encoding Specification (ACES) and is LUT-format agnostic, supporting many popular formats [8].

This work aims to clarify unclear information in OCIO and to analyze the capabilities of the individual Rec.709 transforms in terms of transfer functions and color values of primaries in import and export. The Roundtrip *Rec.709 – Input - Rec.709 – Output* is also part of the investigations.

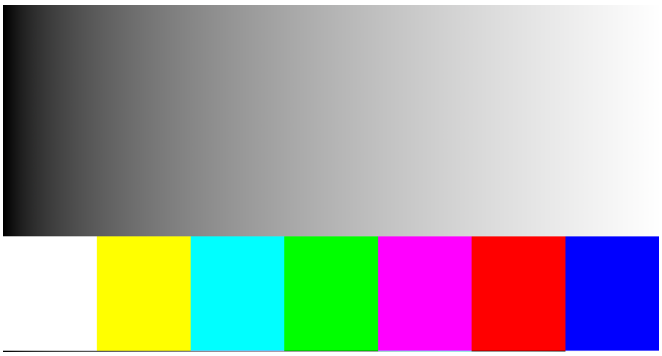


Figure 1. *Test Image (Targa) with grey scale ramp and color bars*

2. Rec.709 transforms in ACES 1.1 OpenColorIO configuration

In the input section of the OpenColorIO configuration (ACES 1.1) [9], which is used here, no Rec.709 transform is specified. Only in the Utility section and in the Output section Rec.709 transforms are indicated.

Utility:

- Curve – Rec.709
- Linear – Rec.709
- Rec.709 – Camera
- Rec.709 – Display

The OCIO ACES1.1 Readme document refers to the utility transforms as "a collection of colorspaces that are used to facilitate the creation of LUTs and other basic functionality"[10]. They are, therefore, not explicitly specified as input or output transforms.

The following transforms appear in the output area:

- Rec.709
- Rec.709 (D60 sim.)

The OCIO ACES1.1 Readme document describes the output transforms as "colorspaces and transforms implementing the ACES Output Transforms. These colorspaces produce code values ready for display on hardware devices calibrated to the standard used to name the colorspace".

As further sources, the transforms of the Color Transform Language are referred to, since these serve as the reference for the OCIO system. The following transforms are available in the Rec.709 section:

- ODT.Academy.Rec709_D60sim_100nits_dim.ctl
- ODT.Academy.Rec709_100nits_dim.ctl
- InvODT.Academy.Rec709_D60sim_100nits_dim.ctl
- InvODT.Academy.Rec709_100nits_dim.ctl

In this paper, we will not discuss the white point simulation (D60 sim.) and leave it for further pipeline investigation.

3. Utility – Curve – Rec.709 as an input transform

In this section, the input capabilities of the four Utility Transforms: *Curve - Rec.709, Rec.709 - Camera, Rec.709 – Display, and Linear - Rec.709* are examined.

3.1. The Utility – Curve – Rec.709 transform

When analyzing the OCIO transforms directly in the config file, only the *Utility – Curve – Rec.709* has a direction to the reference, which is ACES2065-1. It consists of ACES2065-1 primaries (P0), a D60 (ACES) white point, and a linear transfer function. The other three have an implementation that starts from the reference. First, the input capabilities of the *Utility – Curve – Rec.709* transform are examined. The OCIO ACES1.1 Readme document states that "the colorspaces starting with 'Curve -' apply a transfer function but do not convert between gamuts." [10]

```
- !<ColorSpace>
    name: Utility - Curve - Rec.709
    family: Utility
    equalitygroup: Curve - Rec.709
    bitdepth: 32f
    description: |
      The Curve - Rec.709 color space
    isdata: false
    allocation: uniform
    allocationvars: [0, 1]
         to_reference: !<FileTransform>
        {src:
            linear_to_rec709.spi1d,
            interpolation: linear,
            direction: inverse}
```

Code example 1. *OCIO-Utility-Curve-Rec.709 transform [9]*

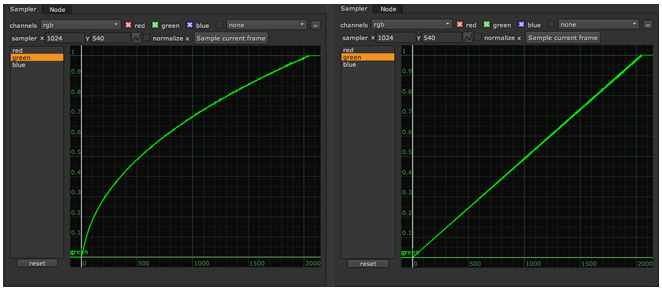


Figure 2*. Rec.709 - transfer function in original image and after transform (grey)*

A 1-D LUT *linear_to_rec709* is used, which applies a Rec.709 transfer function (EOTF) to the input image. When the direction is inverse, it removes it. As indicated in *code example 1*, it is only removing the Rec.709 transfer function but does not perform any primary or white point conversion.

Concerning the transfer function, it works as expected. But the problem is the missing primaries / white point conversion. When no color conversion is taking place, the color values of the input image are passed „as they are.“ That means that the Nuke workspace is ACEScg, but the internal OCIO input is *from reference* (ACES2065-1) (Primaries 0 = P0), so a transform from the OCIO reference (P0) to ACEScg (Primaries 1) is automatically taking place. The algorithm treats the incoming Rec.709 color values as ACES2065-1 values and performs the color conversion to ACEScg. Because the color gamut of ACES2065-1 is much wider than ACEScg, the color values are increased (see Table 1).

	R	G	B
white	1.00000	1.00000	1.00000
yellow	1.21493	1.09967	0.00228
cyan	-0.45144	1.07655	0.99168
green	-0.23651	1.17623	-0.00603
magenta	1.23651	-0.17623	1.00603
red	1.45144	-0.07655	0.00832
blue	-0.21493	-0.09968	0.99772

Table 1. *ACEScg color values after the Utility – Curve – Rec.709 input*

To avoid this and readjust the incoming color values to its original numbers, a transform from ACEScg to ACES2065-1 is applied. This returns the color bars to their initial values (Fig. 3).

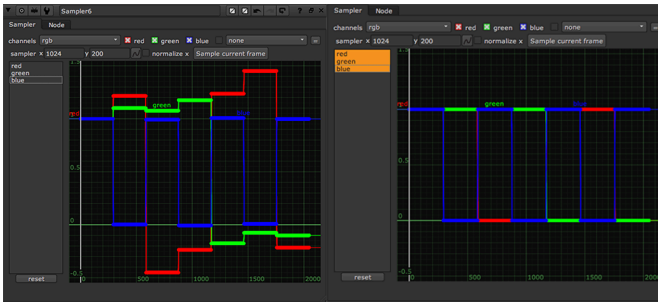


Figure 3*. Wrong and corrected values of the color bars after the input conversion*

But there is another problem that must be taken into account if the original colors are to be preserved. When the color values are passed without conversion, the numbers are interpreted as belonging to the ACEScg color space (see Tab. 2).

	x	y
white	0.32168	0.33767
yellow	0.45741	0.54346
cyan	0.14279	0.35809
green	0.1650	0.83000
magenta	0.38088	0.15164
red	0.71300	0.29300
blue	0.12800	0.04400

Table 2. *CIE xy chromaticity diagram values after the Utility – Curve – Rec.709 input (ACES-WP)*

If the values of the Rec.709 color bars are displayed in the CIE xy chromaticity diagram after the transform, it can be seen that the Rec.709 primaries have been moved to the ACEScg primaries. Therefore a further transform from ACEScg to Rec.709 is necessary if the color bars are to retain their original color (see 4. 2).

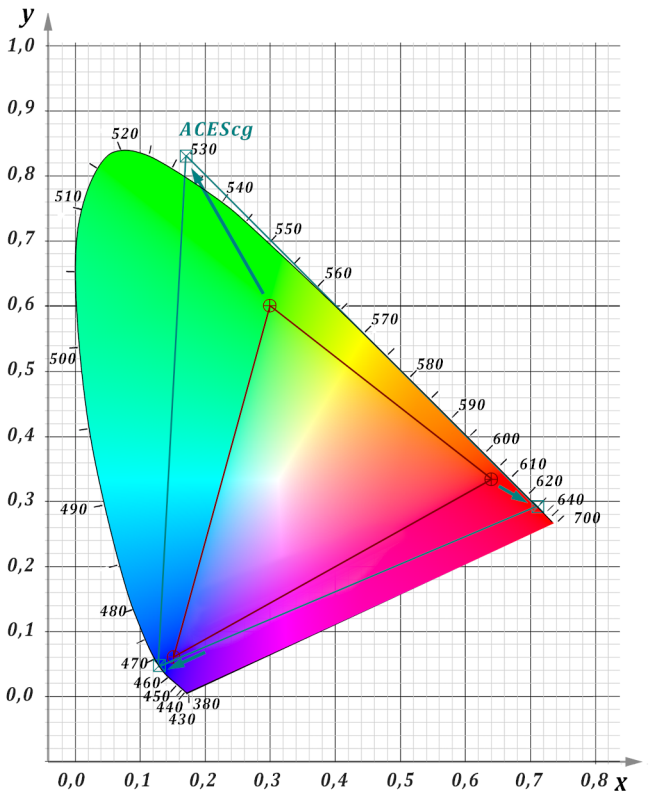


Figure 4*. Primaries of Rec.709 color bars are interpreted as ACESsg primaries*

3.2. The Utility – Camera – Rec.709 transform

The first transform in this section is *Utility – Rec.709 – Camera*. As given in code example 2, the transform direction is from the reference (ACES2065-1). However, due to the implementation in Nuke, it is executed inversely.

```
Utility – Rec.709 – Camera
    The Rec.709 – Camera color space
    from_reference:
         !<MatrixTransform> {matrix:
                 [0.952552, 0, 9.36786e-05, 0,
                  0.343966, 0.728166, -0.0721325, 0,
                  0, 0, 1.00883, 0,
                  0, 0, 0, 1]}
                  //ACES(P0) --> XYZ 60
         !<MatrixTransform> {matrix:
                 [3.2096, -1.55743, -0.495805,
                  0, -0.970989, 1.88517, 0.0394894,
                  0, 0.0597193, 0.210104, 1.14312,0,
                  0, 0, 0, 1]}
                  //XYZ 60 --> sRGB/Rec709
         !<FileTransform>
                 {src: linear_to_rec709.spi1d
```

Code example 2. *OCIO-Utility-Rec.709-Camera transform [9]*

First, the Rec.709 transfer function is removed. Then a transform is performed from the Rec.709 primaries / white point to CIE XYZ D60 (ACES) and from there to ACES2065-1 (ACES-P0) primaries. Nuke then automatically converts to the working color space (ACEScg). In the code examples, only the actual transforms are shown; the comments in all code examples were provided by the authors.

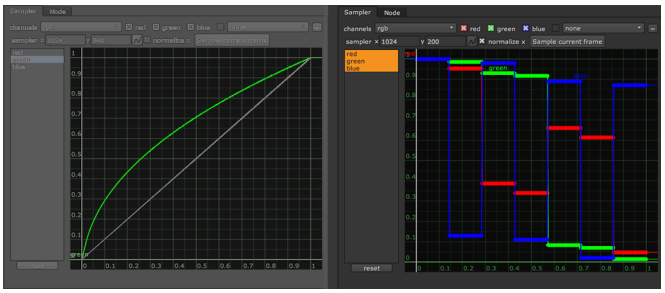


Figure 5. *Utility – Rec.709 – Camera transfer function: original image (green) and after transform (grey) – color values in ACEScg*

The name "Camera" for this transform was chosen because the transfer function is that of a video camera (Rec.709). The transform correctly removes the transfer function and makes the image linear. The color bars shown in the ACEScg color space do not reach the 1.0 positions, as these are reserved for the ACES primaries. Thus the values of the smaller Rec.709 color space are correctly transferred (Tab. 3).

	x	y
white	0.31271	0.32902
yellow	0.41931	0.50525
cyan	0.22465	0.32875
green	0.30000	0.60000
magenta	0.32091	0.15418
red	0.64000	0.33000
blue	0.15000	0.06000

Table 3. *CIE xy chromaticity diagram values after the Utility – Rec.709 – Camera input (D65)*

3.3. The Utility – Display – Rec.709 transform

The second transform in this section is *Utility – Rec.709 – Display*. The name "Display" for this transform was chosen because the transfer function is that of a broadcast monitor (Rec.1886). Here the conversion direction is also inverse.

```
Utility - Rec.709 - Display
   The Rec.709 - Display color space
   from_reference:
        !<MatrixTransform> {matrix:
                [0.952552, 0, 9.36786e-05, 0,
                 0.343966, 0.728166, -0.0721325, 0,
                 0, 0, 1.00883, 0,
                 0, 0, 0, 1]}
                 //ACES(P0) --> XYZ 60
        !<MatrixTransform> {matrix:
                [3.2096, -1.55743, -0.495805, 0,
                 -0.970989, 1.88517, 0.0394894,0,
                 0.0597193, 0.210104, 1.14312, 0,
                 0, 0, 0, 1]}
                 //XYZ 60 --> sRGB/Rec709
        !<FileTransform>
                {src: linear_to_rec1886.spi1d
```

Code example 3. *OCIO-Utility-Rec.709-Display- transform [9]*

The sub-transforms used here are the same as in *Utility – Rec.709 – Camera*. The only difference is that a Rec.1886 transfer function is removed, which is incorrect in this case. Therefore the resulting transfer function is not linear (Fig. 6). The color conversion is the same as in *Utility – Rec.709 – Camera* (Tab. 3).

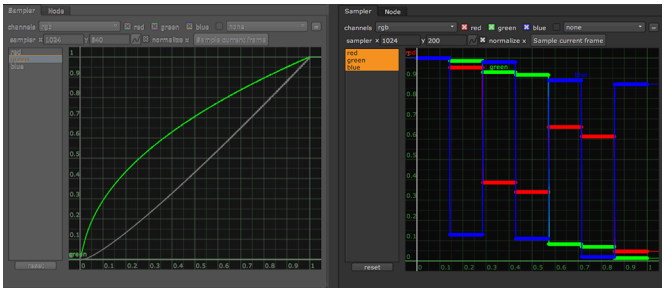


Figure 6. *Utility – Rec.709 – Display transfer function: original image (green) and after transform (grey) – color values in ACEScg*

3.4. The Utility – Rec.709 – linear transform

The last transform in this section is *Utility – Linear – Rec.709.* The colorspaces starting with 'Linear - ' convert to or from a specific gamut but do not apply a transfer function [10].

```
Utility - Linear - Rec.709
   The Linear - Rec.709 color space
   from_reference:
        !<MatrixTransform> {matrix:
                [0.952552, 0, 9.36786e-05, 0,
                 0.343966, 0.728166, -0.0721325, 0,
                 0, 0, 1.00883, 0,
                 0, 0, 0, 1]}
                 //ACES(P0) --> XYZ 60
        !<MatrixTransform> {matrix:
                [3.2096, -1.55743, -0.495805,0,
                -0.970989, 1.88517, 0.0394894, 0,
                 0.0597193, 0.210104, 1.14312, 0,
                 0, 0, 0, 1]}
                 //XYZ 60 --> sRGB/Rec709
```

Code-example 4. *OCIO-Utility-Linear-Rec.709 transform [x]*

Therefore only the transforms from Rec.709 to CIEXYZ (D60-ACES) and from there to ACES2065-1 (P0) are executed. Nuke then automatically converts to the working color space (ACEScg). The transfer function is retained. The color conversion is the same as in *Utility – Rec.709 – Camera* (Tab. 3).

In conclusion, except for the *Utility – Curve – Rec.709* transform (no primaries / white point conversion), all transforms are usable but hidden in the Utility transforms, which is not intuitive. Also, the wrong transform direction needs further implementation. The names “Camera” and “Display” are not recognizable at first glance. For the input of Rec.709 imagery into Nuke *Utility – Rec.709 – Camera* is the choice.

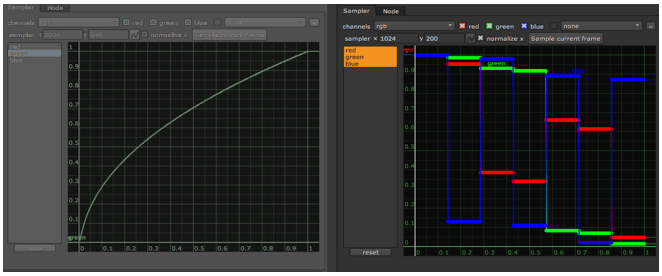


Figure 7. *Utility – Linear – Rec.709 transfer function: original image (green) and after transform (grey) – color values in ACEScg*

4. Creation of a customized Rec.709 – Input Transform

4.1. Aim of the transform

For artists and staff members who are not familiar with the color workflow, it is confusing if there is no Rec.709 transform in the input section. Also, the conversion direction is wrong for some Utility transforms. For this reason, it is useful to create a custom made Rec.709 and insert it into the OCIO configuration so that it appears in the Input section. This transform was first created in Nuke using the discrete compositing nodes and then it was transferred into the OCIO configuration.

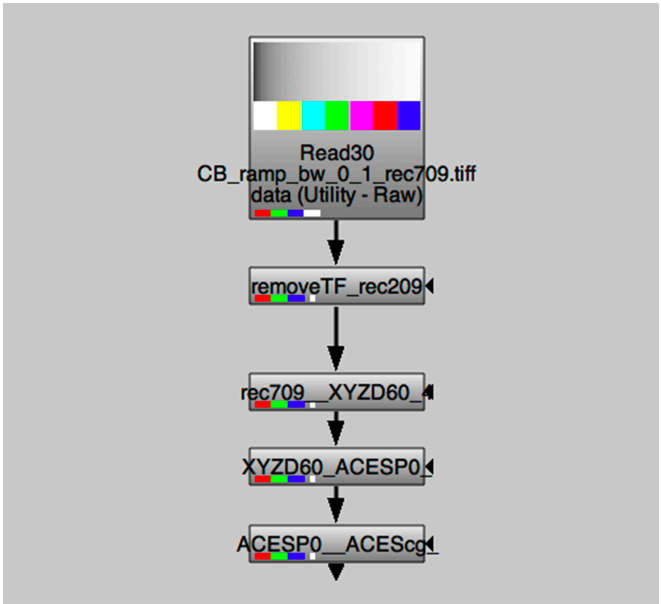


Figure 8. *Nuke nodes for the custom Rec.709 input transform*

The color-scientifically correct order of the individual transforms is as follows:

1) Remove TF Rec.709 transfer function
2) Transform from Rec.709 (D65) to XYZ(D60)
3) Transform from XYZ(D60) to ACES2065-1
4) Transform from ACES2065-1 to ACEScg

The last-mentioned process takes place automatically in an OCIO transform. In case of a setup with Nuke nodes it still has to be inserted.

4.2. Creation of the transform

Fig. 8 shows the corresponding Nuke nodes, including the internal transform from ACES2065-1 to ACEScg.

Figure 9. *Colorspace node settings to remove the Rec.709 transfer function*

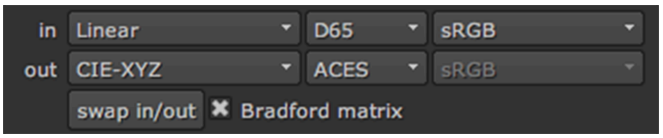


Figure 10. *Colorspace node settings for the transform from Rec.709 to CIEXYZ D60*

Fig. 9 shows the settings of the Nuke *Colorspace* node for removing the transfer function, Fig. 10 shows the settings of the transform from linear Rec.709 (D65) to XYZ(D60). Alternatively, the following color matrix (Formula 1) can be applied.

$$\begin{bmatrix} X_{D60} \\ Y_{D60} \\ Z_{D60} \end{bmatrix} = \begin{bmatrix} 0.41878837 & 0.36482045 & 0.16903733 \\ 0.21533319 & 0.71600521 & 0.06866148 \\ 0.01769946 & 0.11254156 & 0.87858415 \end{bmatrix} \cdot \begin{bmatrix} R_{Rec.709} \\ G_{Rec.709} \\ B_{Rec.709} \end{bmatrix}$$

Formula 1. *Color Matrix from Rec.709 to XYZ D60olor Matrix*

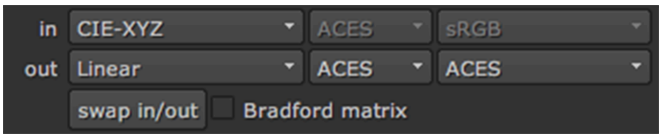


Figure 11. *Colorspace node settings for the transform from XYCD60 to ACES2065-1*

Fig. 11 shows the settings of the transform from XYZ(D60) to ACES2065-1. Alternatively, the following color matrix (Formula 2) can be used.

$$\begin{bmatrix} R_{ACES\,2065-1} \\ G_{ACES\,2065-1} \\ B_{ACES\,2065-1} \end{bmatrix} = \begin{bmatrix} 1.04981 & 0 & -0.00009748 \\ -0.49590296 & 1.37331295 & 0.09824004 \\ 0 & 0 & 0.99125206 \end{bmatrix} \cdot \begin{bmatrix} X_{D60} \\ Y_{D60} \\ Z_{D60} \end{bmatrix}$$

Formula 2. *Color Matrix from CIEXYZ D60 to ACES2065-1*

Finally, a conversion from ACES2065-1 to the Nuke Workspace (ACEScg) has to take place. This was achieved by using an OCIOColorspace node. As shown in Figure 12, the

conversion of the transfer function was performed correctly. The converted color bars have the same values as in Tab. 3.

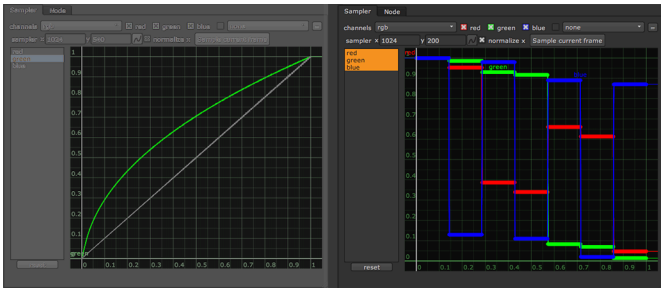


Figure 12. *Utility – Rec.709 – Camera transfer function: original image (green) and after transform (grey) – color values in ACEScg*

4.3 Implementation in OCIO

In the next section, the individual sub-transforms – represented by the nuke nodes – are converted into a correct OCIO transform.

```
- !<ColorSpace>
    name: Input - Rec.709
    family: Input
    equalitygroup: ""
    bitdepth: 32f
    description: |
      The Rec.709 Input color space
    isdata: false
    allocation: uniform
    allocationvars: [0, 1]
    to_reference: !<GroupTransform>
      children:
    -!<FileTransform> {src: linear_to_rec709.spi1d,
          interpolation: linear, direction: inverse}
       - !<MatrixTransform> {matrix:
       [0.41878837, 0.36482045, 0.16903733, 0,
        0.21533319, 0.71600521, 0.06866148, 0,
        0.01769946, 0.11254156, 0.87858415, 0,
               0, 0, 0, 1]}
       - !<MatrixTransform> {matrix:
             [1.04981, 0, -9.74845e-05, 0,
            -0.495903, 1.37331, 0.09824, 0,
             0, 0, 0.991252, 0,
             0, 0, 0, 1]}
```

Code example 5. *OCIO custom made Input-Rec.709 transform*

The *Input – Rec.709* Transform inserted in the OCIO configuration appears in the Nuke Interface as follows (Fig. 13):

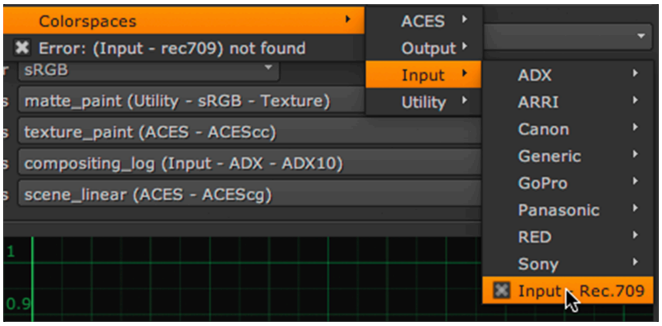


Figure 13*. OCIO Input – Rec.709 appearance in Nuke interface*

If *Input – Rec.709* is used as an input transform of test image 1, the values are the same as in Fig. 12 and Table 3.

5. The Rec709 Utilities as Output Transforms

In this section, the four Utility - Rec.709 transforms are tested as output transforms. With the exception of the *Utility – Curve - Rec.709* transform the conversion direction is correct (from reference).

5.1 Overview

There are the following transforms usable as outputs integrated in OCIO ACES1.1:

- Utility – Curve - Rec.709
- Utility – Linear - Rec.709
- Utility – Rec.709 - Camera
- Utility – Rec.709 - Display
- Output – Rec.709
- Output – Rec.709 (D60 sim.)

The test starts again with the *Utility – Curve - Rec.709* function. This time three test images generated in Nuke are used (Fig. 14): an SDR greyscale gradient (0...1), an HDR greyscale gradient (0 ... 16.3), and the SDR color bar generated by the *Colorbars* node (0 ... 1).

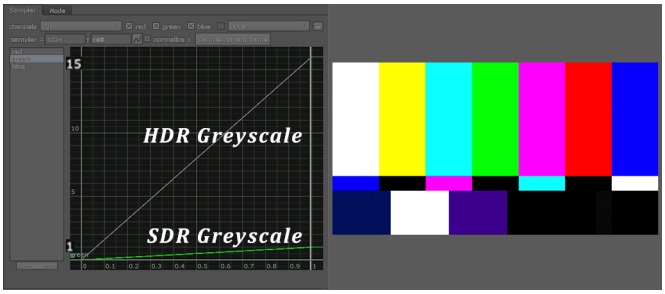


Figure 14*. Test images 2a: sdr greyscale ramp (grey); 2b: hdr greyscale ramp (green); 2c: sdr Color bars (right)*

5.2 Utility – Curve – Rec.709

First, the *Utility – Curve – Rec.709* transform is examined again. This time, the direction of conversion is wrong; the function still adds the Rec.709 transfer function, here called the *Curve*.

```
Utility - Curve - Rec.709
The Curve - Rec.709 color space
to_reference:
          linear_to_rec709.spi1d,
          direction: inverse
```

Code example 6. *OCIO-Utility-Curve-Rec.709 transform [9]*

The *Utility – Curve – Rec.709* Transform adds the Rec.709 transfer function correctly to the SDR greyscale gradient (test image 2a). The greyscale gradient (test image 2b) is truncated at about x=0.06; y = 1.0. In contrary to the *Output – Rec.709* Transform (chapter 6) does not process footage recorded by movie cameras or material generated in computer graphics or compositing

by an S-curve output transform. Since this behavior - except the *Utility – Linear – Rec.709* Transform - applies to all transforms in this chapter, HDR material should not be output with these transforms.

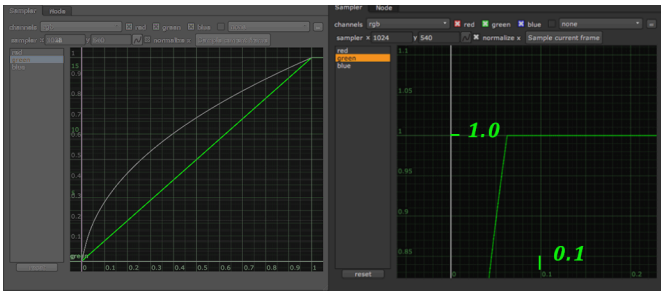


***Figure 15**. Test image 2a (left) and 2b (right) after Utility – Curve – Rec.709 transform*

In *Utility – Curve – Rec.709* no primaries / white point conversion is performed, therefore the values of the color bars remain unchanged. To output the material in Rec.709, another transform would have to take place (see section 3.1). As indicated in the OCIO *Readme.md* document [10] the usage of the *Utility – Curve – Rec.709* is "recommended to facilitate the creation of LUTs and other basic functionality". Other usage should be done with caution.

5.3 Utility – Rec.709 – Camera

Utility – Rec.709 – Camera is the output for standard Rec.709 applications (Internet, computer). The OCIO flow of sub-transforms is here in the right direction from reference to output.

```
Utility - Rec.709 - Camera
    The Rec.709 - Camera color space
    from_reference:
        !<MatrixTransform> {matrix:
                [0.952552, 0, 9.36786e-05, 0,
                 0.343966, 0.728166, -0.0721325, 0,
                 0, 0, 1.00883, 0,
                 0, 0, 0, 1]}
                 //ACES(P0) --> XYZ 60
        !<MatrixTransform> {matrix:
                [3.2096, -1.55743, -0.495805, 0,
                 -0.970989, 1.88517, 0.0394894, 0,
                 0.0597193, 0.210104, 1.14312, 0,
                 0, 0, 0, 1]}
                 //XYZ 60 --> sRGB/Rec709
        !<FileTransform>
                {src: linear_to_rec709.spi1d
```

***Code example 7.** OCIO Utility-Rec.709-Camera transform [9]*

After the ACES2065-12 (ACES Primaries P0) to CIE XYZD60, another transform to linear Rec.709/sRGB is executed. Only as the last step the transform applies the Rec.709 transfer function (with the reciprocal gamma exponent ~1.95).

For the application of the transfer function, the results are the same as for the *Utility – Curve – Rec.709* Transform (Fig. 15). Here too, the HDR gradient is cut off early.

The values of the color bars, which represent the primaries of the color space ACEScg, are not transmitted at their original position. Since they are outside the Rec.709 gamut, the OCIO internal processing chain cuts them off ("limited"). A contrary behavior is described in 5.5. The values of the color bars are precisely reproduced in the Rec.709 color space (Tab. 4).

	Reference		Display/Camera-Transform	
	x	y	x	y
white	0.31271	0.32902	0.31271	0.32902
yellow	0.41931	0.50525	0.41931	0.50525
cyan	0.22465	0.32875	0.22465	0.32875
green	0.30000	0.60000	0.30000	0.60000
magenta	0.32091	0.15418	0.32091	0.15418
red	0.64000	0.33000	0.64000	0.33000
blue	0.15000	0.06000	0.15000	0.06000

***Table 4.** Color bar color values (CIExy D65) after Rec.709 – Display/Camera transforms*

5.4 Utility – Rec.709 – Display

Utility – Rec.709 – Display is intended for output to a 100 nits broadcast master monitor. For this purpose, the Rec.1886 transfer function must be used. Again, the direction of the sub-transforms is correctly specified. The primaries / white point conversions are the same as for *Utility – Rec.709 – Camera*. Only the transfer function is adapted to the 100 nits broadcast mastering monitor.

```
Utility - Rec.709 - Display
    The Rec.709 - Display color space
    from_reference:
        !<MatrixTransform> {matrix:
                [0.952552, 0, 9.36786e-05, 0,
                 0.343966, 0.728166, -0.0721325, 0,
                 0, 0, 1.00883, 0,
                 0, 0, 0, 1]}
                 //ACES(P0) --> XYZ 60
        !<MatrixTransform> {matrix:
                [3.2096, -1.55743, -0.495805, 0,
                 -0.970989, 1.88517, 0.0394894, 0,
                 0.0597193, 0.210104, 1.14312, 0,
                 0, 0, 0, 1]}
                 //XYZ 60 --> sRGB/Rec709
        !<FileTransform>
                {src: linear_to_rec1886.spi1d
```

***Code example 8.** OCIO-Utility-Rec.709-Display transform 9]*

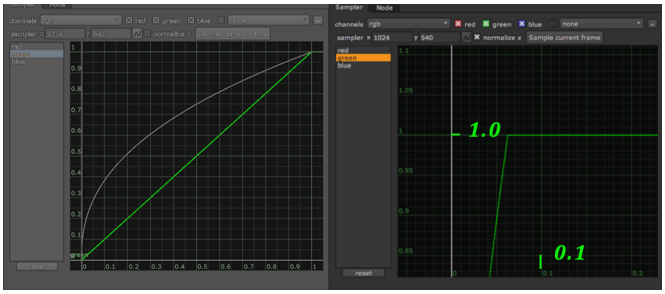


***Figure 16**. Test image 2a (left) and 2b (right) after Utility – Rec.709 – Display transform*

Here the applied transfer function is steeper (Gamma 1/2.4 vs. Gamma 1/~1.95 (Fig. 16). Also, the HDR gradient is cut off early.

The values of the color bars in the Rec.709 color space are the same as in the Rec.709 camera.

5.5 Utility – Linear – Rec.709

Utility – Linear – Rec.709 performs only a primaries / white point conversion from working color space ACEScg to Rec.709. No transfer function is applied.

```
Utility - Linear - Rec.709
    The Linear - Rec.709 color space
    from_reference:
         !<MatrixTransform> {matrix:
                 [0.952552, 0, 9.36786e-05, 0,
                  0.343966, 0.728166, -0.0721325, 0,
                  0, 0, 1.00883, 0,
                  0, 0, 0, 1]}
                  //ACES(P0) --> XYZ 60
         !<MatrixTransform> {matrix:
                 [3.2096, -1.55743, -0.495805, 0,
                  -0.970989, 1.88517, 0.0394894, 0,
                  0.0597193, 0.210104, 1.14312, 0,
                  0, 0, 0, 1]}
                  //XYZ 60 --> sRGB/Rec709
```

Code example 9. *OCIO-Utility-Linear-Rec.709 transform 9x]*

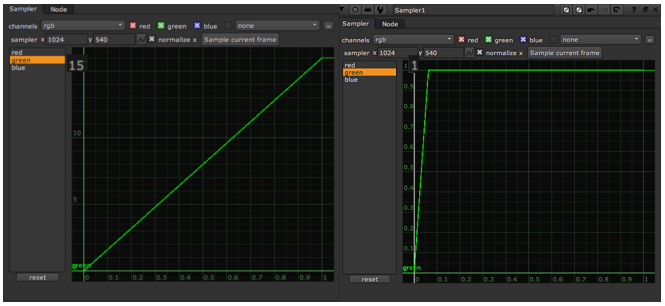


Figure 17*. Test image 2b after Utility – Linear – Rec.709 transform / HDR content cut off after clamping*

The *Utility – Linear – Rec.709* transform passes the HDR content through and does not clip it (Fig. 17 left). However, there is a problem. With the linear transform to the Rec.709 color space, the ACEScg primaries are passed on at their positions in the CIE xy chromaticity diagram. Since the ACEScg gamut is wider than that of Rec.709, the values for the color bars are shifted to the super white range (>1.0) (Fig. 18 left). If the result of the transform is passed on in OpenEXR format, these increased values are retained. The JPEG format cuts these off (Fig. 18 right).

There is no internal clamping like in Display/Camera. So an additional clamping to 0..1 after the transform is recommended to pass the imagery correctly to the pipeline. But this cuts off the HDR content (Fig. 17 right).

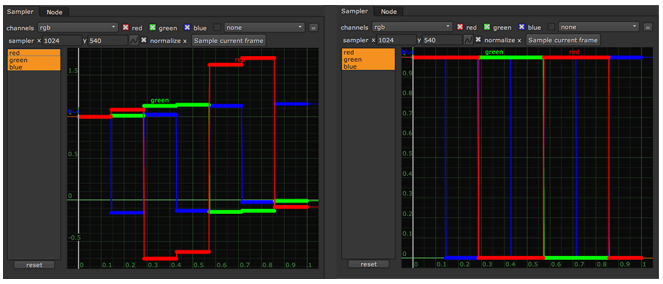


Figure 18*. Test image 2c in Rec.709 color space after Utility – Linear – Rec.709 transform / Primaries in Rec.709 correctly clamped*

6. Using Output - Rec709 for roundtripping

6.1 Overview

Now, the *Output – Rec709* transform is used to output the Nuke created SDR greyscale (test image 2a), and the Nuke created HDR greyscale (test image 2b) images. Also, the Rec.709 test image 1 is imported using the recently created *Input – Rec.709* transform. To this imported image, an *OCIO Output – Rec709* transform is applied to create a round trip. So it is possible to compare the image before and after the transforms.

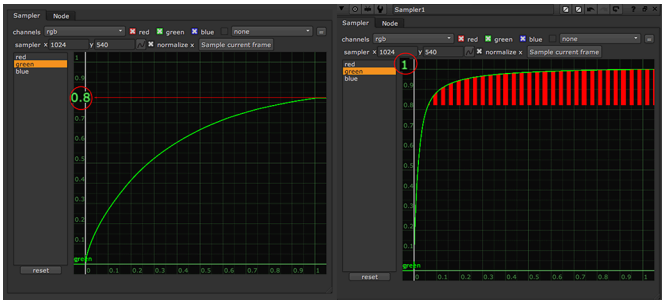


Figure 19*. Test-image-2a after Output – Rec.709 transform*

The output transform darkens the upper luminance range to make space for the high dynamic range content above 1.0 to be rolled in correctly (Fig. 19).

The transform also changes the input color, because there are different modules inside the *Reference Render Transform – Output Device Transform*, which form the *Output Transform*. So this transform is not completely invertible (Tab. 5). This behavior causes problems when the color is needed to be unchanged (logos, brand color).

	Reference		Output – Rec.709 - Transform	
	x	y	x	y
white	0.31271	0.32902	0.31271	0.32902
yellow	0.41931	0.50525	0.38896	0.44454
cyan	0.22465	0.32875	0.23507	0.33746
green	0.30000	0.60000	0.29602	0.52240
magenta	0.32091	0.15418	0.37477	0.14865
red	0.64000	0.33000	0.64615	0.26415
blue	0.15000	0.06000	0.12881	0.04746

Table 5. *Color bar color values (CIExy D65) after Output – Rec.709 transform*

6.2 Inverse ODT+RRT as input

Here, the image content was moved into the HDR range by applying a pre-stretching in the Rec.709 input. The aim of the test in the following section was to verify if this technique keeps the color on the round trip better in place. A combined inverse *Output Transform/Reference Render Transform* was applied when importing the image. First two reference images were created using the Color Transform Language (CTL). These images were then imported into Nuke using the *to reference* - direction in the OCIO implementation of *Output – Rec.709*.

```
Output - Rec.709
ACES 1.0 Output - Rec.709 Output Transform
    ACES Transform ID :
      ODT.Academy.Rec709_100nits_dim.a1.0.3
  to_reference:
    InvRRT.Rec.709.Log2_48_nits_Shaper.spi3d
    Log2_48_nits_Shaper_to_linear.spi1d
  from_reference:
    Log2_48_nits_Shaper_to_linear.spi1d
          Log2_48_nits_Shaper.RRT.Rec.709.spi3d
```

Code example 10. *OCIO-Output-Rec.709 transform [9]*

Two TIFF-coded versions of the 8 bit *test image 1* were created. The first image output was an 8-bit version. The resulting image was then imported into Nuke using the *OCIO Utility – Raw* transform. The CTL calls in the macOS-Terminal are given in *Code example 11*.

```
home $ ctlrender –ctl ctl/odt/rec709/~
InvODT.Academy.Rec709_100nits_dim.ctl
–ctl ctl/rrt/InvRRT.ctl
images/CB_ramp_bw_0_1_rec709.tiff
fullinverse_8bit.tiff –format tiff8
```

Code example 11. *Terminal call of CTL transforms (8-bit image)*

The low resolution prevents the pre-stretching of the image content into the HDR area with the used inverse *Output – Rec.709* transform. At the limit of the resolution, the image content is cut off (Fig. 20).

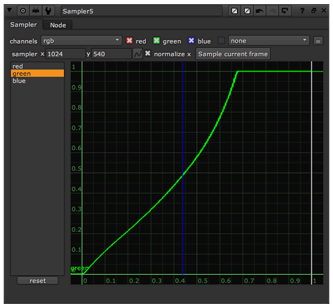


Figure 20*. Transfer function of an 8 bit tiff version clamped*

Then, the second version of *test image 1* was created; this time, the output was a 32-bit-floating-point version. The resulting image was then imported into Nuke using the *OCIO Utility – Raw* transform. The CTL calls in the macOS-Terminal were:

```
home $ ctlrender –ctl ctl/odt/rec709/~
InvODT.Academy.Rec709_100nits_dim.ctl
–ctl ctl/rrt/InvRRT.ctl
images/CB_ramp_bw_0_1_rec709.tiff
fullinverse_32bit.tiff –format tiff32
```

Code example 12. *Terminal call of CTL transforms (32-bit image)*

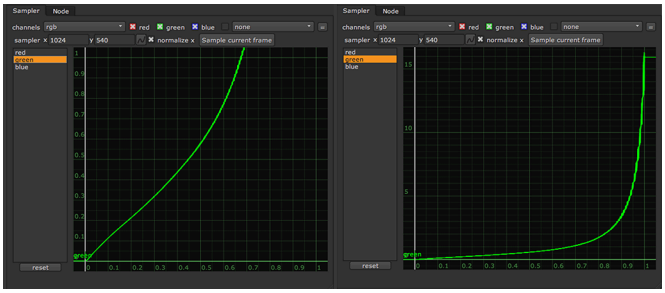


Figure 21. *Transfer function of a 32 bit tiff version clamped, range 0..1 (left), range 0..16 (right)*

Due to the higher resolution, the image content is shifted into the HDR area. It must be noted here that the greyscale gradient of the original image is only between 0 and 1, and its peak is pushed to 16.3 by the transform.

In the third test, *test image 1* was imported into Nuke using the *OCIO-Output-Rec.709* transform. The transfer function after the import was the same as in Fig. 21.

6.3 Round tripping Output - Rec709

It is not apparent at first glance, but the transfer functions in *ODT.Academy.Rec709_100nits_dim* and in the inverse counterpart are retrieved from Rec.1886 and intended to drive a Rec.1886 monitor with 100 nits in a dim surrounding environment, in contrast to the film environment, which is termed a dark surrounding. When this transform is applied inversely, it also expects material encoded with the Rec.1886 transfer function. Such material is rather rare, however, as cameras record with the Rec.709 transfer function.

In this subsection, the deviations in transfer function and color values are investigated by adding the *Output-Rec.709* function in its inverse form as input and the original forward form as output. This procedure is often called *round tripping*. The following CTL-transforms are chained together.

- InvODT.Academy.Rec709_100nits
- InvRRT
- RRT
- ODT.Academy.Rec709_100nits_dim

In Nuke, the functionality is achieved by using the Output-Rec.709 as OCIO Transform for the input of the *Read* node. For the output, an *OCIOColorSpace* node with a transform from ACEScg to Output-Rec.709 is used (Fig. 22).

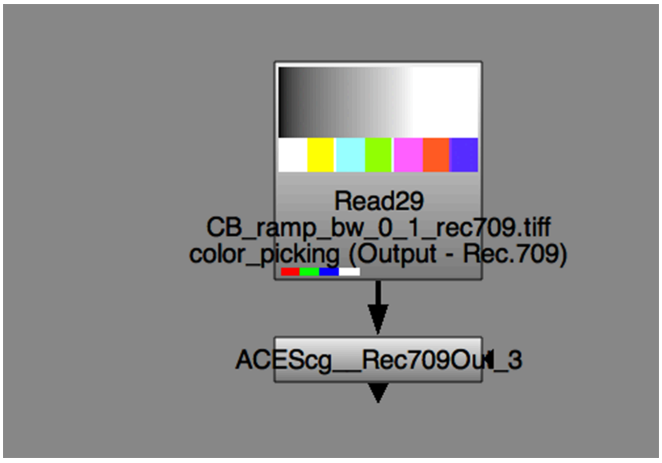


***Figure 22**. Nuke nodes for the Output – Rec.709 round-tripping*

The transfer function during the input and after the round trip is the same. The inverse Rec.1886 and forward Rec.1886 transforms are canceling themselves out. So the original Rec.709 transfer function remains. The result is the correct output for YouTube, but not for a broadcast monitor because it expects a Rec.1886 transfer function to control it

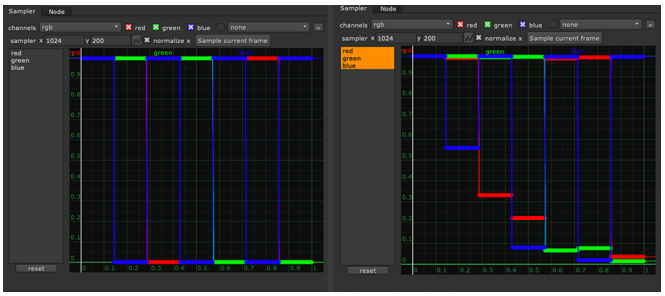


***Figure 23.** Comparison of the color deviations after uncorrected round trip; left reference; right round trip values*

	Reference		After Round Tripping	
	x	y	x	y
white	0.31271	0.32902	0.31271	0.32902
yellow	0.41931	0.50525	0.34656	0.38590
cyan	0.22465	0.32875	0.25870	0.32830
green	0.30000	0.60000	0.32384	0.53628
magenta	0.32091	0.15418	0.31981	0.17202
red	0.64000	0.33000	0.58330	0.35446
blue	0.15000	0.06000	0.16146	0.07291

***Table 6.** Color bar color values (CIExy D65) after Output – Rec.709 round tripping*

More problematic is the comparison of the color values. Due to the internal modular structure of the four CTL transformations, which form the basis for the OCIO transformations, there are considerable color deviations, as shown in Tab. 6, and Fig. 23. The peak values of the color bars are correctly at 1.0, and thus not reduced to about 0.8 if pre-stretching was not applied, but it is noticeable that especially the contribution of the blue channel – which should be zero – for the yellow color bar is very high.

6.4 Correcting the transfer functions

In this section, the transfer functions were corrected. *Test image 1* was read in linearly without any transformation (*OCIO - Utility – RAW*). Then the Rec.709 transfer function was removed and the Rec.1886-transfer function was added, which is expected by the *Output – Rec.709* Transform. Then the inverse *Output - Rec.709* transform was used as the input for an *OCIOColorspace* node, with the output set to Nuke's workspace ACEScg. Then the forward *Output – Rec.709* was applied. The result of the transforms is then suitable for controlling a broadcast master monitor.

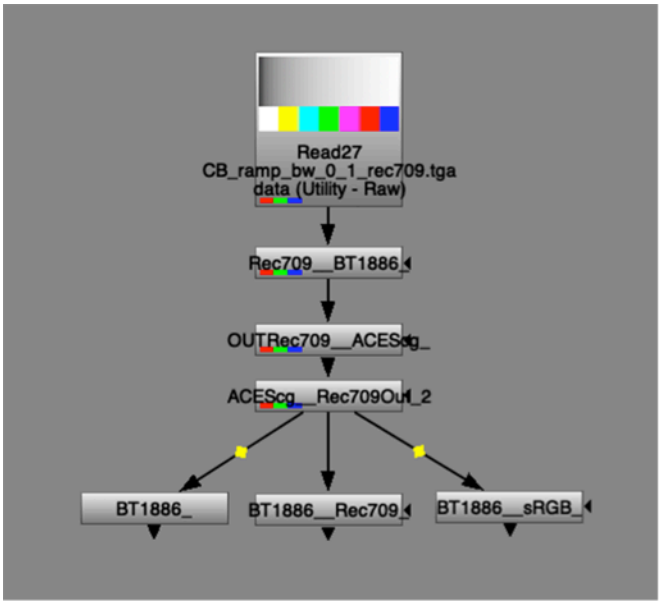


***Figure 24**. Nuke nodes for the corrected round trip*

For Rec.709 encoded material, the Rec.1886 transfer function was removed, and the *Rec.709* transfer function was added. As bonus, a transform from the output to sRGB was applied (Fig. 24). The results of all three procedures were then compared.

If *Output – Rec.709* transform was selected as input for the *OCIOColorSpace* node, and the output was selected as *ACEScg*, the operator generated a pre-stretch. Since *test image 1* was read in unchanged (*Utility – Raw*), it was possible to correct the transfer function in the input section before the invers *Output – Rec.709* transform. The Rec.709 transfer function was removed, and instead, a Rec.1886 transform was applied because the following *Output – Rec.709* conversion expects it as the input.

6.4.1 Conversion to Rec.1886

For this conversion the output took place directly after the *ACEScg__Rec709Out* node. With this procedure, a broadcast monitor (ITU-R BT.1886 [11]), can be controlled because the forward Output – Rec.709 transform outputs an image with a Rec.1886 transfer function applied.

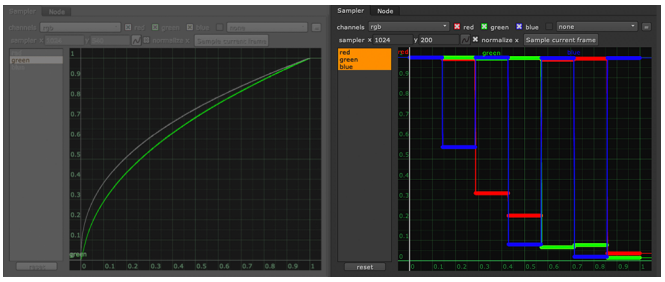


***Figure 25**. Comparison of input transfer function (Rec.709) and output to Rec.1886 (grey) after the round trip / Color bars after the round trip*

The transfer function of the original image (Rec.709) is flatter than the output to Rec.1886. The values of the color bars are shifted further by the round trip and the gamma correction (Fig. 25, Tab 7).

	After corrected round tripping to Rec.1886		Deviations	
	x	y	x	y
white	0.31271	0.32901	0.00000	0.00000
yellow	0.37125	0.42836	0.04805	0.07689
cyan	0.23795	0.32762	0.01330	0.00123
green	0.30879	0.58163	0.00879	0.01837
magenta	0.31890	0.15740	0.00202	0.00322
red	0.62562	0.33620	0.01438	0.00620
blue	0.15262	0.06295	0.00262	0.00295
		Error xy	0.08916	0.10886
		Error sum	**0.19802**	

***Table 7.** Color bar color values (CIExy D65) after corrected Output – Rec.709 round tripping to Rec.1886*

6.4.2 Conversion to Rec.709

To output video material correctly with a Rec.709 transfer function (e.g., for YouTube), it was necessary to remove the Rec.1886 transfer function after the roundtrip and add a Rec.709 transfer function.

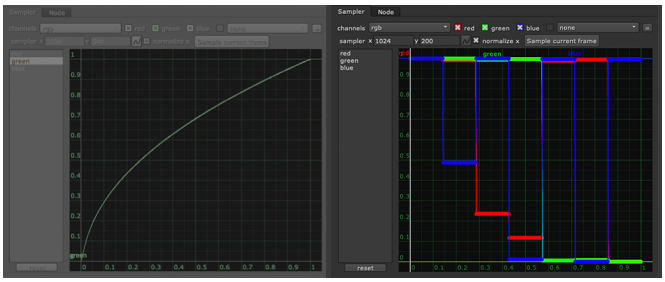


***Figure 26**. Comparison of input transfer function (Rec.709) and output to Rec.709 (grey) after the round trip / Color bars after the round trip*

The output transfer function of the original image (Rec.709) was the same after the output. The values of the color bars are not shifted so far away from the reference like the output to Rec.1886. This is because the applied gamma curve is not so steep for Rec.709 (Fig. 26, Tab. 8).

	After corrected round tripping to Rec.709		Deviations	
	x	y	x	y
white	0.31271	0.32901	0.00000	0.00000
yellow	0.38038	0.4421	0.03892	0.06315
cyan	0.23212	0.32736	0.00747	0.00149
green	0.30455	0.59482	0.00455	0.00518
magenta	0.31872	0.15341	0.00220	0.00077
red	0.63856	0.33104	0.00144	0.00104
blue	0.15010	0.06007	0.00010	0.00007
		Error xy	0.05468	0.07107
		Error sum	**0.12638**	

***Table 8.** Color bar color values (CIExy D65) after corrected Output – Rec.709 round tripping to Rec.709*

6.4.3 Conversion to sRGB

To output video material correctly with an sRGB transfer function (e.g., for images), it is necessary to remove the Rec.1886 transfer function after the round trip and add an sRGB transfer function.

The transfer function of the original image (Rec.709) is flatter than the output to sRGB. The values of the color bars are shifted slightly further away from the reference than the output to Rec.709. This is because the applied gamma curve is a bit steeper than Rec.709 (Fig. 27, Tab. 9).

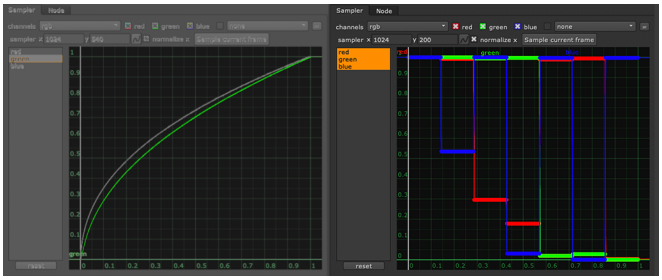


***Figure 27**. Comparison of input transfer function (Rec.709) and output to sRGB (grey) after the round trip / Color bars after the round trip*

	After corrected round tripping to sRGB		Deviations	
	x	y	x	y
white	0.31271	0.32901	0.00000	0.00000
yellow	0.37459	0.43358	0,04471	0.07167
cyan	0.23556	0.32754	0.01091	0.00131
green	0.30744	0.58966	0,00744	0.01034
magenta	0.31892	0.15434	0.00200	0.00016
red	0.63691	0.33296	0.00309	0.00296
blue	0.15029	0.06021	0.00029	0.00021
		Error xy	0.06844	0.08665
		Error sum	**0.15509**	

***Table 9.** Color bar color values (CIExy D65) after corrected Output – Rec.709 round tripping to sRGB*

6.5 Round tripping without RRT+ODT

To retain color values (logos, brand color), we are now testing a workflow that contains neither the RRT nor the ODT, but a simple round trip with our *Input – Rec709* transform and the *Utility – Rec.709 – Camera* (Rec.709) as an output. If the desire is to deliver for a 100 nits broadcast monitor, it is possible to apply as the output the *Utility – Rec.709 – Display* transform instead.

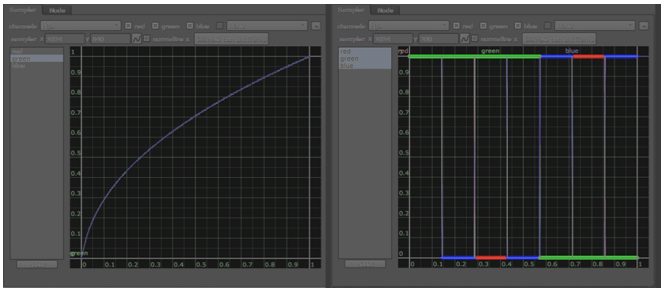


***Figure 28**. Comparison of input transfer function (Rec.709) and output to Rec.709 (grey) after simplified round trip / Color bars after the round trip*

This workflow transmits the transfer function and the color values precisely (Fig. 28). However, this workflow is more useful in Rec.709 stand-alone applications. A combination with the HDR material from film cameras would degrade the latter (see the following section).

7. Conclusion

The color workflow in the cinema area with different color spaces is exceptionally complicated. The transfer of color values between HDR and SDR, wide gamut, and standard gamut, linear, and perceptually uniform oriented color spaces requires precise planning of pipelines. The implementation of OpenColorIO ACES1.1 in Nuke12 is robust. The transforms work error-free as far as the tests here are concerned.

However, the names are sometimes misleading, and essential transforms are hidden in the depth of the interface. Employees with little knowledge of color systems and artists can be irritated. Although there is a system, it is not immediately apparent. For example, at first glance, there is no input for Rec.709 coded material.

Furthermore, the specification of the conversion direction in the OCIO configuration is also irritating. During the implementation of OCIO in Nuke, the conversion direction was obviously determined depending on the placement of the transform (input/output).

To ensure a more intuitive handling, a custom made transform for the import of Rec.709 encoded material was created. The transform includes the correct direction of the sub-transforms. This *Input – Rec.709* transform also appears in the interface under Input, so that artists don't have to worry about choosing the right transform. Of course it is also possible to choose the *Utility - Rec.709 - Camera* which gives the same results. The selection of this function for the input is not immediately obvious. Another reason to create an *Input – Rec.709* Transform was the fact that OCIO is an operating system wide application. It may be that an OCIO implementation in other programs cannot easily change the conversion direction.

The output of Rec.709 encoded material is much more complicated. The biggest problem is the handling of HDR material generated by film cameras and computer graphics. To adapt the larger dynamic range to the end devices with smaller dynamic range, ACES uses a combination of the ACES Reference Render Transform (RRT) and the Output Device Transform (ODT, which consists of two segmented curves and thus transfers the HDR material into the SDR area in an s-curve shape. Clipping is only performed from a value of 16.3 on. Since the HDR material requires space, the SDR material, which also includes Rec.709 encoded material, has to yield and is further compressed from 1.0 to about 0.8. Since this is done in a wide gamut color space, the primary colors are shifted from their position.

This color change is not desirable when it comes to maintaining carefully selected color tones, such as logos and corporate colors. In this case, pre-stretching the material with an inverse combination of (ODT and RRT) does not help, as the modules contained in these transforms change the color.

For stand-alone Rec.709 projects, a simple workflow can be usedwith *Input – Rec.709* (or Utility …– Rec.709 – Camera inverse direction) and as output *Utility – Rec.709 - Camera* (forward). With this workflow, the Rec.709 primary colors are maintained.

If the finishing is in one's own hands and the image combination allows it, it is possible to build two independent

branches in Nuke, which are then joined together before output. One branch is the HDR branch, which is then converted to the Rec.709 color space using RRT+ODT. The other branch contains the Rec.709 material processed according to the simplified workflow described above, which can then be combined with the HDR branch. The material created in this way is then exported untouched.

In part II of this paper the important issue of LMT round tripping between different software packages, displays, and cameras will be described.

References

[1] S. Dyer. LMTs Part 1: What are they and what can they do for me? https://acescentral.com/t/lmts-part-1-what-are-they-and-what-can-they-do-for-me/790. Retrieved 2020-01-09

[2] Technical Bulletin TB-2014-004, Informative Notes on SMPTE ST 2065-1 – Academy Color Encoding Specification (ACES)

[3] Netflix Partner Help Center Color Pipeline Requirements. https://partnerhelp.netflixstudios.com/hc/en-us/articles/360000588548-Color-Pipeline. Retrieved 2020-01-09

[4] S. Cooper, T. Mansencal, K. Wheatley: ACES Retrospective and Enhancements - 2017 - https://goo.gl/F71kvV - DOI: 10.5281/zenodo.345624

[5] Recommendation BT.709. https://www.itu.int/rec/R-REC-BT.709. Retrieved 2020-01-09

[6] The Color Transformation Language. http://ampasctl.sourceforge.net. Retrieved 2020-01-09

[7] The Foundry website: https://www.thefoundry.co.uk. Retrieved 2020-01-09

[8] OpenColorIO OPEN SOURCE COLOR MANAGEMENT https://opencolorio.org/developers/getting_started.html. Retrieved 2020-01-09

[9] ACES 1.1 OpenColorIO configuration /Applications/Nuke12.0v3/Nuke12.0v3.app/Contents/Resources/OCIOConfigs/configs/aces_1.1/config.ocio

[10] OCIO README.md /Applications/Nuke12.0v3/Nuke12.0v3.app/Contents/Resources/OCIOConfigs/configs/aces_1.1/README.md

[11] Recommendation BT.1886. https://www.itu.int/rec/R-REC-BT.1886. Retrieved 2020-01-09

Author Biographies

Eberhard Hasche received his diploma in electro engineering from the Technical University of Dresden (1976). Afterward, he studied double bass, composition and arranging at Hochschule für Musik „Carl Maria von Weber“ in Dresden (state examination 1989). Since 2003 he is professor for audio and video technology at Brandenburg University of Applied Sciences, Germany. He is focused on image compositing (certified Nuke Trainer by The Foundry in 2012). He is member of the Visual Effects Society since 2018)

Dominik Benning graduated as an assistant interior designer in 2015. He is working on his Bachelor degree in computer science at the University of Applied Sciences Brandenburg in Germany, focusing his efforts on digital video and 3D modeling.

Reiner Creutzburg is a retired Professor of Computer Science at the Technische Hochschule Brandenburg in Brandenburg, Germany. He is a member of the IEEE and SPIE and Chairman of the Multimedia on Mobile Device (MOBMU) Conference at the Electronic Imaging conferences since 2005. His research interest is focused on Cybersecurity, Digital Forensics, Open Source Intelligence, Multimedia Signal Processing, eLearning, and Modern Digital Media and Imaging Applications.

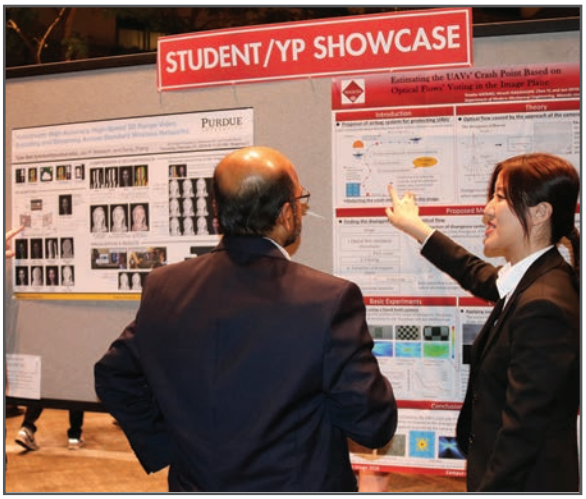
JOIN US AT THE NEXT EI!

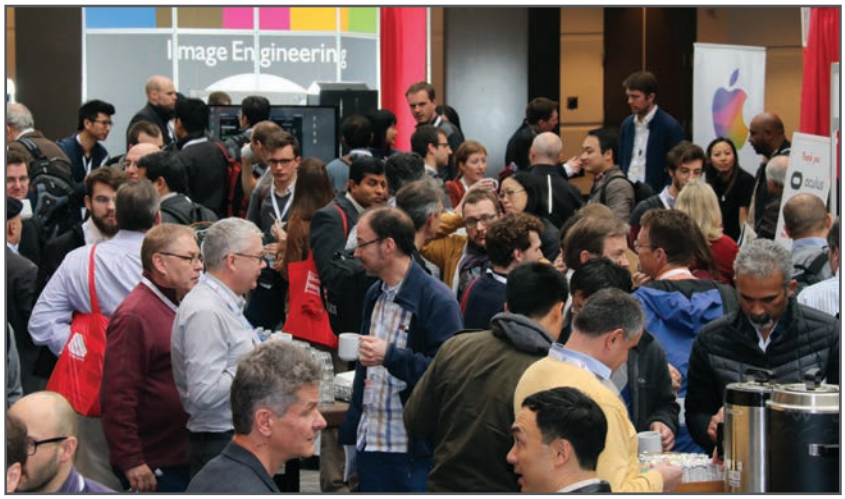
IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!





• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org

IS&T
imaging.org