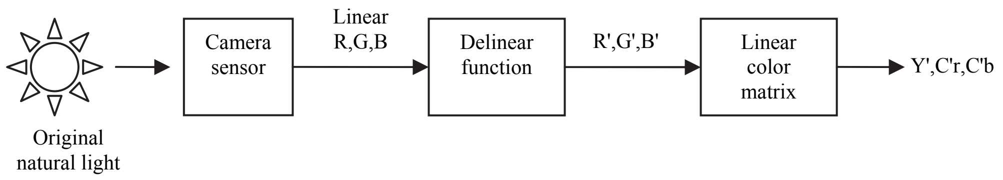
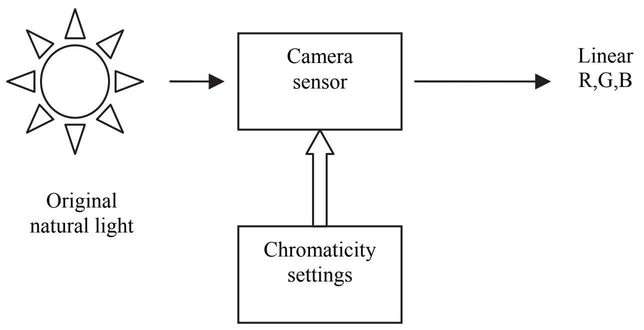
I've struggle to gather the informations i wanted, so i put all what i've found in this document. Informations come from several sources (internet search, etc…), but mostly come from the following documents :
R-REC-BT.601
R-REC-BT.656
R-REC-BT.709
R-REC-BT.1886
R-REC-BT.2020
R-REC-BT.2087
R-REC-BT.2100
R-REP-BT.2390
Sometimes i go into details, sometimes i summerise. Anyway, if you want more details information, i suggest you to read the following documents listed above.

Video recording

The video recording follow the global process :



My idea was : If from Y',C'_r,C'_b i'm able to revert to the *Original natural light* (or to data providing similar informaton), can't i just put back this information in front of another camera, to do, for exemple, an HDR to SDR convertion ? Of course, if it was so simple, it would have been done since a long time.
At first glance, it seems that *Linear R,G,B* would be the data to use for *Original natural light*, unfortunately, there is a trick. When digging more accurately, you discover that in reality, you have this :



The *Linear R,G,B* data provided are functions of the *Chromaticity settings* of the camera (White color T° and this kind of stuff, not realy my field of expertise). Nevertheless, the same *Original natural light* input can provide differents *Linear R,G,B* output according these settings, so it's not what i'm looking for. To get rid of these settings, and have data providing informations realy similar to *Original natural light*, the **ZYZ** colorspace seems the more appropriate to work with.

RGB/XYZ matrix conversion

Your primary colours references are : (x_r,y_r) for Red Primary, (x_g,y_g) for Green Primary, (x_b,y_b) for Blue Primary.
Reference White is $(x_w,y_w)Y$.
The normalized values are :

$$X_w = Y \times \frac{x_w}{y_w}$$

$Y_w=Y$

$$Z_w = Y \times \frac{(1.0 - x_w - y_w)}{y_w}$$

Most of the time, white point is provided with normalized Y=1.0.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [M] \times \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$[M] = \begin{bmatrix} S_r \times X_r & S_g \times X_g & S_b \times X_b \\ S_r \times Y_r & S_g \times Y_g & S_b \times Y_b \\ S_r \times Z_r & S_g \times Z_g & S_b \times Z_b \end{bmatrix}$$

$$X_r = Y_r \times \frac{x_r}{y_r}$$

$Y_r=1.0$

$$Z_r = Y_r \times \frac{(1.0 - x_r - y_r)}{y_r}$$

$$X_g = Y_g \times \frac{x_g}{y_g}$$

$Y_g=1.0$

$$Z_g = Y_g \times \frac{(1.0 - x_g - y_g)}{y_g}$$

$$X_b = Y_b \times \frac{x_b}{y_b}$$

$Y_b=1.0$

$$Z_b = Y_b \times \frac{(1.0 - x_b - y_b)}{y_b}$$

$$\begin{bmatrix} S_r \\ S_g \\ S_b \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix}^{-1} \times \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

For proper XYZ values, the RGB must be what is called *voltage normalized by the reference white level and proportional to the implicit light intensity* or *signal determined by scene light and scaled by camera exposure.* So the linear scene light (R_S, G_S, B_S) the output of the sensor, not the linear displayed values (R_D, G_D, B_D) and even less the non-linear (R', G', B').

Warning :

For an input range of [0.0,1.0] of (R,G,B), you don't have an ouput range of [0.0,1.0] for XYZ.

The few tests i've made with standard value from REC.BT showed a range more like this :

X : [0.0,0.95]

Y : [0.0,1.0]

Z : [0.0,1.1]

The same few tests seems to show that this range doesn't change with the several chromaticity values from standard REC.BT, which all have the same D65 white point. Nevertheless, if you change the value of the white point, the range changes.

If this has no consequences when working with float datas, it has when working with integer values. In this case, you'll need to know what where the chromaticity parameters used to convert from R,G,B, to know the exact range X,Y,Z can have.

If you're with 8 bits for exemple, you have to make your system for the following :

X : $0 \rightarrow X_{min}$, $255 \rightarrow X_{max}$.

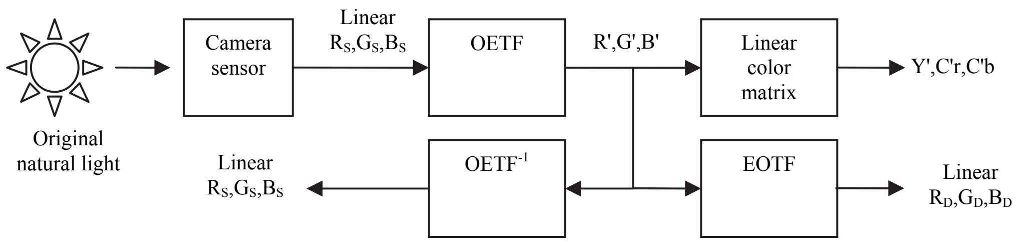
Y : $0 \rightarrow Y_{min}$, $255 \rightarrow Y_{max}$.

Z : $0 \rightarrow Z_{min}$, $255 \rightarrow Z_{max}$.

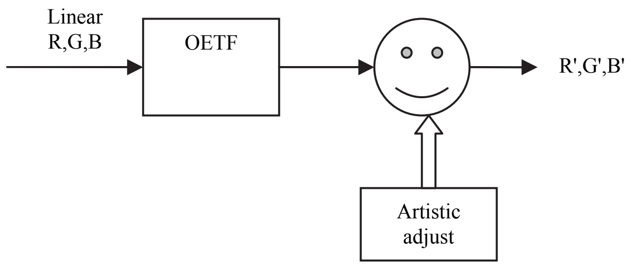
But these limit values, can only be know if you know the chromaticity parameters used. So if you want to program something generic (where you can set the chromaticity parmeters), and work with other things that float values, each time you'll work in XYZ colorspace with integer values, you have to know the chromaticity parameters used to accurately know these limits. You still can consider that the limits will always stays within the "standard" range, but it comes with a risk.

SDR Video

For SDR video, you have the following process :



Again, when you dig, you found that sometimes you have a little trick, and can have in reality the following :



Of course, reversing the *Artistic adjust* is not possible ! So, you can consider that if you goes back until *Original natural light*, you have instead *Light the artistic ajust would have liked.* This step will be ignored in the reverse process, there is no choice anyway.

The OETF function (identical for REC.2020, REC.709 and REC.601) to go from linear sensor to non-linear is :

α=1.09929682680944
β=0.018053968510807
$E' = 4.5 \times E \quad 0 \le E < \beta$
$E' = \alpha \times E^{0.45} - (\alpha - 1) \quad \beta \le E \le 1$
E is voltage normalized by the reference white level and proportional to the implicit light intensity, E' the resulting non-linear.
For luminance, we have E=1 for 100 cd/m².

Reverse path

The $OETF^{-1}$ function to go from non-linear to linear is :

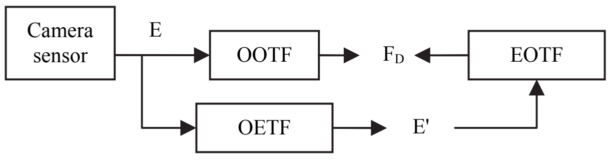
$$E = \frac{E'}{4.5} \quad 0 \le E' < 4.5 \times \beta$$

$$E = \left(\frac{E' + (\alpha - 1)}{\alpha} \right)^{\frac{1}{0.45}} \quad 4.5 \times \beta \le E' \le 1$$

The EOTF function for SDR (REC-BT.1886) to go from non-linear to linear display is :
$E = E'^{2.404}$

HDR Video

Before going to each specific HDR modes (PQ/HLG), the global synopsys of HDR process can be seen like this :



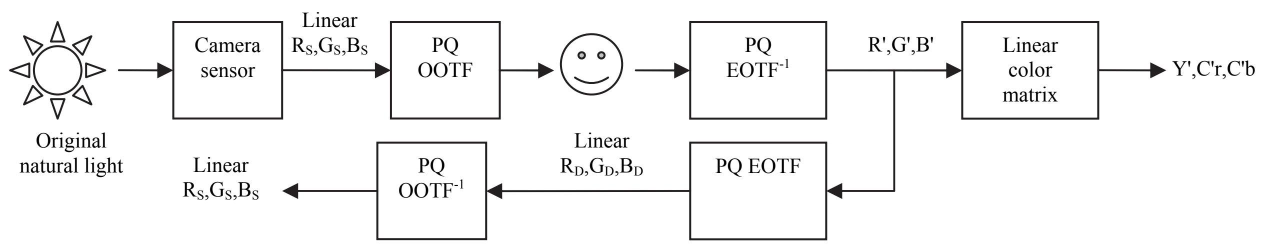
E is voltage normalized by the reference white level and proportional to the implicit light intensity. Also described as signal determined by scene light and scaled by camera exposure. As said (*voltage normalized by the reference white level*, *scaled by camera exposure*), this signal include the chromaticity elements. It's close to the true original scene light, but still not exactly. The closest to the original scene light would be the XYZ colorspace.

F_D=OOTF[E]=EOTF[E']
F_D is the linear displayed luminance, so the value used by the screen to display. As we can see, it's not the real light. According what you want to do, you may want to stop at this step.

E'=OETF[E]
The final E' is the resulting non-linear signal. This is the one which will be encoded, transmitted, etc… This is also the one we are all working on on our PCs !

HDR PQ Video

Now that the trick is knew, for HDR PQ video, you have the following process :



Again, the *Artistic adjust* will be ignored. As it can be seen (and it's logical), this is done at the step we have the linear displayed value.

For PQ HDR, the luminance max is 10000 cd/m² (so 100 times the SDR). For both scene light (camera sensor) and displayed component, 1.0 = 10000 cd/m².

We have :

F_D=EOTF[E']=10000Y

E' denotes a non-linear colour value {R', G', B'} in PQ space [0.0:1.0].
F_D is the luminance of a displayed linear component {R_D, G_D, B_D} in cd/m².
Y denotes the normalized linear colour value, in the range [0.0:1.0].

F_D=OOTF[E]=G_{1886}[G_{709}[E]]= G_{1886}[E']

E={R_S, G_S, B_S} is the signal determined by scene light and scaled by camera exposure, in the range [0.0:1.0].
E′ is a non-linear representation of E (but not in the PQ space).
F_D is the luminance of a displayed linear component {R_D, G_D, B_D} in cd/m².

E'=OETF[E]=$EOTF^{-1}$[OOTF[E]]= $EOTF^{-1}$[F_D]

E′ is the resulting non-linear signal {R', G', B'} in the range [0.0:1.0] in PQ space.
E={R_S, G_S, B_S} is the signal determined by scene light and scaled by camera exposure, in the range [0.0:1.0].
F_D is the luminance of a displayed linear component {R_D, G_D, B_D} in cd/m².

The PQ OOTF function is :

α=1.09929682680944
β=0.018053968510807

$$E' = 4.5 \times 59.5208 \times E \quad 0 \le E \le \frac{\beta}{59.5208}$$

$$E' = \alpha \times (59.5208 \times E)^{0.45} - (\alpha - 1) \quad \frac{\beta}{59.5208} \le E \le 1$$

$$F_D = 100 \times E'^{2.404}$$

The PQ EOTF^{-1} function is :

m1 = 0.1593017578125
m2 = 78.84375
c1 = 0.8359375
c2 = 18.8515625
c3 = 18.6875

$$\mathrm{EOTF}^{-1}[F_D] = Y' = \left(\frac{c1 + c2 \times Y^{m1}}{1 + c3 \times Y^{m1}}\right)^{m2}$$

$$Y = F_D/10000 \rightarrow Y = \frac{E'^{2.404}}{100}$$

<u>Reverse path</u>

The PQ EOTF function is :

m1 = 0.1593017578125
m2 = 78.84375
c1 = 0.8359375
c2 = 18.8515625
c3 = 18.6875

$$Y = \left(\frac{\max\left[(Y'^{\frac{1}{m2}} - c1), 0\right]}{c2 - c3 \times Y'^{\frac{1}{m2}}}\right)^{\frac{1}{m1}}$$

The PQ OOTF^{-1} function is :

$$F_D = 10000Y \rightarrow E' = (100 \times Y)^{\frac{1}{2.404}}$$

$$E = \frac{E'}{59.5208 \times 4.5} \qquad 0 \le E' \le 4.5 \times \beta$$

$$E = \frac{\left(\frac{E' + (\alpha - 1)}{\alpha}\right)^{\frac{1}{0.45}}}{59.5208} \qquad 4.5 \times \beta < E' \le 1$$

So, we have the directs mathematical paths :
$(R_S, G_S, B_S) \leftrightarrow (R_D, G_D, B_D)$
$(R_D, G_D, B_D) \leftrightarrow (R', G', B')$
Allowing us to have :
$(R_S, G_S, B_S) \leftrightarrow (R', G', B')$
Also, all mathematical paths are independant on all channels, allowing easy LUT implementation.

HDR HLG Video

HLG is more tricky. First, it's adaptative, HLG has two others parameters : L_W and L_B which define the maximum and the minimum for linear displayed value, where on PQ maximum is fixed to 10000 cd/m².
HLG has also for E' two range mode [0.0:1.0] and [0.0:12.0]. The formulas will be for the [0.0:1.0] range mode, and we will use this mode.

E'=OETF[E]
E is the signal for each colour component $\{R_S, G_S, B_S\}$ proportional to scene linear light and scaled by camera exposure, normalized to the range [0.0:1.0].
E' is the resulting non-linear signal {R', G', B'} in the range [0.0:1.0].

The HLG OETF function is :
$E' = \sqrt{3 \times E} \quad 0 \le E \le 1/12$
$E' = a \times \ln(12 \times E - b) + c \quad 1/12 < E \le 1$
$a = 0.17883277 \;\; b = 1 - 4 \times a \;\; c = 0.5 - a \times \ln(4 \times a)$

F_D=OOTF[E]=OOTF[OETF^{-1}[E']]
E=OETF^{-1}[E']

$\{R_S, G_S, B_S\}$ are the scene linear light signals, E, for each colour component normalized in the range [0.0:1.0].
F_D is the luminance of a displayed linear component $\{R_D, G_D, \text{or } B_D\}$ in cd/m².

The HLG OOTF function is :
$Y_S = 0.2627 \times R_S + 0.6780 \times G_S + 0.0593 \times B_S$
$R_D = \alpha \times Y_S^{\gamma-1} \times R_S + \beta$
$G_D = \alpha \times Y_S^{\gamma-1} \times G_S + \beta$
$B_D = \alpha \times Y_S^{\gamma-1} \times B_S + \beta$
$\alpha = (L_W - L_B)$
$\beta = L_B$
$\gamma = 1.2 + 0.42 \times \mathrm{Log}_{10}(L_W/1000)$
Or, within the range [0.0,1.0] :
$R_D = Y_S^{\gamma-1} \times R_S$
$G_D = Y_S^{\gamma-1} \times G_S$
$B_D = Y_S^{\gamma-1} \times B_S$

The HLG OETF^{-1} function is :
$E = E'^2/3 \quad 0 \le E' \le 1/2$
$E = (\exp((E' - c)/a) + b)/12 \quad 1/2 < E' \le 1$

The HLG OOTF^{-1} fucntion is :
$Y_D = 0.2627 \times R_D + 0.6780 \times G_D + 0.0593 \times B_D$

$$R_S = \left(\frac{Y_D - \beta}{\alpha}\right)^{\frac{1-\gamma}{\gamma}} \times \left(\frac{R_D - \beta}{\alpha}\right)$$

$$G_S = \left(\frac{Y_D - \beta}{\alpha}\right)^{\frac{1-\gamma}{\gamma}} \times \left(\frac{G_D - \beta}{\alpha}\right)$$

$$B_S = \left(\frac{Y_D - \beta}{\alpha}\right)^{\frac{1-\gamma}{\gamma}} \times \left(\frac{B_D - \beta}{\alpha}\right)$$

Or, within the range [0.0,1.0] :

$$R_S = Y_D^{\frac{1-\gamma}{\gamma}} \times R_D$$

$$G_S = Y_D^{\frac{1-\gamma}{\gamma}} \times G_D$$

$$B_S = Y_D^{\frac{1-\gamma}{\gamma}} \times B_D$$

The Catch !

HLG is adaptative according L_W, and scalling to uniform/normalise must be done with F_D.
So, we have the directs mathematical paths :
$(R_S, G_S, B_S) \leftrightarrow (R', G', B')$
$(R_S, G_S, B_S) \leftrightarrow (R_D, G_D, B_D)$
But we don't have $(R_D, G_D, B_D) \leftrightarrow (R', G', B')$.
This, creates much more diffucult math paths than with PQ.

PQ to HLG :

$$Y=EOTF_{PQ}[E'] \rightarrow F_D = \min\left(\left(\frac{10000 \times Y}{L_W}\right), 1.0\right) \rightarrow E=OOTF_{HLG}^{-1}[F_D] \rightarrow E'=OETF_{HLG}[E]$$

HLG to PQ :

$$E=OETF_{HLG}^{-1}[E'] \rightarrow F_D=OOTF_{HLG}[E] \rightarrow F_D = \min\left(\left(\frac{E_D \times L_W}{10000}\right), 1.0\right) \rightarrow E'=EOTF_{PQ}^{-1}[F_D]$$

HLG to LinearRGB, output normalized 1.0=10000 cd/m² if $L_W \neq 10000$:

$E=OETF_{HLG}^{-1}[E'] \rightarrow F_D=OOTF_{HLG}[E]$ (with L_W = specified value) $\rightarrow$

$$F_D = \min\left(\left(\frac{E_D \times L_W}{10000}\right), 1.0\right) \rightarrow E=OOTF_{HLG}^{-1}[F_D] \text{ (with } L_W=10000)$$

LinearRGB, input normalized 1.0=10000 cd/m² to HLG if $L_W \neq 10000$:

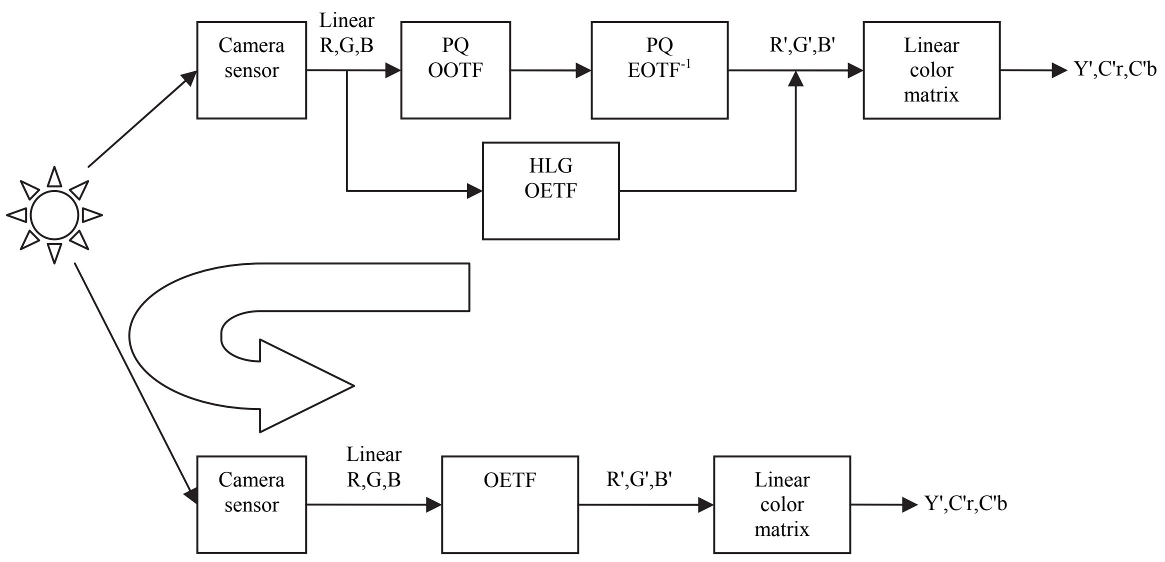
$$F_D=OOTF_{HLG}[E] \text{ (with } L_W=10000) \rightarrow F_D = \min\left(\left(\frac{E_D \times 10000}{L_W}\right), 1.0\right) \rightarrow$$

$E=OOTF_{HLG}^{-1}[F_D]$ (with L_W = specified value) $\rightarrow E'=OETF_{HLG}[E]$.

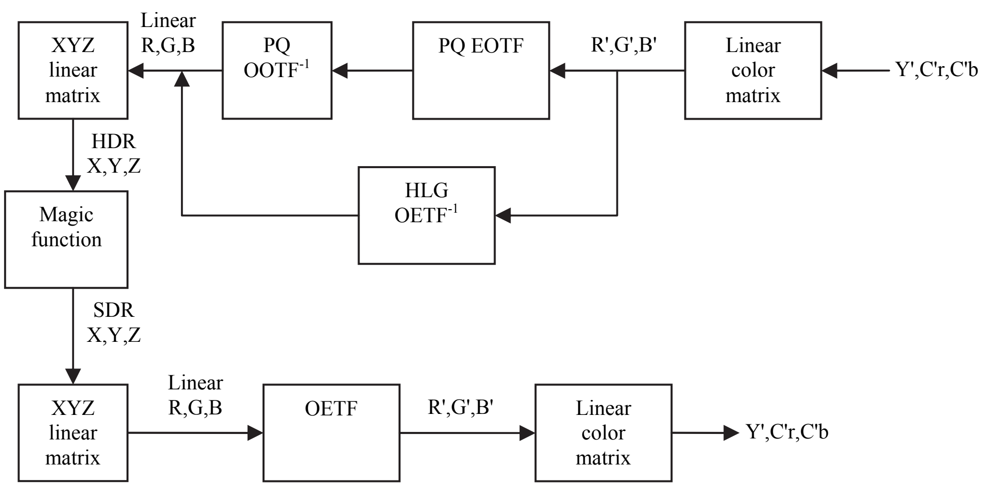
Also, the OOTF function is not independant by channel, preventing easy LUT implementation.

HDR to SDR

As explained, my idea is to try do the following :



So, the following :



As there is a 100 scale factor between HDR and SDR, my first test, out of curiosity for **Magic function** a simple scaling function.
Of course, again, if it was so simple, it would have been done since a long time.