

Gamma Correction

Galina Paskaleva

Institute of Computer Graphics and Algorithms

Vienna University of Technology



Radiance in the direction θ :

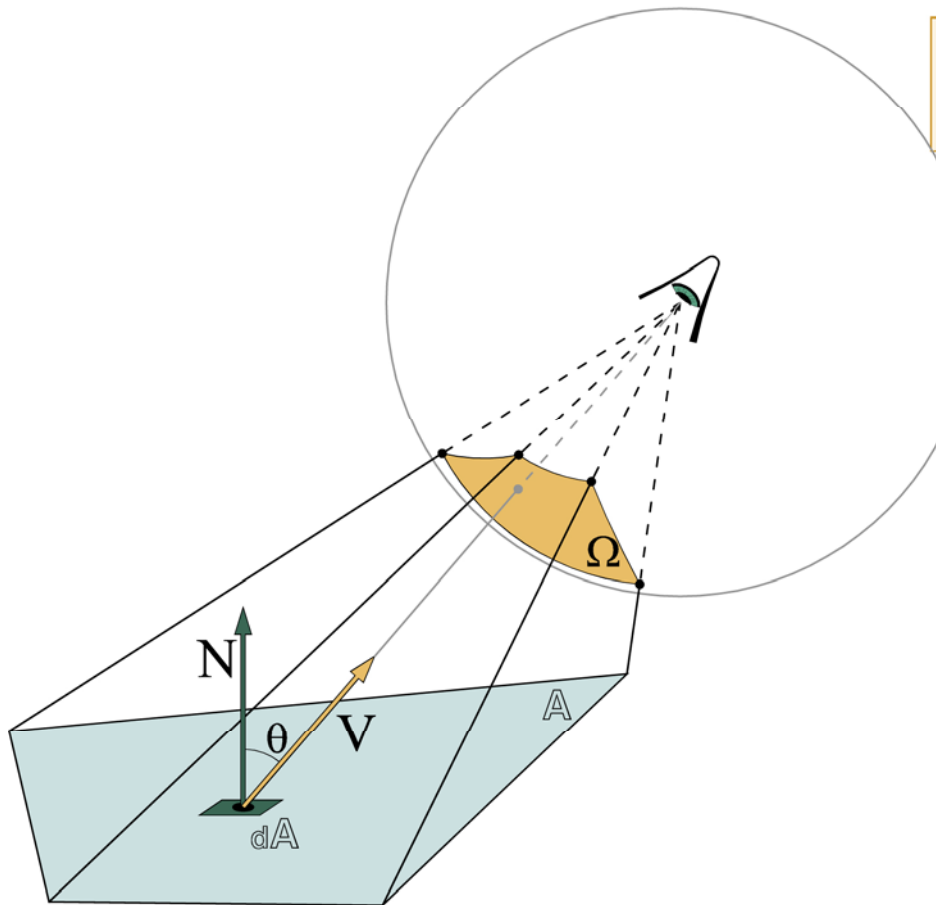
$$R(\theta) = \frac{d^2\Phi}{dA d\Omega \cos\theta} [\text{W}/\text{m}^2\text{sr}]$$

where:

Φ [W]... the total power of electro-magnetic radiation in all frequencies emitted by the surface w. area A ;

θ [1] ... the angle between the surface normal N and the viewing vector V ;

Ω [sr] ... the solid angle obtained by projecting the observed surface onto a unit sphere with center at the point from which the measurement is taken.



Radiance:

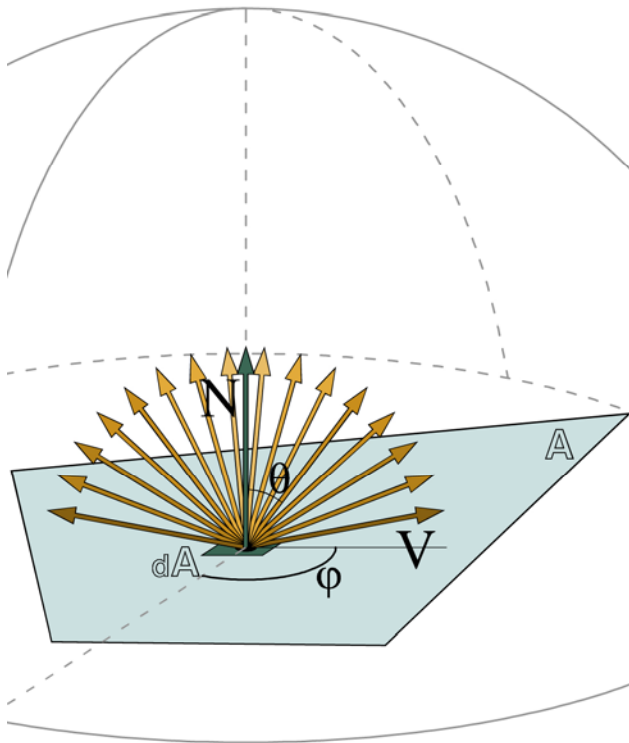
$$R = \int_{\varphi=0}^{2\pi} \int_{\theta=0}^{\pi/2} R(\theta) \cos\theta \sin\theta \, d\theta \, d\varphi \quad [\text{W/m}^2]$$

Spectral Radiance in the direction θ :

$$R(\lambda, \theta) = \frac{d^2\Phi(\lambda)}{dA d\Omega \cos\theta} \quad [\text{W/m}^2\text{sr}]$$

Spectral Radiance:

$$R(\lambda) = \int_{\varphi=0}^{2\pi} \int_{\theta=0}^{\pi/2} R(\lambda, \theta) \cos\theta \sin\theta \, d\theta \, d\varphi \quad [\text{W/m}^2]$$



Luminosity for monochromatic light:

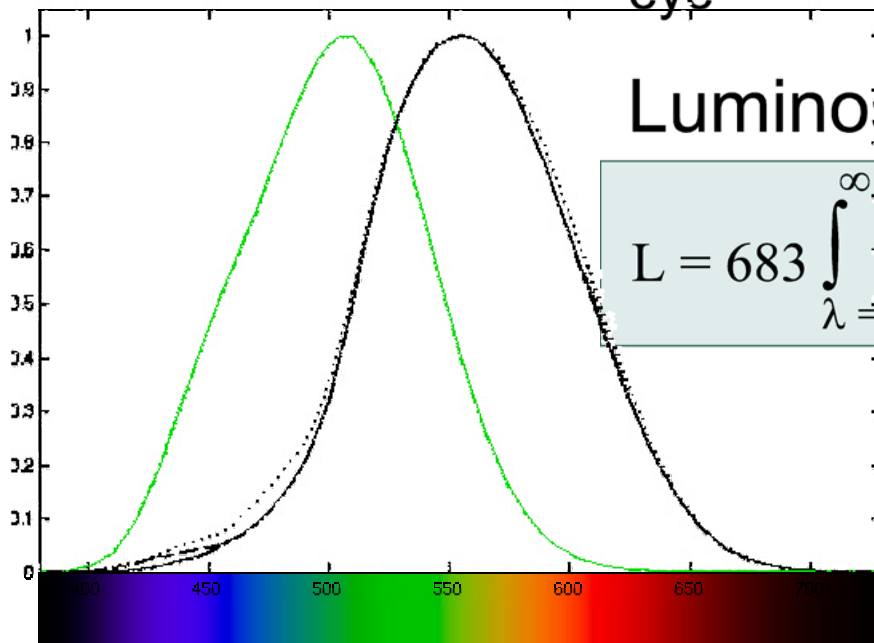
$$L(\lambda) = 683 f(\lambda) R(\lambda) \quad [\text{cd/m}^2]$$

where:

$f(\lambda)$... the standard luminosity function, describing the average sensitivity of the human eye

Luminosity:

$$L = 683 \int_{\lambda=0}^{\infty} f(\lambda) R(\lambda) d\lambda \quad [\text{cd/m}^2]$$



- describes the relation between
 - ◆ Radiance (R) ~ Intensity (I) and
 - ◆ Luminosity (L) ~ Brightness (B)
 - ◆ by approximating the integral

$$L = 683 \int_{\lambda=0}^{\infty} f(\lambda) R(\lambda) d\lambda \quad [\text{cd/m}^2]$$

as

$$dB/B = \gamma \cdot dI / I \quad \rightarrow \quad B = c \cdot I^{\gamma}$$

- ◆ in a PC $\gamma = 0.45$

- ◆ in a Mac $\gamma = 0.55$



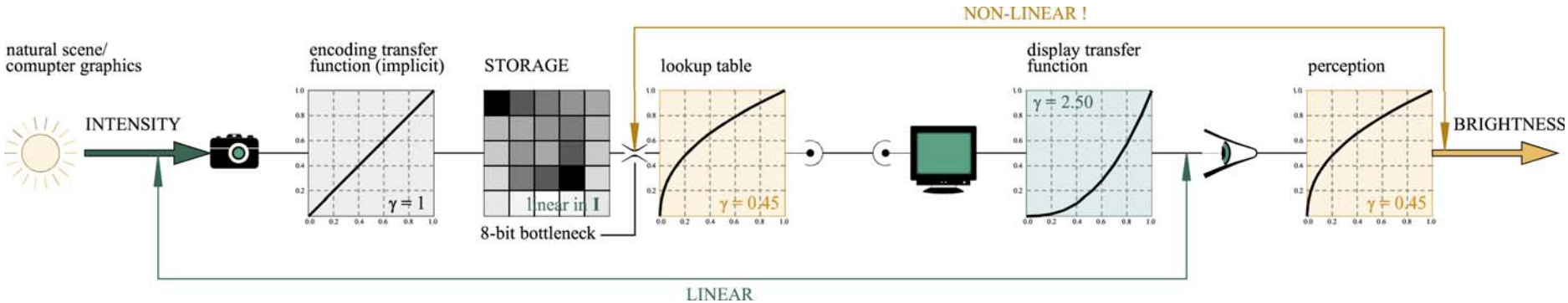
- CRT has non-linear response to the input signal, described as the relation between
 - ◆ Voltage (**V**) and
 - ◆ Intensity (**I**)

$$I = a \cdot V^{\gamma} + b$$

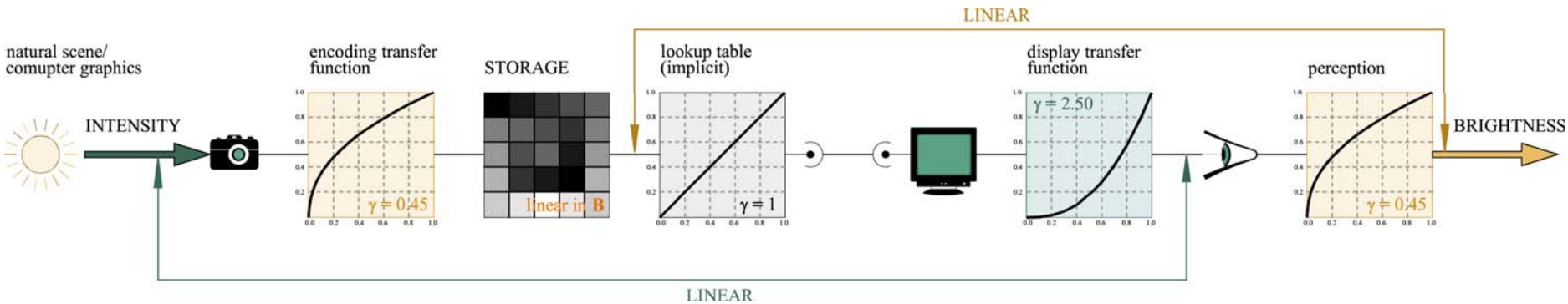
where $\gamma = 2.5$

- ◆ incidentally $0.45 \times 2.5 \approx 1.0$





- storage linear in intensity (practically unused)



- storage linear in brightness (most common)



linear in Brightness



linear in Intensity



0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0



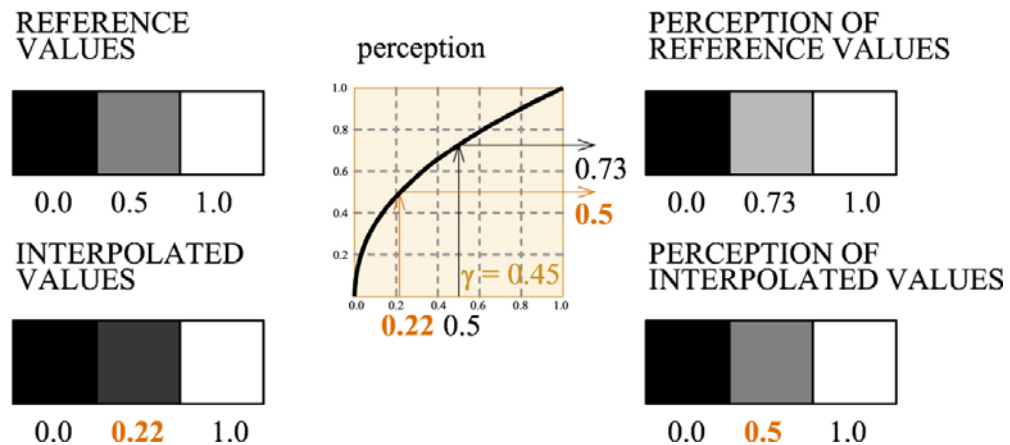
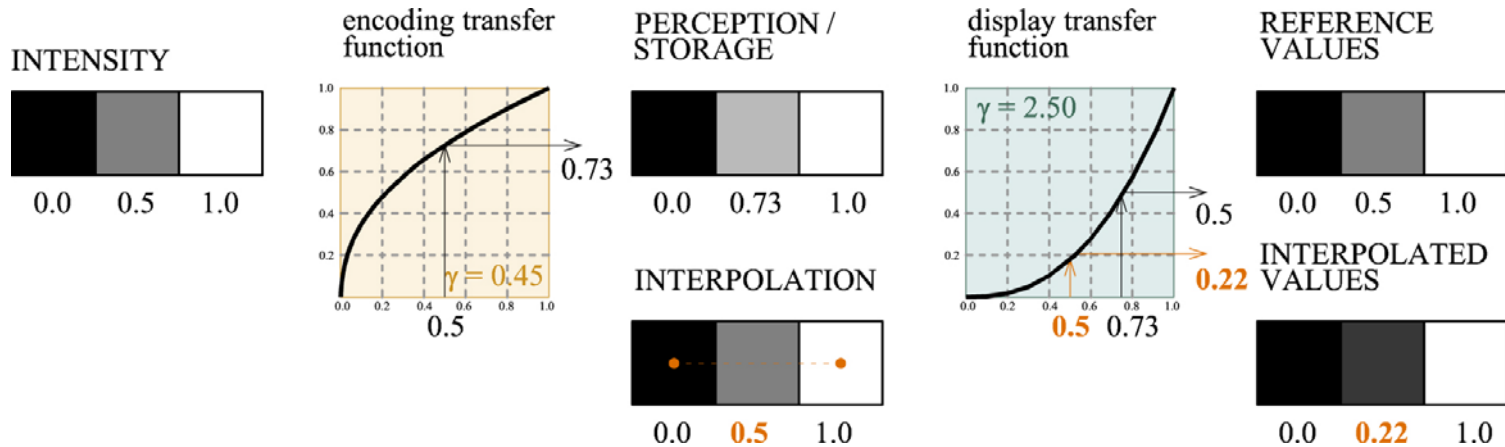
- a linear transformation is characterized by:
 - ◆ $f(x + y) = f(x) + f(y)$ and
 - ◆ $f(\lambda x) = \lambda f(x)$
- light transport is linear
- the contributions of two light sources in a scene are **summed** in the renderer to produce the correct **illumination effect**
- the contribution of a light source is **multiplied** by reflectance coefficients in the renderer to produce the correct **material behaviour**



- If an image looks good in a web browser, it has, in all probability, already been gamma-corrected
- Alpha-channels, Height or Displacement Maps, on the other hand, are not
- JPEG files are pre-corrected for a $\gamma = 2.2$
 - ◆ I.e. they are linear in brightness but non-linear in intensity
 - ◆ lets interpolate btw. the values 0.0 and 1.0
 - we want to achieve a brightness of 0.5 ...



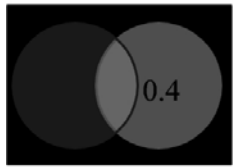
Interpolation w/o Gamma Correction



Lighting w/o Gamma Correction

INTENSITY CONTRIBUTION LIGHT 1

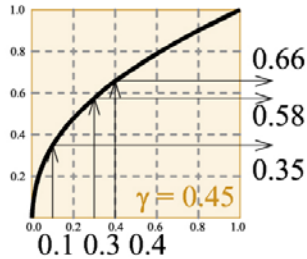
0.3



0.1

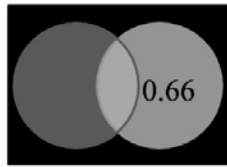
INTENSITY CONTRIBUTION LIGHT 2

encoding transfer function



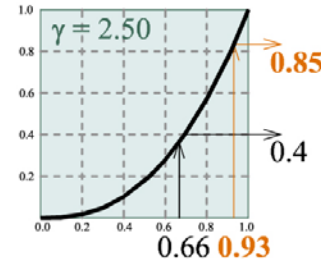
PERCEPTION / STORAGE

0.58



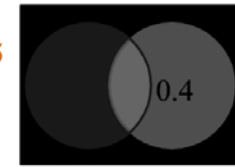
0.35

display transfer function



REFERENCE VALUES

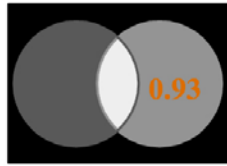
0.3



0.1

SUM

0.58



0.35

SUMMED VALUES

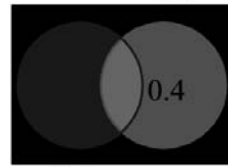
0.3



0.1

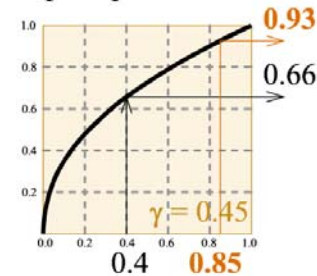
REFERENCE VALUES

0.3



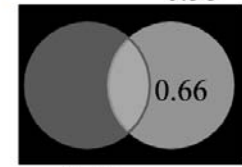
0.1

perception



PERCEPTION OF REFERENCE VALUES

0.58



0.35

SUMMED VALUES

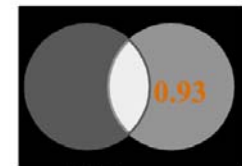
0.3



0.1

PERCEPTION OF SUMMED VALUES

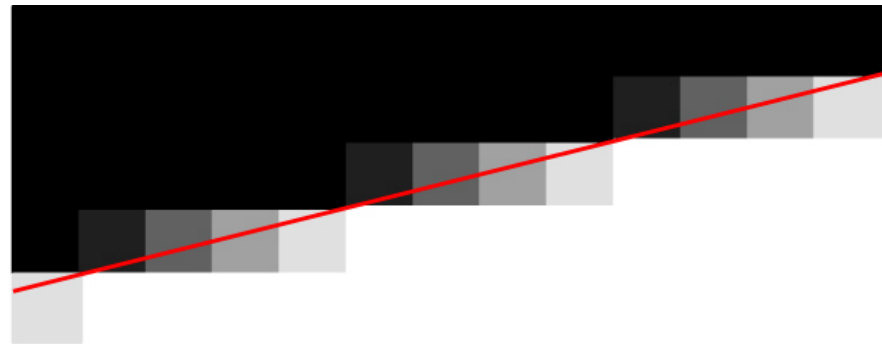
0.58



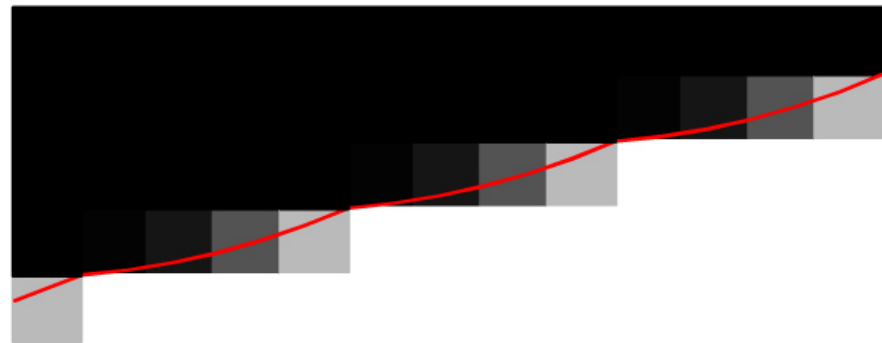
0.35



- “Bending” of anti-aliased edges due to the non-linearity of brightness space
 - ◆ interpolated in intensity space



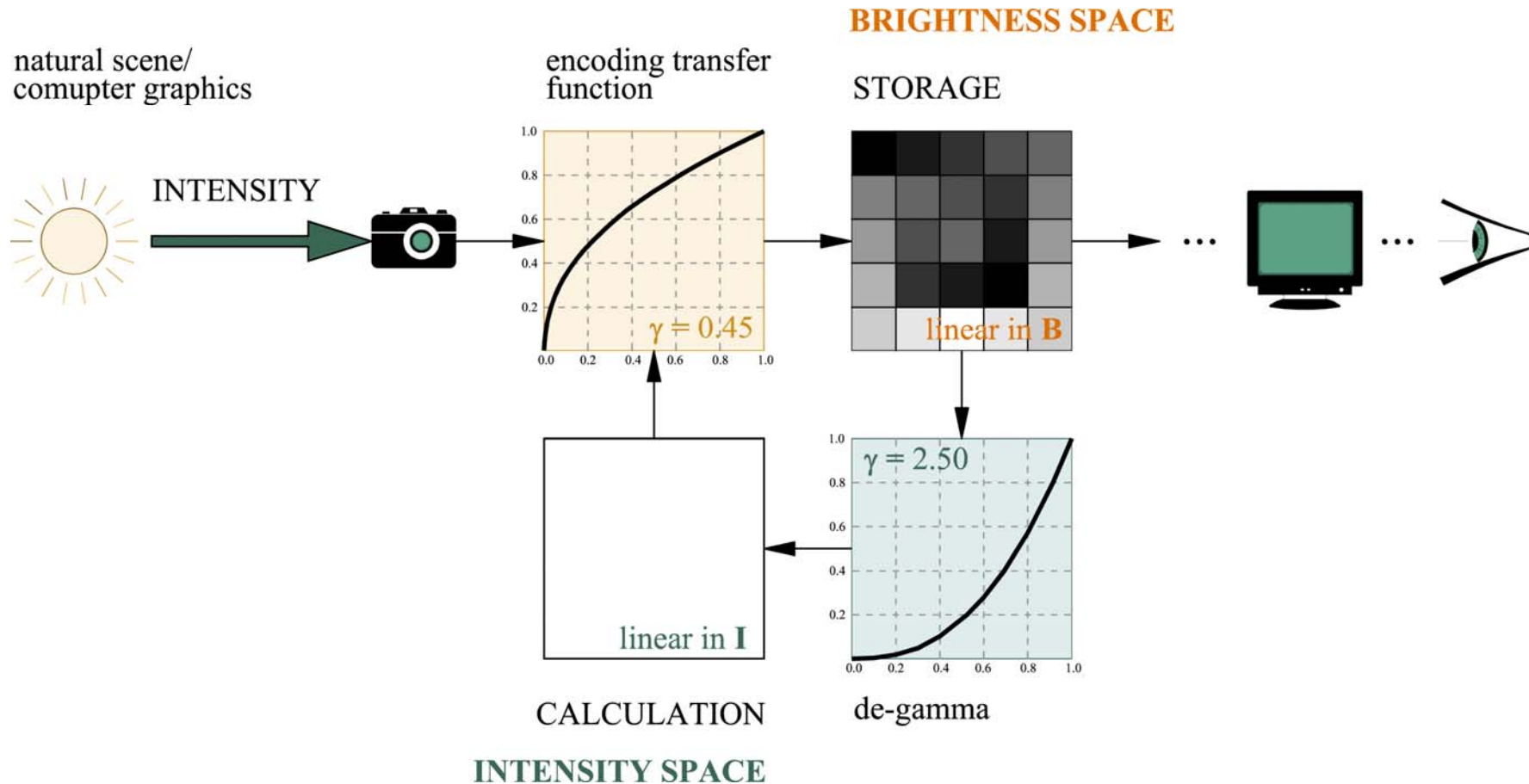
- ◆ interpolated in brightness space





- gamma-corrected / not gamma-corrected

Store and Calculate in different spaces



- sRGB texture format with $\gamma = 2.2$
 - ◆ proper gamma is applied by the **HW** before the results of texture fetches are used for shading
 - **can** be done manually
 - ◆ all samples used in a texture filtering operation are linearized by the **HW** before filtering
 - **cannot** be done manually
 - ◆ SRGB8 and SRGB8_ALPHA8 are stored as 8-bit unsigned fixed-point values



- final re-application of the gamma correction by the **HW** before display on the monitor
- any value returned in the Shader is gamma-corrected by the **HW** before storage in the frame buffer (or render-to-texture buffer)
- blending is automatically performed in linear intensity space and the result is also automatically gamma-corrected
 - ◆ Alpha values are not affected
- use sRGB intermediate color buffers



■ visible in the Application:

```
glActiveTexture(GL_TEXTURE1);  
glBindTexture(GL_ARB_TEXTURE_SRGB, texture);  
  
glEnable(GL_ARB_FRAMEBUFFER_SRGB);  
glBindFramebuffer(GL_DRAW_FRAMEBUFFER, fbuffer);
```

■ no influence in the Shader:

```
in vec2 ex_texCoord;  
uniform sampler2D un_texture;  
vec4 color = texture2D(un_texture, texCoord);
```



■ visible in the Shader:

◆ on fetch:

```
vec4 color = pow(texture2D(un_texture,  
    texCoord), 2.2);
```

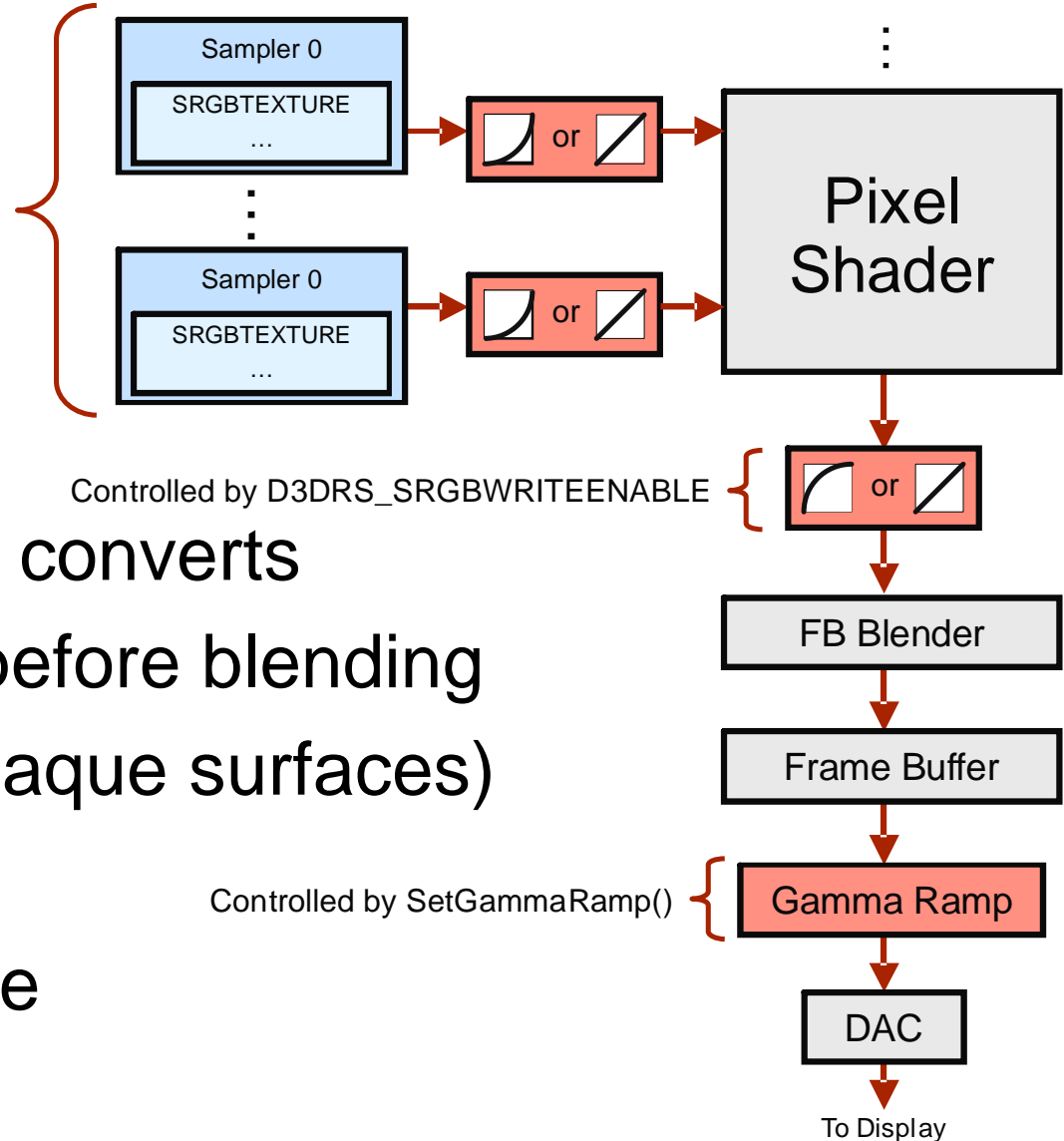
◆ on last-stage output:

```
return float4(pow(finalCol, 1.0 / 2.2),  
    pixelAlpha);
```

◆ **filtering** and **mipmapping** is performed in **non-linear** space!



Texture Samplers



- **DirectX 9** incorrectly converts into non-linear space before blending (not problematic for opaque surfaces)
- **DirectX 10** performs blending in linear space



- [1] Tomas Akenine-Möller, Eric Haines, Naty Hoffman, „Real-Time Rendering“, A. K. Peters, Ltd., Wellesley, MA 02482, 2008.
- [2] http://en.wikipedia.org/wiki/Gamma_correction
- [3] <http://www.poynton.com/GammaFAQ.html>
- [4] OpenGL Version 4.1 Core Profile Specification

