

HDR em Jogos

Alessandro Ribeiro da Silva

asilva@dcc.ufmg.br

Wallace Santos Lages

wsl@dcc.ufmg.br

Sumário

- A Indústria de Jogos
- O Sistema Visual Humano
- Formatos de imagens HDR
- Algoritmos de tone mapping
- Conclusões

A indústria

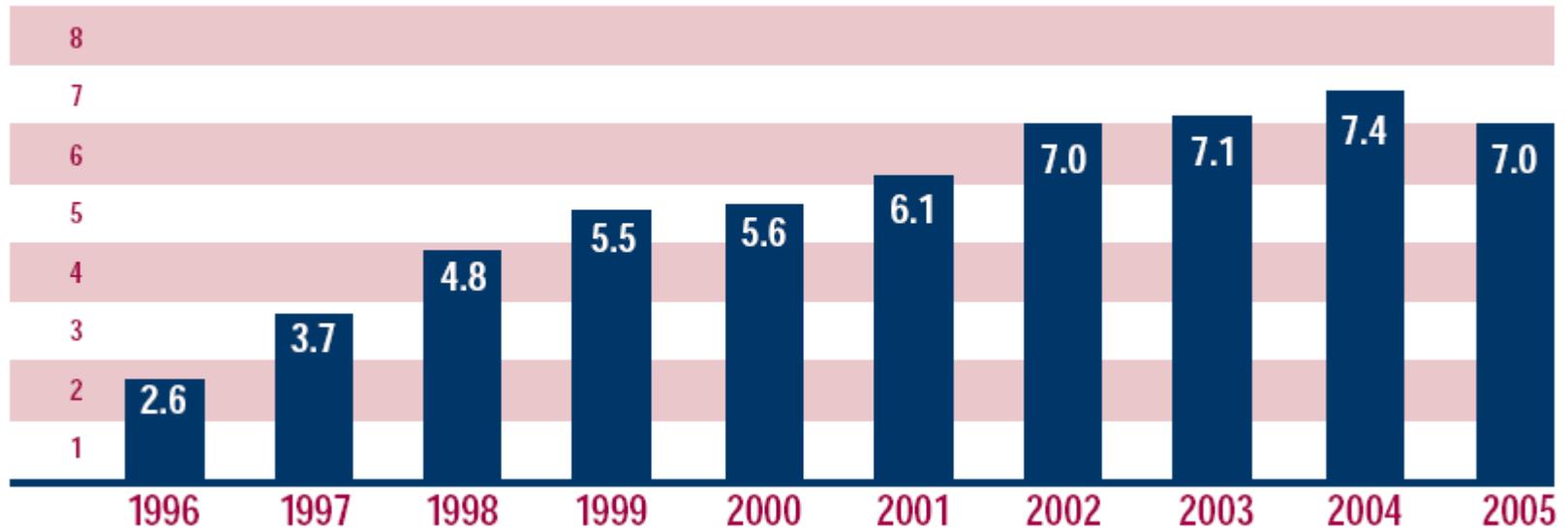
Fatos*:

- Idade média do jogador americano: 33 anos
- 69% dos chefes de família jogam jogos
- 38% são mulheres

*Entertainment Software Association, 2006

A indústria

U.S. COMPUTER AND VIDEO GAME DOLLAR SALES GROWTH DOLLARS IN BILLIONS

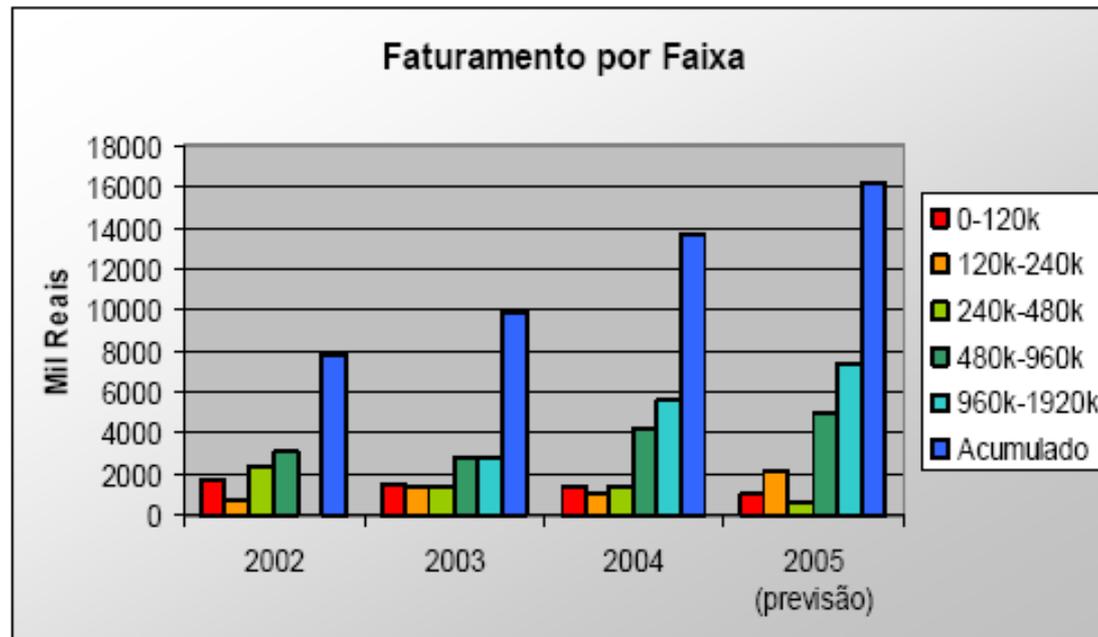


Source: The NPD Group / Point-of-Sale Information

No Brasil ?

Estimado em torno de 100 milhões de reais
Prejuízos de pirataria estimados em 94%

(IDG Consulting)

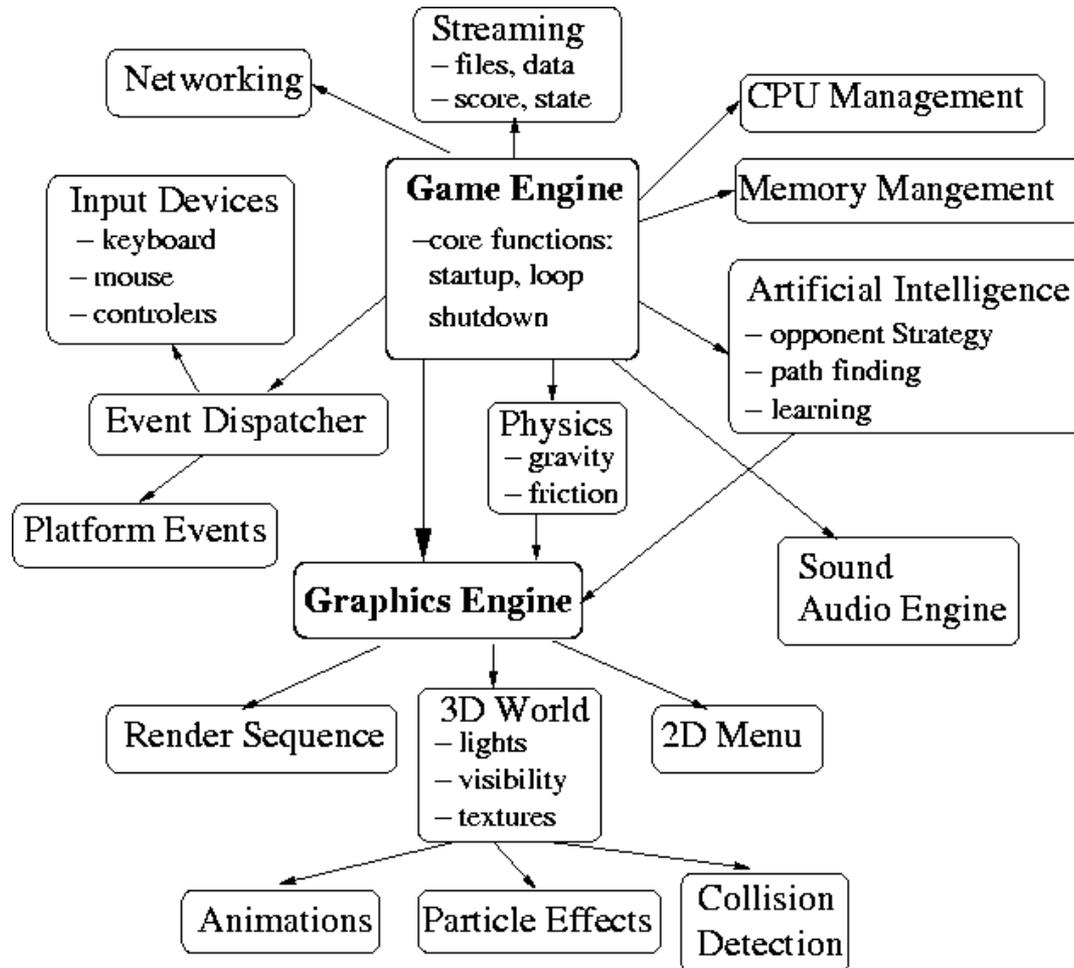


A Indústria de Desenvolvimento de Jogos
Eletrônicos no Brasil Abragames-2005

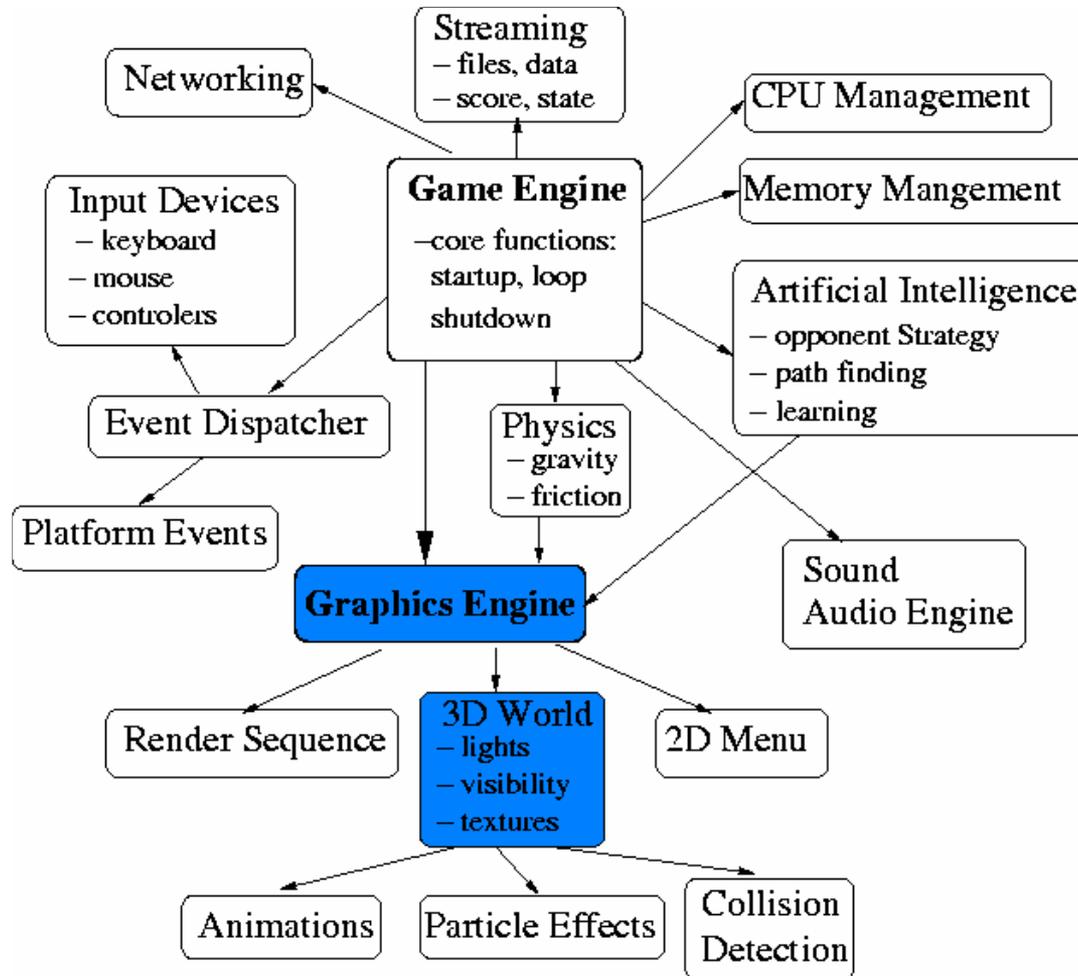
Complexidade

- Computação Gráfica
- Redes
- Simulação física
- Inteligência Artificial

Por dentro do jogo



Por dentro do jogo

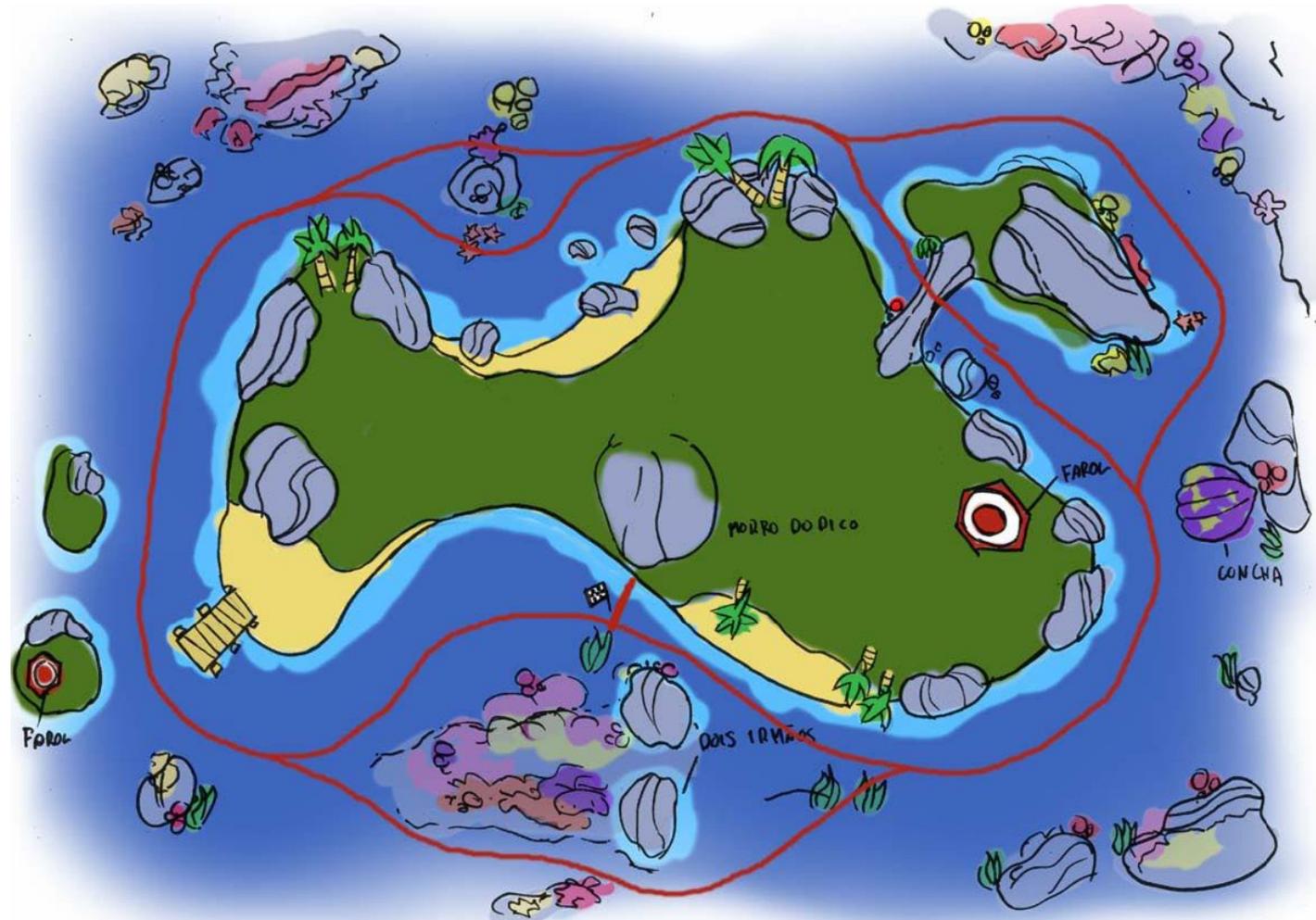


Produção

Peixis!
A disputa subaquática!



Fernando de Noronha



Personagens



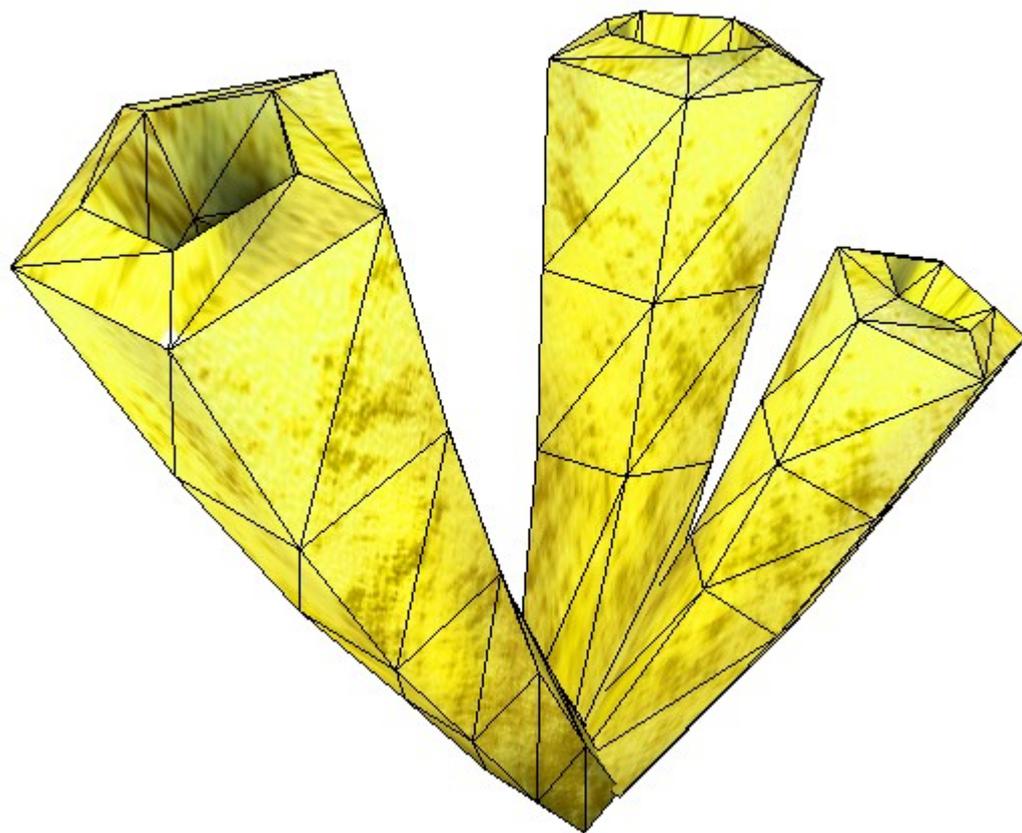
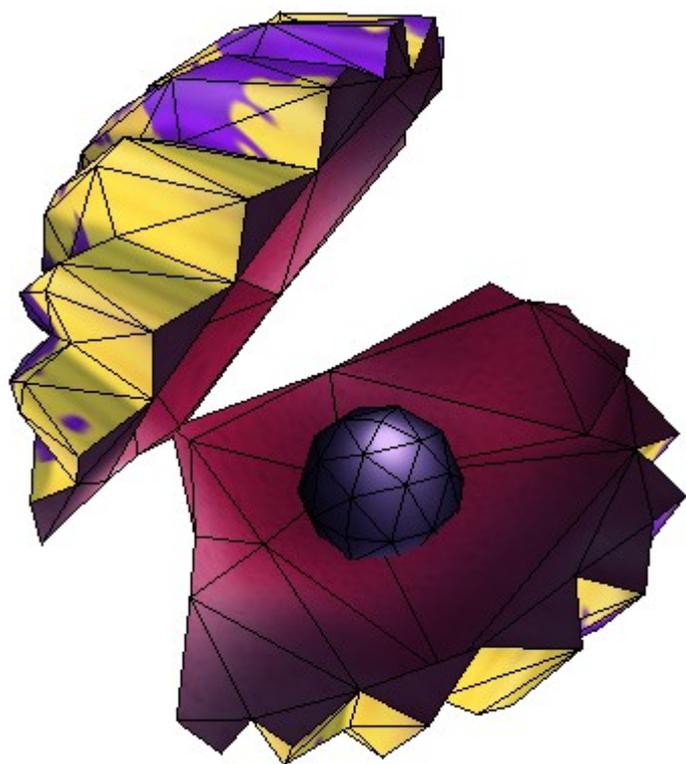
Leo Torpedo



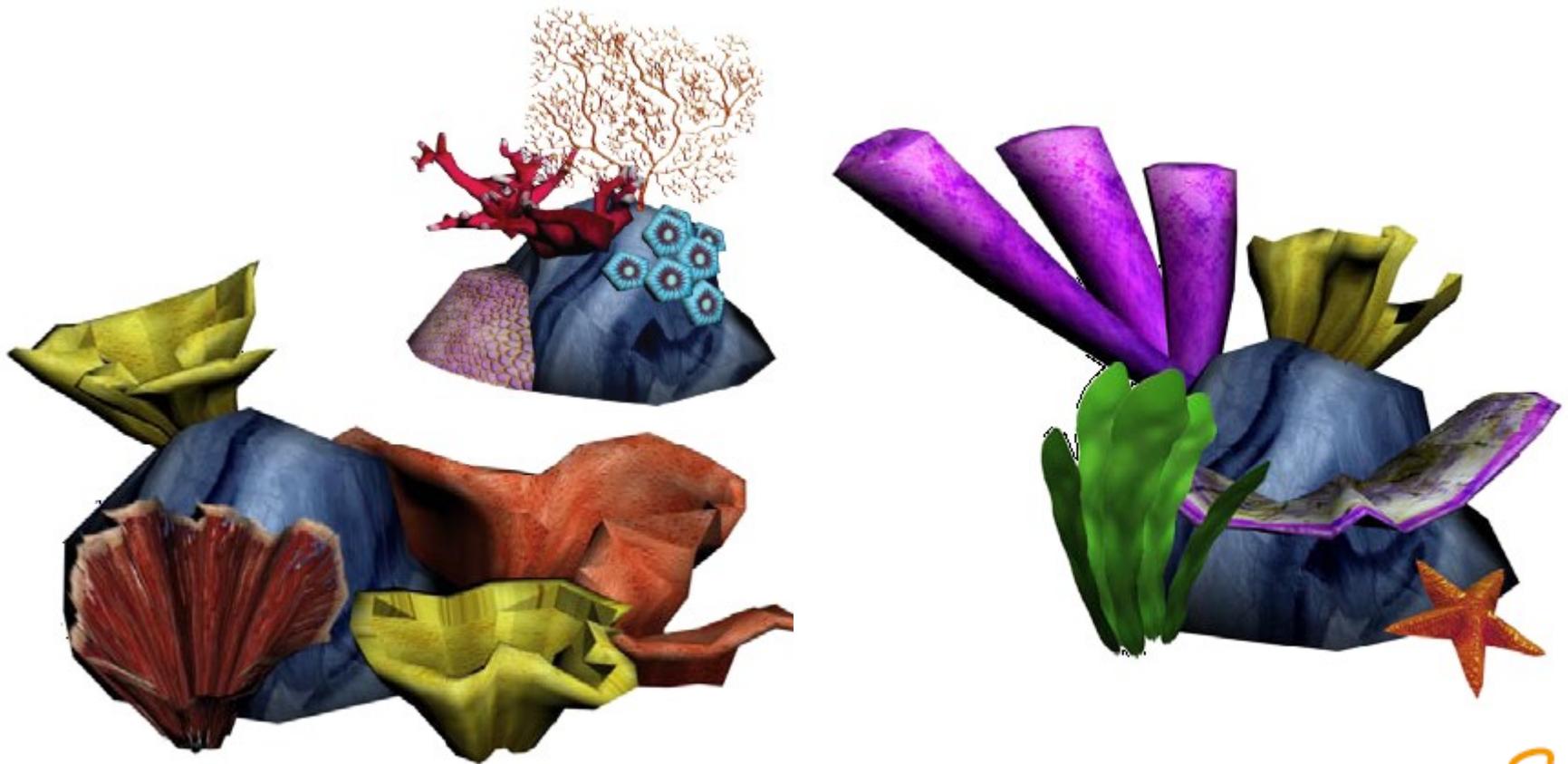
Peixis!
A disputa subaquática!



Objetos



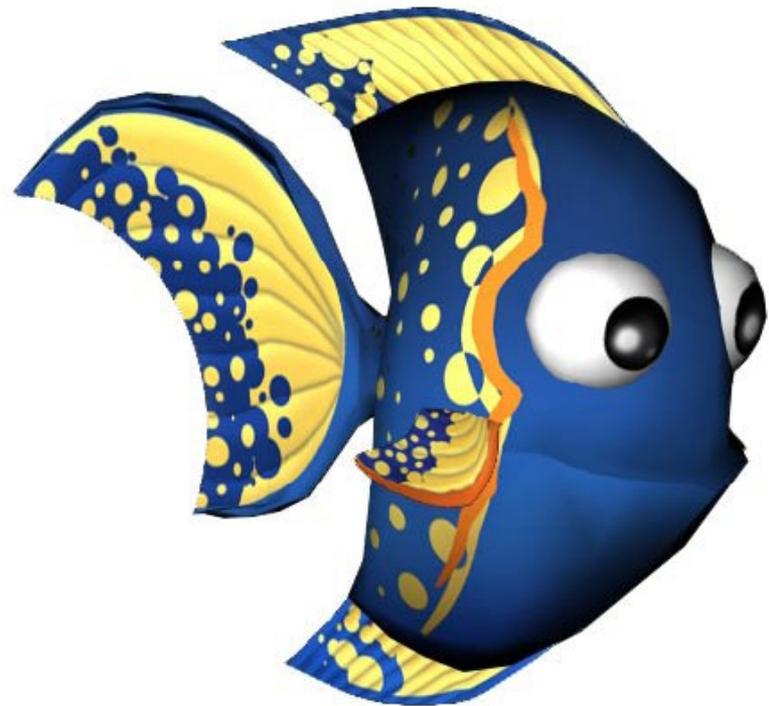
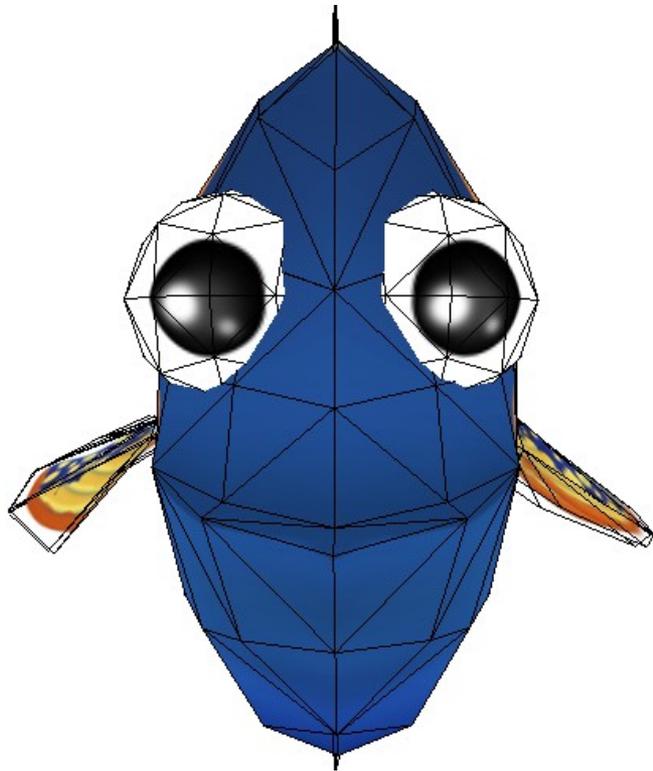
Corais



Peixis!
A disputa subaquática!



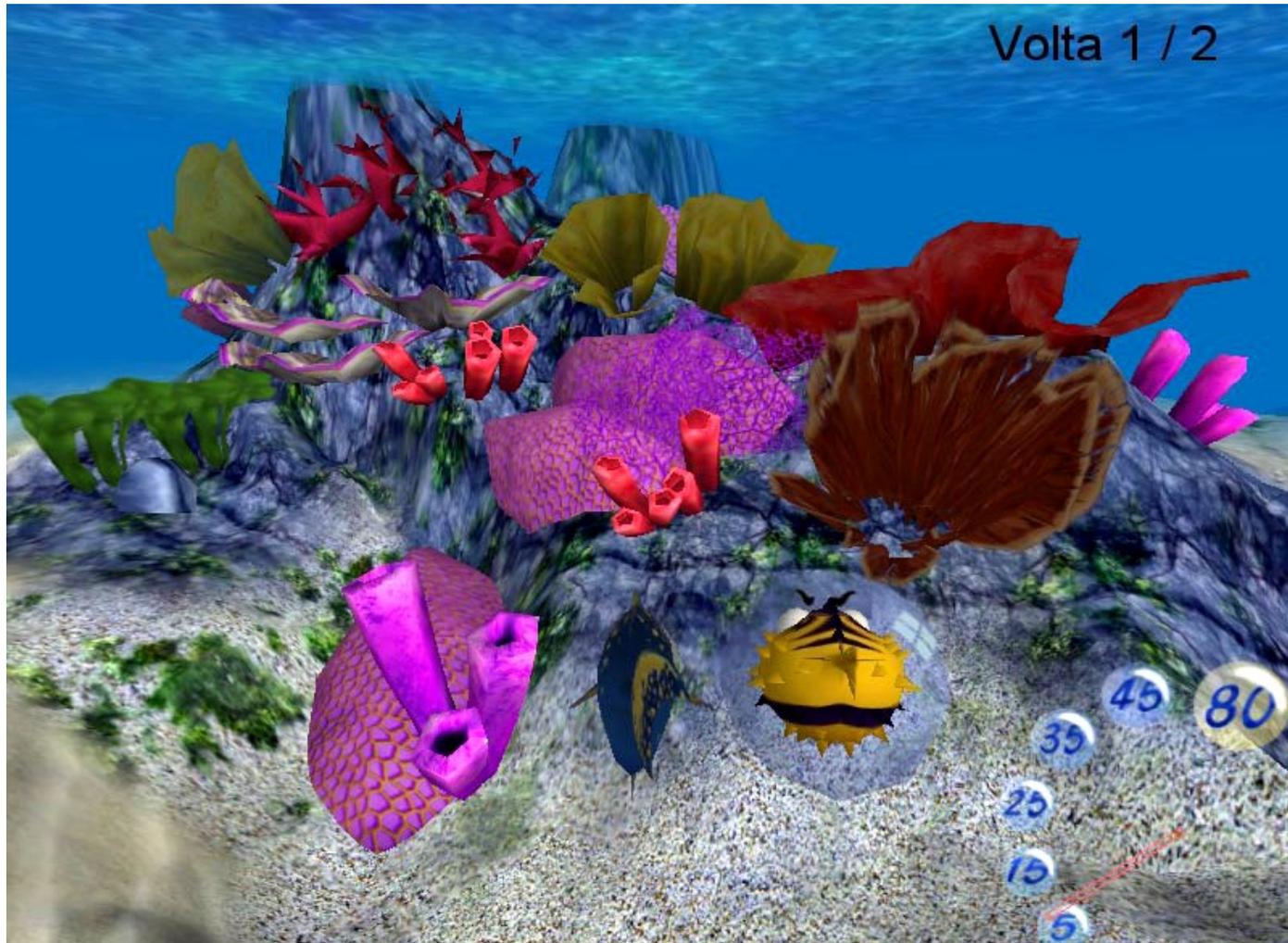
Leo Torpedo



Peixis!
A disputa subaquática!



Resultado Final



Resultado Final



Desafio

- Representar adequadamente a impressão da cena desejada



Nem sempre o fisicamente correto é percebido como correto!

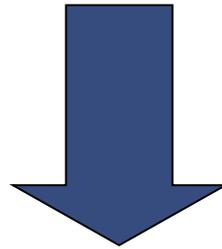
Desafio

- Um bom sistema de reprodução deveria ser capaz de reproduzir cenas escuras e cenas brilhantes.
- Entretanto isso é difícil:
 - Os dispositivos de saída são lineares e possuem um alcance dinâmico pequeno.
 - O sistema visual humano possui peculiaridades

Exemplo

- Murmúrio vs show de rock.
 - A diferença de $10,000,000,000 = 100 \text{ dB}$
- noite sem lua vs dia claro
 - Diferença de $1:1,000,000$ ou 90dB

Idéia

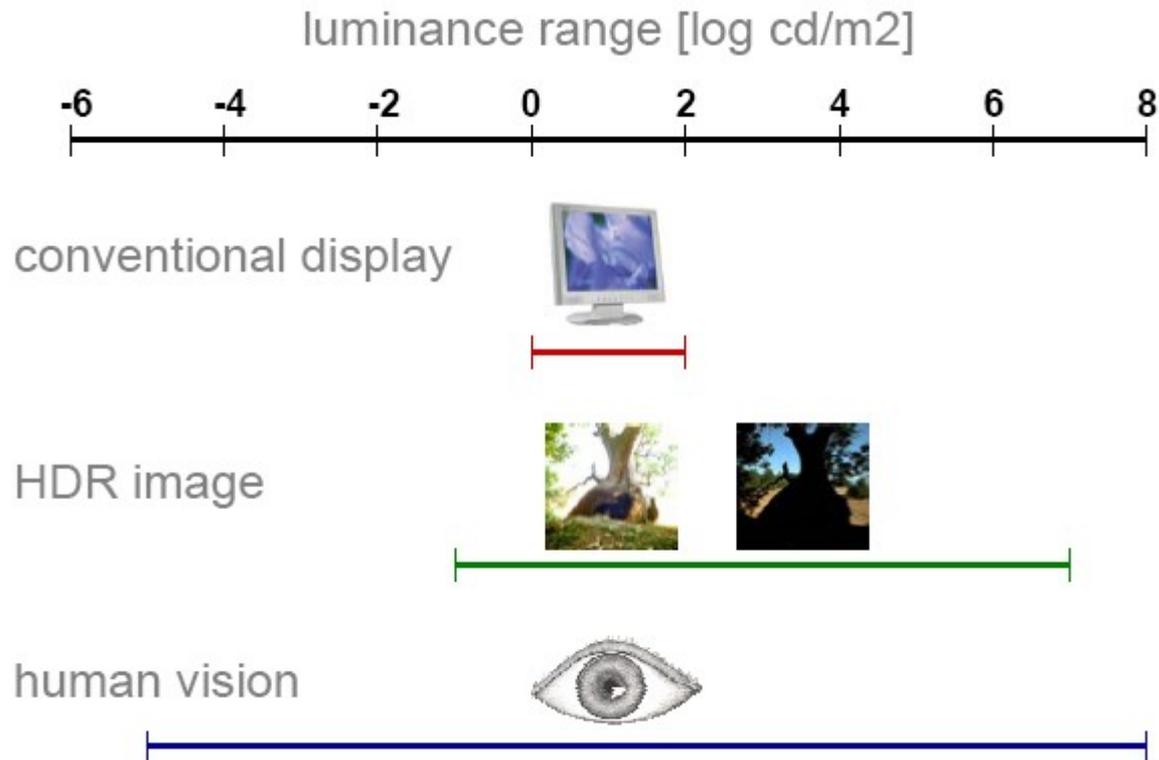


Representar os valores de intensidade medidos ou sintéticos de maneira absoluta ao invés de uma razão da maior intensidade possível.

High Dynamic Range

- Técnicas para obtenção e exibição de imagens com alto alcance dinâmico.
- Alcance dinâmico: Razão entre o menor e o maior valor possíveis de uma grandeza

Luminâncias reproduzidas

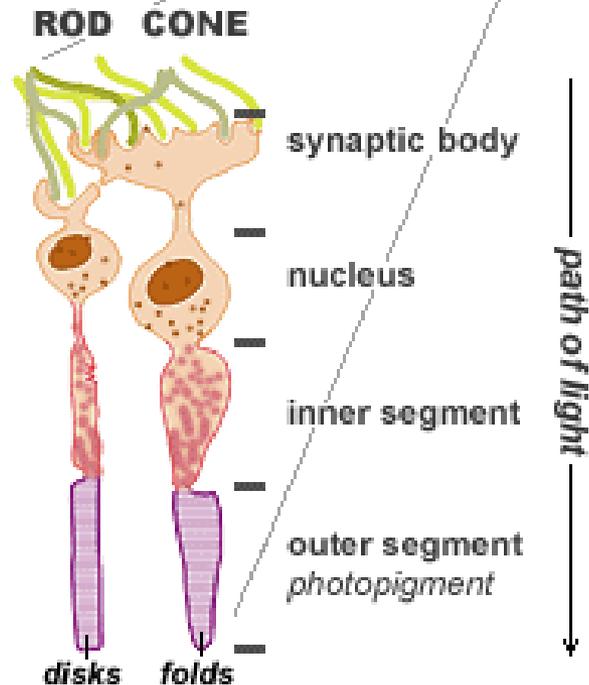
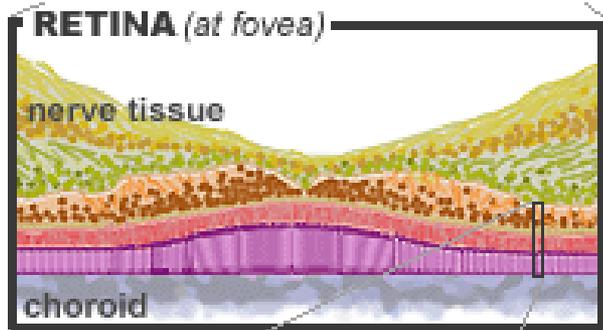


De: A computational model of lightness perception for HDR Imaging
Krawczyk et Al.

Codificando imagens HDR

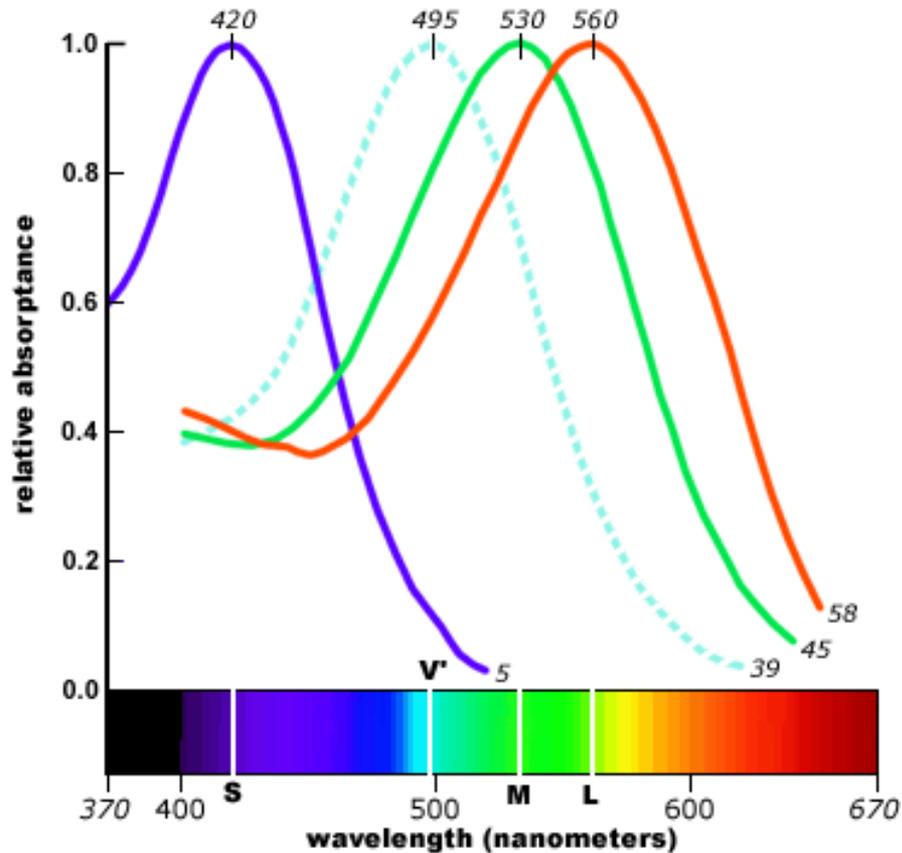
- Imagens HDR -> scene-referred
 - Luminâncias do mundo real
- Imagens convencionais -> device referred
 - Cores que devem ser reproduzidas no monitor

O Sistema Visual Humano



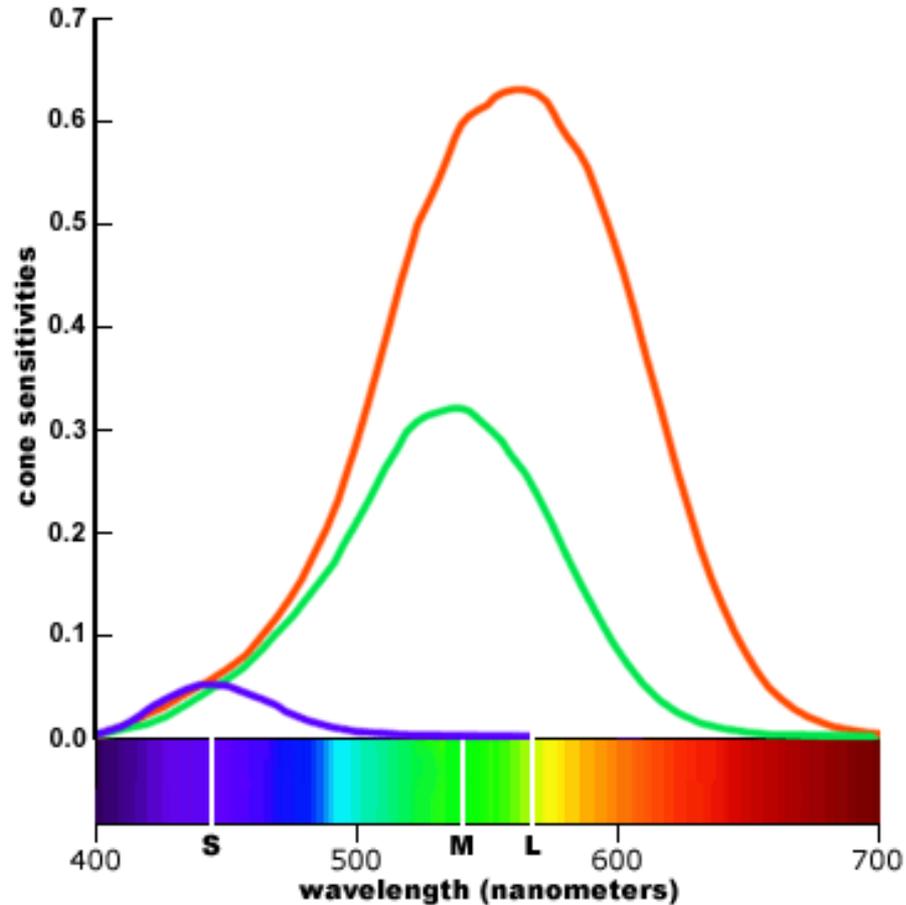
- Quatro tipos de células principais
 - Três tipos de cones e 1 bastonete
 - 100 milhões bastonetes
 - 6 milhões de cones

Curva de eficiência



Curvas de eficiência normalizada dos foto-pigmentos
De: Dartnall, Bowmaker & Mollon (1983)

Curva de eficiência

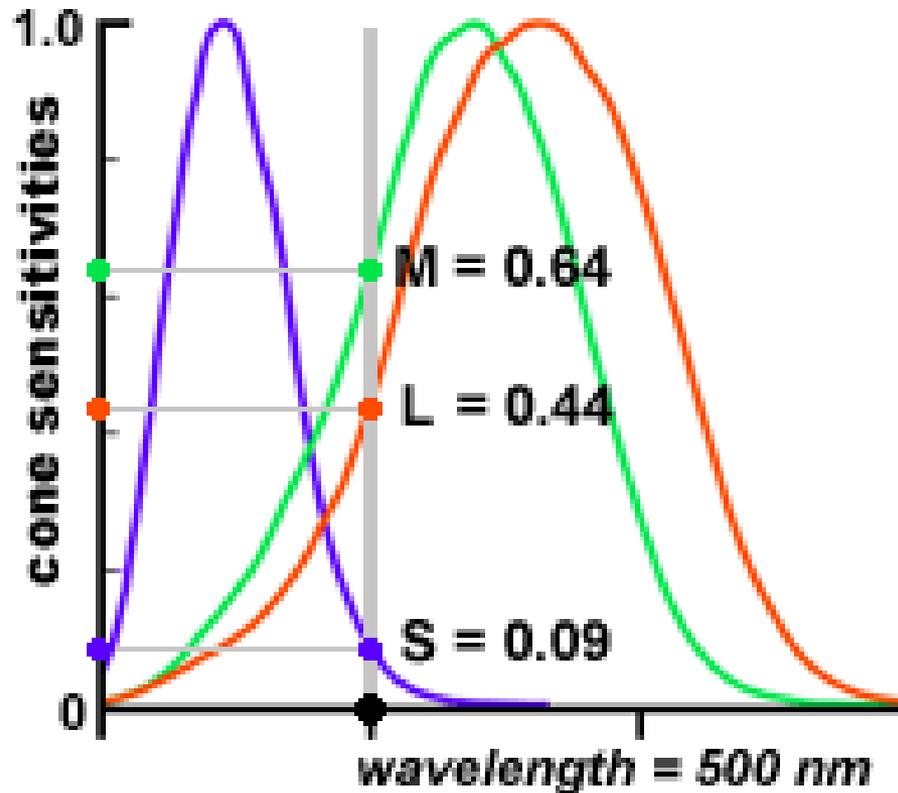


Sensitividade dos cones * proporção das populações de L,S e M Stockman & Sharpe (2000)

Problema

- Um cone sozinho não distingue cor de luminância
- Como é possível ver diferentes cores e brilhos ?
 - As curvas devem se sobrepor no espectro
 - As respostas dos cones devem ser independentes.

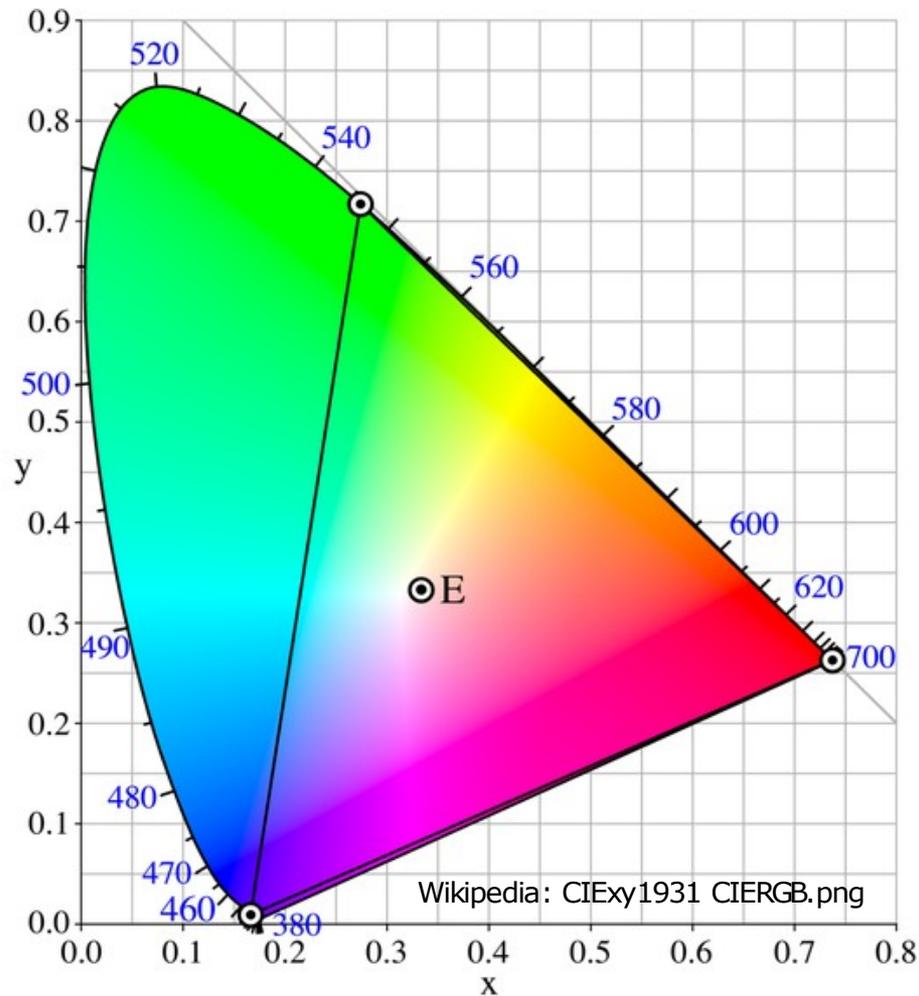
A percepção da cor



$$\text{brightness (B)} = L + M + S$$

$$\text{chromaticity (C)} = L/B, M/B, S/B.$$

Espaços de cor



Quantização

- Interessante utilizar uma codificação não linear, devido a percepção do SVH
- Relação entre tensão e intensidade luminosa no CRT:

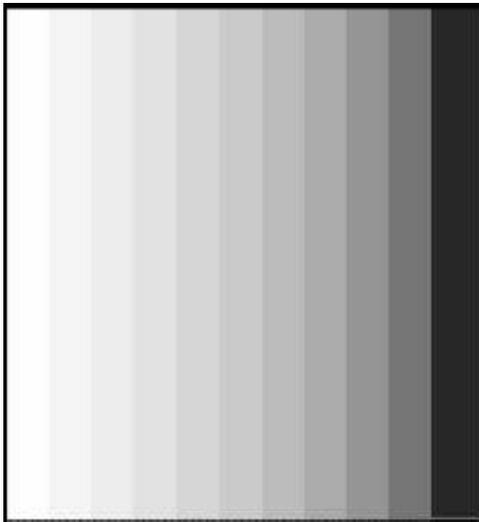
$$I = \text{alfa} (V + \text{epsilon})^{\text{gamma}}$$

- Os gammas variam entre 2.3 e 2.6

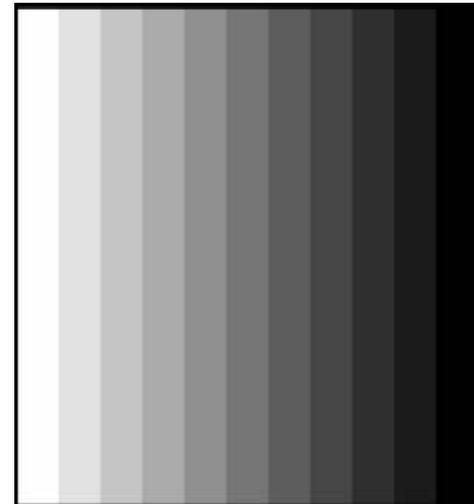
Quantização

- Passo iguais na luminância codificada correspondem a brilhos iguais subjetivamente.

Intervalos iguais de
luminância
($\Delta Y = \text{const.}$)



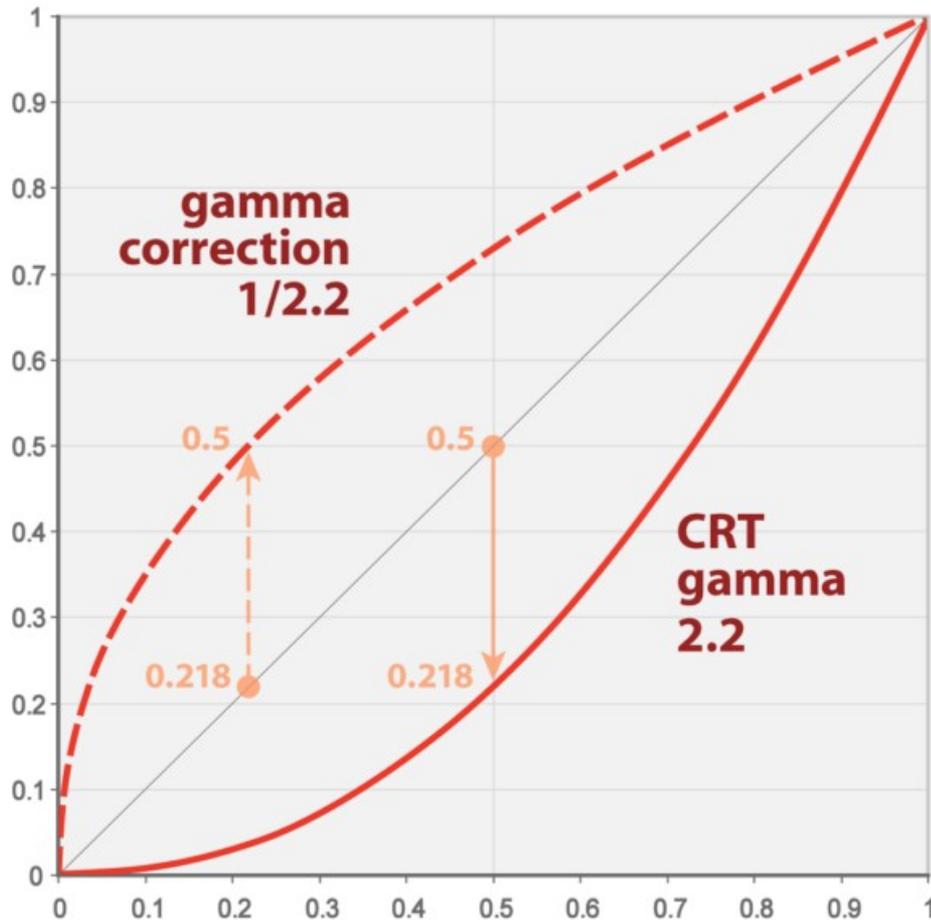
Intervalos iguais de brilho
($\Delta L = \text{const.}$)



Quantização

- Se aplicamos uma quantização linear nos valores de cor, percebemos mais claramente os passos na regiões mais escuras.
- Utilizando uma codificação de potência, a quantização se torna menos perceptível..

Gama



Fotos já utilizam codificação Gamma, produzindo uma imagem adequada para visualização

Computação gráfica utiliza relações lineares então as quantidades devem ser corrigidas com gama.

Resumindo

- O Sistema Visual Humano
 - É mais sensível a variações locais
 - É mais sensível a luminância relativa
 - Percebe mais detalhes nas áreas escuras

Codificações HDR

- Maior número de bits devido a codificação linear e a necessidade de representar todos os valores de luminância visíveis.
- Normalmente ponto flutuante.



Formatos

- Pixar Log Encoding (TIFF)
- Radiance RGBE Encoding (HDR)
- SGI LogLuv (TIFF)
- ILM OpenEXR (EXR)
- Microsoft/HP scRGB Encoding

Pixar Log Encoding (TIFF)

- Primeiro padrão HDR. Criado pela Divisão de computação gráfica da LucasFilm, agora Pixar
- Filme, resposta logarítmica -> capaz de armazenar uma alcance maior que um CRT.
- TIFF com codificação logarítmica, implementado como um codec para a biblioteca TIFF de Sam Leffera
- Armazenado em três canais, mas com codificação logarítmica de 11 bits ao invés da representação gama de 8 bits. Capaz de codificar 3.6 ordem de magnitudes com passos de 1%

Radiance RGBE Encoding

- Saída do sistema de renderização RADIANCE do Lawrence Berkley Laboratory
- Representação de 4 bytes com três mantissas de 8 bits compartilhadas com um expoente de 8 bits
- Precisão absoluta de 1% com um alcance de 76 ordens de magnitude!
- Distribuição do erro não é uniforme perceptualmente

SGI LogLuv (TIFF)

- Agora baseado na percepção visual, com passos de quantização adequados para a percepção humana de contraste e cor.
- Canais de luminância e cromaticidade separados, com codificação log na luminância. Similar com a codificação YCC mas com alcance dinâmico
 - 10 bits de luminância com tabela CIE 14 bits = 24 bits
 - 16 bits de luminância = 38 ordens de magnitude e passos de 0,3 %
 - 16 bits de luminância + 8 bits para cada CIE = 32 bits/pixel com todas as cores visíveis

ILM OpenEXR (EXR)

- Industrial Light and Magic, Código C++ disponível, 2002
- Encapsulamento para o formato de 16 bits derivado do IEEE-754 e adotado nos buffers Nvidia e ATI
- Pode representar valores negativos e portanto todo o gamut visível com um alcance de 10.7 ordens de magnitude e passos de 0.1%.
- Suporte para canais extra de alfa, profundidade ou amostragem espectral.

Microsoft/HP scRGB Encoding

- Extensão da codificação sRGB de 24 bits para 16 bits por cor primária com codificação linear ou 12 por primária com codificação gama.
- A codificação linear gasta a precisão no lugar errado, dos valores baixo de luminância. Apenas 3.5 ordens de magnitude. Permite primárias negativas mas foge do gamut visível na parte superior.
- 36 bits/pixel com gama ramp with a linear subsection near zero. Com 25% menos bits consegue 3.2 ordens de magnitude e também permite primárias negativas.

Comparação de formatos

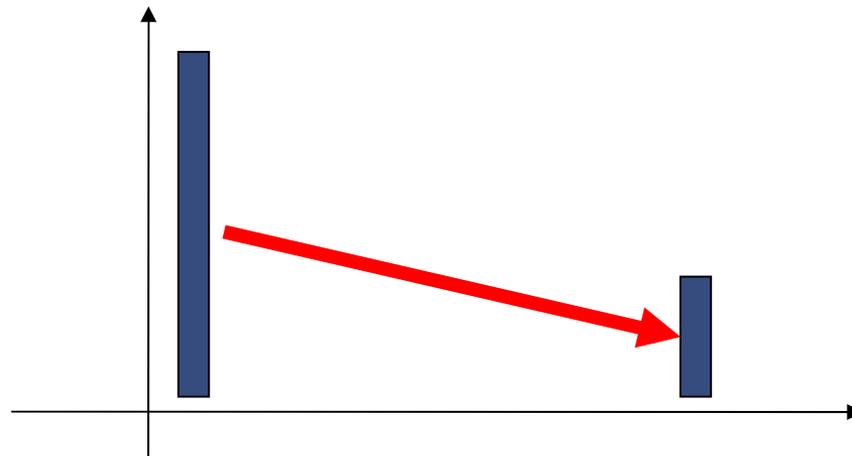
| Encoding | Covers Gamut | Bits / pixel | Dynamic Range | Quant. Step |
|----------------------|---------------------|---------------------|------------------------------|--------------------|
| sRGB | No | 24 | 1.6 (1.0:0.025) | Variable |
| Pixar Log | No | 33 | 3.8 (25.0:0.004) | 0.4% |
| RGBE XYZE | No Yes | 32 | 76 (1038:10 ⁻³⁸) | 1% |
| LogLuv 24 | Yes | 24 | 4.8 (15.9:0.00025) | 1.1% |
| LogLuv 32 | Yes | 32 | 38 (1019:10 ⁻²⁰) | 0.3% |
| EXR | Yes | 48 | 10.7 (65000:0.0000012) | 0.1% |
| scRGB | Yes | 48 | 3.5 (7.5:0.0023) | Variable |
| scRGB-nl scYCC-nl | Yes Yes | 36 | 3.2 (6.2:0.0039) | Variable |

Fontes

- Renderizadores fotométricos
 - Iluminação de cenas reais utilizando unidades reais ([watts/steradian/m2](#))
- Combinação de fotografias em diferentes exposições
 - Paul Debevec, SIGGRAPH 1997 presented his paper entitled

Tone mapping

- Técnica para aproximar a aparência de imagens HDR ($\sim 10000:1$) em meios com alcance mais limitado ($200:1$).
 - Ex: Revistas, monitores, projetores real image



Tone mapping

- Efeitos especiais
- Maximizar o número de detalhes
- Maximizar o contraste
- Imitar a percepção real

Tone Mapping

- **Tone Reproduction Curve**
 - (single scale/ global)
- **Tone Reproduction Operator**
 - (multi scale / local)

Tone Reproduction Curve

- Idéia: Manipular as distribuições de pixels definindo uma função que mapeia as intensidades originais no conjunto final.

Mapeamentos da forma:

$$T(I) = s * I^p \text{ ou } T(I) = s * \log(I)$$

- Comprime o alcance dinâmico mas não preserva as intensidades relativas

Tone Reproduction Curve

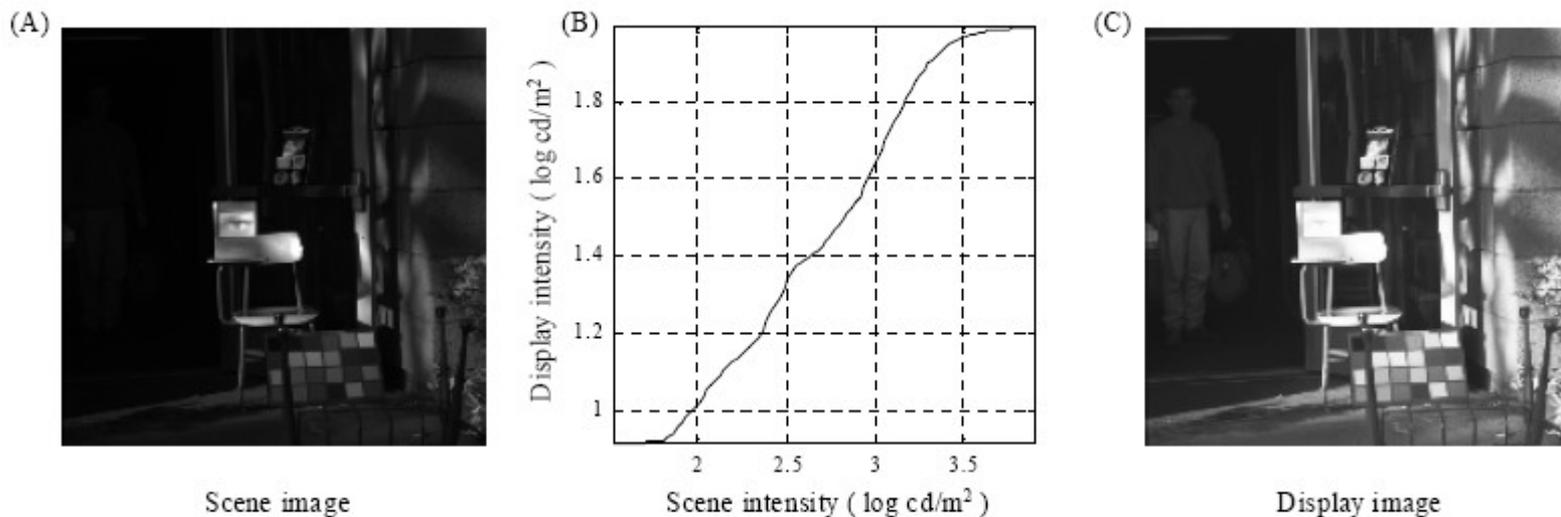


Figure 3. A tone reproduction curve is used to reduce dynamic range. (A) The original image. (B) The tone reproduction curve. TRCs are one-to-one and monotonic mappings. (C) The resulting image after the dynamic range has been compressed.

De: **Rendering high dynamic range images**, Jeffrey M. DiCarlo*^a and Brian A. Wandell

Tone Reproduction Curve

- TRC dependentes da imagem

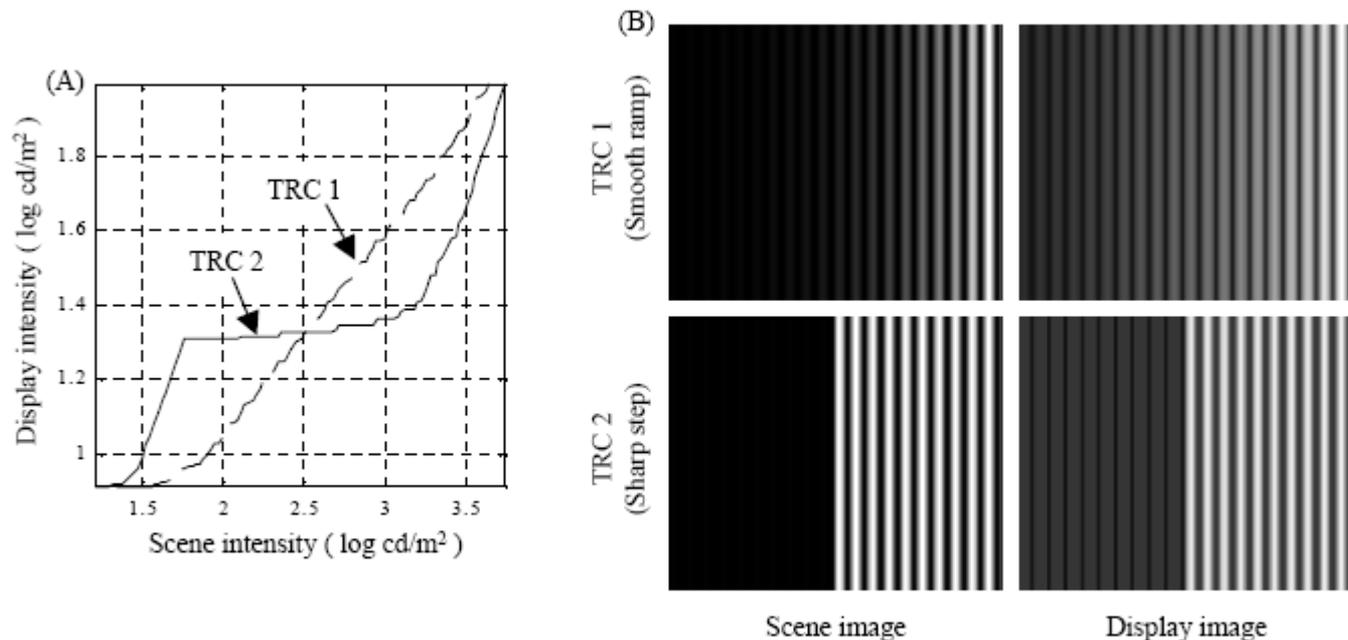


Figure 4. Larson's algorithm is applied to two scene images shown on the right. (A) The TRCs computed for the two images. (B) The scene and display images corresponding to the TRCs. See text for details.

Tone Reproduction Curve

- Trabalhos:
 - **Tone reproduction for realistic images** (J. Tumblin and H. Rushmeier, 1993)
 - **“A visibility matching tone reproduction operator for high dynamic range scenes”**- G. W. Larson, H. Rushmeier and C. Piatko, 1997
 - Incorpora modelos de sensibilidade a cores, contraste, ofuscamento e acuidade espacial

Tone Reproduction Curve

- Problema?

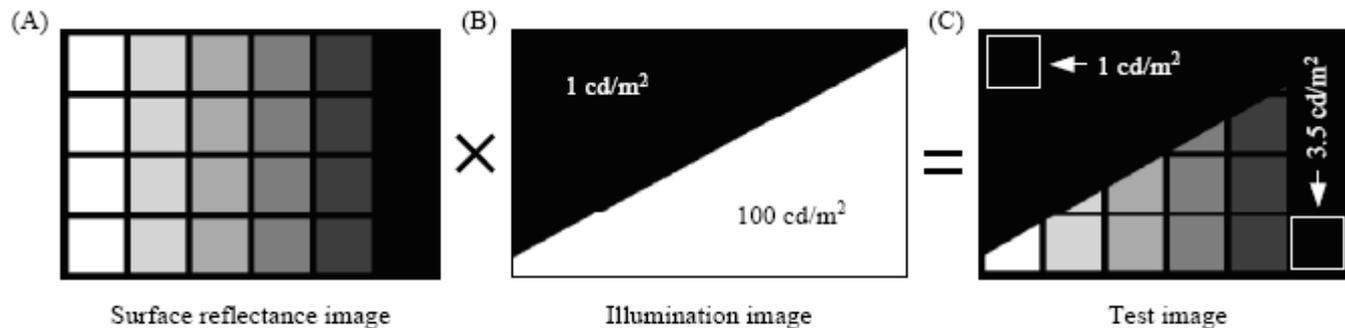


Figure 5. A test image with two spatially distinct illumination regions. (A) The surface reflectance image. (B) The illumination image. (C) The test image. The luminance of the white surface in the upper left corner is 1 cd/m^2 . The luminance of the black surface in the lower right corner is 3.5 cd/m^2 .

Tone Reproduction Operator

- Idéia: Manipular espacialmente os valores de pixels vizinhos
- Permitir que pixels de mesma intensidade sejam mapeados para valores diferentes dependendo do contexto.
- Decompor a imagem em variações de iluminação e variações de reflectância. Compactar a iluminação.

Tone Reproduction Operator

Multiresolution TRO

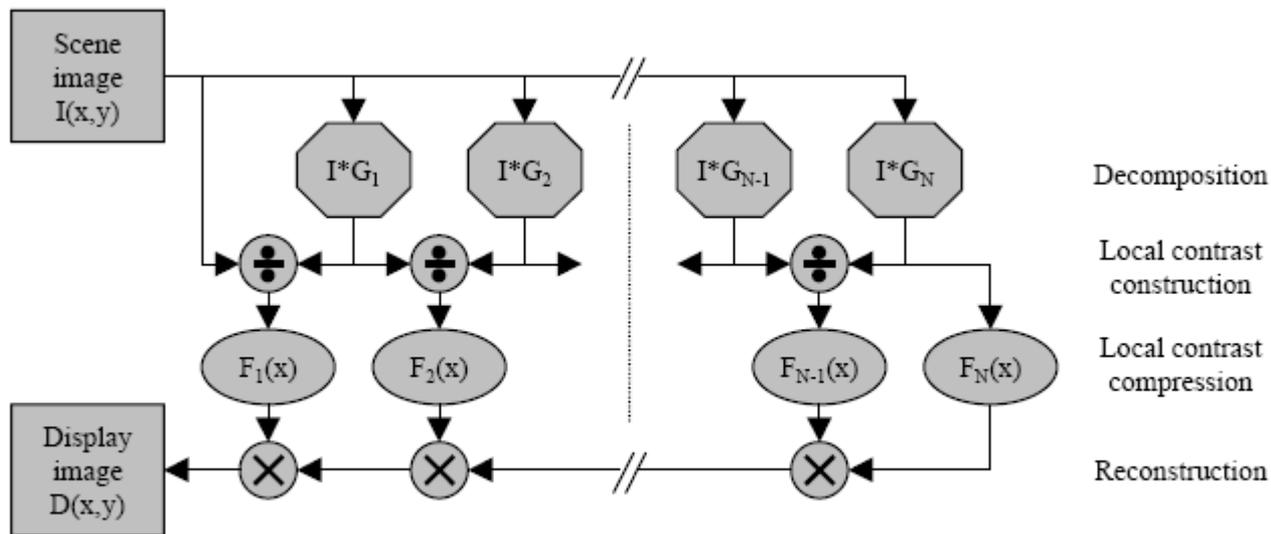


Figure 7. The basic multiresolution decomposition used by TRO algorithms. I is the scene image. G_1, G_2, G_{N-1}, G_N are Gaussian filters of increasing spatial size. The "*" denotes the convolution operator. $F_1(x), F_2(x), F_{N-1}(x), F_N(x)$ are the compression functions.

Tone Reproduction Operator

- Trabalhos

- **Spatially nonuniform scaling functions for high contrast image**
K. Chiu, M. Herf, P. Shirley, S. Swamy, C. Wang and K. Zimmerman, 1993
- **Fast bilateral filtering for the display of high-dynamic-range images",**
F. Durand and J. Dorsey, 2002

Tone Reproduction Operator

- Problema:
 - Assume variações locais como variações de reflectância.
 - Falha em sombras produzindo artefatos.

Algoritmos recentes

- *Lightness Perception in Tone Reproduction*
- Decomposição da imagem HDR em áreas e cálculo local dos valores e iluminação.
- A iluminação final líquida é calculada fundindo as áreas proporcionalmente a sua influência.
- Não afeta o contraste local e preserva as cores naturais.

Tone mapping methods

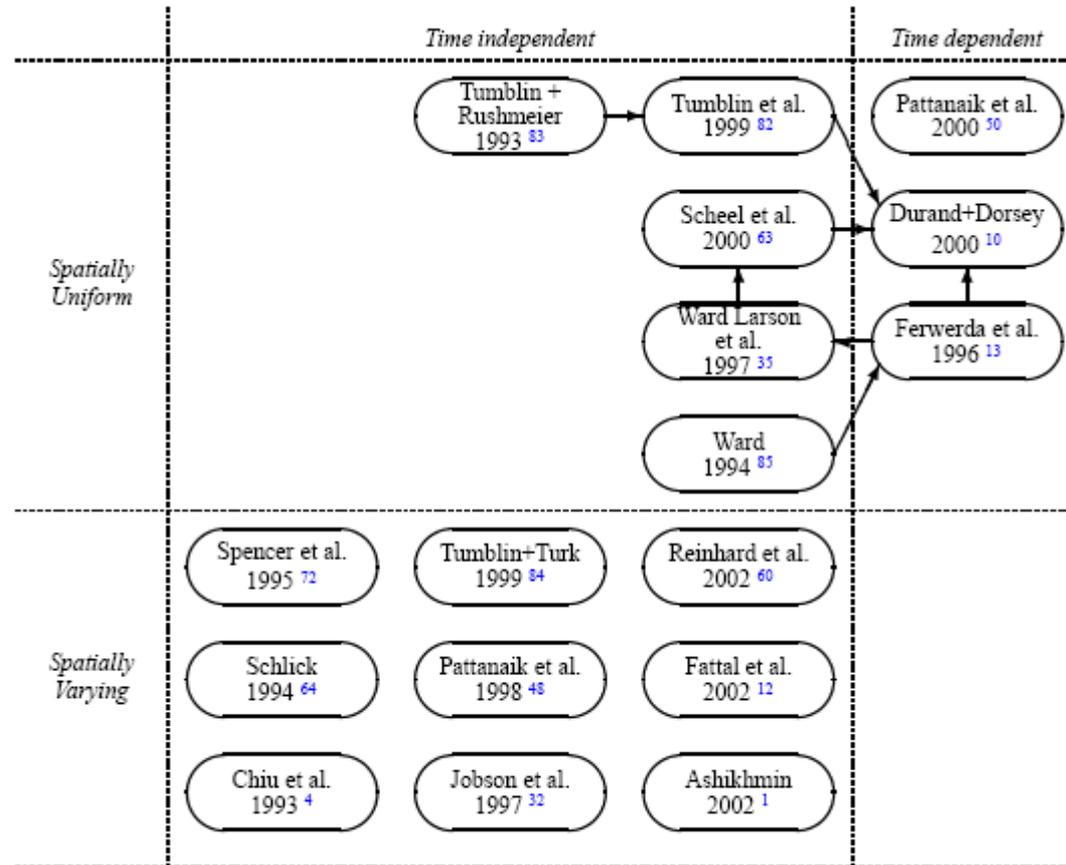


Figure 11: Taxonomy of Tone Reproduction Methods

From: Devling, Tone reproduction and Physically based Spectral rendering

Factor 5's Lair



Factor 5's Lair



Factor 5's Lair

- PS3
- Progressive mesh,
- HDR
- Luzes e sombras em tempo real
- Dynamics fluidos, personagens, objetos, tecido
- Nuvens volumétricas,
- Resoluções de 1920x1080
- Até 4 gigabytes de dados por fase

Factor 5's Lair



Lair © Sony Computer Entertainment America

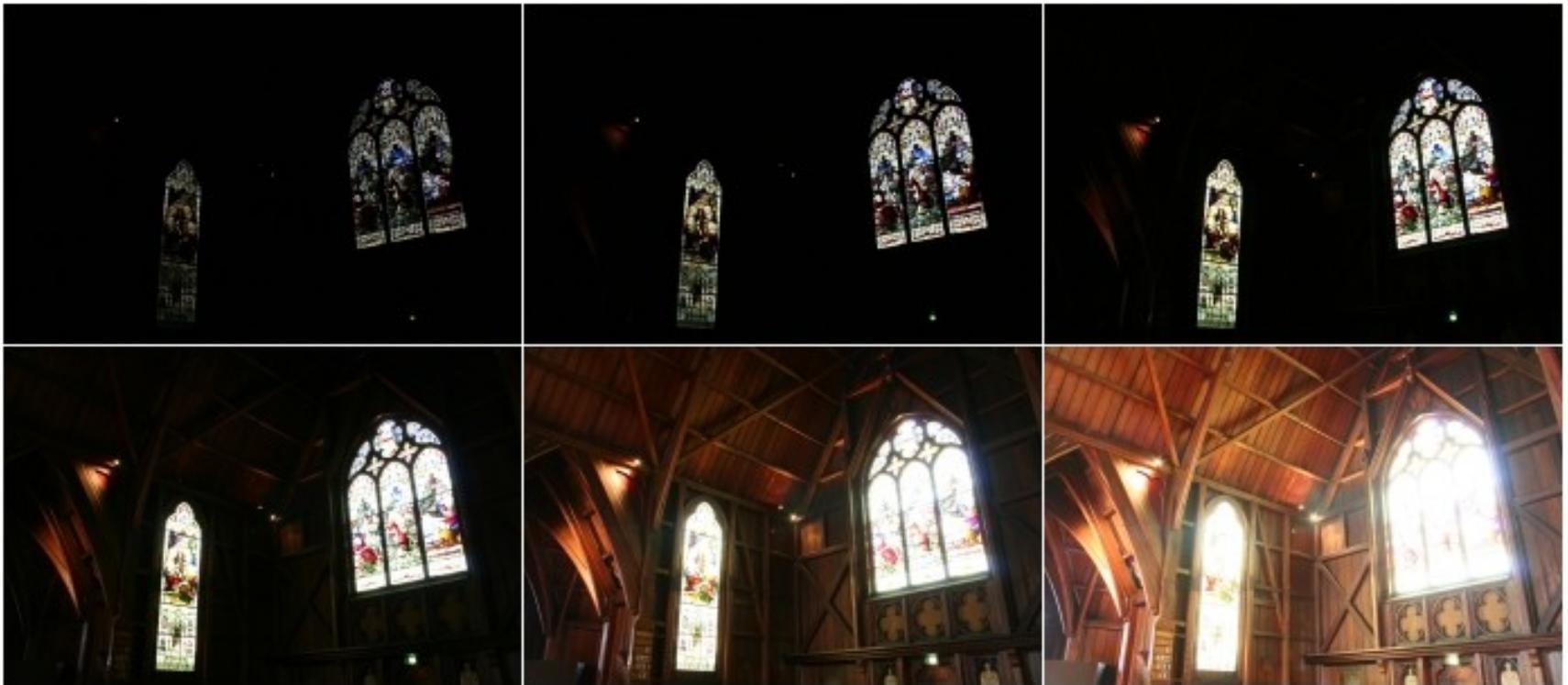
HDR - Opções

- Visualizar Imagens (HDRI)
- Renderizar Imagens (HDRR)

HDR – Visualizar

- Obtendo Imagens:
 - Utilizar várias imagens da mesma cena com tempos de exposições distintos

HDR – Obtendo Imagens



Wikipedia

HDR – Tone Mapping



Wikipedia

HDR - Renderizar



FarCry sem HDR

FarCry com HDR



HDR – Renderizar

- Valores de pixels podem ultrapassar intensidade de 100%.

HDR - Renderizar



HalfLife 2

HDR – Renderizar

- Geralmente em renderizações HDR se acrescenta o efeito de desfoque de imagens
 - Bloom ou Glow

HDR - Renderizar

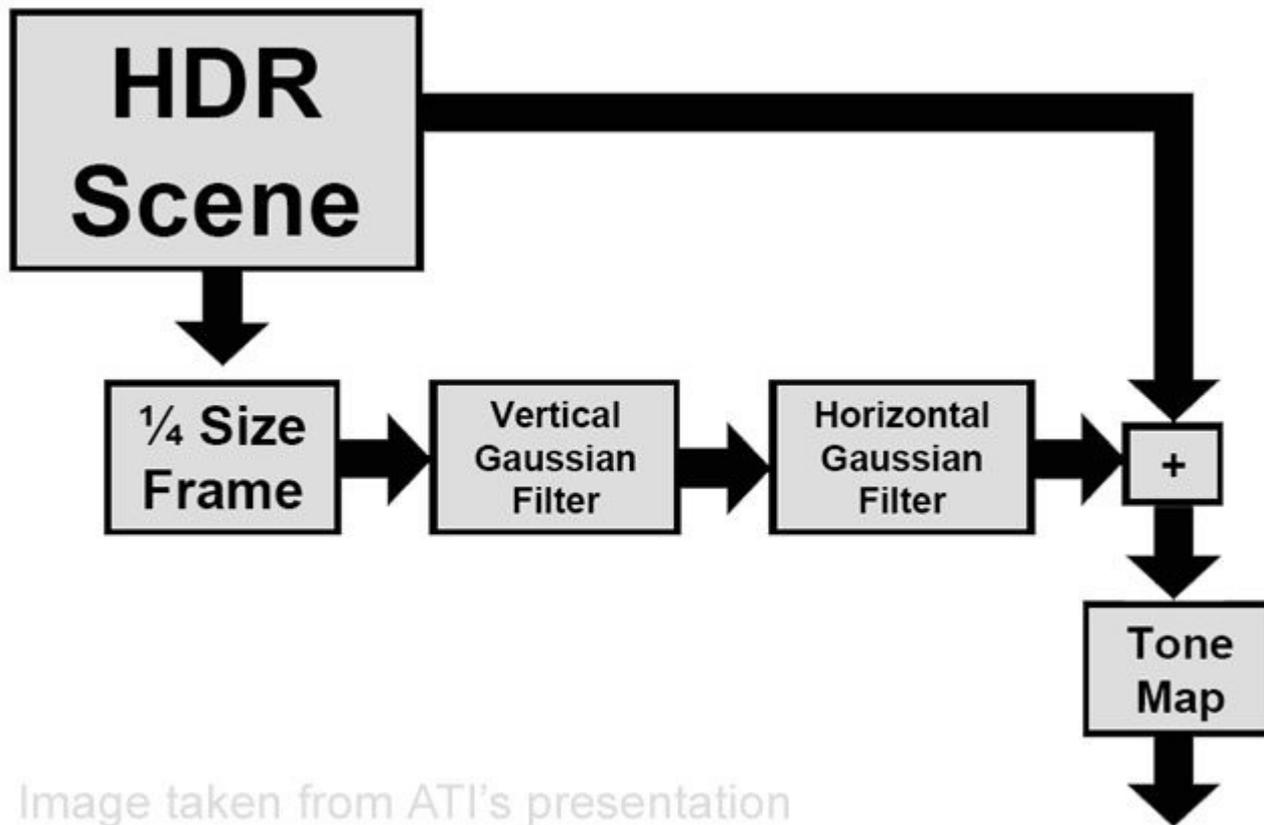
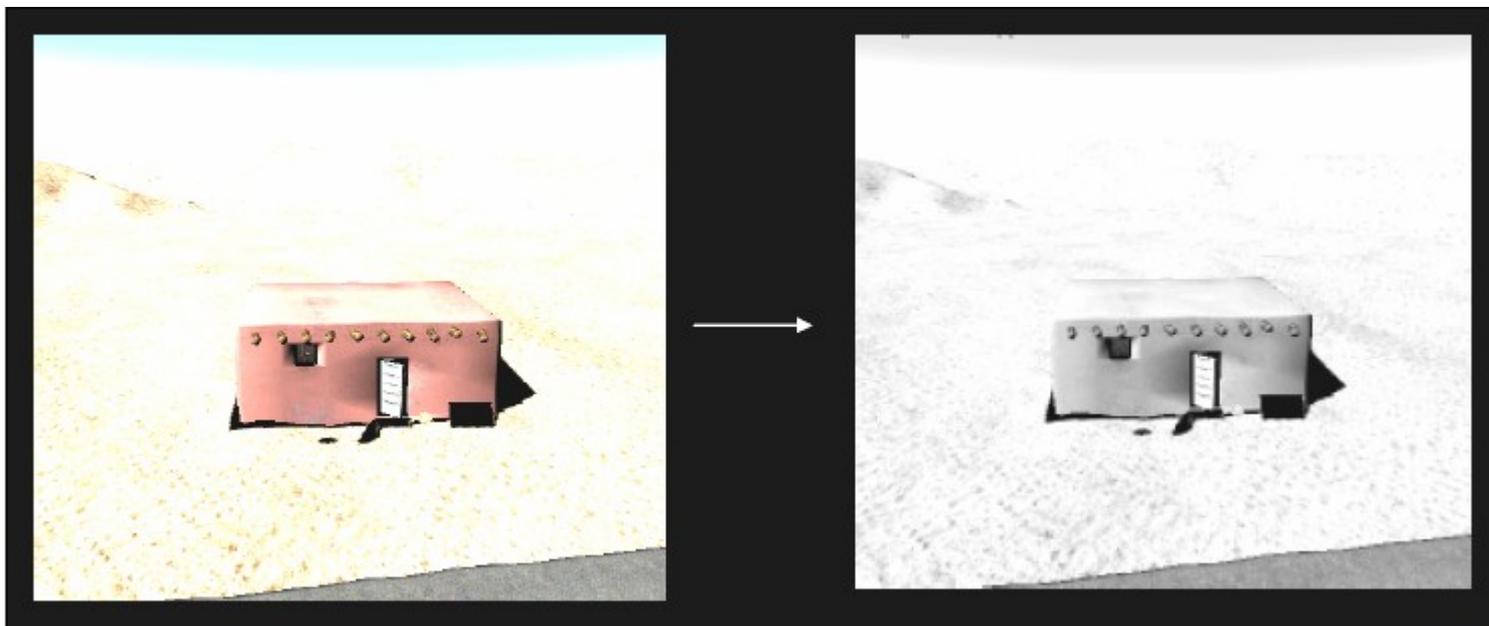


Image taken from ATI's presentation

Bloom

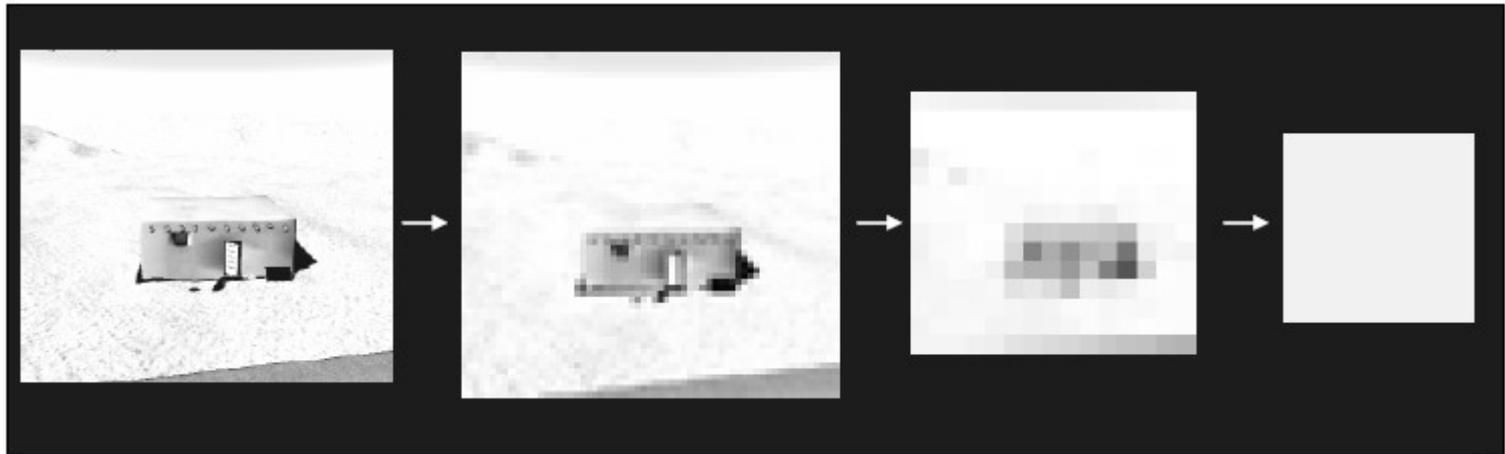
- Efeitos obtidos:
 - Partes claras da cena sobrepoem pixels vizinhos
 - Contorno dos objetos parecem mais suaves

HDR -> Luminância



Nvidia

Obtendo a Luminância



Nvidia

HDR – Como implementar?

- Atualmente HDR pode ser implementado através de Shaders, para visualização ou renderização.

HDR – Como implementar?

- São necessários vários passos de renderização para se gerar uma cena
- Para implementar HDR é necessário no mínimo Shader Model 2.0

HDR – Shaders

- Shaders são programas que podem ser executados em uma GPU.
 - Transformam vertices e modificam pixels

HDR – Shaders

- Linguagens de programação:
 - GLSL (Khronos Group)
 - Portável
 - Funciona com OpenGL
 - HLSL (Microsoft)
 - Não portável
 - Funciona com DirectX (PC, XBOX, etc...)
 - Cg (Nvidia)
 - Portável
 - Pode ser utilizada juntamente com OpenGL ou DirectX

HDR – Ambientes de desenvolvimento

- FXComposer (HLSL)
Nvidia
- RenderMonkey (GLSL e HLSL)
ATI
- ShaderDesign (GLSL)
ThyphoonLabs3D

HDR – Ambientes de desenvolvimento

The screenshot displays the NVIDIA FX Composer interface for a project named "tonemap.fx". The central editor shows the following HLSL code:

```
v2f TonemapVS(a2v IN)
{
    v2f OUT;
    OUT.position = IN.position;
    OUT.texcoord = IN.texcoord;
    OUT.exposure = pow(2.0, exposure);
    return OUT;
}

half4 TonemapPS(v2f IN,
                uniform half3 defog,
                uniform half gamma) : COLOR
{
    half3 c = tex2D(ImageSam
c = max(0, c - defog);
c *= IN.exposure;
// gamma correction - co
c = pow(c, gamma);
return half4(c.rgb, 1.0)
}
```

The Properties panel on the right shows the following parameters for the "tonemap" effect:

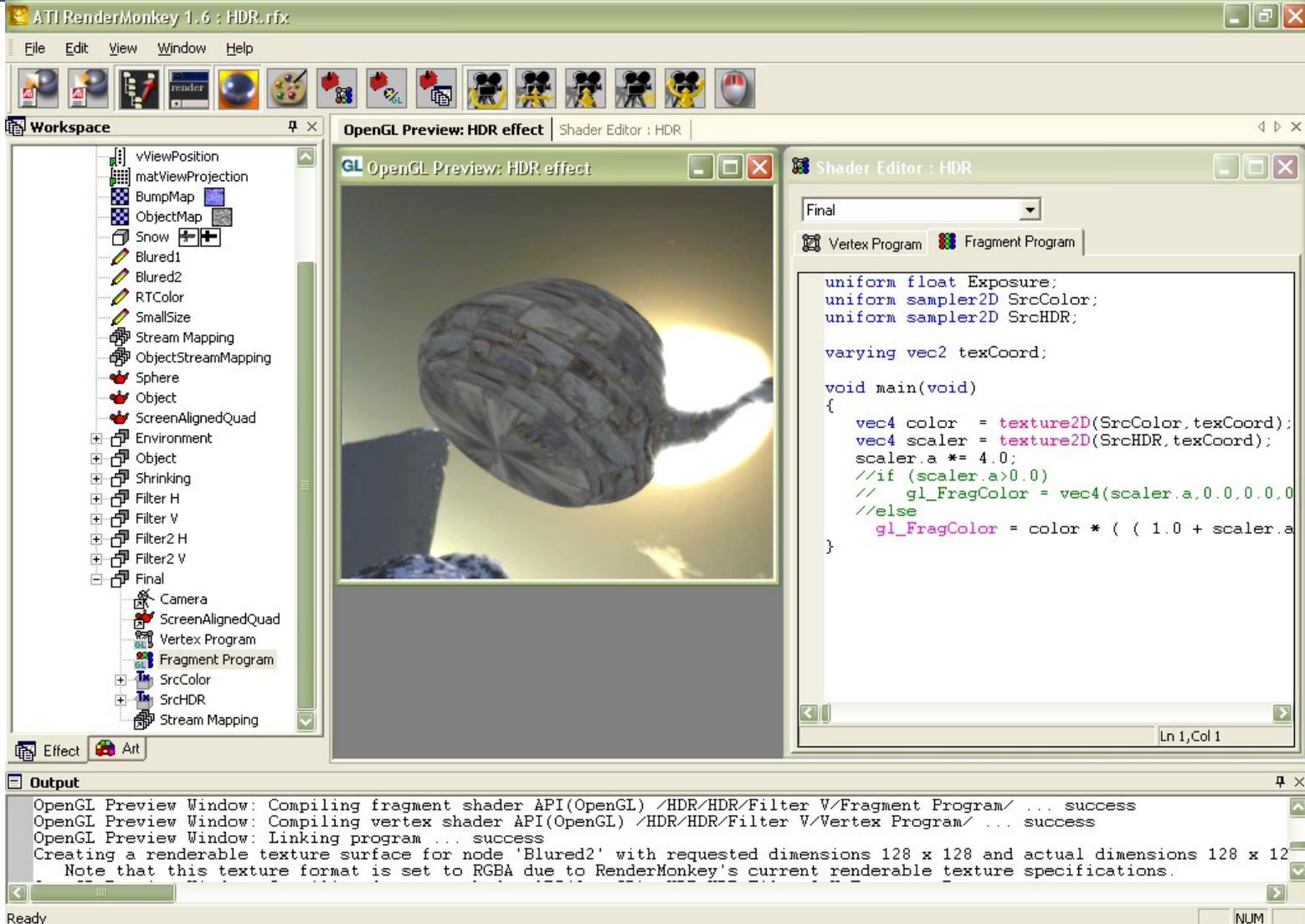
| Parameter | Value |
|-----------|------------------------------|
| Image | C:\Arquivos de programas ... |
| Exposure | 2.3 |
| De-fog | 0.000 |
| Gamma | 0.68 |
| fogColor | 1.00, 1.00, 1.00 |

The Scene panel at the bottom shows a 3D render of a church interior with the text "Frame: 73 Perspective". The Log panel at the bottom left contains the following messages:

```
Build Complete - Updated: 2 materials associated w
Errors: 0, Warnings: 0
Created HLSL Effect from EffectCompiler: C:\Arqui
Found valid technique: Main
Technique: Main Validated for this device
Created HLSL Effect from EffectCompiler: C:\Arqui
Found valid technique: Main
Technique: Main Validated for this device
```

The interface also includes a Materials panel on the left, a Textures panel at the bottom left, and a Log/Task panel at the bottom.

HDR – Ambientes de desenvolvimento



The screenshot displays the ATI RenderMonkey 1.6 interface for an HDR effect. The main workspace shows a 3D scene with a textured sphere and a bright light source. The Shader Editor window is open, showing the following GLSL code for a Fragment Program:

```
Final
Vertex Program Fragment Program
uniform float Exposure;
uniform sampler2D SrcColor;
uniform sampler2D SrcHDR;

varying vec2 texCoord;

void main(void)
{
    vec4 color = texture2D(SrcColor, texCoord);
    vec4 scaler = texture2D(SrcHDR, texCoord);
    scaler.a *= 4.0;
    //if (scaler.a>0.0)
    //    gl_FragColor = vec4(scaler.a,0.0,0.0,0.0)
    //else
    gl_FragColor = color * ( ( 1.0 + scaler.a
```

The Output window at the bottom shows the following log messages:

```
OpenGL Preview Window: Compiling fragment shader API(OpenGL) /HDR/HDR/Filter V/Fragment Program/ ... success
OpenGL Preview Window: Compiling vertex shader API(OpenGL) /HDR/HDR/Filter V/Vertex Program/ ... success
OpenGL Preview Window: Linking program ... success
Creating a renderable texture surface for node 'Blured2' with requested dimensions 128 x 128 and actual dimensions 128 x 128
Note that this texture format is set to RGBA due to RenderMonkey's current renderable texture specifications.
```

HDR – Ambientes de desenvolvimento

The screenshot displays the ShaderDesigner software interface, version 1.5.9.4, with a project path of C:\Arquivos de programas\TyphoonLabs\Shader Designer\shaders\envmap.gdp. The interface is divided into several panels:

- Texture Preview:** Shows a preview of the environment map texture, which is a detailed, high-resolution scene of a forest or jungle.
- Code Editor:** Displays the GLSL code for the environment mapping fragment shader. The code includes comments and uniform variables for BaseColor and MixRatio, and varying variables for Normal, EyeDir, and LightIntensity. The main function computes reflection and altitude.
- Light States:** Shows the configuration for Light 0, including FrontMatShininess (0), FrontMatSpecular (0, 0, 0), and LightModelAmbient (0, 0, 0, 0).
- Shader Render:** A window showing the rendered output of the shader, featuring a large, semi-transparent blue sphere overlaid on the environment map texture. The FPS is 39.
- Uniform Variables:** A list of uniform variables with their IDs, types, names, and values. The BaseColor variable is highlighted.
- Native Compilation Results:** Shows the compilation status, indicating that the fragment shaders were compiled successfully and the linking was successful.

```
1 //
2 // Fragment shader for environment mapping with
3 // equirectangular 2D texture
4 //
5 // Authors: John Kessenich, Randi Rost
6 //
7 // Copyright (c) 2002-2004 3Dlabs Inc. Ltd.
8 //
9 // See 3Dlabs-License.txt for license informati
10 //
11
12 const vec3 Xunitvec = vec
13 const vec3 Yunitvec = vec
14
15 uniform vec3 BaseColor;
16 uniform float MixRatio;
17
18 uniform sampler2D EnvMap;
19
20 varying vec3 Normal;
21 varying vec3 EyeDir;
22 varying float LightIntens
23
24 void main (void)
25 {
26     // Compute reflection
27     vec3 reflectDir = ref
28
29     // Compute altitude
```

Uniform Variables

| | |
|-------------------------------------|---|
| <input checked="" type="checkbox"/> | Id: 6, Type: float [3], Name: Xunitvec, Values: 1,0,0 |
| <input checked="" type="checkbox"/> | Id: 7, Type: float [3], Name: Yunitvec, Values: 0,1,0 |
| <input checked="" type="checkbox"/> | Id: 8, Type: int, Name: EnvMap, Values: 0 |
| <input checked="" type="checkbox"/> | Id: 9, Type: float, Name: MixRatio, Values: 0.8 |
| <input checked="" type="checkbox"/> | Id: 10, Type: float [3], Name: BaseColor, Values: 0.4,0.4,1 |

Native Compilation Results

```
Compiling Fragment Shaders...
envmap.frag
OK

+++++
Linking all shaders....
Link successful
```

Compilation: Ok.

HDR – Shaders

Fxcomposer ToneMapping:

- Feito para suportar imagem .hdr

```
v2f TonemapVS(a2v IN){  
    v2f OUT;  
    OUT.position = IN.position;  
    OUT.texcoord = IN.texcoord;  
    OUT.exposure = pow(2.0, exposure);  
    return OUT;  
}
```

HDR – Shaders

Fxcomposer ToneMapping:

```
half4 TonemapPS(v2f IN, uniform half3 defog,  
                uniform half gamma) : COLOR {  
    half3 c = tex2D(ImageSampler, IN.texcoord);  
    c = max(0, c - defog);  
    c *= IN.exposure;  
    // gamma correction - could use texture lookups for this  
    c = pow(c, gamma);  
    return half4(c.rgb, 1.0);  
}
```

HDR – Shaders

Fxcomposer ToneMapping:

VIDEO – tonemapping e bloom

HDR – Shaders

Implementações HDR:

RenderMonkey

Utiliza textura .dss e possui implementação Bloom. Utiliza buffer comum de renderização com 8bits para cada canal.

Implementa bump mapping.

DX-SDK

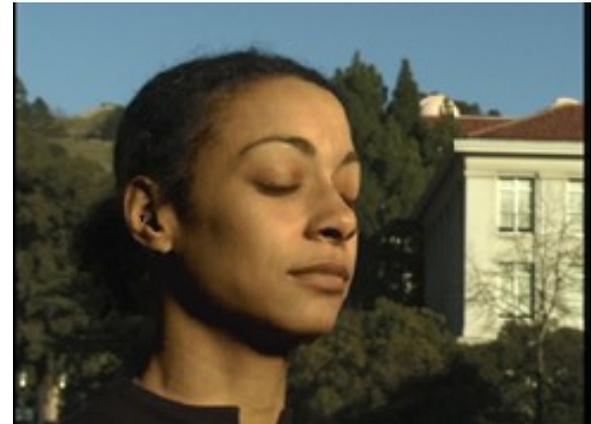
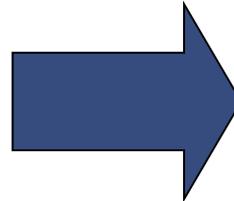
Implementa Bloom + HDR + controle de exposição. Não utiliza textura.

HDR – Shaders

Implementações HDR:

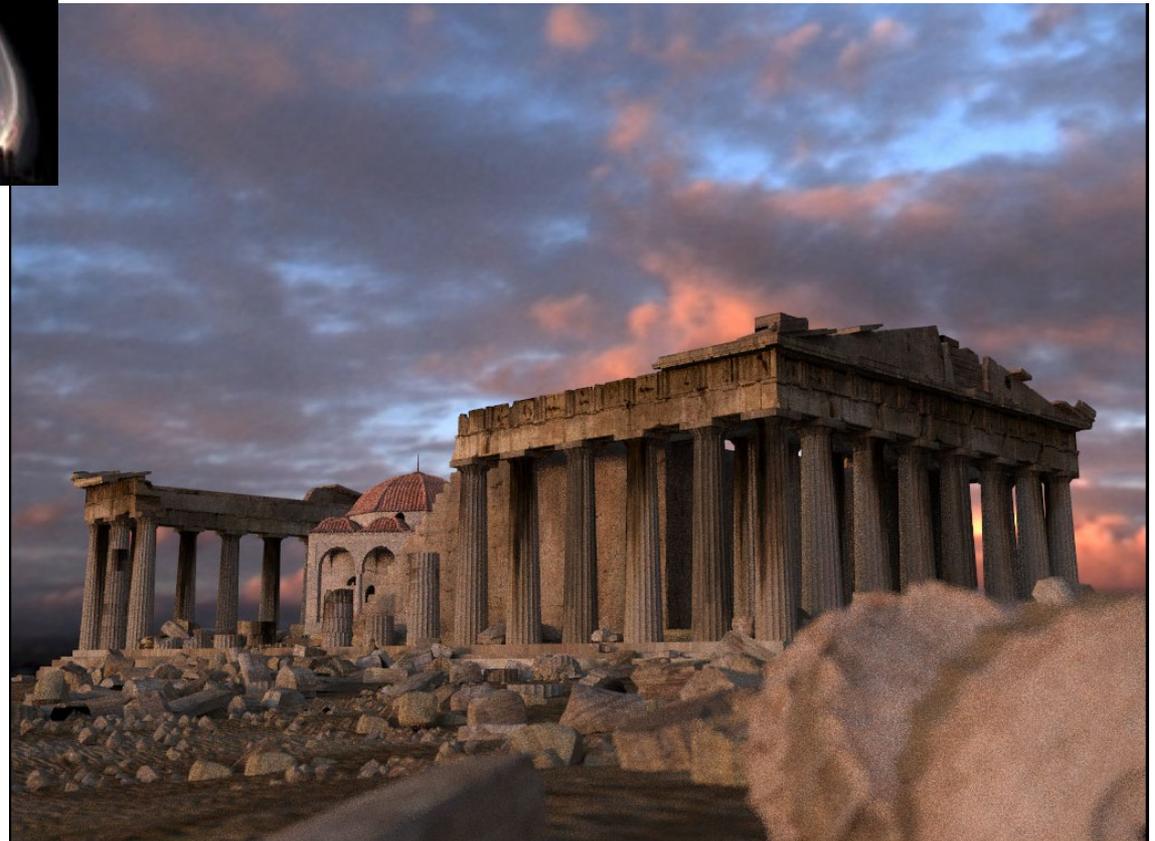
VIDEO

Outros usos?



HDRI and Image-Based Lighting
Paul Debevec - SIGGRAPH 2003 Course #19

Iluminação baseada em Imagens



HDRI and Image-Based
Lighting Paul Debevec
-SIGGRAPH 2003

Iluminação baseada em Imagens



HDRI and Image-Based
Lighting Paul Debevec
-SIGGRAPH 2003



Obrigado!

- Perguntas ?

Alessandro Silva: asilva@dcc.ufmg.br

Wallace Lages: wsl@dcc.ufmg.br

Enquanto isso assistir Video Final

Quest 3D