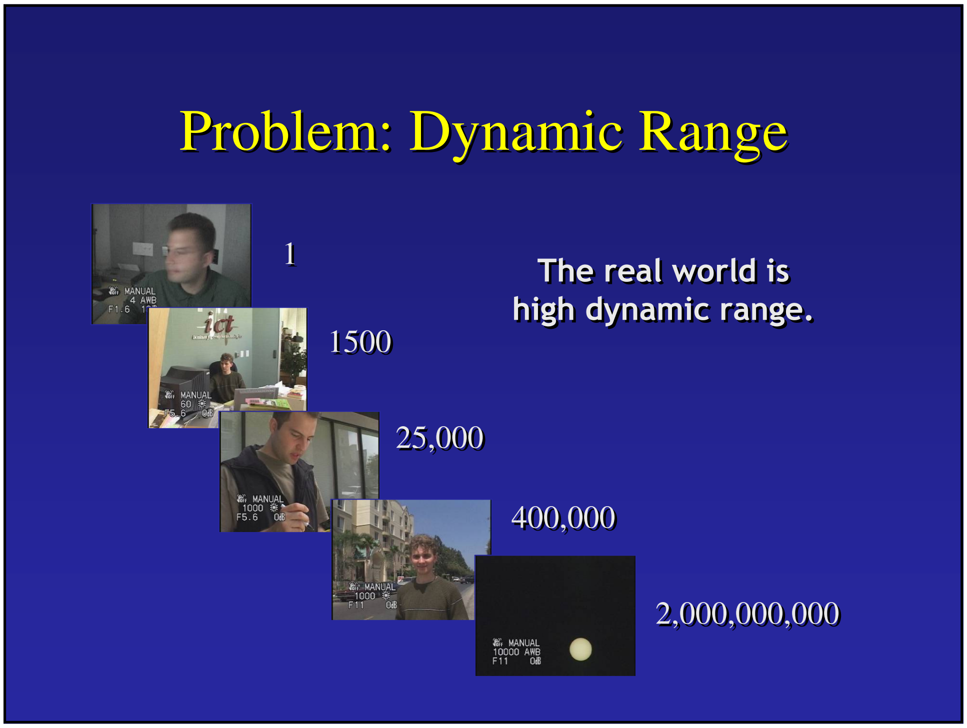
High Dynamic Range Images

15-463: Rendering and Image Processing
Alexei Efros

The Grandma Problem


Problem: Dynamic Range
1
1500
25,000
400,000
2,000,000,000
The real world is
high dynamic range.

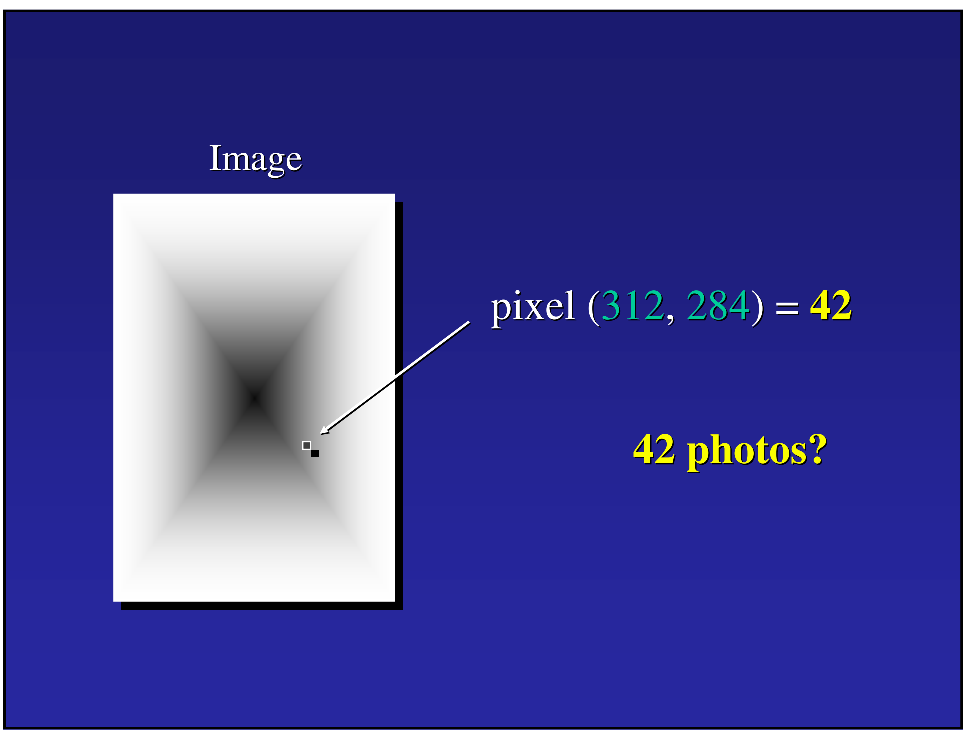
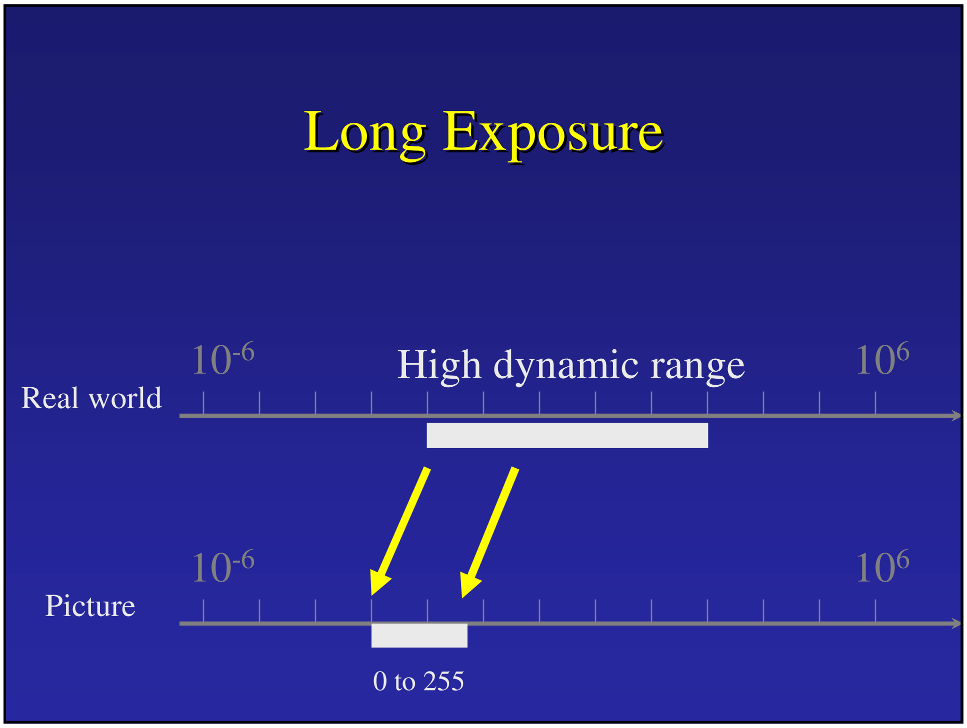
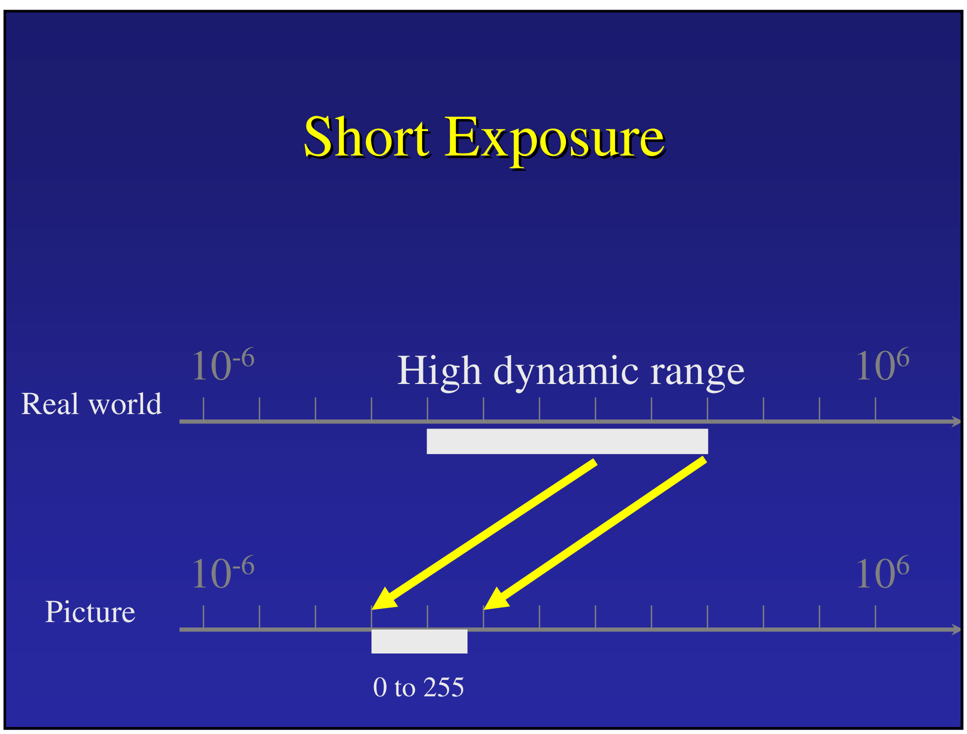


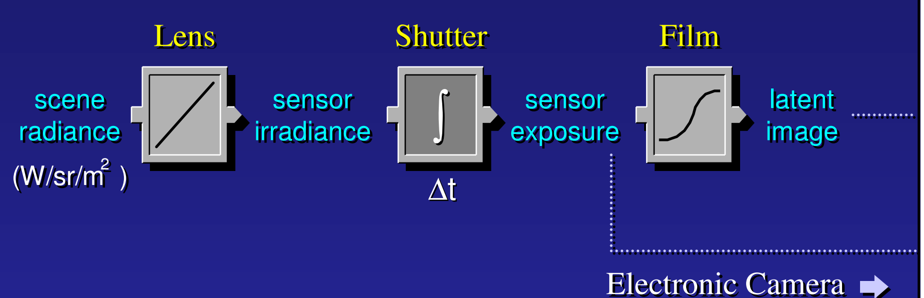
Image
pixel (312, 284) = 42
42 photos?


Long Exposure
Real world
10^{-6}
High dynamic range
10^{6}
Picture
10^{-6}
10^{6}
0 to 255

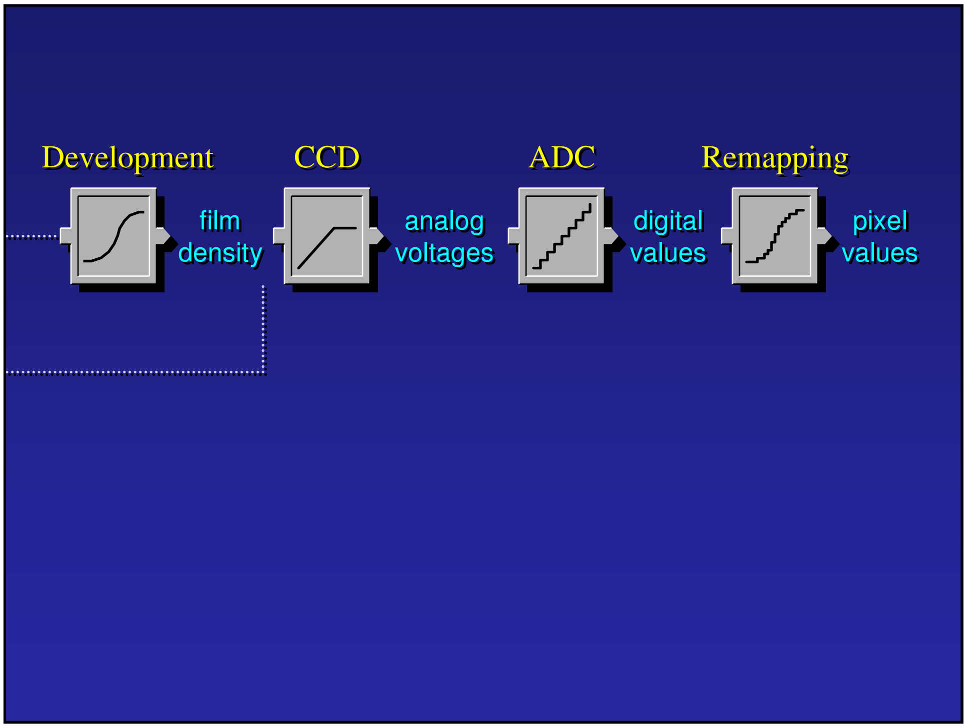

Short Exposure
Real world
10^{-6}
High dynamic range
10^{6}
Picture
10^{-6}
10^{6}
0 to 255

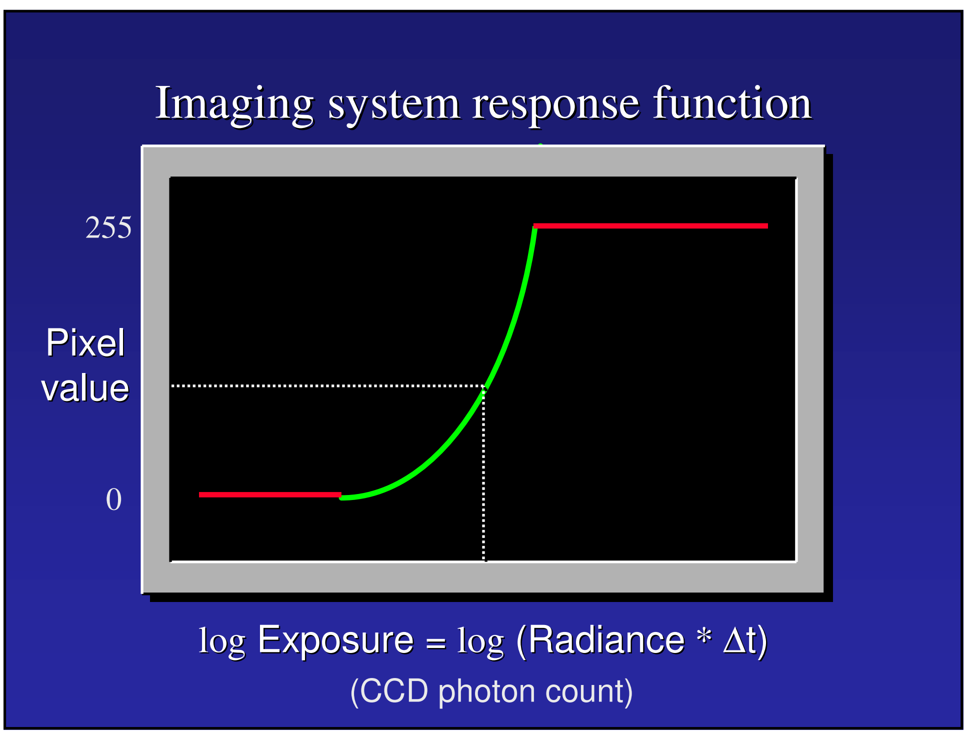
Camera Calibration

- Geometric
  - How pixel **coordinates** relate to **directions** in the world
- Photometric
  - How pixel **values** relate to **radiance** amounts in the world



The Image Acquisition Pipeline


Development
CCD
ADC
Remapping
film density
analog voltages
digital values
pixel values


Imaging system response function
255
Pixel value
0
log Exposure = log (Radiance * Δt)
(CCD photon count)

Varying Exposure

Camera is not a photometer!

- Limited dynamic range
  - ⇒ Perhaps use multiple exposures?
- Unknown, nonlinear response
  - ⇒ Not possible to convert pixel values to radiance
- Solution:
  - – Recover response curve from multiple exposures, then reconstruct the ***radiance map***

Recovering High Dynamic Range Radiance Maps from Photographs

Paul Debevec
Jitendra Malik

Computer Science Division
University of California at Berkeley

August 1997

Ways to vary exposure

- Shutter Speed (*)
- F/stop (aperture, iris)


- Neutral Density (ND) Filters

Shutter Speed

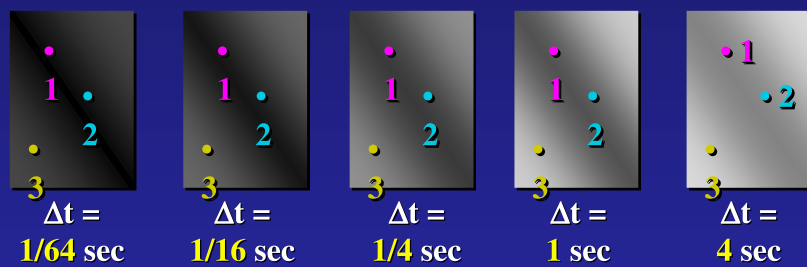
- Ranges: Canon D30: 30 to 1/4,000 sec.
- Sony VX2000: ¼ to 1/10,000 sec.
- Pros:
- Directly varies the exposure
- Usually accurate and repeatable
- Issues:
- Noise in long exposures

Shutter Speed

- Note: shutter times usually obey a power series – each "stop" is a factor of 2
- ¼, 1/8, 1/15, 1/30, 1/60, 1/125, 1/250, 1/500, 1/1000 sec
- Usually really is:
- ¼, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256, 1/512, 1/1024 sec

The Algorithm

Image series



Pixel Value Z = f(Exposure)

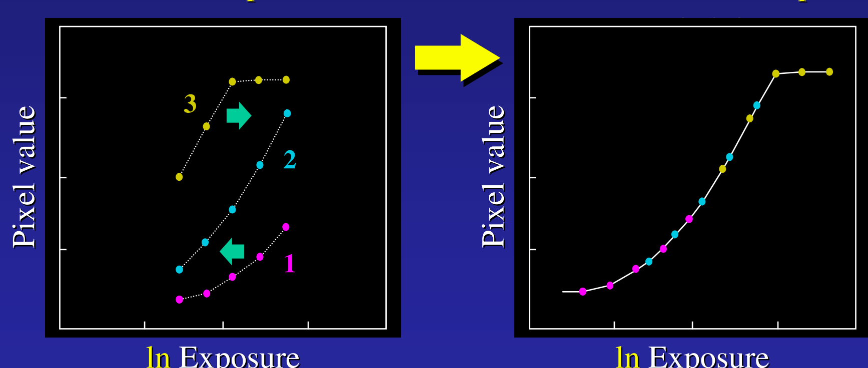
Exposure = Radiance × Δt

log Exposure = log Radiance + log Δt

Response Curve

Assuming unit radiance for each pixel

After adjusting radiances to obtain a smooth response



The Math

- Let *g(z)* be the *discrete* inverse response function
- For each pixel site *i* in each image *j*, want:

$$\ln Radiance_i + \ln \Delta t_j = g(Z_{ij})$$

- Solve the overdetermined linear system:

$$\sum_{i=1}^{N} \sum_{j=1}^{P} \left[\ln Radiance_i + \ln \Delta t_j - g(Z_{ij}) \right]^2 + \lambda \sum_{z=Z_{min}}^{Z_{max}} g''(z)^2$$

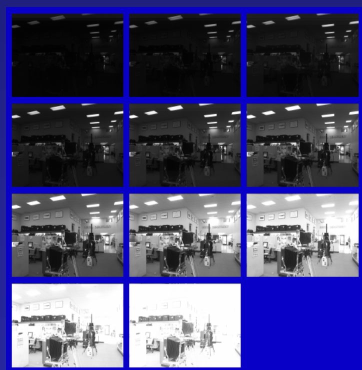
fitting term smoothness term

Matlab Code

```
function [g,lE]=gsolve(Z,B,l,w)

n = 256;
A = zeros(size(Z,1)*size(Z,2)+n+1,n+size(Z,1));
b = zeros(size(A,1),1);

k = 1;                  %% Include the data-fitting equations
for i=1:size(Z,1)
  for j=1:size(Z,2)
    wij = w(Z(i,j)+1);
    A(k,Z(i,j)+1) = wij; A(k,n+i) = -wij; b(k,1) = wij * B(i,j);
    k=k+1;
  end
end

A(k,129) = 1;           %% Fix the curve by setting its middle value to
k=k+1;

for i=1:n-2             %% Include the smoothness equations
  A(k,i)=l*w(i+1); A(k,i+1)=-2*l*w(i+1); A(k,i+2)=l*w(i+1);
  k=k+1;
end

x = A\b;                %% Solve the system using SVD

g = x(1:n);
lE = x(n+1:size(x,1));
```

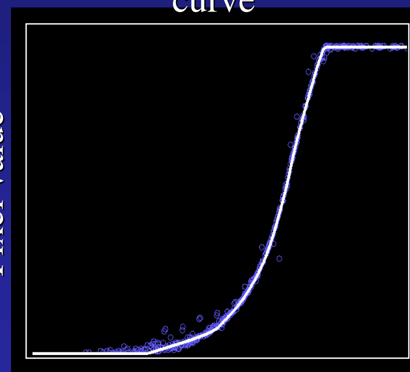
Results: Digital Camera

Kodak DCS460
1/30 to 30 sec



Recovered response curve

Pixel value

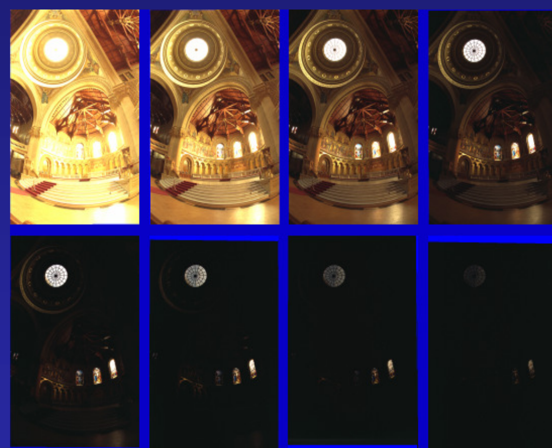


log Exposure

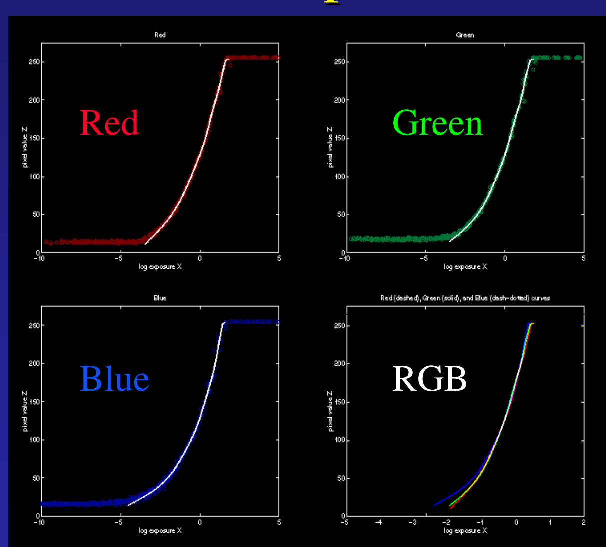
Reconstructed radiance map

Results: Color Film

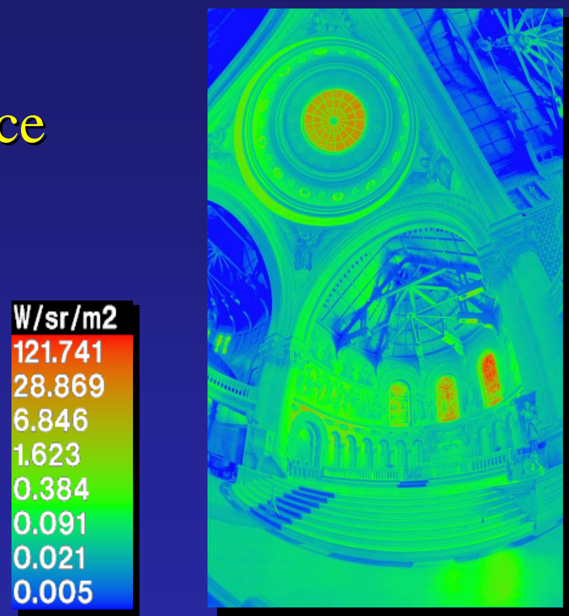
- Kodak Gold ASA 100, PhotoCD



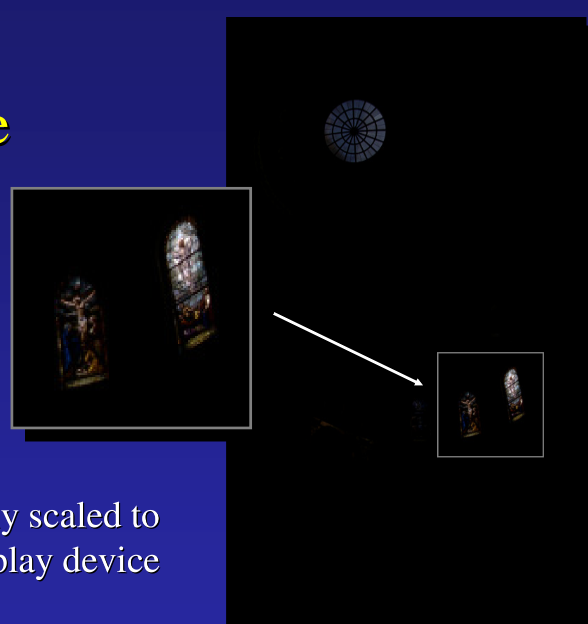
Recovered Response Curves



The Radiance Map



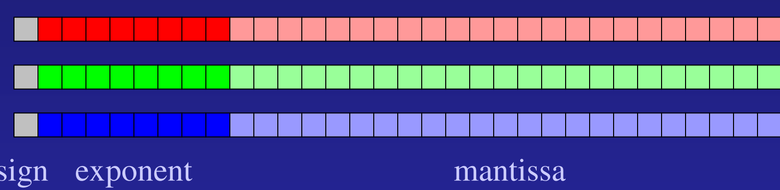
The Radiance Map



Linearly scaled to display device

Portable FloatMap (.pfm)

- 12 bytes per pixel, 4 for each channel



Text header similar to Jeff Poskanzer's .ppm image format:

```
PF
768 512
1
<binary image data>
```

Floating Point TIFF similar

Radiance Format (.pic, .hdr)

(145, 215, 87, 149) =

(145, 215, 87) * 2^(149-128) =

(1190000, 1760000, 713000)

(145, 215, 87, 103) =

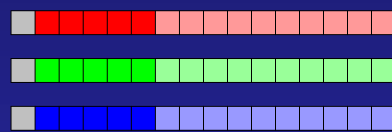
(145, 215, 87) * 2^(103-128) =

(0.00000432, 0.00000641, 0.00000259)

Ward, Greg. "Real Pixels," in Graphics Gems IV, edited by James Arvo, Academic Press, 1994

ILM's OpenEXR (.exr)

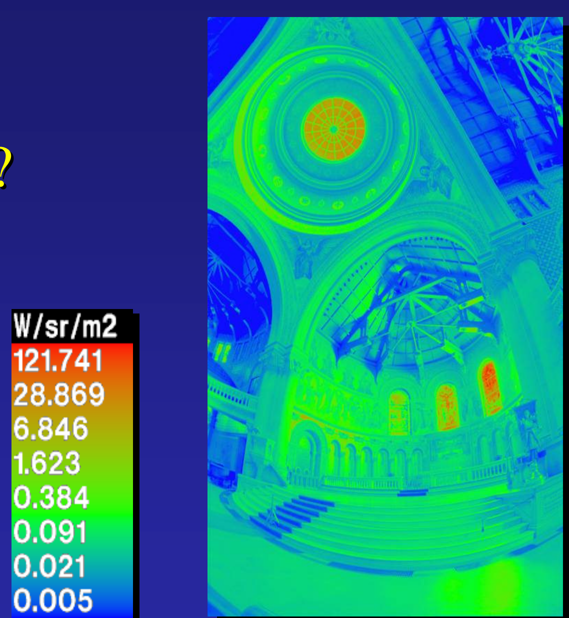
- 6 bytes per pixel, 2 for each channel, compressed



- Several lossless compression options, 2:1 typical
- Compatible with the "half" datatype in NVidia's Cg
- Supported natively on GeForce FX and Quadro FX

- Available at http://www.openexr.net/

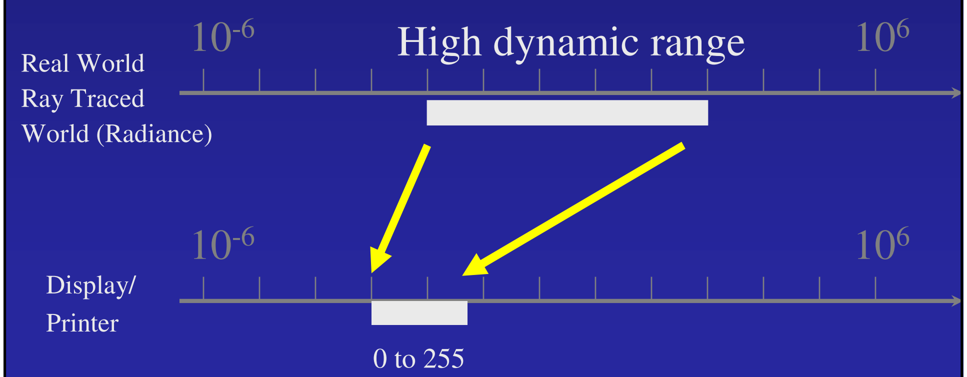
Now What?



Tone Mapping

- How can we do this?

  Linear scaling?, thresholding? Suggestions?

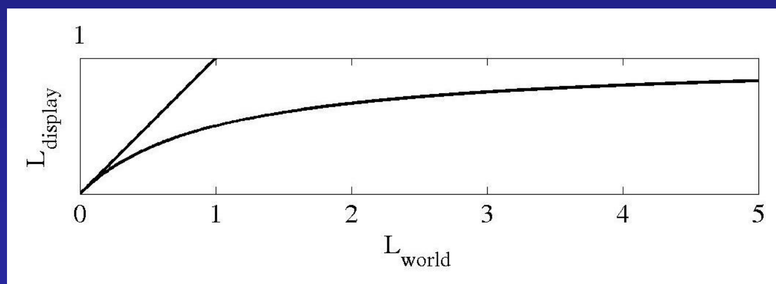


Simple Global Operator

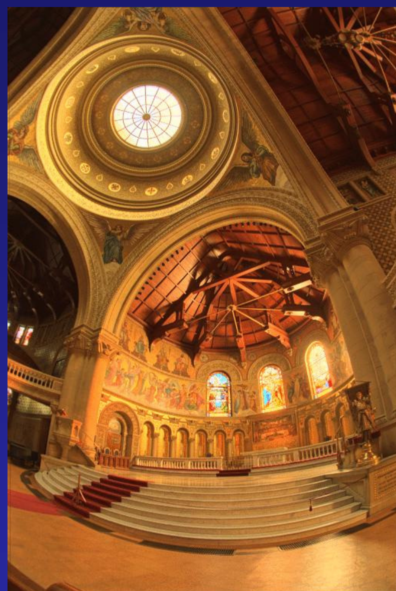
- Compression curve needs to
  - Bring everything within range
  - Leave dark areas alone
- In other words
  - Asymptote at 255
  - Derivative of 1 at 0

Global Operator (Reinhart et al)

$$L_{display} = \frac{L_{world}}{1 + L_{world}}$$



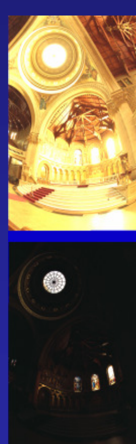
Global Operator Results



Reinhart Operator

Darkest 0.1% scaled to display device

What do *we* see?



Vs.

What does the eye sees?

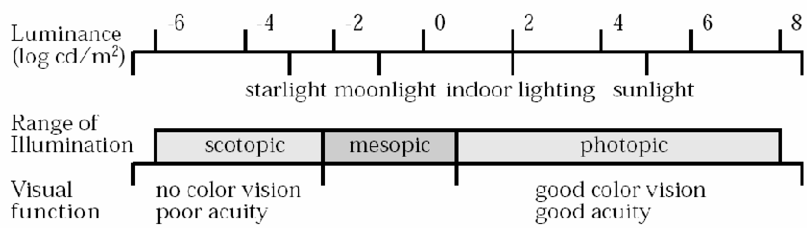
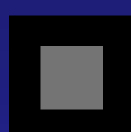


Figure 1: The range of luminances in the natural environment and associated visual parameters. After Hood (1986).

The eye has a huge dynamic range
Do we see a true radiance map?

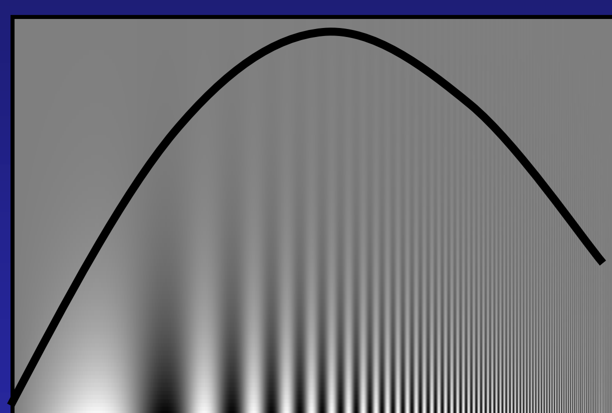
Eye is not a photometer!



- *"Every light is a shade, compared to the higher lights, till you come to the sun; and every shade is a light, compared to the deeper shades, till you come to the night."*
- — John Ruskin, 1879

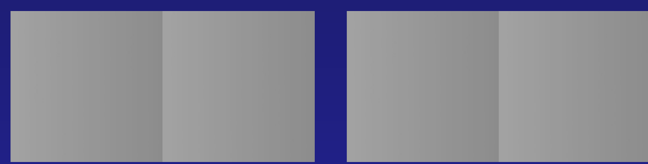
Cornsweet Illusion

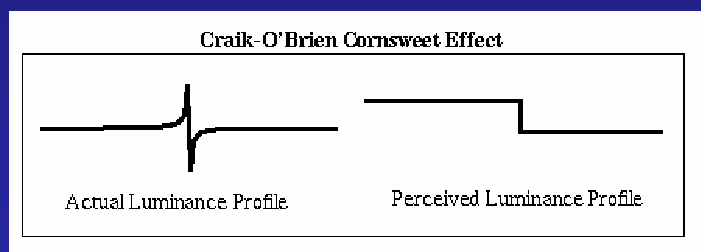
Sine wave



Campbell-Robson contrast sensitivity curve

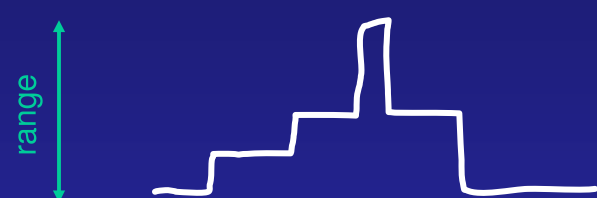
Metamores





Can we use this for range compression?

Compressing Dynamic Range



Fast Bilateral Filtering for the Display of High-Dynamic-Range Images

Frédo Durand & Julie Dorsey
Laboratory for Computer Science
Massachusetts Institute of Technology

High-dynamic-range (HDR) images

- CG Images

- Multiple exposure photo [Debevec & Malik 1997]

Recover response curve → HDR value for each pixel

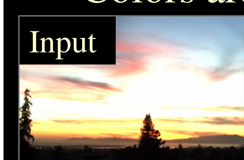
- HDR sensors

A typical photo

- Sun is overexposed
- Foreground is underexposed

Gamma compression

- $X \to X^{\gamma}$
- Colors are washed-out

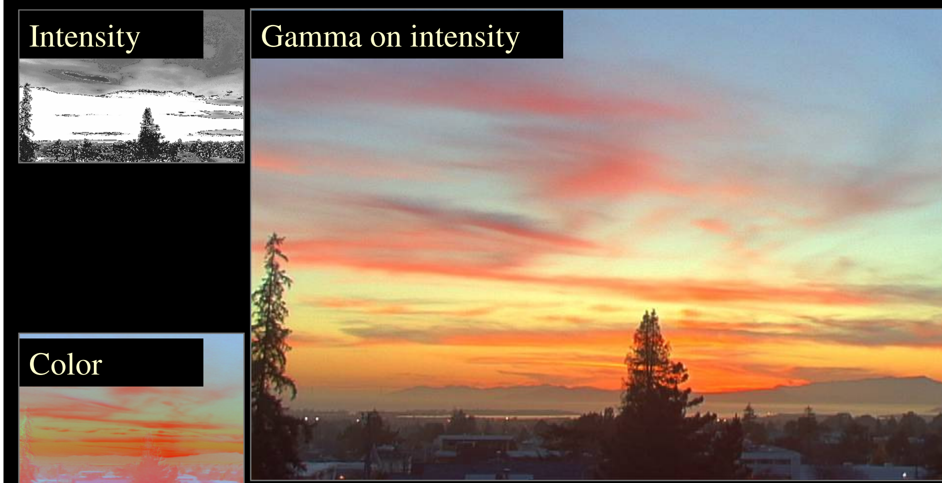


Input

Gamma

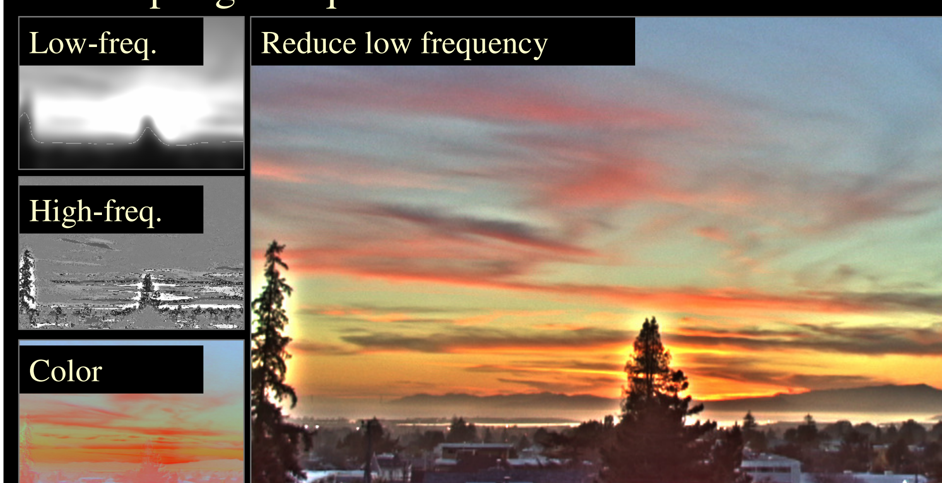
Gamma compression on intensity

- Colors are OK,
  but details (intensity high-frequency) are blurred



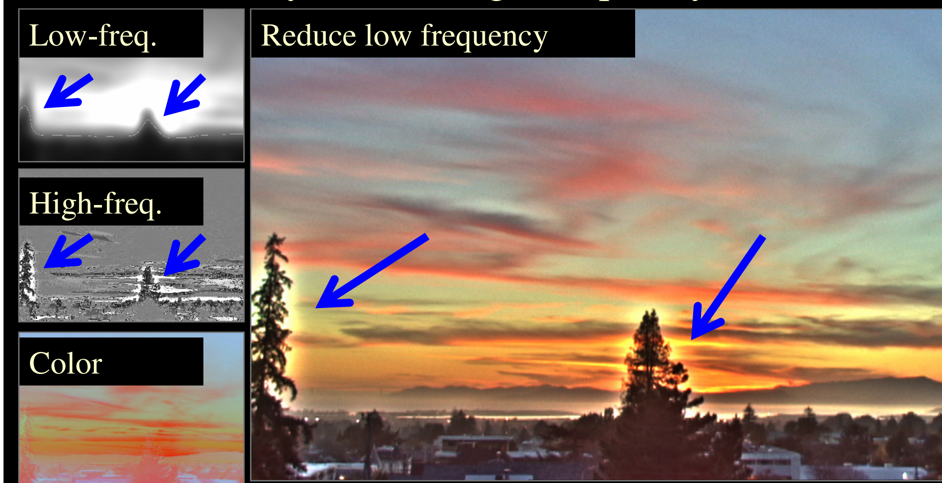
Chiu et al. 1993

- Reduce contrast of low-frequencies
- Keep high frequencies



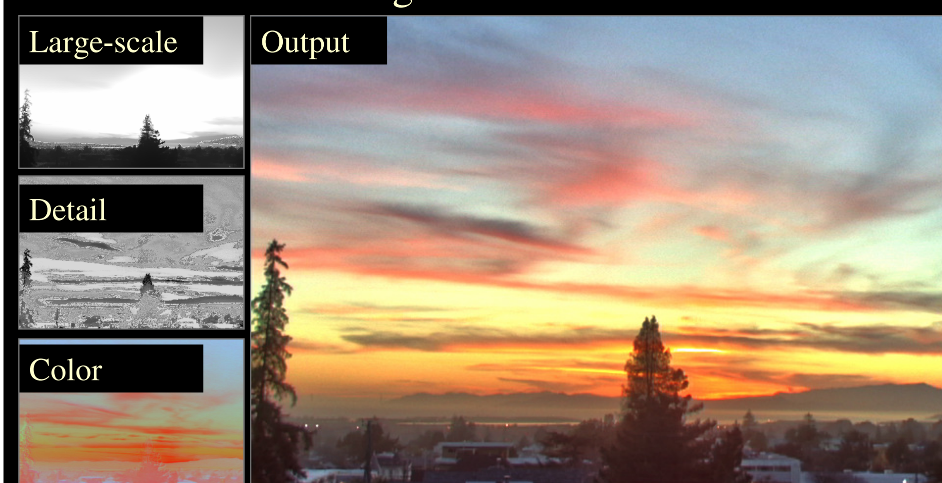
The halo nightmare

- For strong edges
- Because they contain high frequency



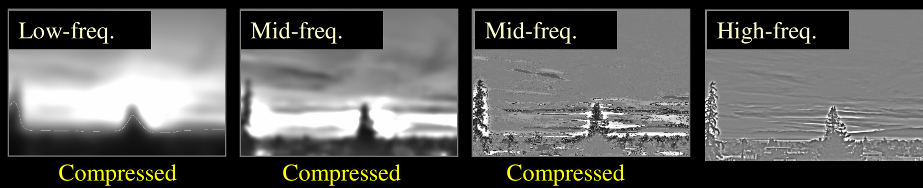
Our approach

- Do not blur across edges
- Non-linear filtering



Multiscale decomposition

- Multiscale retinex [Jobson et al. 1997]



Edge-preserving filtering

- Blur, but not across edges

- Anisotropic diffusion [Perona & Malik 90]
  - Blurring as heat flow
  - LCIS [Tumblin & Turk]
- Bilateral filtering [Tomasi & Manduci, 98]

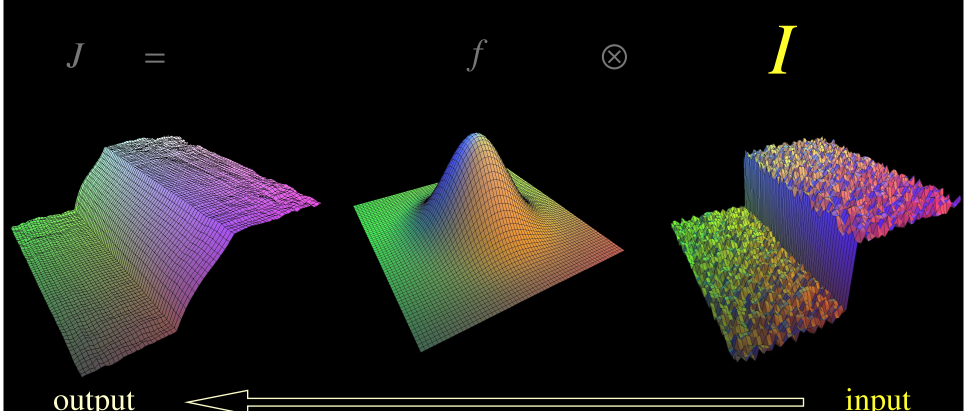
Comparison with our approach

- We use only 2 scales
- Can be seen as illumination and reflectance
- Different edge-preserving filter from LCIS

Compressed

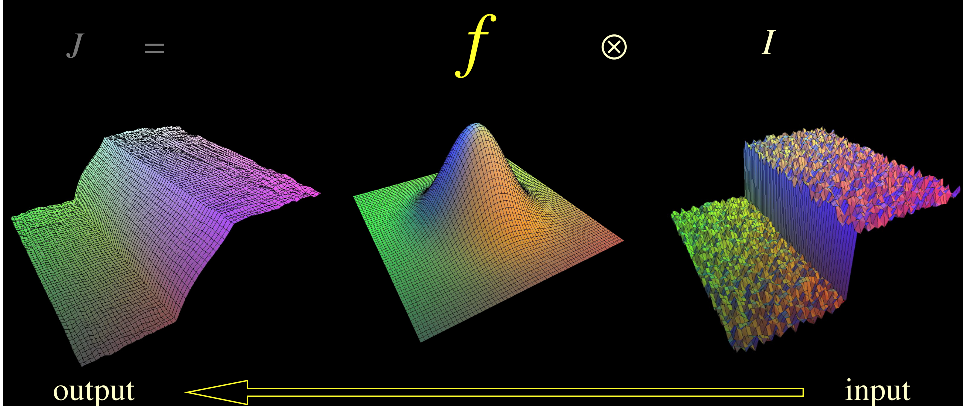
Start with Gaussian filtering

- Here, input is a step function + noise



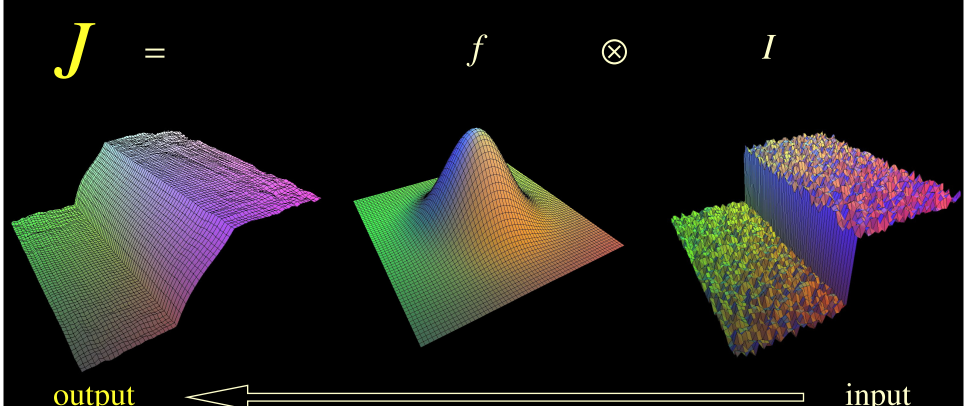
Start with Gaussian filtering

- Spatial Gaussian f



Start with Gaussian filtering

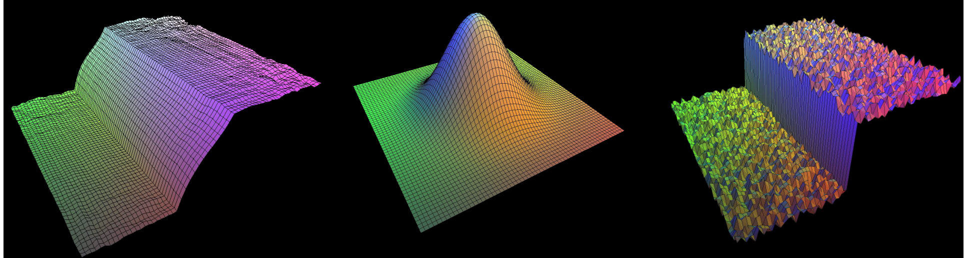
- Output is blurred



Gaussian filter as weighted average

- Weight of ξ depends on distance to x

$$J(x) = \sum_{\xi} f(x,\xi) \; I(\xi)$$

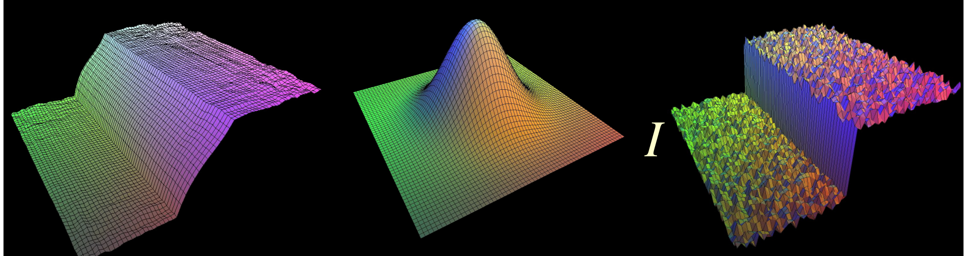


output ⟵ input

The problem of edges

- Here, $I(\xi)$ "pollutes" our estimate J(x)
- It is too different

$$J(x) = \sum_{\xi} f(x,\xi) \; I(\xi)$$

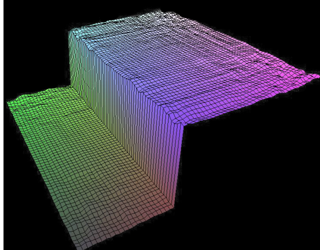


output ⟵ input

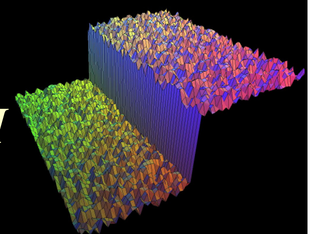
Principle of Bilateral filtering

- [Tomasi and Manduchi 1998]
- Penalty g on the intensity difference

$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi) - I(x)) \quad I(\xi)$$



I

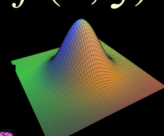


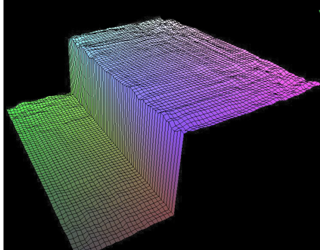
output ⟵ input

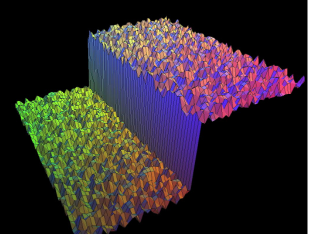
Bilateral filtering

- [Tomasi and Manduchi 1998]
- Spatial Gaussian f

$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi) - I(x)) \quad I(\xi)$$





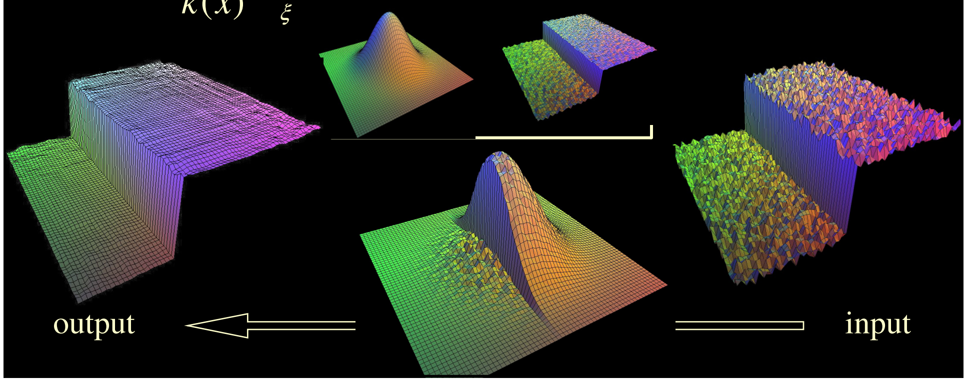


output ⟵ input

Bilateral filtering

- [Tomasi and Manduchi 1998]
- Spatial Gaussian f
- Gaussian g on the intensity difference

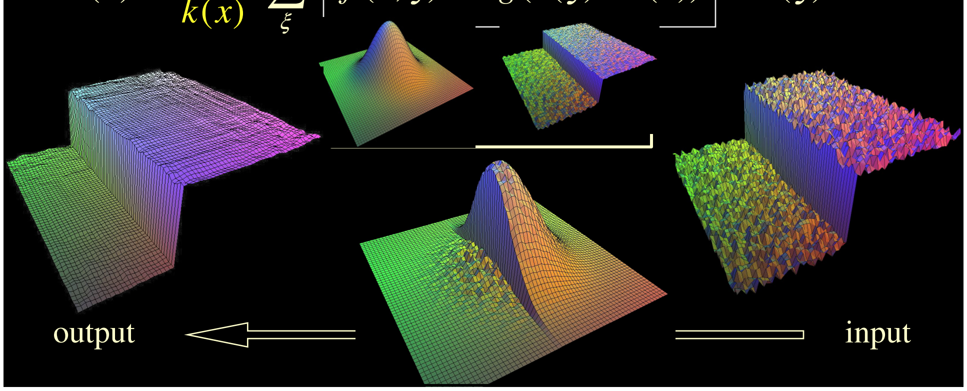
$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi)-I(x)) \quad I(\xi)$$



Normalization factor

- [Tomasi and Manduchi 1998]
- k(x)= $\sum_{\xi} \boxed{f(x,\xi) \quad g(I(\xi)-I(x))}$

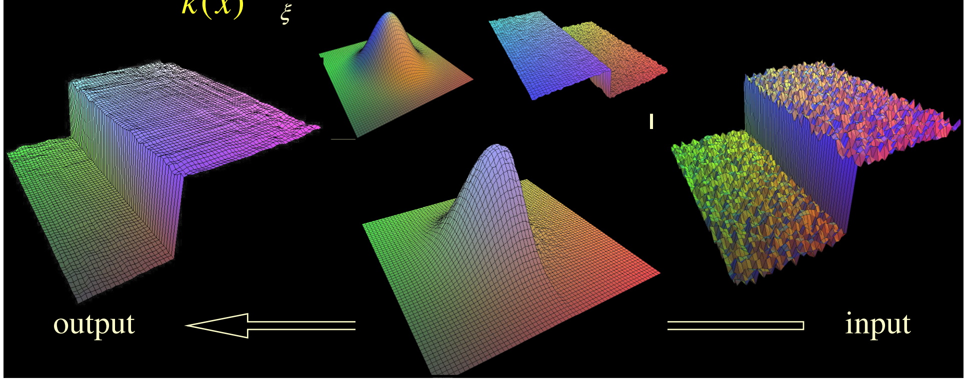
$$J(x) = \frac{1}{k(x)} \sum_{\xi} \boxed{f(x,\xi) \quad g(I(\xi)-I(x))} \quad I(\xi)$$



Bilateral filtering is non-linear

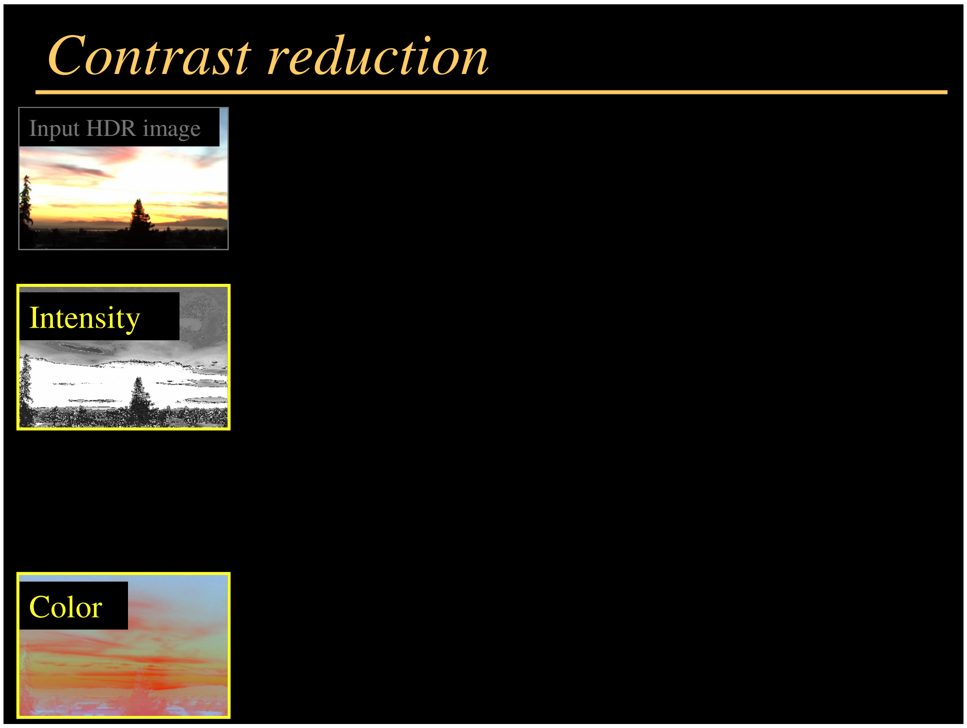
- [Tomasi and Manduchi 1998]
- The weights are different for each output pixel

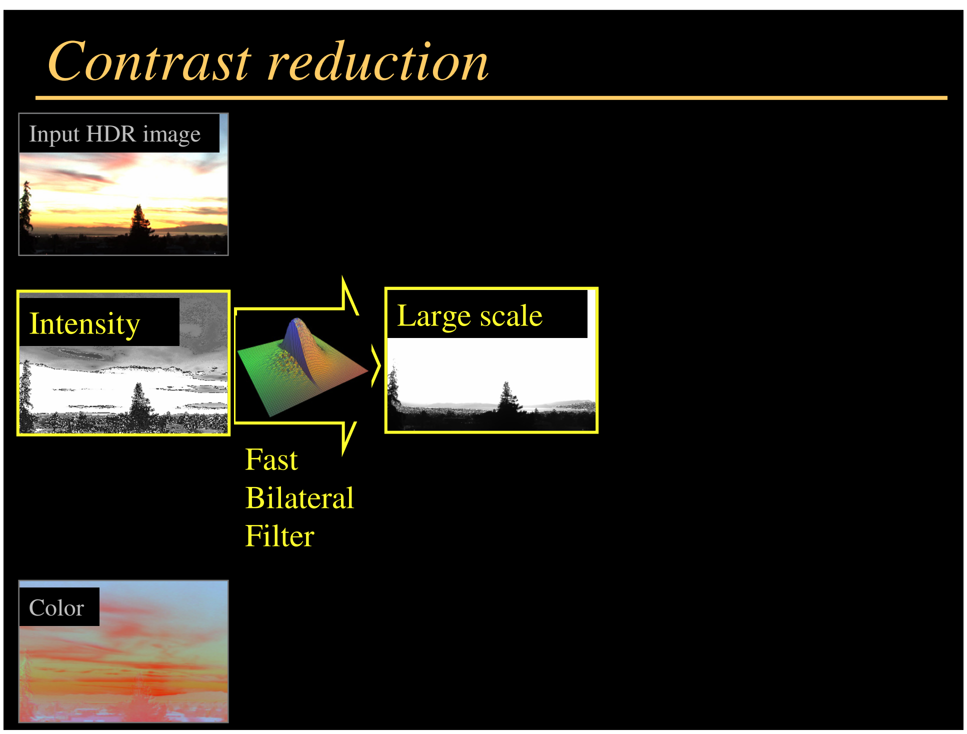
$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi)-I(x)) \quad I(\xi)$$

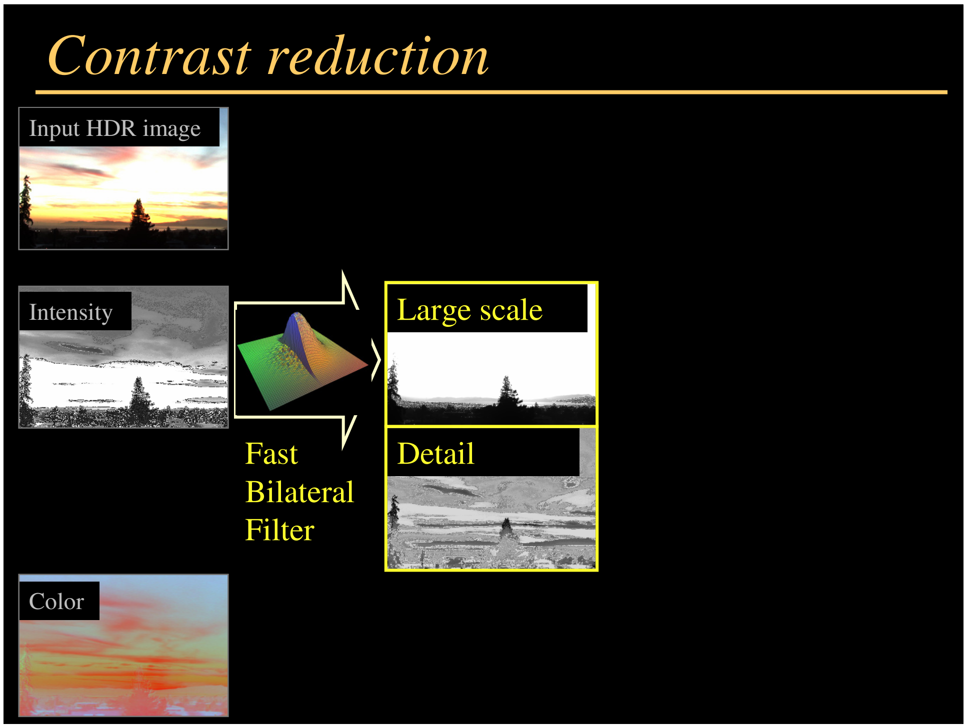


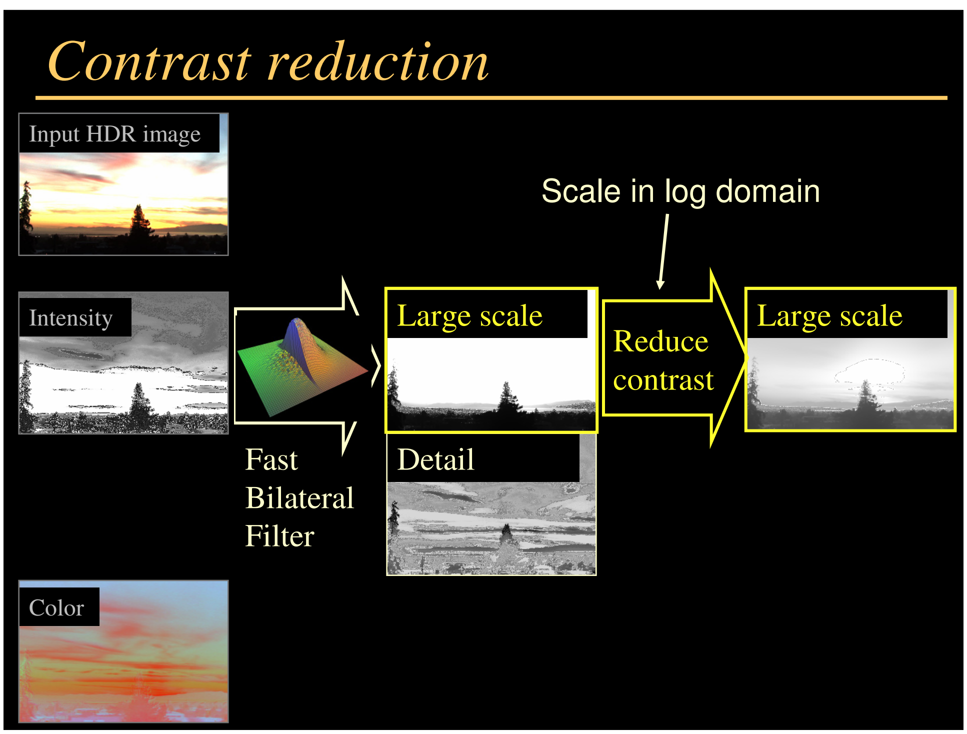
Contrast reduction

Contrast
too high!

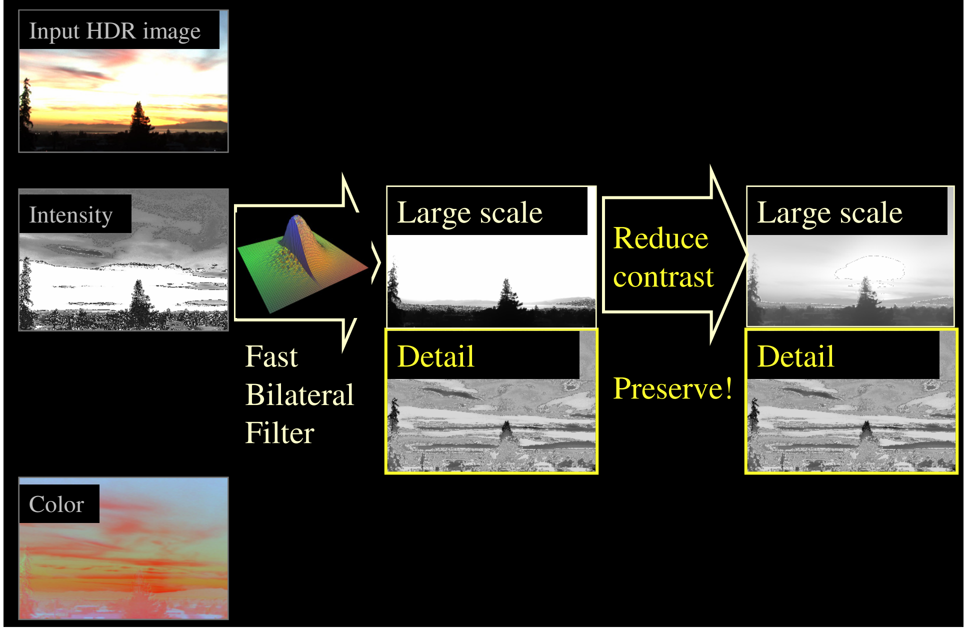

Contrast reduction
Input HDR image
Intensity
Color


Contrast reduction
Input HDR image
Intensity
Large scale
Fast
Bilateral
Filter
Color

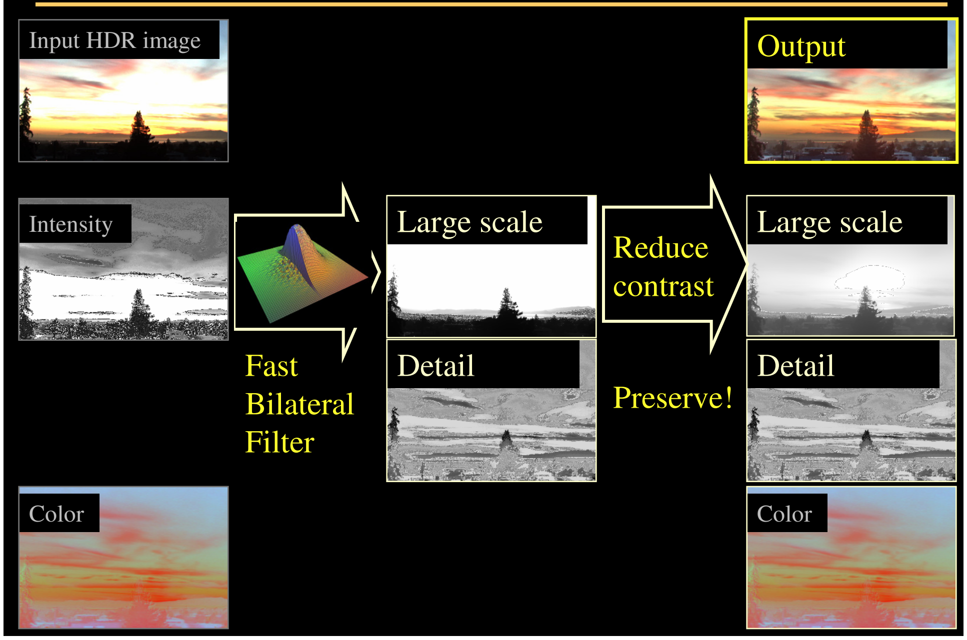

Contrast reduction
Input HDR image
Intensity
Large scale
Detail
Fast
Bilateral
Filter
Color


Contrast reduction
Input HDR image
Scale in log domain
Intensity
Large scale
Reduce
contrast
Large scale
Detail
Fast
Bilateral
Filter
Color

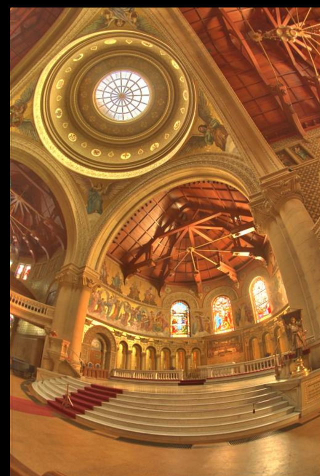
Contrast reduction



Contrast reduction



Informal comparison



Bilateral
[Durand et al.]

Photographic
[Reinhard et al.]

Informal comparison

Bilateral
[Durand et al.]

Photographic
[Reinhard et al.]