Tonemapping and bilateral filtering

http://graphics.cs.cmu.edu/courses/15-463

15-463, 15-663, 15-862
Computational Photography
Fall 2018, Lecture 6
Course announcements

• Homework 2 is out.
  - Due September 28th.
  - Requires camera *and* tripod.
  - Still five cameras left if anybody needs one.
  - Start early! Large programming component and generous bonus.

• Any issues with Homework 1?
  - How did you find homework 1 in general?
  - Which part of homework 1 did you enjoy the most?
Overview of today’s lecture

• Leftover from lecture 5: optimal weights for HDR merging.
• Color calibration and homography estimation.
• Tonemapping.
• Edge-aware filtering and bilateral filtering.
• Back to tonemapping.
• Some notes about HDR and tonemapping.
Slide credits

Many of these slides were inspired or adapted from:

• James Hays (Georgia Tech).
• Fredo Durand (MIT).
• Gordon Wetzstein (Stanford).
• Sylvain Paris (MIT).
• Sam Hasinoff (Google).
Color calibration and homography estimation
Many different spectral sensitivity functions

Each camera has its more or less unique, and most of the time secret, SSF.
- Makes it very difficult to correctly reproduce the color of sensor measurements.

Images of the same scene captured using 3 different cameras with identical sRGB settings.
Color calibration

Apply linear scaling and translation to RGB vectors in the image:

\[ c' = M \cdot c + b \]

transformed RGB vector

original RGB vector

What are the dimensions of each quantity in this equation?
Color calibration

Apply linear scaling and translation to RGB vectors in the image:

\[ c' = M \cdot c + b \]

What are the dimensions of each quantity in this equation?

How do we decide what transformed vectors to map to?
Using (again) a color checker

Color patches manufactured to have pre-calibrated XYZ coordinates.

Calibration chart can be used for:
1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row
Using (again) a color checker

Calibration chart can be used for:

1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row

Color patches manufactured to have pre-calibrated XYZ coordinates.

Can we use any color chart image for color calibration?
Using (again) a color checker

1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row

Color patches manufactured to have pre-calibrated XYZ coordinates.

Can we use any color chart image for color calibration?
- It needs to be a *linear* image!
- Do radiometric calibration first.
Color calibration

Apply linear scaling and translation to RGB vectors in the image:

\[ c' = M \cdot c + b \]

- transformed RGB vector
- original RGB vector

What are the dimensions of each quantity in this equation?

How do we decide what transformed vectors to map to?

How do we solve for matrix M and vector b?
Color calibration

Apply linear scaling and translation to RGB vectors in the image:

\[
\begin{bmatrix}
    c' \\
    1
\end{bmatrix} =
\begin{bmatrix}
    M & b \\
    0 & 1
\end{bmatrix}
\begin{bmatrix}
    c \\
    1
\end{bmatrix}
\]
Color calibration

Apply linear scaling and translation to RGB vectors in the image:

\[
\begin{bmatrix}
    c' \\
    1
\end{bmatrix} = \begin{bmatrix}
    M & b \\
    0 & 1
\end{bmatrix} \begin{bmatrix}
    c \\
    1
\end{bmatrix}
\]

\( C' \quad H \quad C \)
Color calibration

Apply a homography to homogeneous RGB vectors in the image:

\[ C' = H \cdot C \]

How do we solve for a homography transformation?
Determining the homography matrix

Write out linear equation for each color vector correspondence:

\[
C' = H \cdot C \quad \text{or} \quad \begin{bmatrix} r' \\ g' \\ b' \\ 1 \end{bmatrix} = a \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \\ h_5 & h_6 & h_7 & h_8 \\ h_9 & h_{10} & h_{11} & h_{12} \\ h_{13} & h_{14} & h_{15} & h_{13} \end{bmatrix} \begin{bmatrix} r \\ g \\ b \\ 1 \end{bmatrix}
\]
Determining the homography matrix

Write out linear equation for each color vector correspondence:

\[ C' = H \cdot C \quad \text{or} \quad \begin{bmatrix} r' \\ g' \\ b' \\ 1 \end{bmatrix} = a \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \\ h_5 & h_6 & h_7 & h_8 \\ h_9 & h_{10} & h_{11} & h_{12} \\ h_{13} & h_{14} & h_{15} & h_{16} \end{bmatrix} \begin{bmatrix} r \\ g \\ b \\ 1 \end{bmatrix} \]

Expand matrix multiplication:

\[ r' = a(h_1 r + h_2 g + h_3 b + h_4) \]
\[ g' = a(h_5 r + h_6 g + h_7 b + h_8) \]
\[ b' = a(h_9 r + h_{10} g + h_{11} b + h_{12}) \]
\[ 1 = a(h_{13} r + h_{14} g + h_{15} b + h_{16}) \]
Determining the homography matrix

Divide out unknown scale factor:

\[ r'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_1r + h_2g + h_3b + h_4) \]
\[ g'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_5r + h_6g + h_7b + h_8) \]
\[ b'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_9r + h_{10}g + h_{11}b + h_{12}) \]
Determining the homography matrix

Divide out unknown scale factor:

\[ r'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_1r + h_2g + h_3b + h_4) \]
\[ g'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_5r + h_6g + h_7b + h_8) \]
\[ b'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_9r + h_{10}g + h_{11}b + h_{12}) \]

Rearrange as a linear constraint on entries of H:

\[ r'rh_{13} + r'gh_{14} + r'bh_{15} + r'h_{16} - rh_1 - gh_2 - bh_3 - h_4 = 0 \]
\[ g'rh_{13} + g'gh_{14} + g'bh_{15} + g'h_{16} - rh_5 - gh_6 - bh_7 - h_8 = 0 \]
\[ b'rh_{13} + b'gh_{14} + b'bh_{15} + b'h_{16} - rh_9 - gh_{10} - bh_{11} - h_{12} = 0 \]
Determining the homography matrix

Re-write in matrix form:

\[ A_i h = 0 \]

What are the dimensions of each variable in this system?

How many equations from one color vector correspondence?

How many color vector correspondences do we need?
Determining the homography matrix

Re-write in matrix form:

\[ A_i h = 0 \]

Stack together constraints from additional color vector correspondences row-wise:

\[ A h = 0 \]

*Homogeneous* linear least squares system.

- How do we solve such systems?
Determining the homography matrix

Re-write in matrix form:

\[ A_i h = 0 \]

Stack together constraints from additional color vector correspondences row-wise:

\[ A h = 0 \]

*Homogeneous* linear least squares system.

- How do we solve such systems? \( \rightarrow \) Use singular value decomposition (SVD)
General form of total least squares

(Warning: change of notation. \( x \) is a vector of parameters!)

\[
E_{\text{TLS}} = \sum_i (a_i x)^2
\]

\[
= \| A x \|^2
\]

\( \| x \|^2 = 1 \) \hspace{1cm} \text{(matrix form)}

minimize \( \| A x \|^2 \)

subject to \( \| x \|^2 = 1 \)

Solution is the eigenvector corresponding to smallest eigenvalue of \( A^T A \)

(equivalent)

Solution is the column of \( V \) corresponding to smallest singular value

\[
A = U \Sigma V^T
\]
An example

original

color-corrected
Quick note

If you cannot do calibration, take a look at the image’s EXIF data (if available).

Often contains information about tone reproduction curve and color space.
Tonemapping
How do we display our HDR images?

adaptation range of our eyes

common real-world scenes

HDR image

display
Linear scaling

Scale image so that maximum value equals 1

Can you think of something better?
Photographic tonemapping

Apply the same non-linear scaling to all pixels in the image so that:

• Bring everything within range \( \rightarrow \) asymptote to 1
• Leave dark areas alone \( \rightarrow \) slope = 1 near 0

\[
I_{\text{display}} = \frac{I_{\text{HDR}}}{1 + I_{\text{HDR}}}
\]

(exact formula more complicated)

• Photographic because designed to approximate film zone system.
• Perceptually motivated, as it approximates our eye’s response curve.
What is the zone system?

- Technique formulated by Ansel Adams for film development.
- Still used with digital photography.

<table>
<thead>
<tr>
<th>Zone</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Pure black</td>
</tr>
<tr>
<td>I</td>
<td>Near black, with slight tonality but no texture</td>
</tr>
<tr>
<td>II</td>
<td>Textured black; the darkest part of the image in which slight detail is recorded</td>
</tr>
<tr>
<td>III</td>
<td>Average dark materials and low values showing adequate texture</td>
</tr>
<tr>
<td>IV</td>
<td>Average dark foliage, dark stone, or landscape shadows</td>
</tr>
<tr>
<td>V</td>
<td>Middle gray: clear north sky; dark skin, average weathered wood</td>
</tr>
<tr>
<td>VI</td>
<td>Average Caucasian skin; light stone; shadows on snow in sunlit landscapes</td>
</tr>
<tr>
<td>VII</td>
<td>Very light skin; shadows in snow with acute side lighting</td>
</tr>
<tr>
<td>VIII</td>
<td>Lightest tone with texture: textured snow</td>
</tr>
<tr>
<td>IX</td>
<td>Slight tone without texture; glaring snow</td>
</tr>
<tr>
<td>X</td>
<td>Pure white: light sources and specular reflections</td>
</tr>
</tbody>
</table>
Examples
Examples

photographic tonemapping

linear scaling (map 10% to 1)
Compare with LDR images
Dealing with color

If we tonemap all channels the same, colors are washed out

Can you think of a way to deal with this?
Intensity-only tonemapping

How would you implement this?
Comparison

Color now OK, but some details are washed out due to loss of contrast

Can you think of a way to deal with this?
Low-frequency intensity-only tonemapping

tonemap low-frequency intensity component
leave high-frequency intensity component the same
leave color the same

How would you implement this?
Comparison

We got nice color and contrast, but now we’ve run into the halo plague.

Can you think of a way to deal with this?
Edge-aware filtering and bilateral filtering
Motivational example

Let’s say I want to reduce the amount of detail in this picture. What can I do?
Motivational example

What is the problem here?

original  Gaussian filtering
Motivational example

How to smooth out the details in the image without losing the important edges?
Motivational example

original  Gaussian filtering  bilateral filtering
The problem with Gaussian filtering

Why is the output so blurry?
The problem with Gaussian filtering

Input

Gaussian kernel

Blur kernel averages across edges

Output
The bilateral filtering solution

Do not blur if there is an edge! How does it do that?
Bilateral filtering

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k, l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]
Bilateral filtering

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k,l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]

Spatial weighting

\[ \sigma_s \]

Assign a pixel a large weight if:
1) it’s nearby
Bilateral filtering

\[
h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]
\]

Assign a pixel a large weight if:

1) it’s nearby and
2) it looks like me
Bilateral filtering

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k,l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]

- Normalization factor
- Spatial weighting
- Intensity range weighting

Assign a pixel a large weight if:

1) it’s nearby and
2) it looks like me
Bilateral filtering vs Gaussian filtering

Which is which?

\[ h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l] \]

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k,l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]
Bilateral filtering vs Gaussian filtering

Gaussian filtering

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l] \]

Bilateral filtering

\[ h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k, l] f[m+k, n+l] \]
Bilateral filtering vs Gaussian filtering

Gaussian filtering

\[ h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l] \]

Bilateral filtering

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k,l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]

Spatial weighting: favor nearby pixels
Bilateral filtering vs Gaussian filtering

Gaussian filtering

\[ h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l] \]

Bilateral filtering

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k,l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]

Spatial weighting: favor nearby pixels

Intensity range weighting: favor similar pixels
Bilateral filtering vs Gaussian filtering

Gaussian filtering

$$h[m, n] = \sum_{k, l} g[k, l] f[m + k, n + l]$$

Bilateral filtering

$$h[m, n] = \frac{1}{W_{mn}} \sum_{k, l} g[k, l] r_{mn}[k, l] f[m + k, n + l]$$

Spatial weighting: favor nearby pixels

Intensity range weighting: favor similar pixels

Normalization factor
Bilateral filtering vs Gaussian filtering

Gaussian filtering

- Smooths everything nearby (even edges)
- Only depends on spatial distance

Bilateral filtering

- Smooths ‘close’ pixels in space and intensity
- Depends on spatial and intensity distance
Gaussian filtering visualization

\[ h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l] \]
Bilateral filtering visualization

\[ h[m, n] = \frac{1}{W_{mn}} \sum_{k,l} g[k, l] r_{mn}[k, l] f[m + k, n + l] \]

Spatial range

Intensity range

Output

Bilateral Filter

Input
Exploring the bilateral filter parameter space

\[ \sigma_r = 0.1 \quad \sigma_r = 0.25 \quad \sigma_r = \infty \] (Gaussian blur)

\[ \sigma_s = 2 \quad \sigma_s = 6 \quad \sigma_s = 18 \]
Does the bilateral filter respect all edges?
Does the bilateral filter respect all edges?

Bilateral filter crosses (and blurs) thin edges.
Denoising

- Noisy input
- Bilateral filtering
- Median filtering
How would you use Gaussian or bilateral filtering for sharpening?
Photo retouching
Photo retouching

original

digital pore removal (aka bilateral filtering)
Before
After
Close-up comparison

original

digital pore removal (aka bilateral filtering)
Cartoonization

input  
cartoon rendition
Cartoonization

How would you create this effect?
Cartoonization

edges from bilaterally filtered image  bilaterally filtered image  cartoon rendition

Note: image cartoonization and abstraction are very active research areas.
Is the bilateral filter:

Linear?

Shift-invariant?
Is the bilateral filter:

Linear?
- No.

Shift-invariant?
- No.

Does this have any bad implications?
The bilateral grid

Real-time Edge-Aware Image Processing with the Bilateral Grid
Jiawen Chen  Sylvain Paris  Frédéric Durand
Computer Science and Artificial Intelligence Laboratory Massachusetts Institute of Technology

Figure 1: The bilateral grid enables edge-aware image manipulations such as local tone mapping on high resolution images in real time. This 15 megapixel HDR panorama was tone mapped and locally refined using an edge-aware brush at 50 Hz. The inset shows the original input. The process used about 1 MB of texture memory.

Data structure for fast edge-aware image processing.
Modern edge-aware filtering: local Laplacian pyramids
Modern edge-aware filtering: local Laplacian pyramids

input

texture decrease

large texture increase

texture increase
Tonemapping with edge-aware filtering
Tonemapping with edge-aware filtering

local Laplacian pyramids  bilateral filter
Back to tonemapping
Comparison

We got nice color and contrast, but now we’ve run into the halo plague

Can you think of a way to deal with this?
Tonemapping with bilateral filtering
Comparison

We fixed the halos without losing contrast
Gradient-domain merging and tonemapping

Compute gradients, scale and merge them, then integrate (solve Poisson problem).
• More in lecture 7.
Gradient-domain merging and tonemapping
Comparison (which one do you like better?)

photographic  bilateral filtering  gradient-domain
Comparison (which one do you like better?)

photographic  bilateral filtering  gradient-domain
Comparison (which one do you like better?)

photographic  bilateral filtering  gradient-domain
Comparison (which one do you like better?)

There is no ground-truth: which one looks better is entirely subjective

photographic  bilateral filtering  gradient-domain
Tonemapping for a single image

Modern DSLR sensors capture about 3 stops of dynamic range.

- Tonemap single RAW file instead of using camera’s default rendering.

result from image processing pipeline (basic tone reproduction)

tonemapping using bilateral filtering (I think)
Tonemapping for a single image

Modern DSLR sensors capture about 3 stops of dynamic range.
• Tonemap single RAW file instead of using camera’s default rendering.

Careful not to “tonemap” noise.
• Why is this not a problem with multi-exposure HDR?
Some notes about HDR and tonemapping
A note of caution

• HDR photography can produce very visually compelling results
A note of caution

• HDR photography can produce very visually compelling results

• It is also a very routinely abused technique, resulting in awful results
A note of caution

- HDR photography can produce very visually compelling results
- It is also a very routinely abused technique, resulting in awful results
- The problem is tonemapping, not HDR itself
A note about HDR today

• Most cameras (even phone cameras) have automatic HDR modes/apps

• Popular-enough feature that phone manufacturers are actively competing about which one has the best HDR

• The technology behind some of those apps (e.g., Google’s HDR+) is published in SIGGRAPH and SIGGRAPH Asia conferences
References

Basic reading:

  The photographic tonemapping paper, including a very nice discussion of the zone system for film.
• Durand and Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” SIGGRAPH 2002.
  The paper on tonemapping using bilateral filtering.
  Short course on the bilateral filter, including discussion of fast implementations, 
  https://people.csail.mit.edu/sparis/bf_course/
• Fattal et al., “Gradient Domain High Dynamic Range Compression,” SIGGRAPH 2002.
  The paper on gradient-domain tonemapping.

Additional reading:

  A very comprehensive book about everything relating to HDR imaging and tonemapping.
  One of many, many papers trying to do a perceptual evaluation of different tonemapping algorithms.
• Hasinoff et al., “Burst photography for high dynamic range and low-light imaging on mobile cameras,” SIGGRAPH Asia 2016.
  The paper describing Google’s HDR+.
  The paper on local Laplacian pyramids.