Digital photography
Course announcements

• Homework 4 has been posted.
  - Due Friday March 23rd (one-week homework!)
  - Any questions about the homework?
  - How many of you have looked at/started/finished homework 4?

• Talk this week: Katie Bouman, “Imaging the Invisible”.
  - Wednesday, March 21st 10:00 AM GHC6115.
  - How many of you attended this talk?
Overview of today’s lecture

• Leftover from color lecture.
• Imaging sensor primer.
• Color sensing in cameras.
• In-camera image processing pipeline.
• Some general thoughts on the image processing pipeline.
• Radiometric calibration (a.k.a. HDR imaging)
• Color calibration.

Take-home message: The values of pixels in a photograph and the output of your camera’s sensor are two very different things.
Slide credits

A lot of inspiration and quite a few examples for these slides were taken directly from:

- Kayvon Fatahalian (15-769, Fall 2016).
- Michael Brown (CVPR 2016 Tutorial on understanding the image processing pipeline).
Human visual system

\[ c(\ell(\lambda)) = (c_s, c_m, c_l) \]

retinal color: linear radiance measurement

\[ c_s = \int k_s(\lambda)\ell(\lambda)d\lambda \]

\[ \ell(\lambda) = r(\lambda)e(\lambda) \]
spectral radiance

e(\lambda) illuminant spectrum

\[ r(\lambda) \]
spectral reflectance

spectral reflectance
What functional of radiance are we measuring?

\[ \ell(\lambda) = r(\lambda)e(\lambda) \]

spectral radiance

illuminant spectrum

spectral reflectance
The modern photography pipeline
The modern photography pipeline

post-capture processing
(16-385, 15-463)

optics and optical controls
(15-463)

sensor, analog front-end, and color filter array
(this lecture)

in-camera image processing pipeline
(this lecture)
Imaging sensor primer
Imaging sensors

- Very high-level overview of digital imaging sensors.
- We could spend an entire course covering imaging sensors.

Canon 6D sensor (20.20 MP, full-frame)
What does an imaging sensor do?

When the camera shutter opens...

... exposure begins...

array of photon buckets

close-up view of photon buckets

... photon buckets begin to store photons...

... until the camera shutter closes. Then, they convert stored photons to intensity values.
Nobel Prize in Physics

Do you know who this is?
Photoelectric effect

incident photons

emitted electrons

Einstein’s Nobel Prize in 1921 “for his services to Theoretical Physics, and especially for his discovery of the law of the photoelectric effect”
Basic imaging sensor design

- Made of silicon, emits electrons from photons.
- Photodiode helps collect more light (also called lenslet).
- Color filter helps photodiode collect more light.
- Microlens helps collect more light.

- Potential well stores emitted electrons.
- Silicon for read-out etc. circuitry.

- Lenslets also filter the image to avoid resolution artifacts.
- Lenslets are problematic when working with coherent light.
- Many modern cameras do not have lenslet arrays.

We will discuss these issues in more detail at a later lecture.

We will see what the color filters are for later in this lecture.
Photodiode quantum efficiency (QE)

How many of the incident photons will the photodiode convert into electrons?

\[
\text{QE} = \frac{\# \text{ electrons}}{\# \text{ photons}}
\]

- Fundamental optical performance metric of imaging sensors.
- Not the only important optical performance metric!
- We will see a few more later in the lecture.
Photodiode response function

For silicon photodiodes, usually linear, but:

- non-linear when potential well is saturated (over-exposure)
- non-linear near zero (due to noise)

We will see how to deal with these issues in a later lecture (high-dynamic-range imaging).

Photodiode full well capacity

How many electrons can photodiode store before saturation?

- Another important optical performance metric of imaging sensors.
Two main types of imaging sensors

**Charged Coupled Device (CCD):**
converts electrons to voltage using readout circuitry separate from pixel

**Complementary Metal Oxide Semiconductor (CMOS):**
converts electrons to voltage using per-pixel readout circuitry

Can you think of advantages and disadvantages of each type?
Two main types of imaging sensors

**Charged Coupled Device (CCD):** converts electrons to voltage using readout circuitry separate from pixel

- ✓ higher sensitivity
- ✓ lower noise

**Complementary Metal Oxide Semiconductor (CMOS):** converts electrons to voltage using per-pixel readout circuitry

- ✓ faster read-out
- ✓ lower cost
### CCD vs CMOS

- **Modern CMOS sensors** have optical performance comparable to CCD sensors.

- **Most modern commercial and industrial cameras** use CMOS sensors.

---

Can you guess what the QE of the human eye is?
CMOS sensor (very) simplified layout

- photodiode (pixel)
- row selection register
- active pixel sensor (2D array of pixels)
- exposed region (light gets here)
- optically black region (no light gets here)

Can anyone guess why there are pixels in the optically black region?

row buffer

analog front-end

bits
Analog front-end

**analog amplifier (gain):**
- Gets voltage in range needed by A/D converter.
- Accommodates ISO settings.
- Accounts for vignetting.

**analog-to-digital converter (ADC):**
- Depending on sensor, output has 10-16 bits.
- Most often (?) 12 bits.

**look-up table (LUT):**
- Corrects non-linearities in sensor’s response function (within proper exposure).
- Corrects defective pixels.

A diagram illustrates the flow from analog voltage to discrete signal, with each block representing a component of the analog front-end system.
Vignetting

Fancy word for: pixels far off the center receive less light

white wall under uniform light

more interesting example of vignetting
Vignetting

Four types of vignetting:

- Mechanical: light rays blocked by hoods, filters, and other objects.
- Lens: similar, but light rays blocked by lens elements.
- Natural: due to radiometric laws (“cosine fourth falloff”).
- Pixel: angle-dependent sensitivity of photodiodes.
What does an imaging sensor do?

When the camera shutter opens, the sensor:

- at every photodiode, converts incident photons into electrons
- stores electrons into the photodiode’s potential well until it is full

... until camera shutter closes. Then, the analog front-end:

- reads out photodiodes’ wells, row-by-row, and converts them to analog signals
- applies a (possibly non-uniform) gain to these analog signals
- converts them to digital signals
- corrects non-linearities

... and finally returns an image.
Remember these?

- Lenslets also filter the image to avoid resolution artifacts.
- Lenslets are problematic when working with coherent light.
- Many modern cameras do not have lenslet arrays.

We will discuss these issues in more detail at a later lecture.

- Silicon for read-out etc. circuitry
- Stores emitted electrons
- Helps photodiode collect more light (also called lenslet)
- Made of silicon, emits electrons from photons

We will see what the color filters are for later in this lecture.
Color sensing in cameras
Color is an artifact of human perception

- “Color” is not an *objective* physical property of light (electromagnetic radiation).
- Instead, light is characterized by its wavelength.

What we call “color” is how we *subjectively* perceive a very small range of these wavelengths.
Spectral Sensitivity Function (SSF)

- Any light sensor (digital or not) has different sensitivity to different wavelengths.
- This is described by the sensor’s spectral sensitivity function $f(\lambda)$.
- When measuring light of a some SPD $\Phi(\lambda)$, the sensor produces a scalar response:

\[
R = \int_{\lambda} \Phi(\lambda) f(\lambda) d\lambda
\]

Weighted combination of light’s SPD: light contributes more at wavelengths where the sensor has higher sensitivity.
Spectral Sensitivity Function of Human Eye

- The human eye is a collection of light sensors called cone cells.
- There are three types of cells with different spectral sensitivity functions.
- Human color perception is three-dimensional (tristimulus color).

\[
\begin{align*}
\text{“short”} & : S = \int_{\lambda} \Phi(\lambda) S(\lambda) d\lambda \\
\text{“medium”} & : M = \int_{\lambda} \Phi(\lambda) M(\lambda) d\lambda \\
\text{“long”} & : L = \int_{\lambda} \Phi(\lambda) L(\lambda) d\lambda
\end{align*}
\]

cone distribution for normal vision (64% L, 32% M)
Color filter arrays (CFA)

• To measure color with a digital sensor, mimic cone cells of human vision system.
• “Cones” correspond to pixels that are covered by different color filters, each with its own spectral sensitivity function.
What color filters to use?

Two design choices:

- What spectral sensitivity functions $f(\lambda)$ to use for each color filter?
- How to spatially arrange ("mosaic") different color filters?

Bayer mosaic

Why more green pixels?

SSF for Canon 50D

Generally do not match human LMS.

$f(\lambda)$
Many different CFAs

Finding the “best” CFA mosaic is an active research area.

How would you go about designing your own CFA? What criteria would you consider?
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.
Aside: can you think of other ways to capture color?
Aside: can you think of other ways to capture color?

- **Field sequential**
- **Multiple sensors**
- **Vertically stacked**

[Slide credit: Gordon Wetzstein]
What does an imaging sensor do?

When the camera shutter opens, the sensor:

• at every photodiode, converts incident photons into electrons using mosaic’s SSF
• stores electrons into the photodiode’s potential well until it is full

... until camera shutter closes. Then, the analog front-end:

• reads out photodiodes’ wells, row-by-row, and converts them to analog signals
• applies a (possibly non-uniform) gain to these analog signals
• converts them to digital signals
• corrects non-linearities

... and finally returns an image.
After all of this, what does an image look like?

- Kind of disappointing.
- We call this the *RAW* image.
The modern photography pipeline

- Post-capture processing (lectures 3-12)
- Optics and optical controls (lectures 13-16)
- Sensor, analog front-end, and color filter array (this lecture)
- In-camera image processing pipeline (this lecture)
The in-camera image processing pipeline
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

- RAW image (mosaiced, linear, 12-bit)
- analog front-end
- white balance
- CFA demosaicing
- tone reproduction
- compression
- final RGB image (non-linear, 8-bit)
- color transforms
- denoising
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

- Image signal processor
- Analog front-end
- RAW image (mosaiced, linear, 12-bit)
- Denoising
- CFA demosaicing
- White balance
- Tone reproduction
- Compression
- Final RGB image (non-linear, 8-bit)

See color lecture
See 18-793

Color transforms
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

- RAW image (mosaiced, linear, 12-bit)
- Analog front-end
- White balance
- Tone reproduction
- Compression
- Final RGB image (non-linear, 8-bit)
- Demosaicing
- Color transforms
- Denoising
White balancing

Human visual system has *chromatic adaptation*:

- We can perceive white (and other colors) correctly under different light sources.

Retinal vs perceived color.
White balancing

Human visual system has *chromatic adaptation*:
- We can perceive white (and other colors) correctly under different light sources.
- Cameras cannot do that (there is no “camera perception”).

White balancing: The process of removing color casts so that colors that we would *perceive* as white are *rendered* as white in final image.

different whites

image captured under fluorescent

image white-balanced to daylight
White balancing presets

Cameras nowadays come with a large number of presets: You can select which light you are taking images under, and the appropriate white balancing is applied.

<table>
<thead>
<tr>
<th>WB SETTINGS</th>
<th>COLOR TEMPERATURE</th>
<th>LIGHT SOURCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10000 - 15000 K</td>
<td>Clear Blue Sky</td>
</tr>
<tr>
<td></td>
<td>6500 - 8000 K</td>
<td>Cloudy Sky / Shade</td>
</tr>
<tr>
<td></td>
<td>6000 - 7000 K</td>
<td>Noon Sunlight</td>
</tr>
<tr>
<td></td>
<td>5500 - 6500 K</td>
<td>Average Daylight</td>
</tr>
<tr>
<td></td>
<td>5000 - 5500 K</td>
<td>Electronic Flash</td>
</tr>
<tr>
<td></td>
<td>4000 - 5000 K</td>
<td>Fluorescent Light</td>
</tr>
<tr>
<td></td>
<td>3000 - 4000 K</td>
<td>Early AM / Late PM</td>
</tr>
<tr>
<td></td>
<td>2500 - 3000 K</td>
<td>Domestic Lighting</td>
</tr>
<tr>
<td></td>
<td>1000 - 2000 K</td>
<td>Candle Flame</td>
</tr>
</tbody>
</table>
Manual vs automatic white balancing

Manual white balancing:
• Manually select object in photograph that is color-neutral and use it to normalize.
• Select a camera preset based on lighting.

How can we do automatic white balancing?
Manual vs automatic white balancing

Manual white balancing:
• Manually select object in photograph that is color-neutral and use it to normalize.
• Select a camera preset based on lighting.

Automatic white balancing:
• Grey world assumption: force average color of scene to be grey.
• White world assumption: force brightest object in scene to be white.
• Sophisticated histogram-based algorithms (what most modern cameras do).
Automatic white balancing

Grey world assumption:
- Compute per-channel average.
- Normalize each channel by its average.
- Normalize by green channel average.

White world assumption:
- Compute per-channel maximum.
- Normalize each channel by its maximum.
- Normalize by green channel maximum.

\[
\begin{bmatrix}
  R' \\
  G' \\
  B'
\end{bmatrix}
= \begin{bmatrix}
  \frac{G_{avg}}{R_{avg}} & 0 & 0 \\
  0 & 1 & 0 \\
  0 & 0 & \frac{G_{avg}}{B_{avg}}
\end{bmatrix}
\begin{bmatrix}
  R \\
  G \\
  B
\end{bmatrix}
\]

\[
\begin{bmatrix}
  R' \\
  G' \\
  B'
\end{bmatrix}
= \begin{bmatrix}
  \frac{G_{max}}{R_{max}} & 0 & 0 \\
  0 & 1 & 0 \\
  0 & 0 & \frac{G_{max}}{B_{max}}
\end{bmatrix}
\begin{bmatrix}
  R \\
  G \\
  B
\end{bmatrix}
\]
Automatic white balancing example

input image  grey world  white world
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

- Analog front-end
- RAW image (mosaiced, linear, 12-bit)
- Final RGB image (non-linear, 8-bit)
- Denoising
- CFA demosaicing
- White balance
- Color transforms
- Tone reproduction
- Compression
CFA demosaicing

Produce full RGB image from mosaiced sensor output.

Any ideas on how to do this?
CFA demosaicing

Produce full RGB image from mosaiced sensor output.

Interpolate from neighbors:
• Bilinear interpolation (needs 4 neighbors).
• Bicubic interpolation (needs more neighbors, may overblur).
• Edge-aware interpolation.
Large area of research.
Demosaicing by bilinear interpolation

Bilinear interpolation: Simply average your 4 neighbors.

$$G_? = \frac{G_1 + G_2 + G_3 + G_4}{4}$$

Neighborhood changes for different channels:
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

1. RAW image (mosaiced, linear, 12-bit)
2. denoising
3. CFA demosaicing
4. white balance
5. tone reproduction
6. compression
7. final RGB image (non-linear, 8-bit)
Noise in images

Can be very pronounced in low-light images.
Three types of sensor noise

1) (Photon) shot noise:
   • Photon arrival rates are a random process (Poisson distribution).
   • The brighter the scene, the smaller the variance of the distribution.

2) Dark-shot noise:
   • Emitted electrons due to thermal activity (becomes worse as sensor gets hotter.)

3) Read noise:
   • Caused by read-out and AFE electronics (e.g., gain, A/D converter).

Bright scene and large pixels: photon shot noise is the main noise source.
How to denoise?
How to denoise?

Simple denoising: look at the neighborhood around you.

- Mean filtering (take average):
  \[
  l'_5 = \frac{l_1 + l_2 + l_3 + l_4 + l_5 + l_6 + l_7 + l_8 + l_9}{9}
  \]

- Median filtering (take median):
  \[
  l'_5 = \text{median}(l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8, l_9)
  \]

Large area of research.
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

1. **RAW image (mosaiced, linear, 12-bit)**
2. **analog front-end**
3. **white balance**
4. **CFA demosaicing**
5. **tone reproduction**
6. **compression**
7. **final RGB image (non-linear, 8-bit)**
8. **color transforms**
9. **denoising**
Tone reproduction

- Also known as gamma correction.
- Without tone reproduction, images look very dark.

Why does this happen?
Perceived vs measured brightness by human eye

We have already seen that sensor response is linear.

Human-eye response (measured brightness) is also linear.

However, human-eye perception (perceived brightness) is non-linear:
• More sensitive to dark tones.
• Approximately a Gamma function.
What about displays?

We have already seen that sensor response is linear.

Human-eye *response* (measured brightness) is also linear.

However, human-eye *perception* (perceived brightness) is *non-linear*:
- More sensitive to dark tones.
- Approximately a Gamma function.

Displays have a response opposite to that of human perception.
Tone reproduction

- Because of mismatch in displays and human eye perception, images look very dark.

How do we fix this?
Tone reproduction

• Because of mismatch in displays and human eye perception, images look very dark.

• Pre-emptively cancel-out the display response curve.
• Add inverse display transform here.
• This transform is the tone reproduction or gamma correction.
The exact tone reproduction curve depends on the camera.

- Often well approximated as $L^\gamma$, for different values of the power $\gamma$ ("gamma").
- A good default is $\gamma = 1/2.2$.

Warning: Our values are no longer linear relative to scene radiance!
Tone reproduction

Question: Why not just keep measurements linear and do gamma correction right before we display the image?
Tone reproduction

Question: Why not just keep measurements linear and do gamma correction right before we display the image?

Answer: After this stage, we perform compression, which includes change from 12 to 8 bits. • Better to use our available bits to encode the information we are going to need.
The (in-camera) image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

- RAW image (mosaiced, linear, 12-bit)
- analog front-end
- CFA demosaicing
- white balance
- tone reproduction
- compression
- final RGB image (non-linear, 8-bit)
- color transforms
- denoising
Some general thoughts on the image processing pipeline
Do I ever need to use RAW?
Do I ever need to use RAW?

Emphatic yes!

- Every time you use a physics-based computer vision algorithm, you need linear measurements of radiance.
- Examples: photometric stereo, shape from shading, image-based relighting, illumination estimation, anything to do with light transport and inverse rendering, etc.
- Applying the algorithms on non-linear (i.e., not RAW) images will produce completely invalid results.
What if I don’t care about physics-based vision?
What if I don’t care about physics-based vision?

You often still *want* (rather than need) to use RAW!

• If you like re-finishing your photos (e.g., on Photoshop), RAW makes your life much easier and your edits much more flexible.
Is it even possible to get access to RAW images?
Is it even possible to get access to RAW images?

Quite often yes!

• Most DSLR cameras provide an option to store RAW image files.

• Certain phone cameras allow, directly or indirectly, access to RAW.

• Sometimes, it may not be “fully” RAW. The Lightroom app provides images after demosaicking but before tone reproduction.
I forgot to set my camera to RAW, can I still get the RAW file?

Nope, tough luck.

- The image processing pipeline is lossy: After all the steps, information about the original image is lost.

- Sometimes we may be able to reverse a camera’s image processing pipeline if we know exactly what it does (e.g., by using information from other similar RAW images).

- The conversion of PNG/JPG back to RAW is know as “de-rendering” and is an active research area.
Derendering

Spectral scene radiance → ? → Output RGB image

RAW
JPEG/sRGB

Panasonic DMC-LX3
Why did you use italics in the previous slide?

What I described today is an “idealized” version of what we *think* commercial cameras do.

- Almost all of the steps in both the sensor and image processing pipeline I described earlier are camera-dependent.

- Even if we know the basic steps, the implementation details are proprietary information that companies actively try to keep secret.

- I will go back to a few of my slides to show you examples of the above.
The hypothetical image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

- RAW image (mosaiced, linear, 12-bit)
- analog front-end?
- denoising?
- CFA demosaicing?
- white balance?
- color transforms?
- tone reproduction?
- compression?
- final RGB image (non-linear, 8-bit)
Various curves

All of these sensitivity curves are different from camera to camera and kept secret.
Radiometric calibration
(a.k.a. high dynamic range imaging)
(a.k.a. capturing linear images)
The image processing pipeline

Which parts of the image processing pipeline introduce non-linearities?

RAW image → analog front-end

denoising ← CFA demosaicing

color transforms ← tone reproduction

denoising ← white balance

color transforms ← compression

tone reproduction ← final RGB image

denoising ← final RGB image

CFA demosaicing → tone reproduction

denoising → white balance

color transforms → tone reproduction

denoising → compression

color transforms → compression

color transforms → final RGB image

color transforms → final RGB image

color transforms → final RGB image
The image processing pipeline

Is using RAW images sufficient to get linear images?

- denoising
- CFA demosaicing
- white balance
- color transforms
- tone reproduction
- compression

RAW image (mosaiced, linear, 12-bit)

final RGB image (non-linear, 8-bit)
Photodiode response function

For silicon photodiodes, usually linear, but:

- **non-linear when potential well is saturated (over-exposure)**
- **non-linear near zero (due to noise)**

We will see how to deal with these issues in a later lecture (high-dynamic-range imaging).
Over/under exposure

in shadows we are limited by noise

in highlights we are limited by clipping
Our devices do not match the world
The world has a high dynamic range.
The world has a high dynamic range

- Common real-world scenes
- Adaptation range of our eyes
(Digital) sensors also have a low dynamic range

- Sensor dynamic range: $10^{-6}$ to $10^6$
- Common real-world scenes: $10^{-6}$ to $10^6$
- Adaptation range of our eyes: $10^{-6}$ to $10^6$
(Digital) images have an even lower dynamic range than common real-world scenes and the adaptation range of our eyes.
(Digital) images have an even lower dynamic range

adaptation range of our eyes

common real-world scenes

high exposure

$10^{-6}$ $10^6$
Our devices do not match the real world

- 10:1 photographic print (higher for glossy paper)
- 20:1 artist's paints
- 200:1 slide film
- 500:1 negative film
- 1000:1 LCD display
- 2000:1 digital SLR (at 12 bits)
- 100000:1 real world

Two challenges:

1. HDR imaging – which parts of the world to include to the 8-12 bits available to our device?
2. Tonemapping – which parts of the world to display in the 4-10 bits available to our device?
Key idea

1. Capture multiple LDR images at different exposures

2. Merge them into a single HDR image
Key idea

1. Capture multiple LDR images at different exposures

2. Merge them into a single HDR image
Ways to vary exposure

1. Shutter speed

2. F-stop (aperture, iris)

3. ISO

4. Neutral density (ND) filters

Pros and cons of each?
Ways to vary exposure

1. Shutter speed
   - Range: about 30 sec to 1/4000 sec (6 orders of magnitude)
   - Pros: repeatable, linear
   - Cons: noise and motion blur for long exposure

2. F-stop (aperture, iris)
   - Range: about f/0.98 to f/22 (3 orders of magnitude)
   - Pros: fully optical, no noise
   - Cons: changes depth of field

3. ISO
   - Range: about 100 to 1600 (1.5 orders of magnitude)
   - Pros: no movement at all
   - Cons: noise

3. Neutral density (ND) filters
   - Range: up to 6 densities (6 orders of magnitude)
   - Pros: works with strobe/flash
   - Cons: not perfectly neutral (color shift), extra glass (interreflections, aberrations), need to touch camera (shake)
Shutter speed

Note: shutter times usually obey a power series – each “stop” is a factor of 2

1/4, 1/8, 1/15, 1/30, 1/60, 1/125, 1/250, 1/500, 1/1000 sec

usually really is

1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256, 1/512, 1/1024 sec

Questions:
1. How many exposures?
2. What exposures?
Shutter speed

Note: shutter times usually obey a power series – each “stop” is a factor of 2

1/4, 1/8, 1/15, 1/30, 1/60, 1/125, 1/250, 1/500, 1/1000 sec

usually really is

1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256, 1/512, 1/1024 sec

Questions:
1. How many exposures?
2. What exposures?

Answer: Depends on the scene, but a good default is 5 exposures, metered exposure and +/- 2 stops around that
Key idea

1. Capture multiple LDR images at different exposures

2. Merge them into a single HDR image
RAW images have a linear response curve

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

No need for calibration in this case
Over/under exposure

In highlights we are limited by clipping.

In shadows we are limited by noise.
RAW (linear) image formation model

Real scene radiance for image pixel \((x,y)\): \(L(x, y)\)

Exposure time:

\[ t_5 \quad t_4 \quad t_3 \quad t_2 \quad t_1 \]

What is an expression for the image \(I(x,y)\) as a function of \(L(x,y)\)?
RAW (linear) image formation model

Real scene radiance for image pixel \((x, y)\): \(L(x, y)\)

Exposure time:

\[ t_5 \quad t_4 \quad t_3 \quad t_2 \quad t_1 \]

What is an expression for the image \(I_{\text{linear}}(x, y)\) as a function of \(L(x, y)\)?

\[
I_{\text{linear}}(x, y) = \text{clip} \left[ t_i \cdot L(x, y) + \text{noise} \right]
\]

How would you merge these images into an HDR one?
Merging RAW (linear) exposure stacks

For each pixel:

1. Find “valid” images
2. Weight valid pixel values appropriately
3. Form a new pixel value as the weighted average of valid pixel values

How would you implement steps 1-2?
Merging RAW (linear) exposure stacks

For each pixel:

1. Find “valid” images

2. Weight valid pixel values appropriately

3. Form a new pixel value as the weighted average of valid pixel values

(noise) $0.05 < \text{pixel} < 0.95$ (clipping)
Merging RAW (linear) exposure stacks

For each pixel:

1. Find “valid” images

2. Weight valid pixel values appropriately

3. Form a new pixel value as the weighted average of valid pixel values

\[
\text{new pixel value} = \frac{\text{pixel value}}{t_i}
\]

(noise) 0.05 < pixel < 0.95 (clipping)
Merging result (after tonemapping)
What if I cannot use raw?
The image processing pipeline

The sequence of image processing operations applied by the camera’s image signal processor (ISP) to convert a RAW image into a “conventional” image.

RAW image (mosaiced, linear, 12-bit) → analog front-end

→ white balance

→ tone reproduction

→ compression

→ final RGB image (non-linear, 8-bit)

RAW image (mosaiced, linear, 12-bit)
Processed images have a non-linear response curve

We must calibrate the response curve

Calibration chart can be used for:
1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row
The image processing pipeline

Which part of the pipeline does the non-linear response curve correspond to?

- denoising
- CFA demosaicing
- white balance
- tone reproduction
- compression

RAW image (mosaiced, linear, 12-bit)

final RGB image (non-linear, 8-bit)
The image processing pipeline

Which part of the pipeline does the non-linear response curve correspond to?
• The tone reproduction (mostly).
Non-linear image formation model

Real scene radiance for image pixel \((x,y)\): \(L(x,y)\)

Exposure time: \(t_i\)

\[
I_{\text{linear}}(x,y) = \text{clip}(t_i \cdot L(x,y) + \text{noise})
\]

\[
I_{\text{non-linear}}(x,y) = f[I_{\text{linear}}(x,y)]
\]

How would you merge the non-linear images into an HDR one?
Non-linear image formation model

Real scene radiance for image pixel \((x,y)\): \(L(x, y)\)

Exposure time: \(t_i\)

\[
I_{\text{linear}}(x,y) = \text{clip}[ t_i \cdot L(x,y) + \text{noise} ]
\]

\[
I_{\text{non-linear}}(x,y) = f[I_{\text{linear}}(x,y)]
\]

\[
I_{\text{est}}(x,y) = f^{-1}[ I_{\text{non-linear}}(x,y) ]
\]

Use inverse transform to estimate linear image, then proceed as before
Linearization

\[ I_{\text{non-linear}}(x,y) = f[I_{\text{linear}}(x,y)] \]

\[ I_{\text{est}}(x,y) = f^{-1}[I_{\text{non-linear}}(x,y)] \]
Merging non-linear exposure stacks

1. Calibrate response curve

2. Linearize images

For each pixel:

3. Find “valid” images

4. Weight valid pixel values appropriately

5. Form a new pixel value as the weighted average of valid pixel values

Note: many possible weighting schemes
What if I cannot measure response curve?
The exact tone reproduction curve depends on the camera.

- Often well approximated as $L^\gamma$, for different values of the power $\gamma$ ("gamma").
- A good default is $\gamma = 1 / 2.2$.

If nothing else, take the square of your image to approximately remove effect of tone reproduction curve.
Relative vs absolute radiance

Final fused HDR image gives radiance only up to a global scale

• If we know exact radiance at one point, we can convert relative HDR image to absolute radiance map
Basic HDR approach

1. Capture multiple LDR images at different exposures
2. Merge them into a single HDR image

Any problems with this approach?
Basic HDR approach

1. Capture multiple LDR images at different exposures
2. Merge them into a single HDR image

Problem: Very sensitive to movement

- Scene must be completely static
- Camera must not move

Most modern automatic HDR solutions include an alignment step before merging exposures
Another type of HDR images

Light probes: place a chrome sphere in the scene and capture an HDR image
• Used to measure real-world illumination environments ("environment maps")

Application: image-based relighting (later lecture)
Another way to create HDR images

Physics-based renderers simulate radiance maps (relative or absolute)

• Their outputs are very often HDR images
A note about HDR today

- Most cameras (even phone cameras) have automatic HDR modes/apps
- The technology behind some of those apps (e.g., Google’s HDR+) is published in SIGGRAPH and SIGGRAPH Asia conferences

Figure 1: A comparison of a conventional camera pipeline (left), middle and our burst photography pipeline (right) mapping on the same cell phone camera. In this low-light setting (about 0 lux), the conventional camera pipeline underexposes (left). Refocusing the image (middle) results in heavy spatial denoising, which results in loss of detail and an unpleasant blocky appearance. For a true low-light image, increasing the signal-to-noise ratio using a spatial-domain denoising algorithm is necessary. We encourage the reader to walk in a wide pipeline exercise in low-light and high dynamic range scene as an example of the latter (see figure 11), it is computationally efficient and relatively artifact-free, as it can be deployed on a mobile camera and used as a substitute for the conventional pipeline in almost all circumstances. For readability, the figure has been made uniformly brighter than the original photograph.

Abstract

Cell phone cameras have small apertures, which limits the number of photons they can gather. Scaling to today’s images is hard. They also have small sensor sizes, which limits the number of electrons that can be read out. Scaling to human dynamic range, we describe a computational photography pipeline that captures, aligns, and merges a large number of frames to reduce sensor and lens-based dynamic range. The system has several key features that help scale to robust and efficient. First, we do not use burst recording. Instead, we capture a series of constant exposure, which allows alignment, motion, and lens distortion enough to avoid motion on highlights. The resulting merged image has deep shadows and high depth, allowing us to apply standard HDR image mapping methods. Second, we begin from captured frames rather than the downscaled RGB or YUV frames produced by hardware Image Signal Processors (ISP) common on mobile platforms. This gives us very high per-pixel and allows us to circumvent the ISP’s saturated noise mapping and spatial denoising. Third, we use a novel, RFP-based alignment algorithm and a hybrid (TFF) Warren filter to detect and merge the frames in a set. Our implementation is built atop Android’s Camera API, which provides pre-frame camera control and access to new image, and is written in the Rust programming-language. Overall, it takes 4 seconds on device (for a 12 Mpix image). requires no user intervention, and ships on several major-produced cell phones.

Keywords: computational photography, high dynamic range

Concepts: Imaging and computer vision; Computational photography; Image processing;

1 Introduction

The main technical impediment to burst photography is lack of light. In bright or average-light scenes, the scene is a simple way to produce meaningful light. The natural solution is either to apply analog or digital gain, which amplifies noise, or to lengthen exposure time, which increases motion blur due to camera shake or subject motion. Surprisingly, daytime shots with high dynamic range may also suffer from lack of light. In particular, if exposure time is increased to avoid blur, Complementary metal-oxide-semiconductor image sensors cannot capture enough light.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions Dept, ACM, fax +1 (212) 869-0481.
Color calibration
(a.k.a., measuring your camera’s color space)
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.
Linear color spaces

basis for retinal color ⇔ color matching functions ⇔ primary colors ⇔ color space

\[
\begin{bmatrix}
\mathbf{c}(\ell(\lambda)) \\
\mathbf{c}(\ell_{435}) & \mathbf{c}(\ell_{545}) & \mathbf{c}(\ell_{625})
\end{bmatrix}
= \begin{bmatrix}
\mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3
\end{bmatrix}
\begin{bmatrix}
\int k_1(\lambda)\ell(\lambda)d\lambda \\
\int k_2(\lambda)\ell(\lambda)d\lambda \\
\int k_3(\lambda)\ell(\lambda)d\lambda
\end{bmatrix}
\]

\[
\begin{bmatrix}
\mathbf{c}(\ell(\lambda)) \\
\mathbf{c}(\ell_{435}) & \mathbf{c}(\ell_{545}) & \mathbf{c}(\ell_{625})
\end{bmatrix} M^{-1} = \begin{bmatrix}
k_1(\lambda) \\
k_2(\lambda) \\
k_3(\lambda)
\end{bmatrix} = M \begin{bmatrix}
k_{435}(\lambda) \\
k_{545}(\lambda) \\
k_{625}(\lambda)
\end{bmatrix}
\]

\[
\begin{bmatrix}
\mathbf{c}(\ell(\lambda)) \\
\mathbf{c}(\ell_{435}) & \mathbf{c}(\ell_{545}) & \mathbf{c}(\ell_{625})
\end{bmatrix} M^{-1} M \text{ can insert any invertible } M
\]

representation of retinal color in LMS space

change of basis matrix

representation of retinal color in space of primaries
Linear color spaces

Change of color space:

\[ c' = H \cdot c \]

desired reference color space (i.e., XYZ)  
camera color space

What does this look like?
Linear color spaces

Change of color space:

\[ c' = H \cdot c \]

- desired reference color space (i.e., XYZ)
- camera color space

What does this look like?
- It’s a homography!

How do we compute homographies?
Linear color spaces

Change of color space:

\[ c' = H \cdot c \]

desired reference color space (i.e., XYZ)

camera color space

What does this look like?
- It’s a homography!

How do we compute homographies?
- We use SVD and the DLT!

How many colors do we need to match?
Linear color spaces

Change of color space:

\[ c' = H \cdot c \]

desired reference color space (i.e., XYZ)  
camera color space

What does this look like?  
- It’s a homography!

How do we compute homographies?  
- We use SVD and the DLT!

How many colors do we need to match?  
- We need at least four colors.
Using (again) a color chart

Color patches manufactured to have pre-calibrated XYZ coordinates.

Can we use any color chart image for color calibration?

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

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?
- It needs to be a *linear* image!
- Do radiometric calibration first.
An example

original
color-corrected
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.
Take-home messages

The values of pixels in a photograph and the values output by your camera’s sensor are two very different things.

The relationship between the two is complicated and unknown, and we often need to account for it when doing computer vision.
Basic reading:

• Szeliski textbook, Section 2.3


• Nine Degrees Below, https://ninedegreesbelow.com/ amazing resource for color photography, reproduction, and management.