

Demosaicing & HDR

From Sensor to Image & More

Mathias Seuret

Pattern Recognition Lab, Friedrich-Alexander-Universität Erlangen-Nürnberg



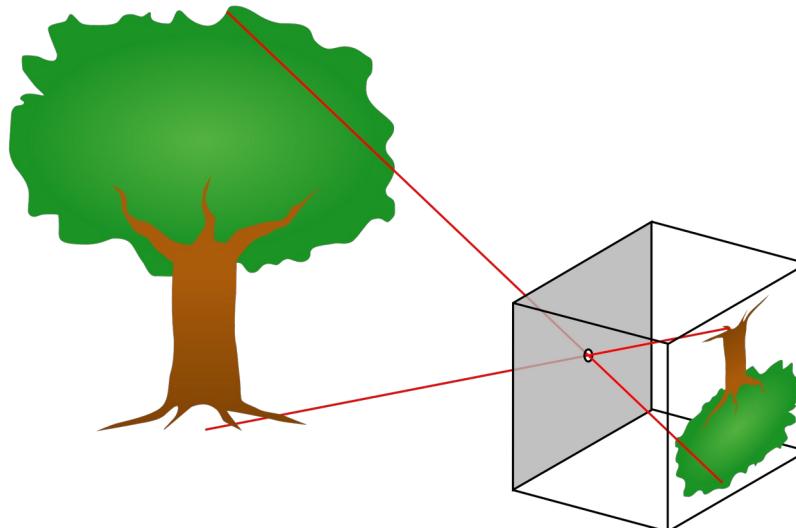
Part 1

Demosaicing

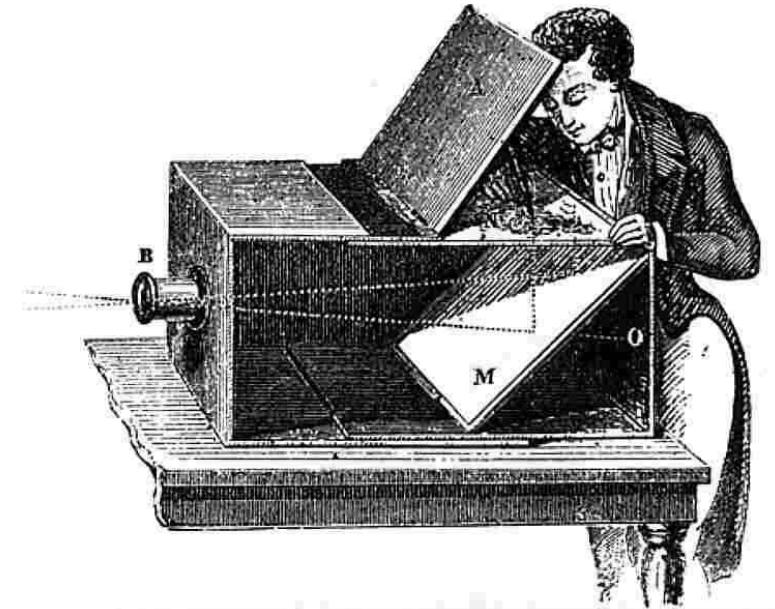
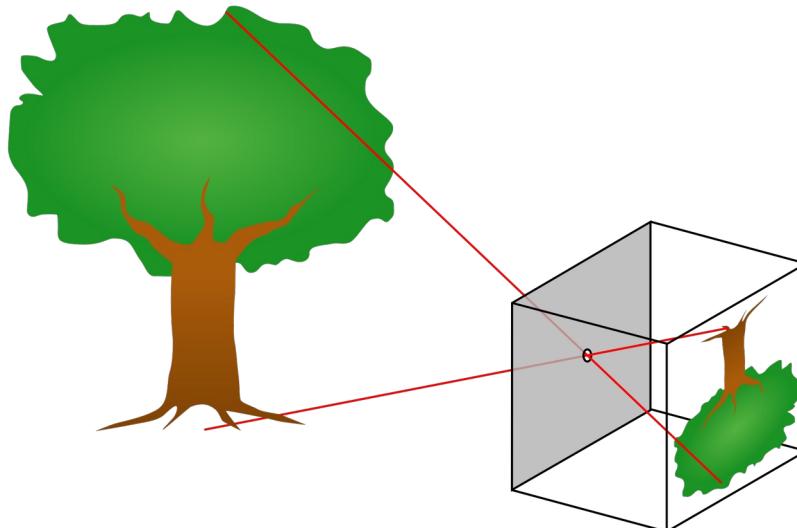
How can we do this?



Back to the Future Past



Back to the Future Past



https://en.wikipedia.org/wiki/Camera_obscura

Half a Camera

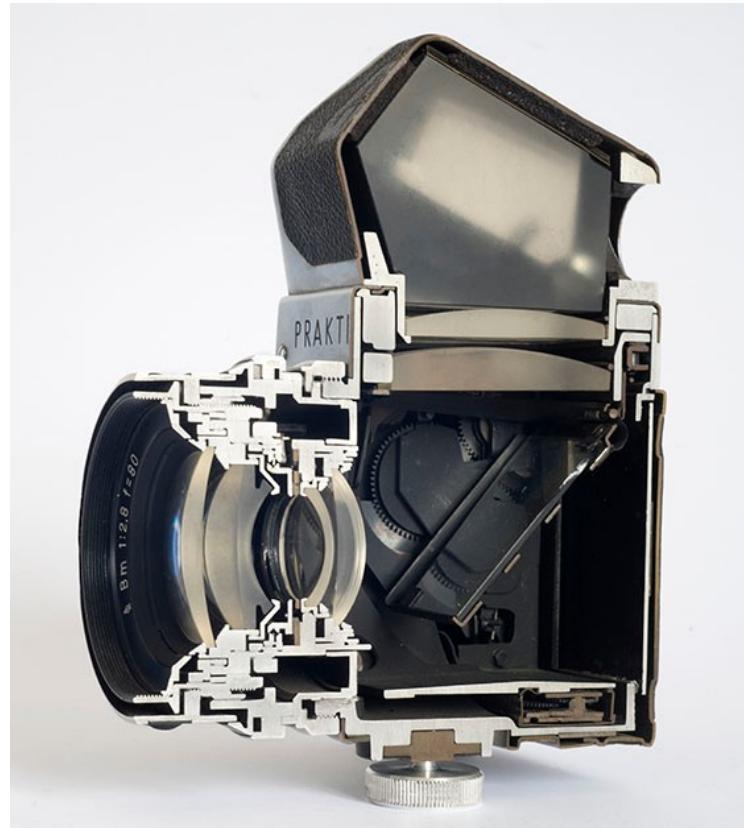


https://fr.wikipedia.org/wiki/Pellicule_photographique
<https://www.pentaconsix.com/slr.htm>

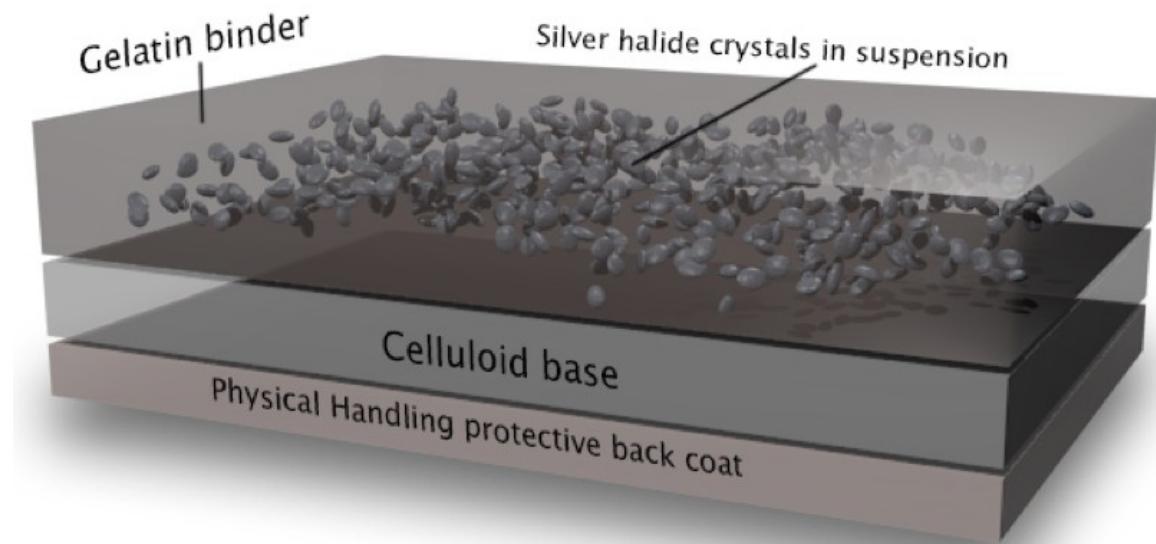
Half a Camera



https://fr.wikipedia.org/wiki/Pellicule_photographique
<https://www.pentaconsix.com/slr.htm>



Back to the Future Past

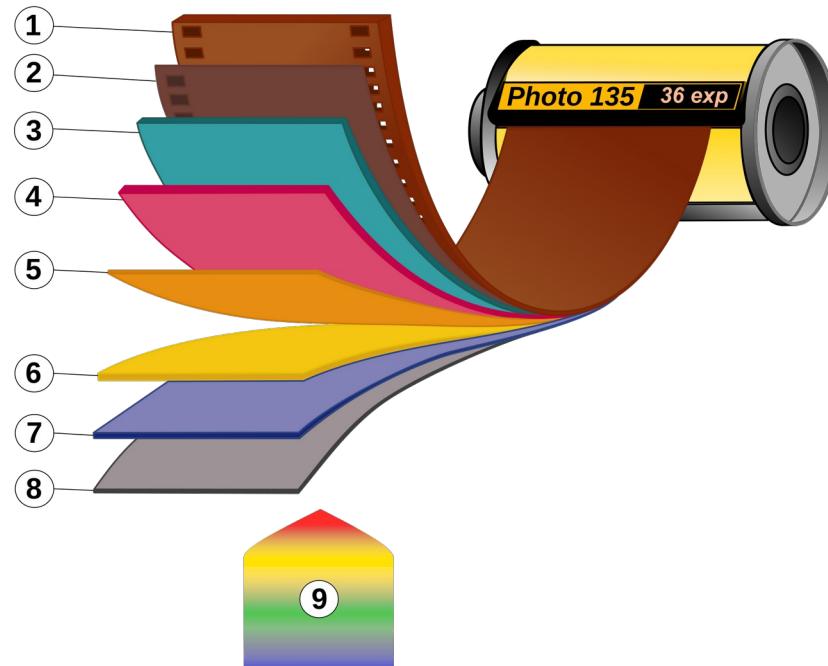


https://fr.wikipedia.org/wiki/Pellicule_photographique

<http://images.math.cnrs.fr/Un-modele-aleatoire-pour-le-grain-photographique.html>

https://en.wikipedia.org/wiki/Photographic_film

Back to the Future Past



https://fr.wikipedia.org/wiki/Pellicule_photographique

<http://images.math.cnrs.fr/Un-modele-aleatoire-pour-le-grain-photographique.html>

https://en.wikipedia.org/wiki/Photographic_film

Early 20th Century

Simple question:

A. Einstein got a nobel prize. Why?

Early 20th Century

Simple question:
A. Einstein got a nobel prize. Why?

*6. Über einen
die Erzeugung und Verwandlung des Lichtes
betrreffenden heuristischen Gesichtspunkt;
von A. Einstein.*

Zwischen den theoretischen Vorstellungen, welche sich die Physiker über die Gase und andere ponderable Körper gebildet haben, und der Maxwell'schen Theorie der elektromagnetischen Prozesse im sogenannten leeren Raum besteht ein tiefgreifender formaler Unterschied. Während wir uns nämlich den Zustand eines Körpers durch die *Lagen und Ge-*

Concerning an Heuristic Point of View Toward
the Emission and Transformation of Light

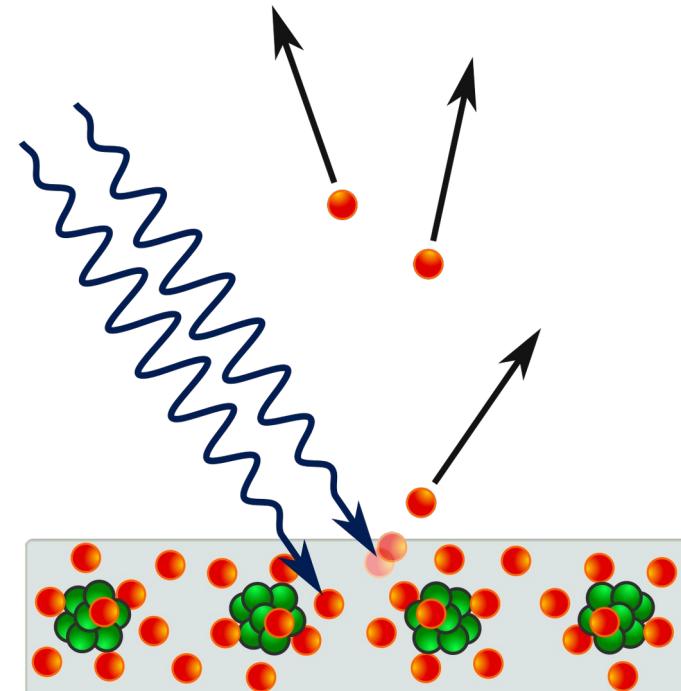
A. Einstein
Bern, 17 March 1905
(Received March 18, 1905)

Translation into English
American Journal of Physics, v. 33, n. 5, May 1965

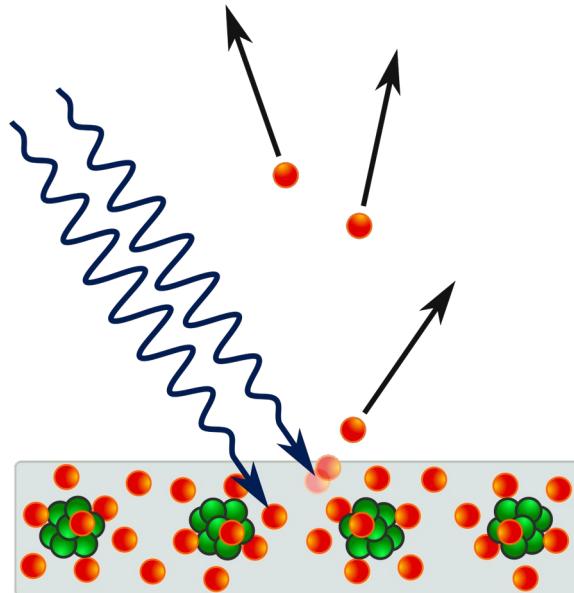
Early 20th Century

Simple question:

A. Einstein got a nobel prize. Why?

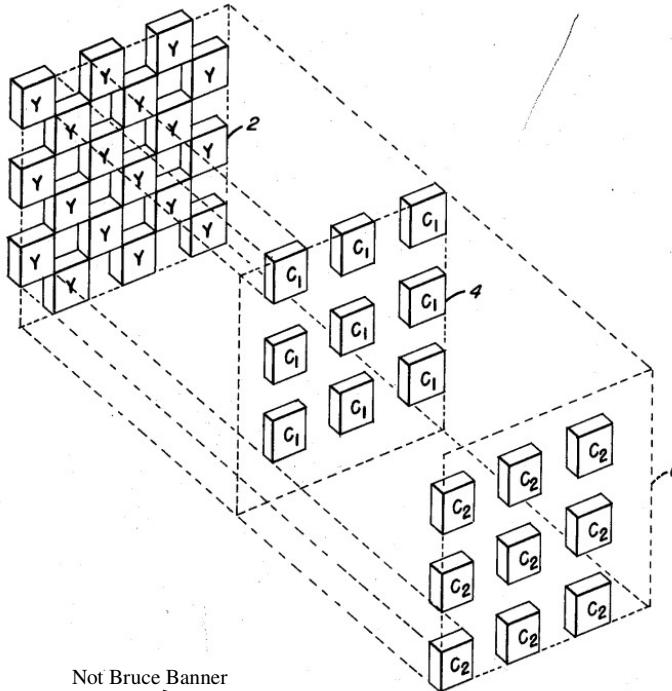


Light sensors are color blind



https://en.wikipedia.org/wiki/Photoelectric_effect
https://en.wikipedia.org/wiki/Solar_panel

The Bayer Filter

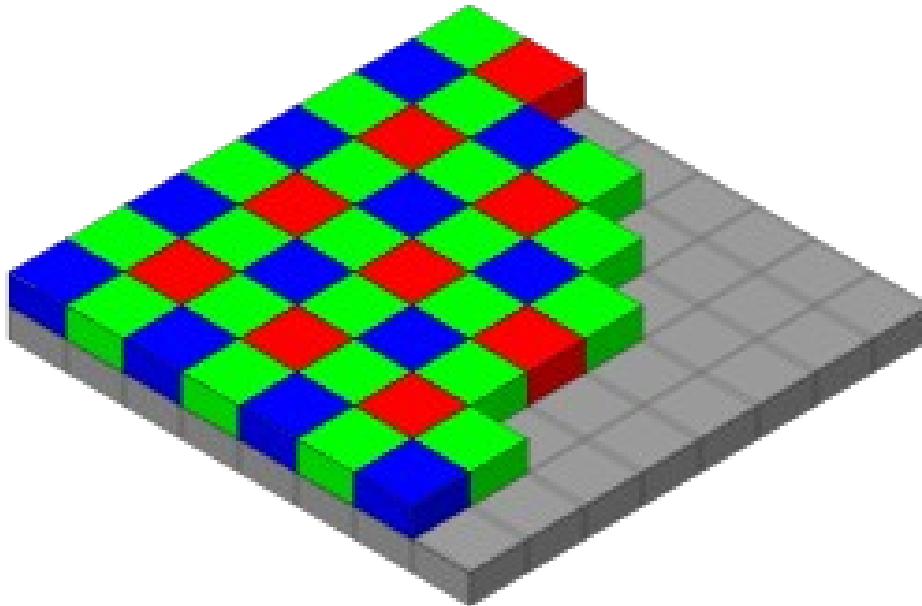


Color imaging array, US Patent, Bryce E. Bayer, 1976
https://en.wikipedia.org/wiki/Bayer_filter

Y	C ₁	Y	C ₁	Y	C ₁
C ₂	Y	C ₂	Y	C ₂	Y
Y	C ₁	Y	C ₁	Y	C ₁
C ₂	Y	C ₂	Y	C ₂	Y
Y	C ₁	Y	C ₁	Y	C ₁
C ₂	Y	C ₂	Y	C ₂	Y

FIG. 1B

The Bayer Filter



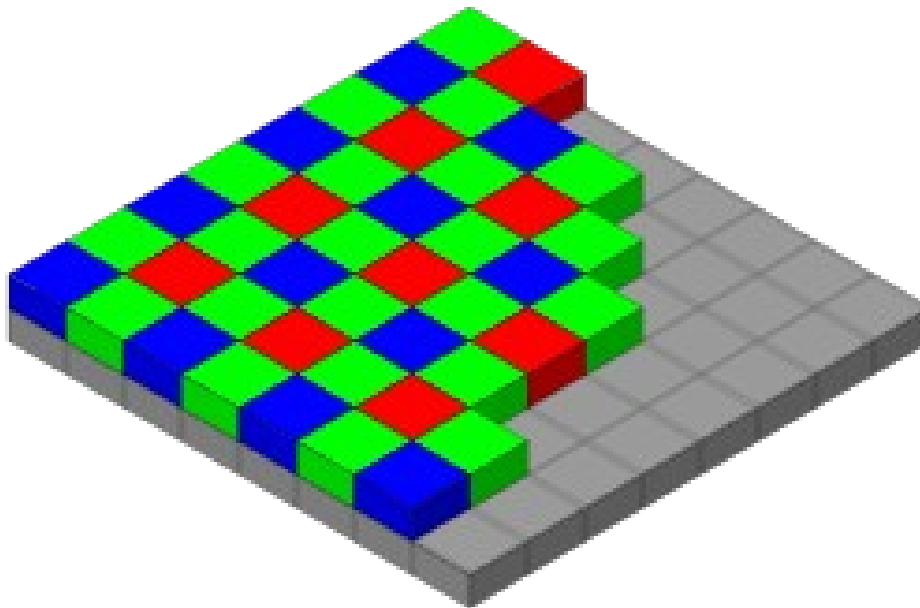
Not Bruce Banner

Color imaging array, US Patent, Bryce E. Bayer, 1976
https://en.wikipedia.org/wiki/Bayer_filter

Y	C ₁	Y	C ₁	Y	C ₁
C ₂	Y	C ₂	Y	C ₂	Y
Y	C ₁	Y	C ₁	Y	C ₁
C ₂	Y	C ₂	Y	C ₂	Y
Y	C ₁	Y	C ₁	Y	C ₁
C ₂	Y	C ₂	Y	C ₂	Y

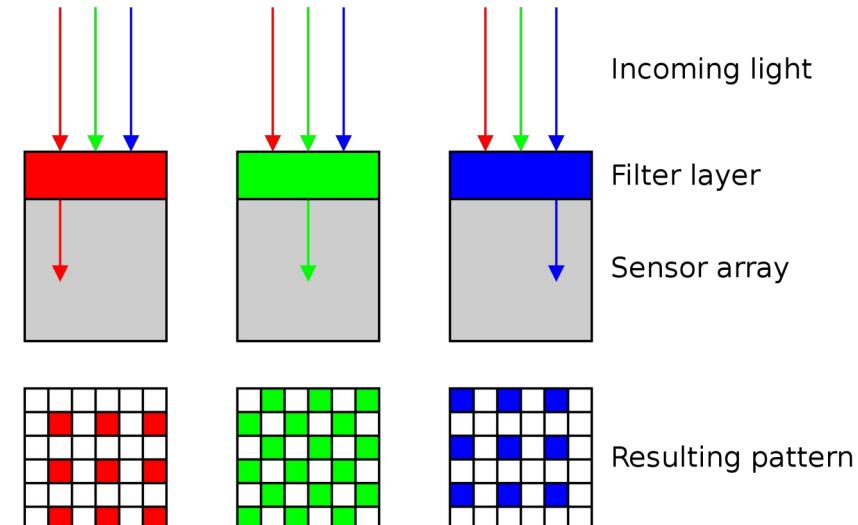
FIG. 1B

The Bayer Filter



Not Bruce Banner

Color imaging array, US Patent, Bryce E. Bayer, 1976
https://en.wikipedia.org/wiki/Bayer_filter



The Bayer Filter & Sensor Values

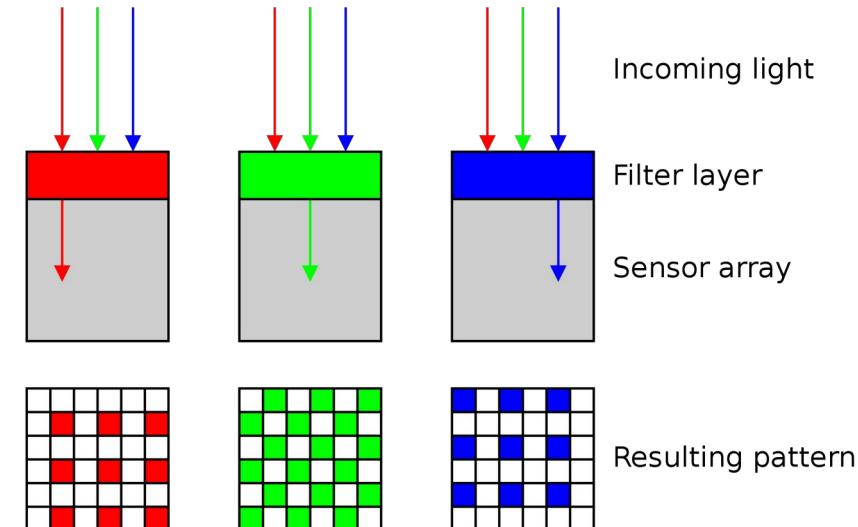
Some comments:

- Sensor values are not in [0,255]
- Example:
 - Canon 90D: 14 bits
 - Canon R3: 12 bits
 - More bits \neq higher quality
- Sensor values \propto collected light
 - Double exposure, double values
 - Double light, double values
 - ...
- It is **not** the case for JPG images
- The green channel is usually brighter
- No white balance in sensor data!

The Bayer Filter & Sensor Values

Some comments:

- Sensor values are not in [0,255]
- Example:
 - Canon 90D: 14 bits
 - Canon R3: 12 bits
 - More bits \neq higher quality
- Sensor values \propto collected light
 - Double exposure, double values
 - Double light, double values
 - ...
- It is **not** the case for JPG images
- The green channel is usually brighter
- No white balance in sensor data!



Exercise 1

Investigate the Bayer patterns.

On StudOn, you will find two numpy arrays. One of them has the Bayer pattern seen on the right. The other one has a slightly different Bayer pattern.

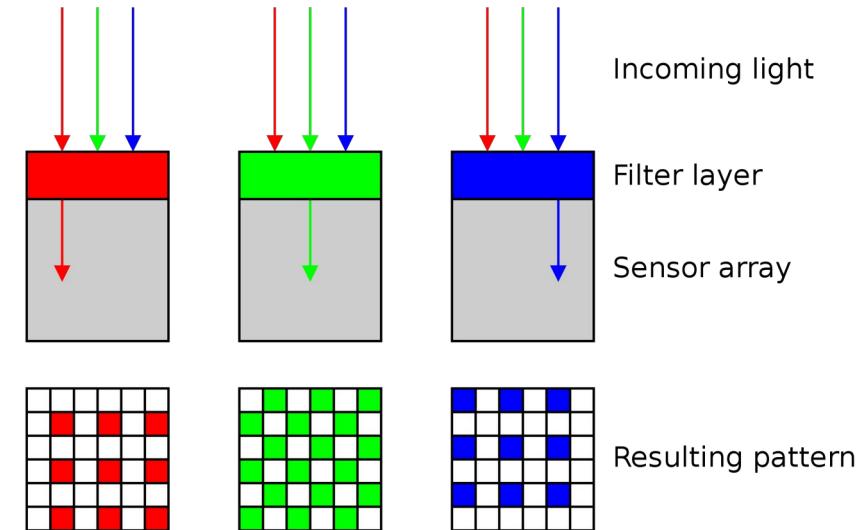
Find out which pattern it is.

Exercise 1

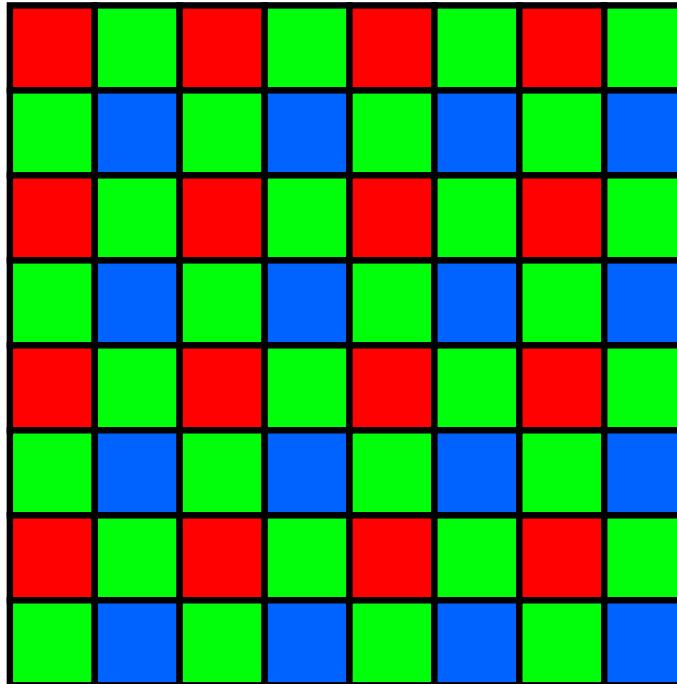
Investigate the Bayer patterns.

On StudOn, you will find two numpy arrays. One of them has the Bayer pattern seen on the right. The other one has a slightly different Bayer pattern.

Find out which pattern it is.



Demosaicing: a simple approach



Demosaicing: a simple approach

1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0

1	1	1
1	1	1
1	1	1

Let X be your data.

Create a mask M_c for each color channel. Example for red channel.

Create a convolution kernel K (at least 3x3)

Compute color channel C as

$$C = ((M_c \times X) \otimes K) / (M_c \otimes K)$$

Demosaicing: a simple approach

1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0

1	1	1
1	1	1
1	1	1

Let X be your data.

Create a mask M_c for each color channel. Example for red channel.

Create a convolution kernel K (at least 3x3)

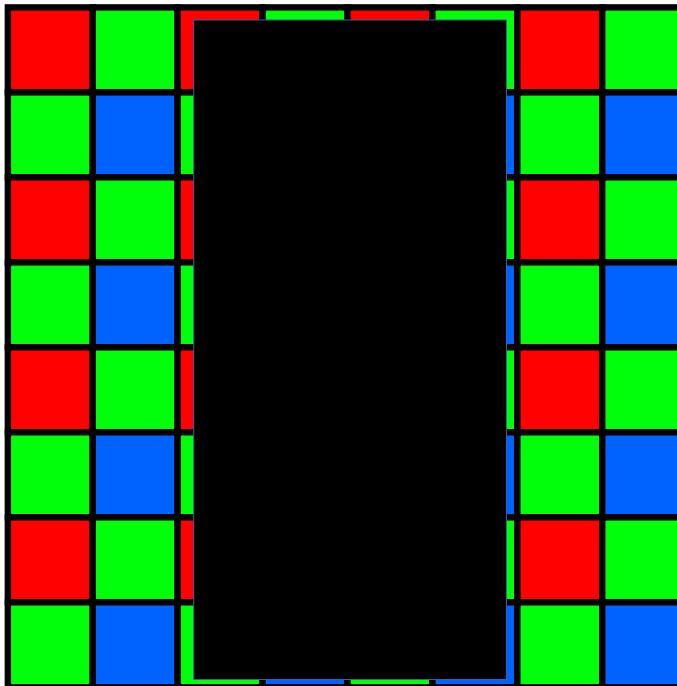
Compute color channel C as

$$C = \frac{(M_c \times X) \otimes K}{(M_c \otimes K)}$$

Filters out other channels
→ weighted sum

Division by the weights
→ weighted average

Demosaicing: a simple approach



Risk of color artifact along sharp edges!

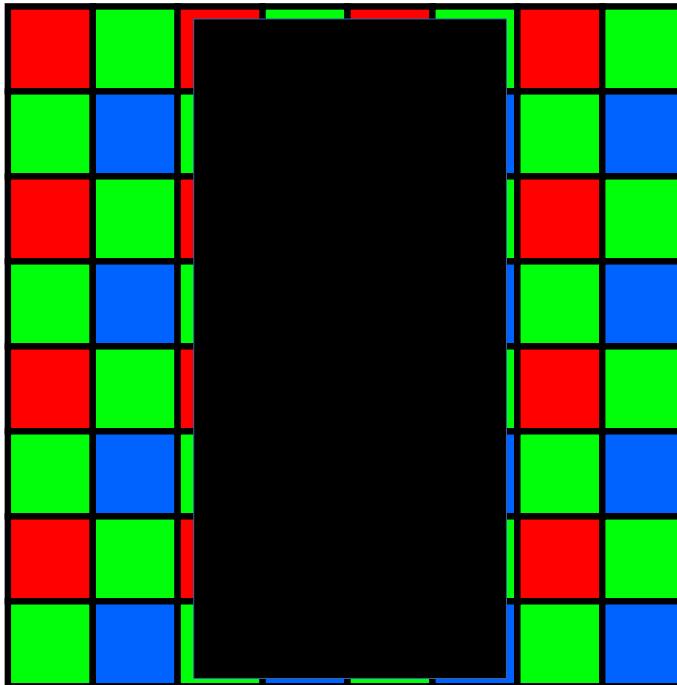
Here:

- 3rd column would be blueish
- 6th column would be reddish

This might be a problem in case of:

- Low resolution sensor
- And/or very sharp image

Demosaicing: a simple approach



<https://de.wikipedia.org/wiki/YUV-Farbmodell>

Risk of color artifact along sharp edges!

Here:

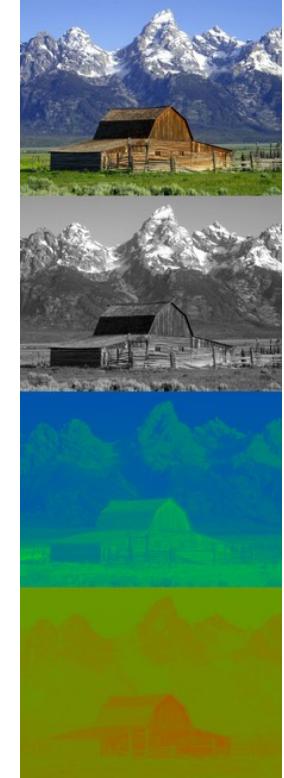
- 3rd column would be blueish
- 6th column would be reddish

This might be a problem in case of:

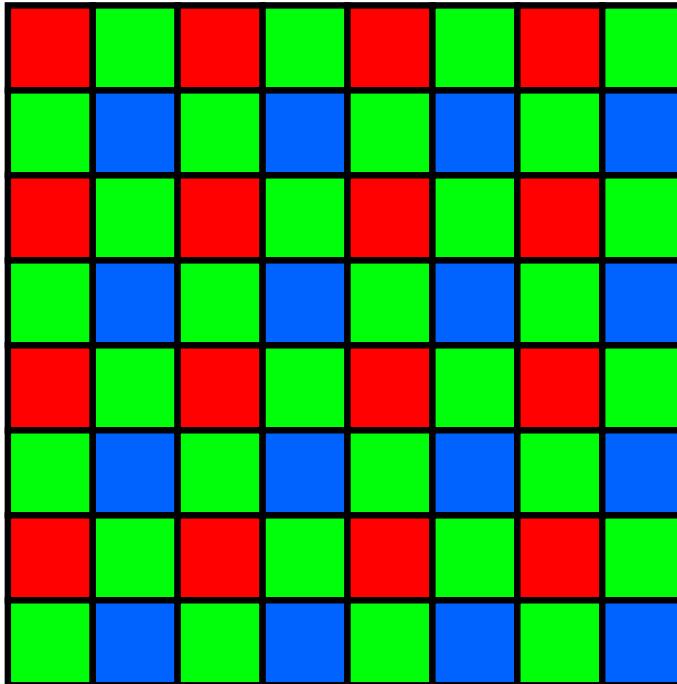
- Low resolution sensor
- And/or very sharp image

Possible fix:

- Convert to YUV
- Apply median filters on U and V
- Revert to RGB



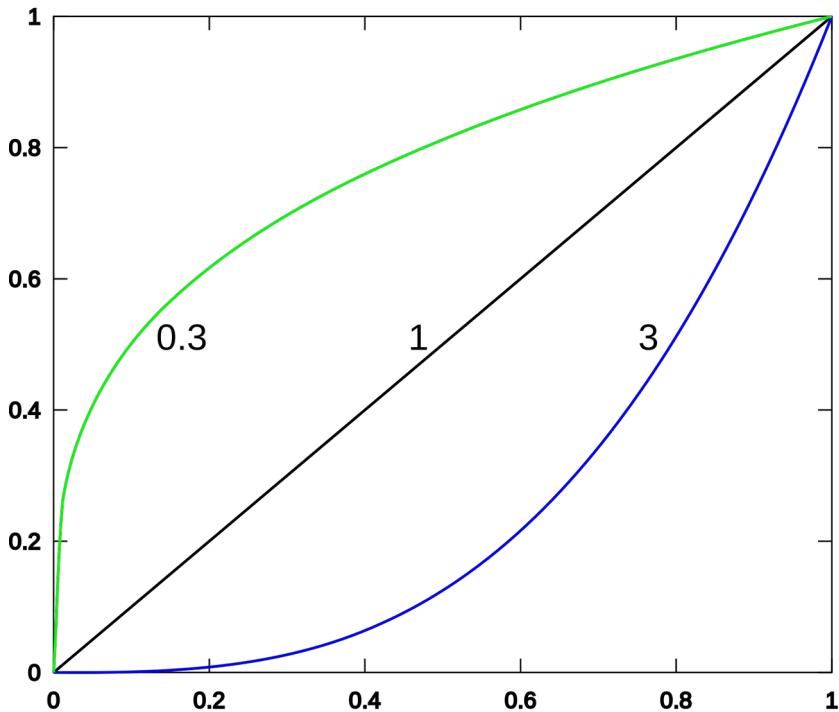
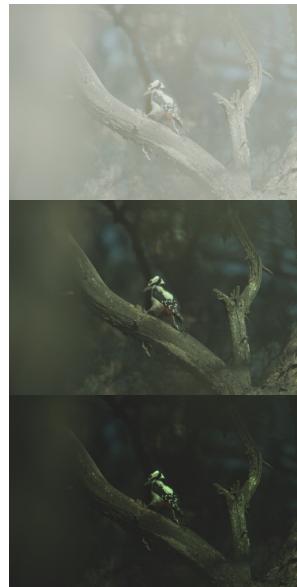
Exercise 2



Implement this demosaicing approach.

It's fine if you end-up with dark and greenish images.
Correcting this comes in later exercises!

Gamma Correction in a hazelnut shell



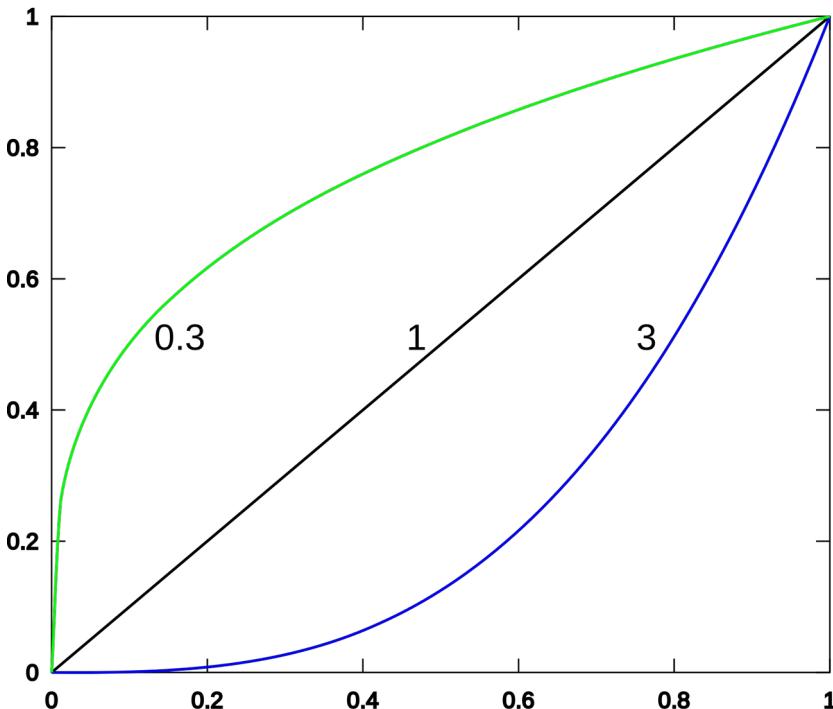
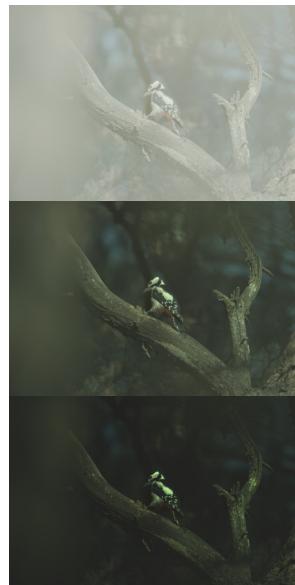
Gamma Correction in a hazelnut shell

A simple image luminosity correction is the gamma correction, defined as:

$$y = x^\gamma$$

There is no magic formula to get the best gamma. Try different ones.

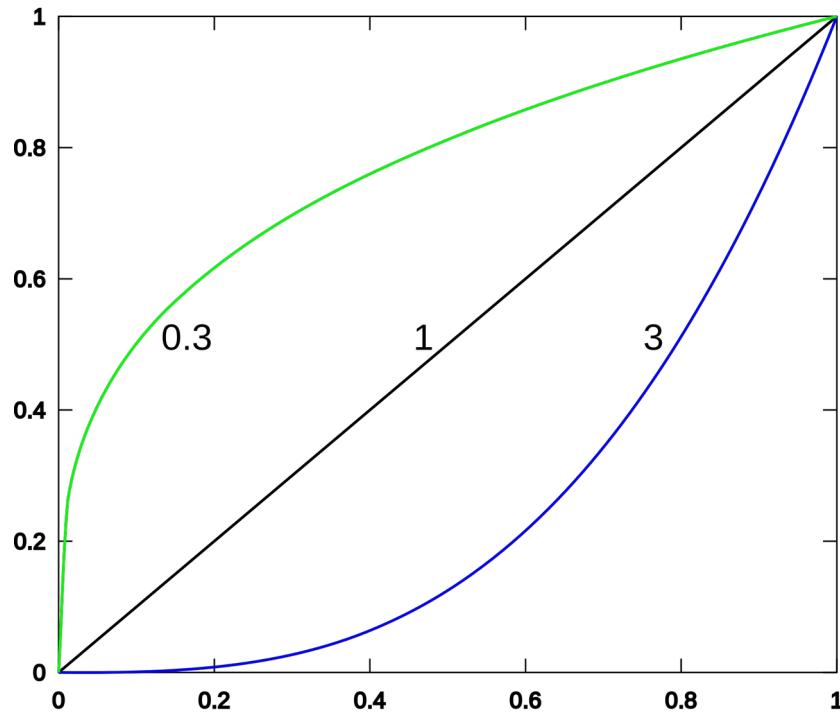
0.3 is usually a good starting point.



Exercise 3

Implement the gamma correction, and apply it to IMG_4782. Try out a few different values, including 0.3, and keep the one you prefer.

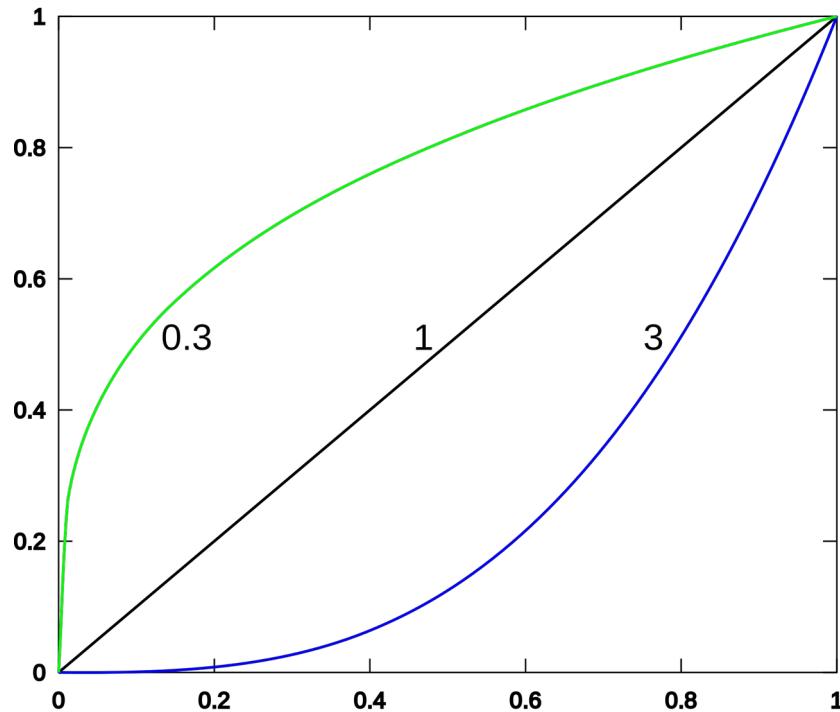
Try at least one more type of curve (be creative).



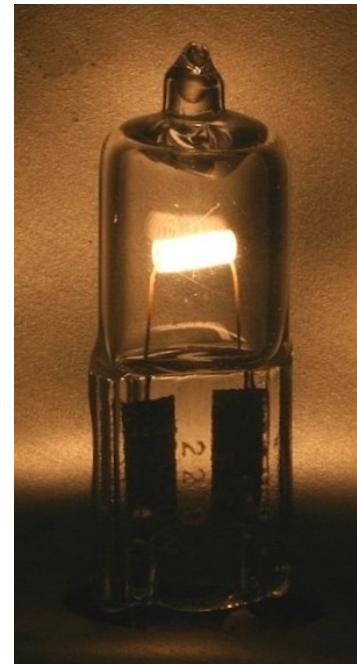
Exercise 3

Implement the gamma correction, and apply it to IMG_4782. Try out a few different values, including 0.3, and keep the one you prefer.

Try at least one more type of curve (be creative).



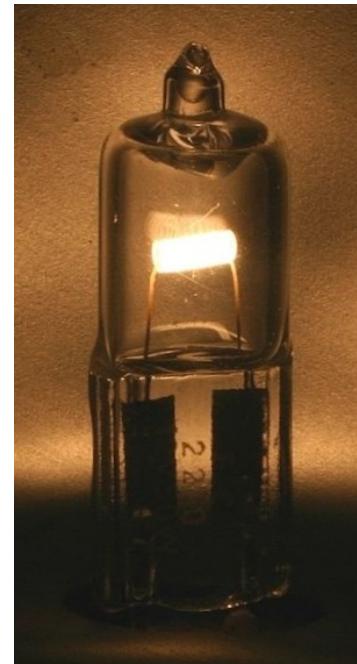
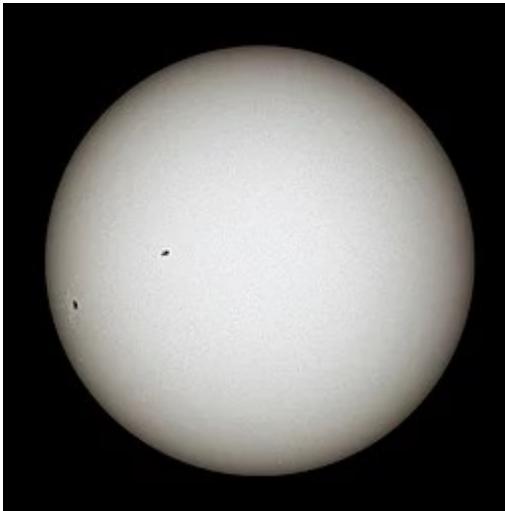
White Balance



What is the color of a white piece of paper?
Does it depend on the light?

<https://en.wikipedia.org/wiki/Sun>
https://en.wikipedia.org/wiki/Halogen_lamp
https://en.wikipedia.org/wiki/Light-emitting_diode

White Balance



What is the color of a white piece of paper?
Does it depend on the light?

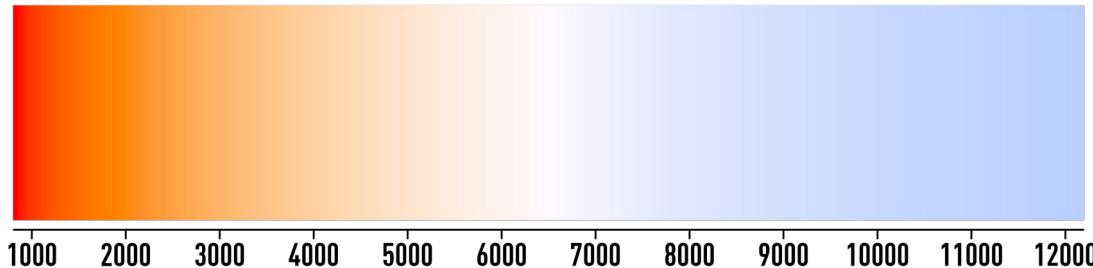
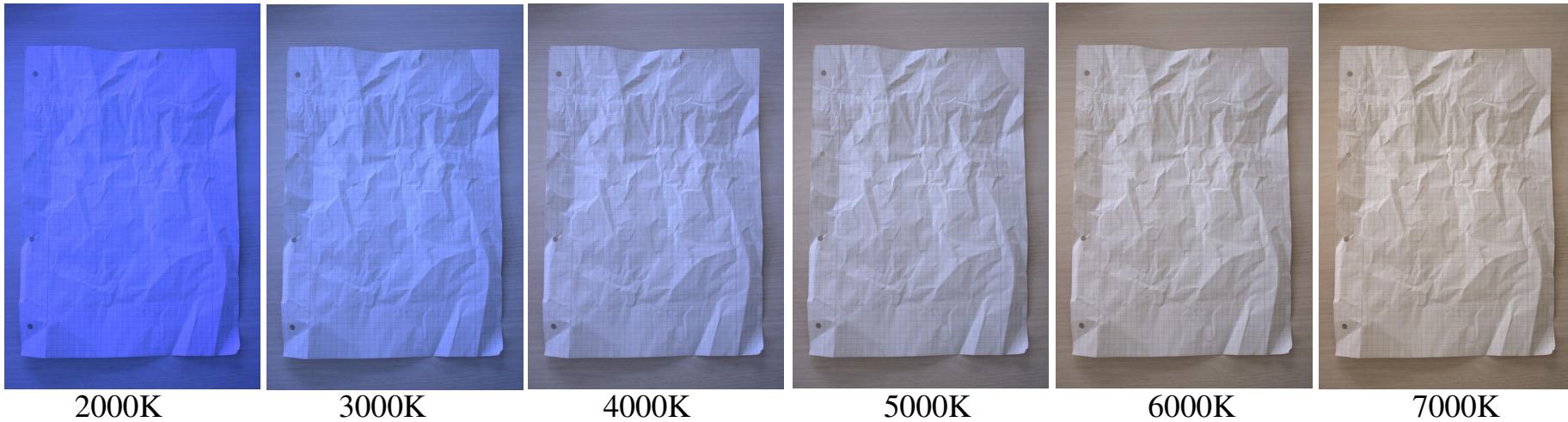
<https://en.wikipedia.org/wiki/Sun>
https://en.wikipedia.org/wiki/Halogen_lamp
https://en.wikipedia.org/wiki/Light-emitting_diode

White Balance



https://en.wikipedia.org/wiki/Elton_John
<https://fr.wikipedia.org/wiki/Ski>

White Balance



https://en.wikipedia.org/wiki/Color_temperature

Color temperature: color of light emitted by a black body at a given temperature in Kelvin.

Daylight: typically 5600K

Balance can be obtained by matching red, green and blue values of a white/gray point.

White Balance – Gray World

Simple algorithm assuming that the average color of the image is gray.

mean value of the image: m_i

mean value of channel c: m_c

multiply values of channel c by m_i / m_c

Take care of out-of-bounds values!

White Balance – Gray World

Simple algorithm assuming that the average color of the image is gray.

mean value of the image: m_i

mean value of channel c: m_c

multiply values of channel c by m_i / m_c

Take care of out-of-bounds values!

The gray world method does not always perform well. Can you guess when? Can you find a solution?

White Balance – Gray World



Multiply channels such that they share the same mean.

Exercise 4

Include the gray world method to your pipeline.

Part 2

High Dynamic Range

Dynamic Range



Two steps for producing HDR images:

- Measure/estimate *real* pixels luminosity in high range
- Compress this range such that it can be displayed

Let's do this!

Dynamic Range



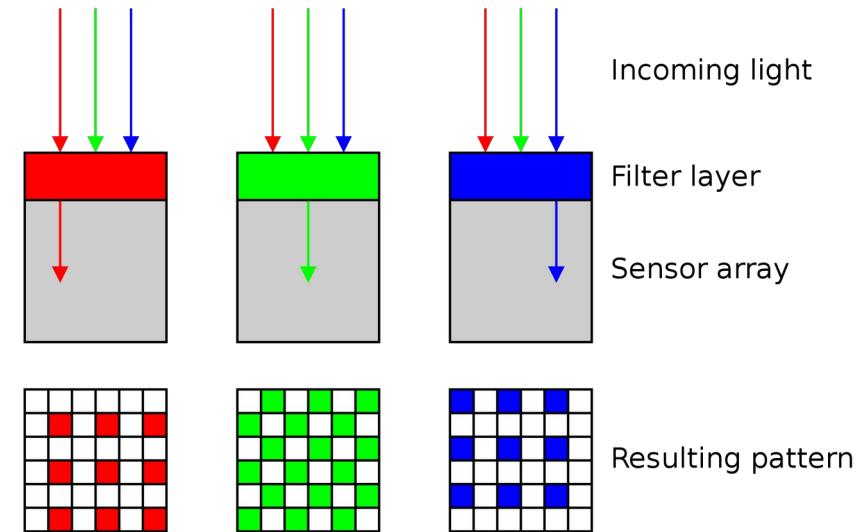
Two steps for producing HDR images:

- Measure/estimate *real* pixels luminosity in high range
- Compress this range such that it can be displayed

Let's do this!

Exercise 5

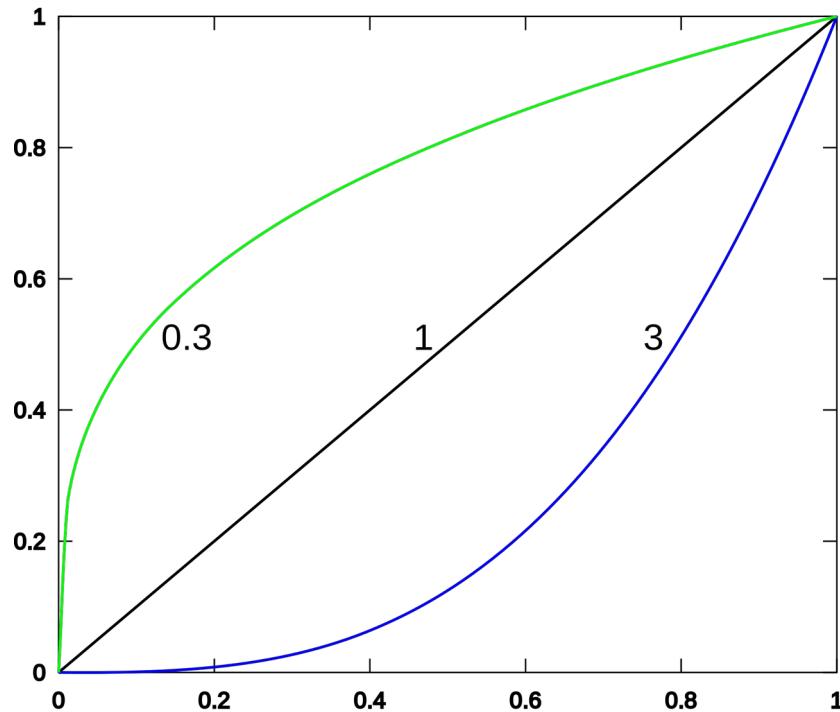
Show that sensor data is linear – i.e., unless values reach the limits of the sensor, then multiplying the amount of light collected by x multiplies the sensor values by x as well.



Dynamic Range

Cameras apply heavy data processing to produce JPG images.

Assuming that a pixel has a value of y in a JPG photo with an exposure of 1 second, we can bet it will not have a value of $2y$ in a photo taken with an exposure of 2 seconds.

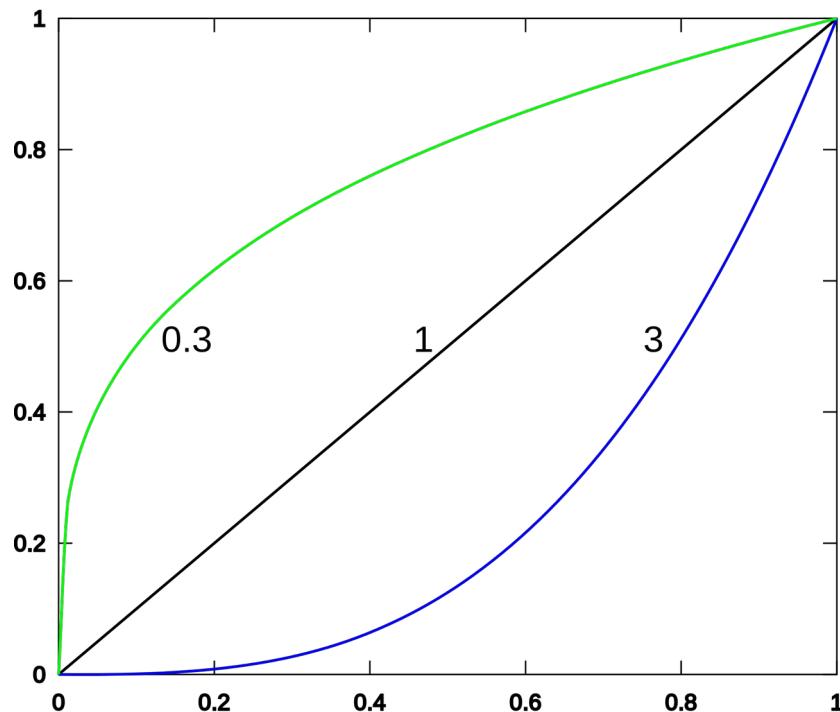


Dynamic Range

Cameras apply heavy data processing to produce JPG images.

Assuming that a pixel has a value of y in a JPG photo with an exposure of 1 second, we can bet it will not have a value of $2y$ in a photo taken with an exposure of 2 seconds.

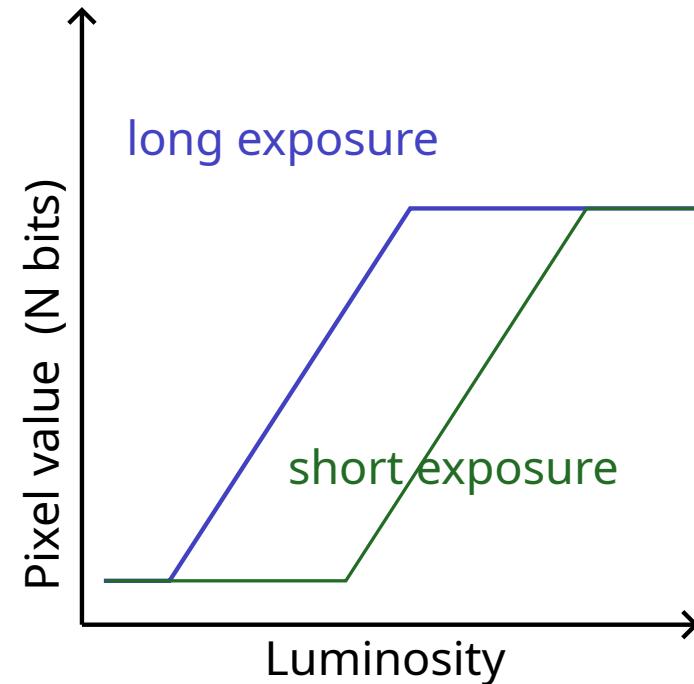
This problem vanishes when using raw data!



From Low to High Dynamic Range

By changing exposure time, we can ‘reveal’ different luminosity ranges.

Anything darker is black, anything brighter is white.

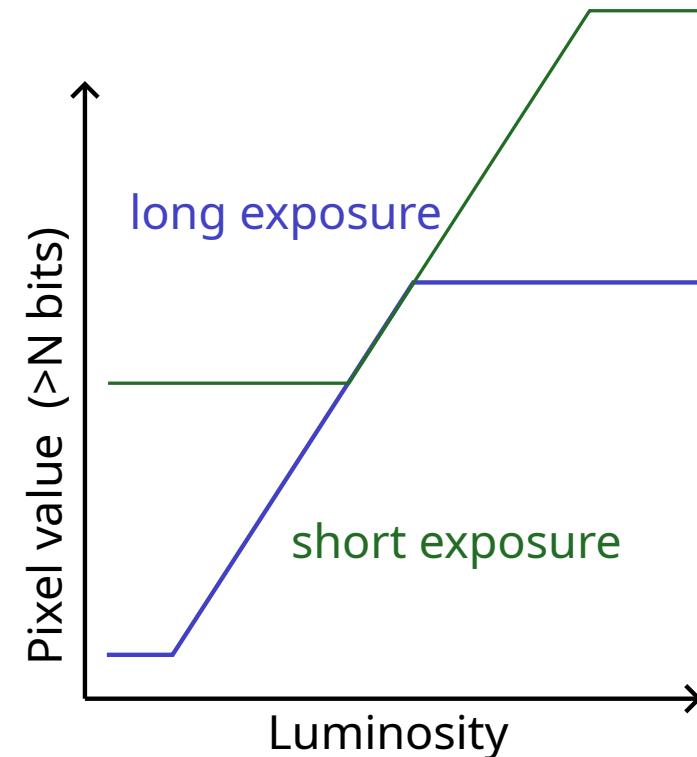


From Low to High Dynamic Range

By changing exposure time, we can ‘reveal’ different luminosity ranges.

Anything darker is black, anything brighter is white.

If these low dynamic ranges are overlapping, then they can be combined to form high dynamic ranges!



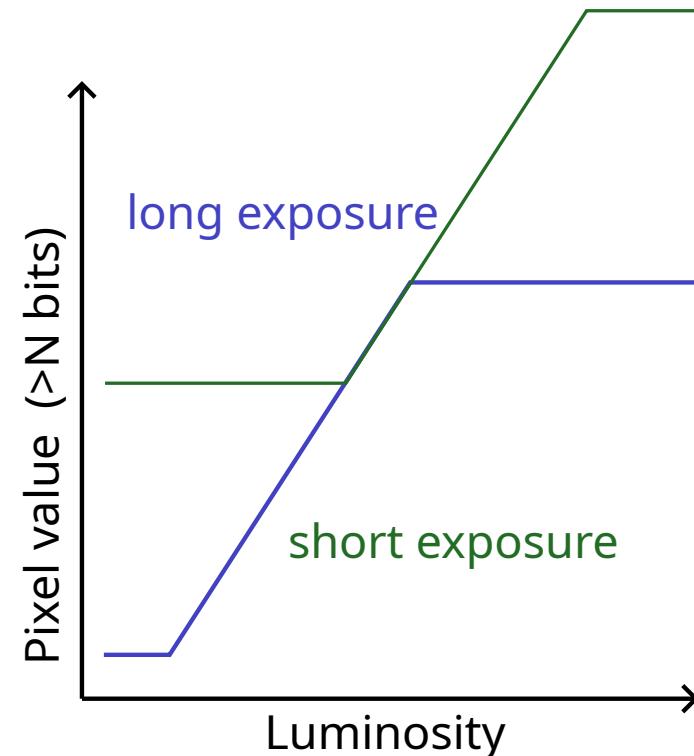
From Low to High Dynamic Range

By changing exposure time, we can ‘reveal’ different luminosity ranges.

Anything darker is black, anything brighter is white.

If these low dynamic ranges are overlapping, then they can be combined to form high dynamic ranges!

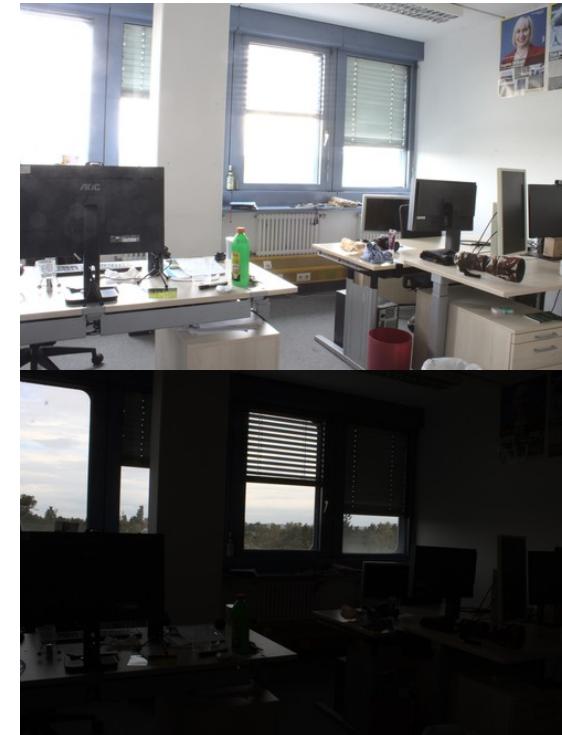
Naïve approach: downscale this range to [0,255]



From Low to High Dynamic Range

Simple combination approach:

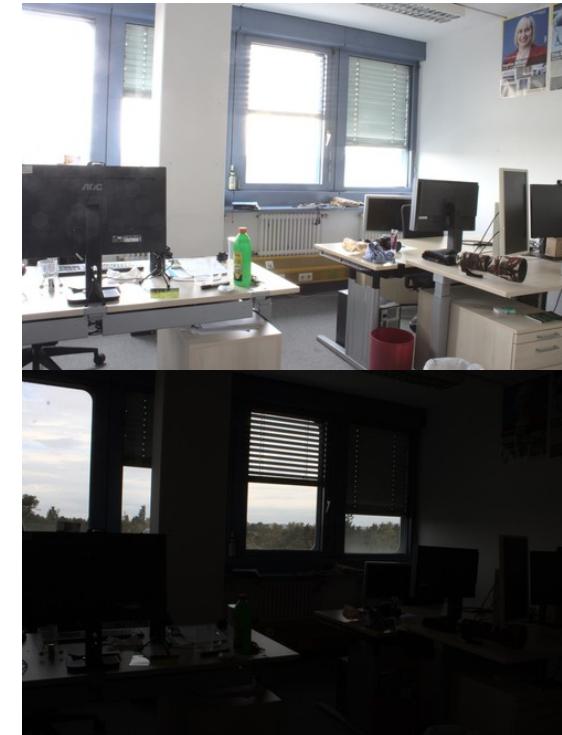
- Load brightest raw data (longest exposure)
- Let's call it h



From Low to High Dynamic Range

Simple combination approach:

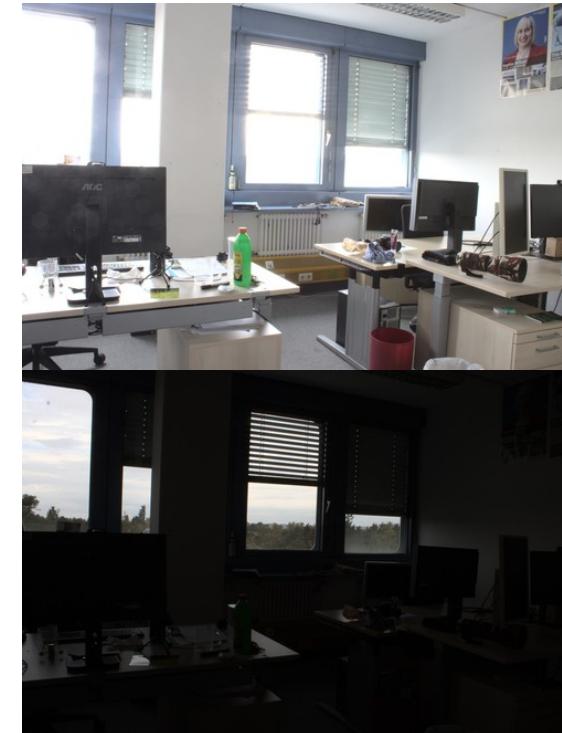
- Load brightest raw data (longest exposure)
- Let's call it h
- For each next raw file:
 - Load it and name it i
 - Multiply i by the exposure difference to the first photo
 - E.g., first photo 1s, second photo 0.5s, multiply by 2



From Low to High Dynamic Range

Simple combination approach:

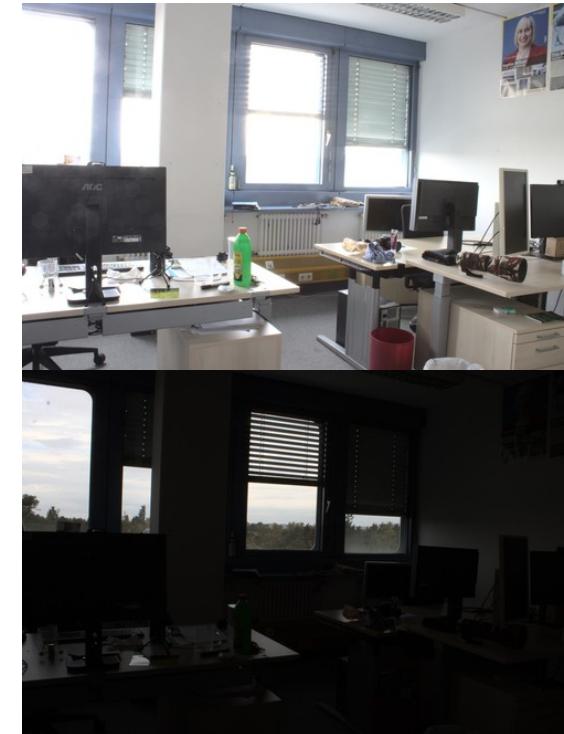
- Load brightest raw data (longest exposure)
- Let's call it h
- For each next raw file:
 - Load it and name it i
 - Multiply i by the exposure difference to the first photo
 - E.g., first photo 1s, second photo 0.5s, multiply by 2
 - Values in h which are above a threshold t get replaced by the corresponding values in i



From Low to High Dynamic Range

Simple combination approach:

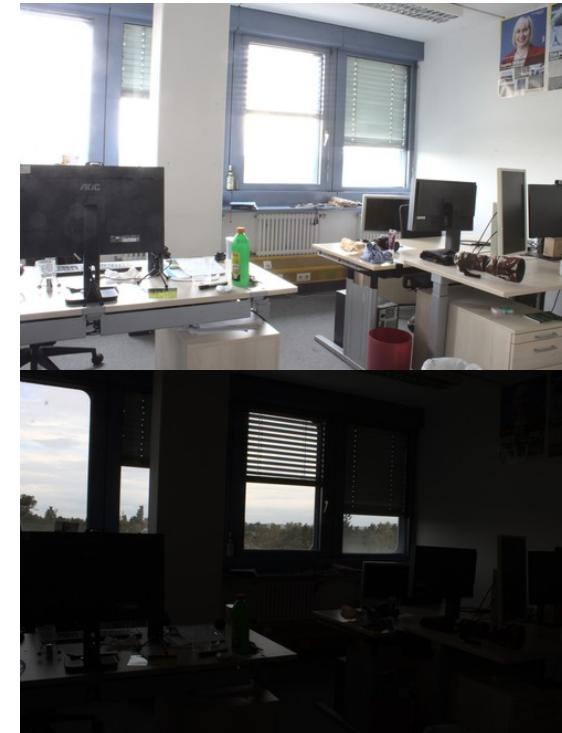
- Load brightest raw data (longest exposure)
- Let's call it h
- For each next raw file:
 - Load it and name it i
 - Multiply i by the exposure difference to the first photo
 - E.g., first photo 1s, second photo 0.5s, multiply by 2
 - Values in h which are above a threshold t get replaced by the corresponding values in i
 - t is not that important
 - Avoid the ‘plateau’ in i
 - A value of $0.8 \cdot \max(h)$ should be fine



From Low to High Dynamic Range

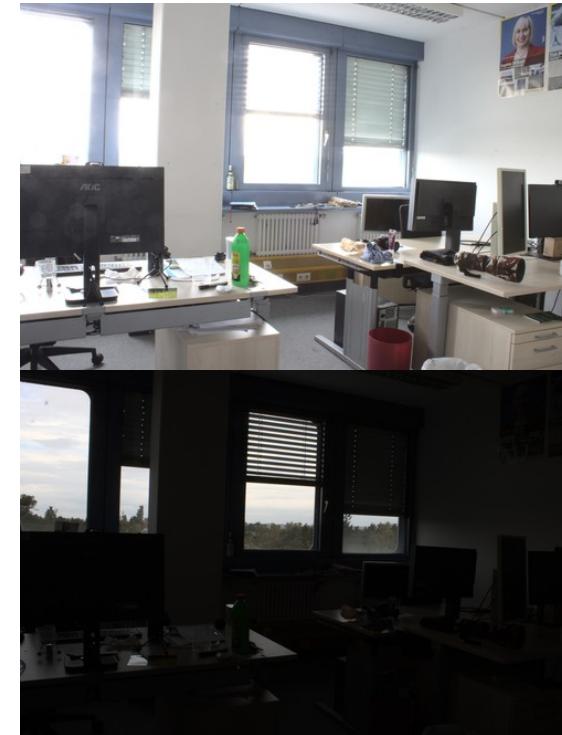
Simple combination approach:

- Load brightest raw data (longest exposure)
- Let's call it h
- For each next raw file:
 - Load it and name it i
 - Multiply i by the exposure difference to the first photo
 - E.g., first photo 1s, second photo 0.5s, multiply by 2
 - Values in h which are above a threshold t get replaced by the corresponding values in i
 - t is not that important
 - Avoid the ‘plateau’ in i
 - A value of $0.8 \cdot \max(h)$ should be fine



Back From High Dynamic Range

One does not simply rescale values to [0,255]



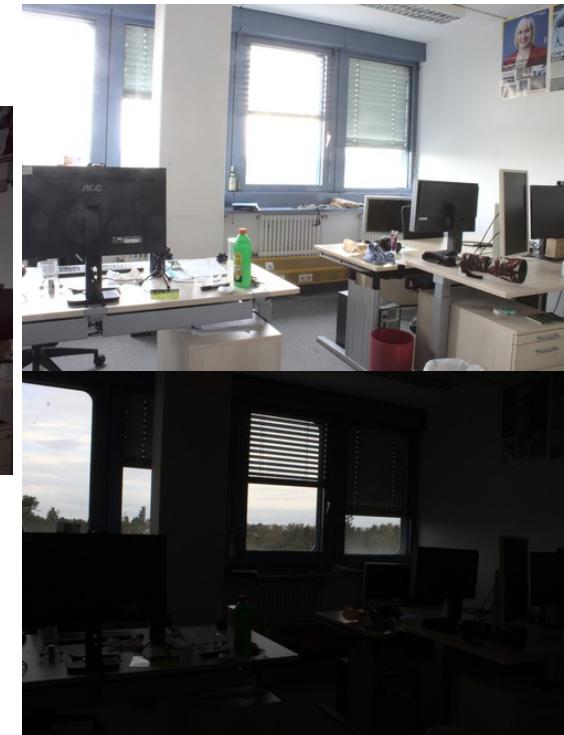
Back From High Dynamic Range

One does not simply rescale values to [0,255]



Human light perception is logarithmic – let's do this!

(approximately)



Exercise 6

Implement the combination of several images to produce HDR data, as seen during the lecture.

Apply a log scale to these values, normalize the result in $[0, 255]$, and display it.

Tone Mapping

Applying a log scale to the image decreases local contrasts.

There are many better (and slower) methods.

- Increase local contrast
 - Unsharp filter
 - Small Gaussian kernel: sharpens the image
 - Large kernel: increases local contrast



Tone Mapping

Applying a log scale to the image decreases local contrasts.

There are many better (and slower) methods.

- Increase local contrast
 - Unsharp filter
 - Small Gaussian kernel: sharpens the image
 - Large kernel: increases local contrast
 - **Problems at edges**

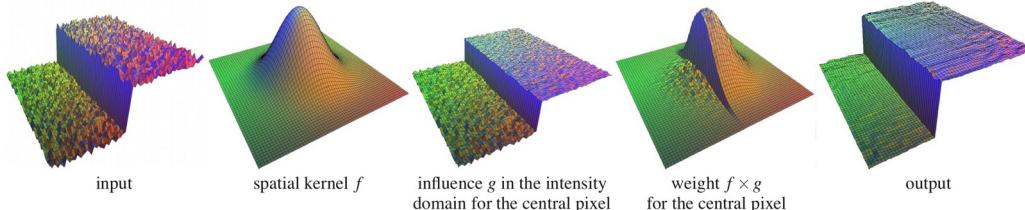


Tone Mapping

Applying a log scale to the image decreases local contrasts.

There are many better (and slower) methods.

- Increase local contrast
 - Unsharp filter
 - Small Gaussian kernel: sharpens the image
 - Large kernel: increases local contrast
 - **Problems at edges**
 - Bilateral filter?



Fast Bilateral Filtering for the Display of High-Dynamic-Range Images, Durand and Dorsey.

iCAM06

iCAM06 aims at preserving details, without halo effect, and compresses more the ‘base’ of the image:

Pseudocode:

```
output_range = 4
input_intensity = 1/61 · (20·red + 40·green + blue)
r, g, b = rgb / input_intensity
log_base = bilat_filt(log(input_intensity))
log_details = log(input_intensity) - log_base
compression = log(output_range) / (max(log_base)-min(log_base))
log_offset = -max(log_base) · compression
output_intensity = exp(log_base · compression + log_offset + log_detail)
rgb = r·output_intensity, g·output_intensity, b·output_intensity
```

iCAM06

iCAM06 aims at preserving details, without halo effect, and compresses more the ‘base’ of the image:

Pseudocode:

```

output_range = 4
input_intensity = 1/61 · (20·red + 40·green + blue)
r, g, b = rgb / input_intensity
log_base = bilat_filt(log(input_intensity))
log_details = log(input_intensity) - log_base
compression = log(output_range) / (max(log_base)-min(log_base))
log_offset = -max(log_base) · compression
output_intensity = exp(log_base · compression + log_offset + log_detail)
rgb = r·output_intensity, g·output_intensity, b·output_intensity

```



iCAM06: A refined image appearance model for HDR image rendering, Kuang et al., 2007

Exercise 7

Implement the iCAM06 method, and apply it to the provided data.

Exercise 8 – final one

Create a demosaicing function named process_raw which:

- Takes as input the path of a CR3 raw file,
- Outputs an RGB image to a given path
- Runs in an acceptable amount of time

You are entirely free about how it works. You can use what you implemented in the other exercises, but you can also do anything else that you want (denoising, local contrast enhancing, histogram equalization or matching, saturation correction, ...)

I will run them on a raw photo, and upload (anonymous) results on StudOn.

You will then vote for the one that looks best :-)

Question Time

