What Are The Important Things? Full Well and Saturation Capacity

- Each pixel has a maximum capacity to collect electrons.
- But we will limit that to a lower maximum, because non-linear effects occur.

- A pixel has a saturation capactity of 50000 electrons.
- A glass of wheat beer contains approx. 50000 drops of beer.
- A glass of wine or champagne might have 10000 drops.



What Are The Important Things? Signal to Noise Ratio

- When refilling the pixel there is a slight jitter, although you try to get the same amount of electrons in. The „noise" is equivalent to the square root of the number of electrons.
- Noise = sqrt (number of electrons)

Max. Signal to Noise Ratio (SNR)
=(max number of elec.)/(sqrt(max number of elec.))
=Saturation Capacity/(sqrt(Saturation Capacity))
=sqrt(Saturation Capacity)

- Sqrt(50000) = approx. 224
- Refilled beer glasses do not have the same content.



AUTOMATE • 2011

What Are The Important Things? Dark Noise

- There is some remaining noise, although no light hits the sensor.
- Depending on the sensor, this is between 8 to 110 electrons.
- Compare this to the remaining drops of beer in an "empty" glass.



What Are The Important Things? Dynamic Range

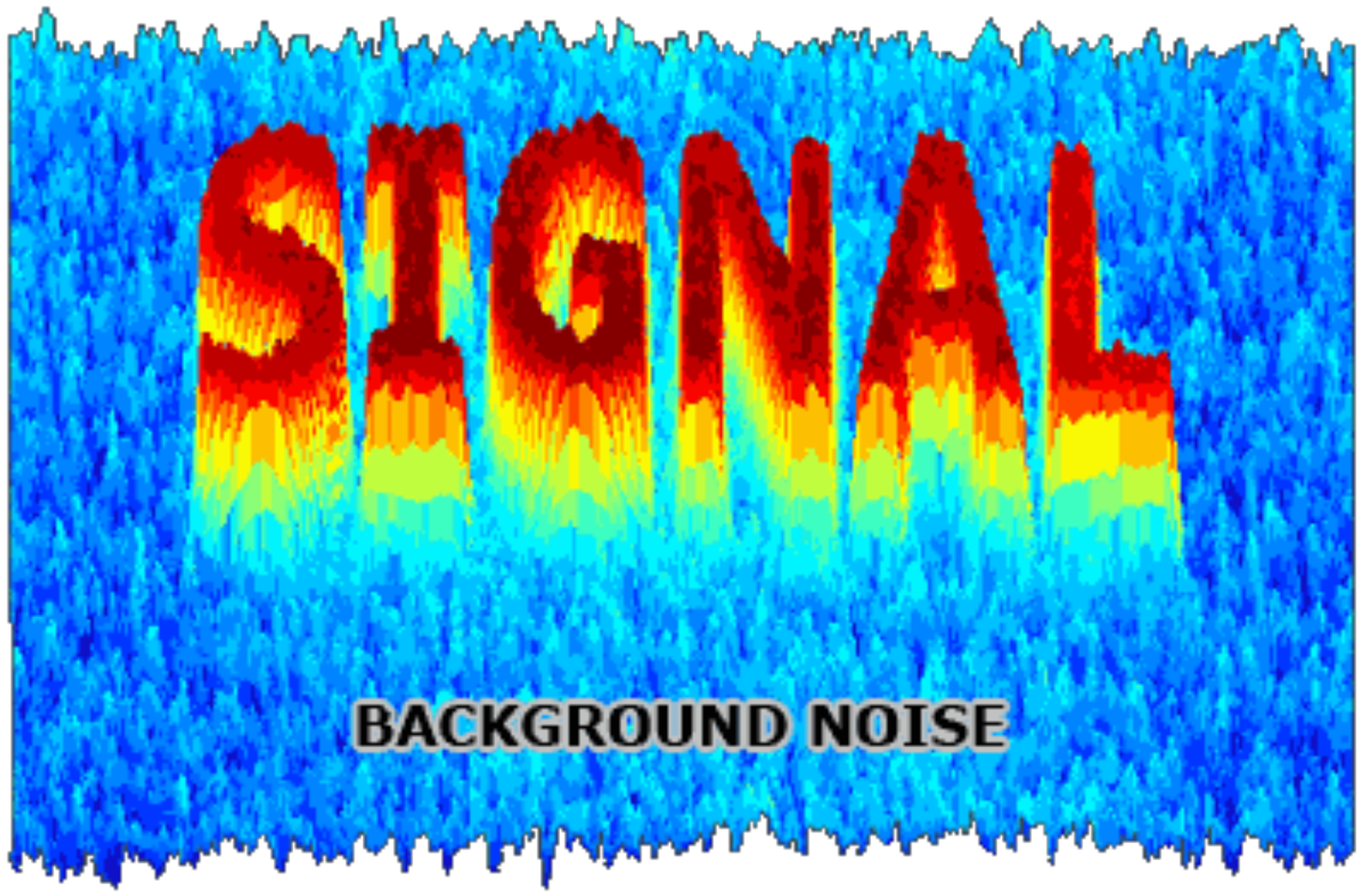
- Dynamic range is the ratio between a full and an empty glass of beer.

Dynamic range
= Full / Empty
= Saturation Cap. / Dark Noise
= 50000 / 25
= 200

- A glass of beer will always have dynamics!



- http://www.petapixel.com/2012/04/17/advanced-camera-sensor-technologies-explained-with-beer/



Noise and denoising

Fredo Durand
MIT EECS 6.8370/6.8371

http://www.cambridgeincolour.com/tutorials/image-noise.htm

Pset 1

- Due Wednesday 9pm
- Actually counts

- PSet partners: https://psetpartners.mit.edu/

Noise

Noisy image

- Usually for dark conditions

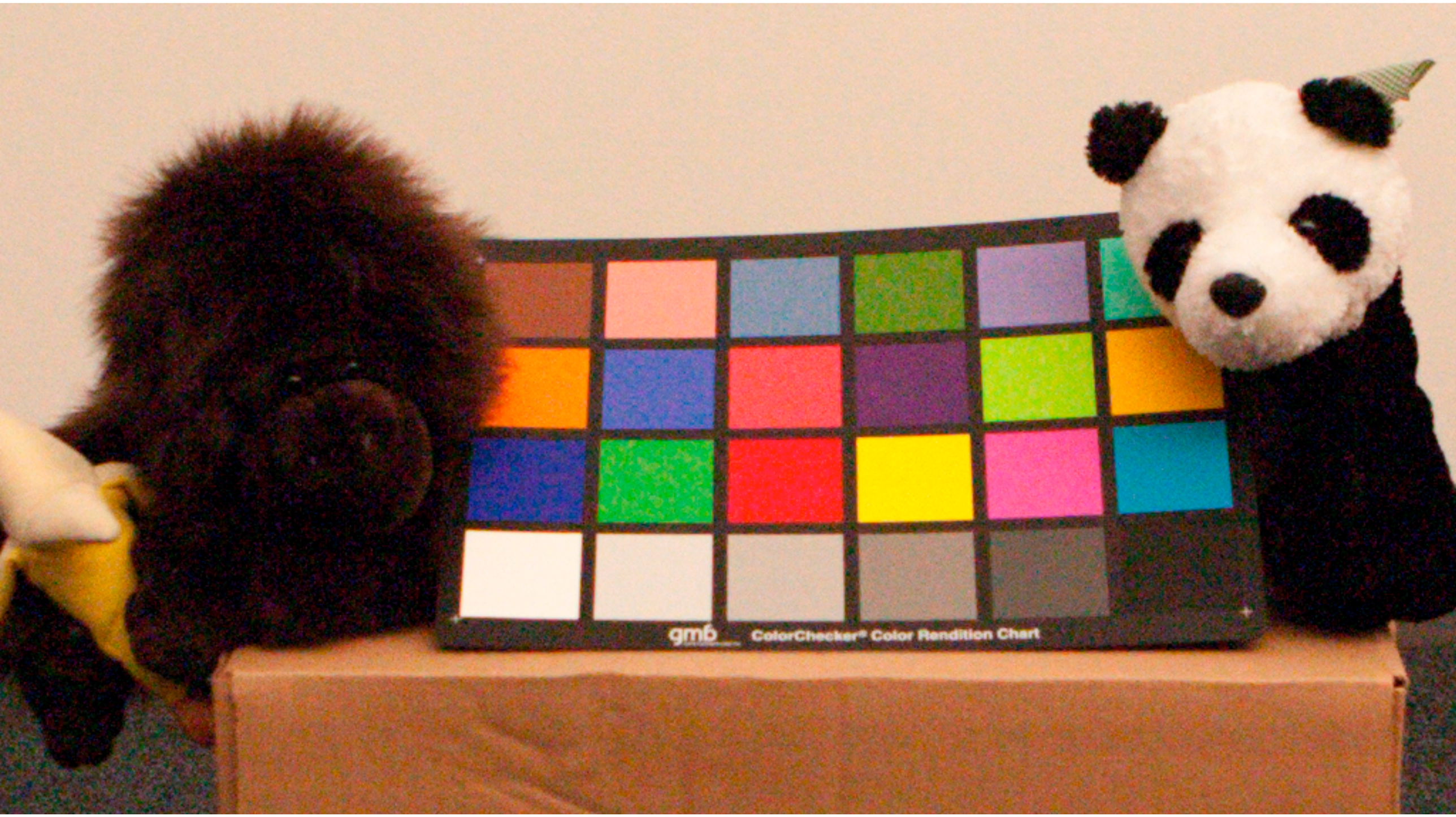


Noise

- Fluctuation when taking multiple shots

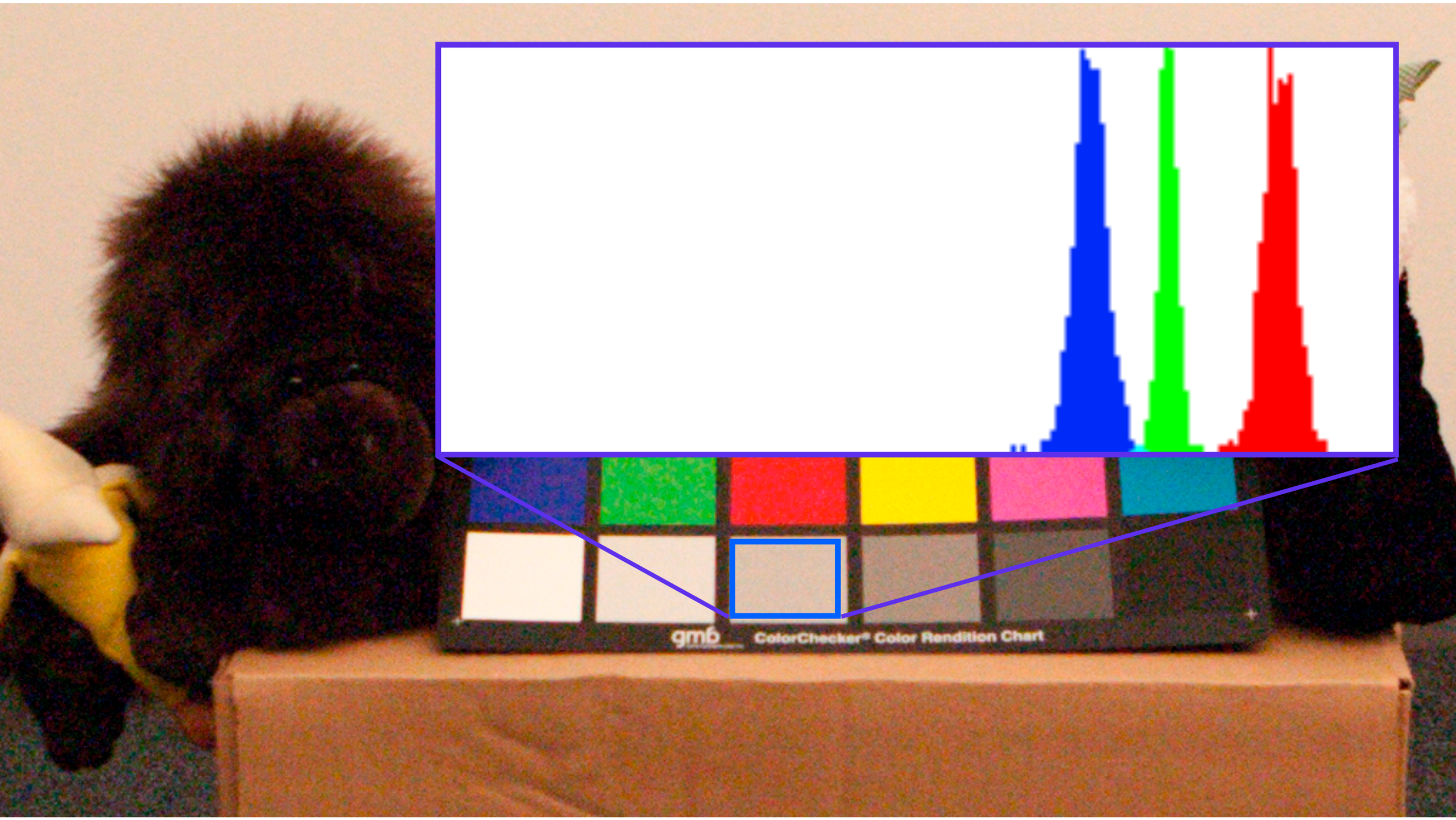


Canon 1D mark IIN at ISO 3200



Histogram of grey patch

- Should be single values for RGB (constant color)



Recap

- Noise exists
- Noise can be observed:
  - fluctuation over time
  - fluctuation over space when should be constant
- Noise is annoying

Denoising by averaging

Averaging

```
mean=constantIm(height, width, [0, 0, 0])
for i in imageList:
        mean+=imSeq[i]
mean/=len(imageList)
```

1 image



3 images



5 images



Statical/ probabilistic perspective

Noise statistics / probability

- Denote pixel values like random variables X

Poll about basic definitions

Noise statistics / probability

- Denote pixel values like random variables X
- Mean: E[x], the true measurement
- Variance = ~ average squared error
  - more precisely: average squared difference to mean

$$\sigma^2[x] = E[(E[x] - x)^2]$$

$$\sigma^2 = E[x^2] - E[x]^2$$

- Standard deviation σ: square root of variance
  - In same unit as measurement

Estimating the variance

$$\sigma^2[x] = E[(E[x] - x)^2]$$

Estimating the variance

$$\mu_N = \frac{1}{N} \sum x_i$$

$$\sigma_N^2 = \frac{1}{N-1} \sum (x_i - \mu_N)^2$$

Estimating the variance

$$\mu_N = \frac{1}{N}\sum x_i$$

$$\sigma_N^2 = \frac{1}{N-1}\sum (x_i - \mu_N)^2$$

- **Divide by N-1, not by N**
  - called Bessel correction: Removes bias
    - Otherwise,
      variance would be underestimated on average
    - Intuition: we use the same samples for estimating the mean and variance, introduces correlation that underestimate variance

Example with coin flip, N=2

- We do 2 coin flips
- Try to estimate mean & **variance**
- Sometimes will be wrong
  - e.g. if we get 0 twice, we'll think variance is zero
  - but we'd like to be right on average
    - called unbiased

Example with coin flip, N=2

- 4 scenarios: (0,0) ; (0, 1) ; (1 , 0) ; (1 , 1)
- => mean estimates: 0 ; 0.5 ; 0.5 ; 1
  - average of the mean estimations: 0.5, equal to true mean, estimator is unbiased
- true variance: 0.25
- sum square difference to mean: 0; 0.5; 0.5; 0
- estimator $\sigma_N^2 = \frac{1}{N}\sum(x_i - \mu_N)^2$ 0, 0.25, 0.25, 0
  - 0.125 on average, biased
- estimator $\sigma_N^2 = \frac{1}{N-1}\sum(x_i - \mu_N)^2$ 0, 0.5, 0.5, 0
  - 0.25 on average, unbiased

SNR

- Signal noise ratio

$$E[x^2]/\sigma^2[x]$$

- Often shown as log, in db:

$$10log_{10}\left(E[x^2]/\sigma^2[x]\right)$$

- For calibration, 30db is a good PSNR

PSNR

- in terms of the max value of the signal/sensor



PSNR=29.4477 db



PSNR=25.8187 db



PSNR=17.1598db

http://www.darktable.org/2012/12/profiling-sensor-and-photon-noise/

SNR in practice

- **Careful, sometimes variance is zero (for no good reason) and will break things.**
  - practical hack: take the max of sigma^2 and a small number, e.g. 1e-6

Questions

Convergence

Convergence of averaging

- Assume images are IID random measurements
- variance for one pixel i: $\sigma^2[x_i]$
- Average: $\frac{1}{N}\sum_{i=1}^{N} x_i$
- What is the variance of the average?
  - i.e. how fast does it converge wrt N
  - As often: more human-readable value is stddev but derivation is easier with variance

Preliminary: Variance identity

$$\sigma^2[x] = E[(E[x] - x)^2]$$

Variance identity

$$\sigma^2[x] = E[(E[x] - x)^2]$$

$$= E[E[x]^2 - 2xE[x] + x^2]$$

$$= E[x]^2 - 2E[x]E[x] + E[x^2]$$

$$= -E[x]^2 + E[x^2]$$

Variance properties

- Multiplication by k:

- Addition of two independent random variables

Variance properties

- Multiplication by k:

$$\sigma^2[kx] = E[(kx)^2] - E[kx]^2 = k^2\sigma^2[x]$$

- Addition of two independent random variables

$$\begin{aligned}
\sigma^2[x+y] &= E[(x+y)^2] - E[x+y]^2 \\
&= E[x^2 + 2xy + y^2] - (E[x] + E[y])^2 \\
&= E[x^2] + E[y^2] + \cancel{E[2xy]} - E[x]^2 - E[y]^2 - \cancel{2E[x]E[y]} \\
&= E[x^2] - E[x]^2 + E[y^2] - E[y]^2 \\
&= \sigma^2[x] + \sigma^2[y]
\end{aligned}$$

uncorrelated:
E[xy]=E[x]E[y]

Convergence

- Assume images are IID random measurements
- variance for one image: $\sigma^2[x_i]$
- Average: $\frac{1}{N}\sum_{i=1}^{N} x_i$

$$\sigma^2\left[\frac{1}{N}\sum_{i=1}^{N} x_i\right] = \left(\frac{1}{N}\right)^2 \sum_{i=1}^{N} \sigma^2[x_i]$$

$$= \left(\frac{1}{N}\right)^2 N\sigma^2[x_i]$$

$$= \frac{1}{N}\sigma^2[x_i]$$

IMPORTANT RESULT

- Denoising by averaging:
  - variance is reduced as 1/N
  - “error” (in stddev) is reduced by sqrt(N)

Questions?

Alignment

Brute force

- Pset 4!
- Try all possible shifts within /- maxOffset
- Keep the one with minimum sum of square differences

Casio EXF1, Google HDR+, etc.

- Can do denoising by aligning and averaging N images
- HDR+, Nightsight
- E.g. https://ai.googleblog.com/2014/10/hdr-low-light-and-high-dynamic-range.html
- https://ai.googleblog.com/2018/11/night-sight-seeing-in-dark-on-pixel.html

Denoising from a single image

Denoising from 1 image

- We can’t take average over multiple images



Denoising from 1 image

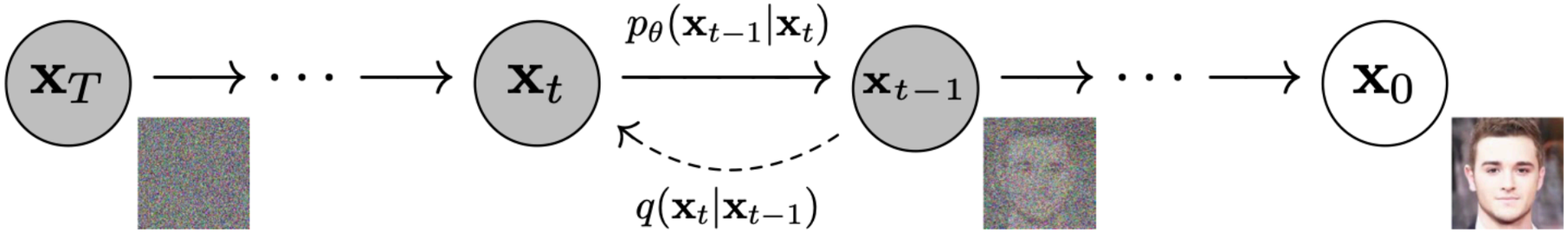
- We can't take average over multiple images

- Very vast topics
- Lots of approaches
- Most practically used:
  - pyramid-based
  - Bilateral-filter adjacent
- In research
  - Patch similarity (BM3D)
  - Deep learning



Denoising Diffusion models

- Dall-E and stable diffusion are based on denoising ideas.





https://arxiv.org/pdf/2006.11239v2.pdf

Denoising from 1 image

- We can't take average over multiple images
- Idea 1: take a spatial average
  - Most pixels have roughly the same color as their neighbor
  - Noise looks high frequency => do a low pass
- First, let's try Gaussian blur



Gaussian blur

- Noise is mostly gone
- But image is blurry
  - duh!

- (to be fair, median filtering would look better)



Bilateral filtering

Gaussian blur

- Noise is mostly gone
- But image is blurry
  - duh!

- Problem: not all neighbors have the same color

- Bilateral filter idea: only consider neighbors that have values similar enough



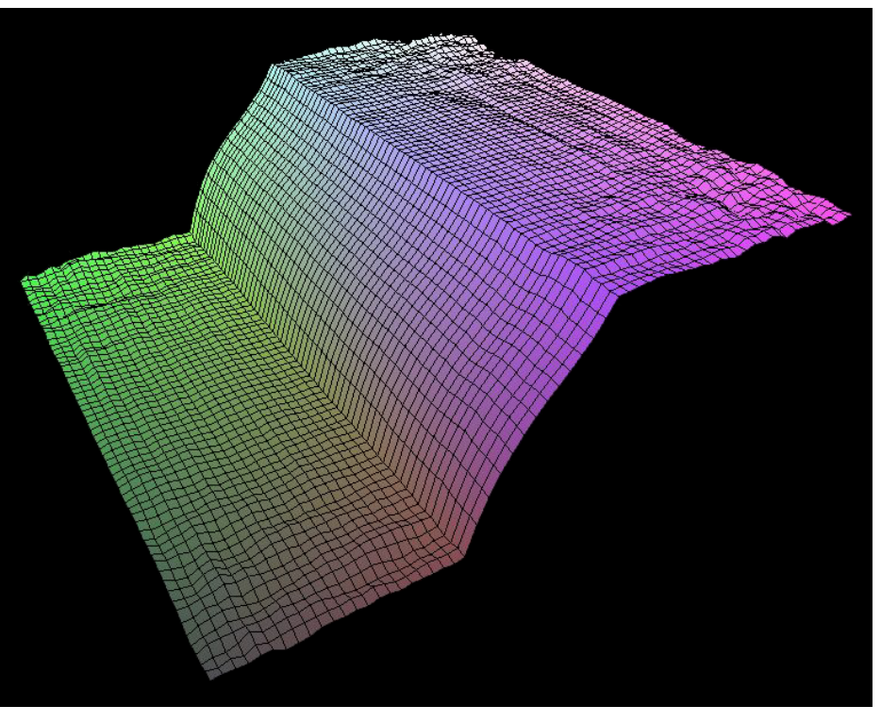
Bilateral filter

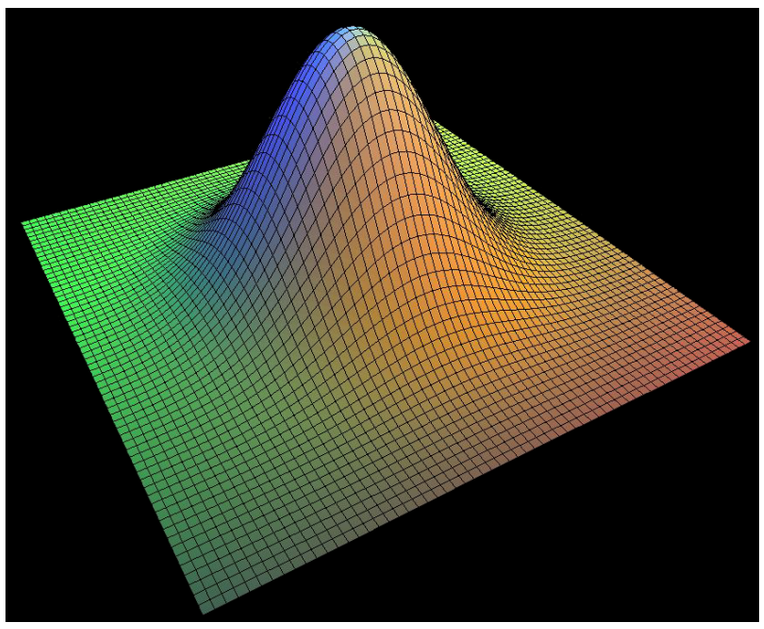
- **Tomasi and Manduci 1998]**
  - –http://www.cse.ucsc.edu/~manduchi/Papers/ICCV98.pdf
- **Developed for denoising**
- **Related to**
  - –SUSAN filter [Smith and Brady 95] http://citeseer.ist.psu.edu/smith95susan.html
  - –Digital-TV [Chan, Osher and Chen 2001] http://citeseer.ist.psu.edu/chan01digital.html
  - –sigma filter http://www.geogr.ku.dk/CHIPS/Manual/f187.htm
- **Full survey: http://people.csail.mit.edu/sparis/publi/2009/fntcgv/Paris_09_Bilateral_filtering.pdf**

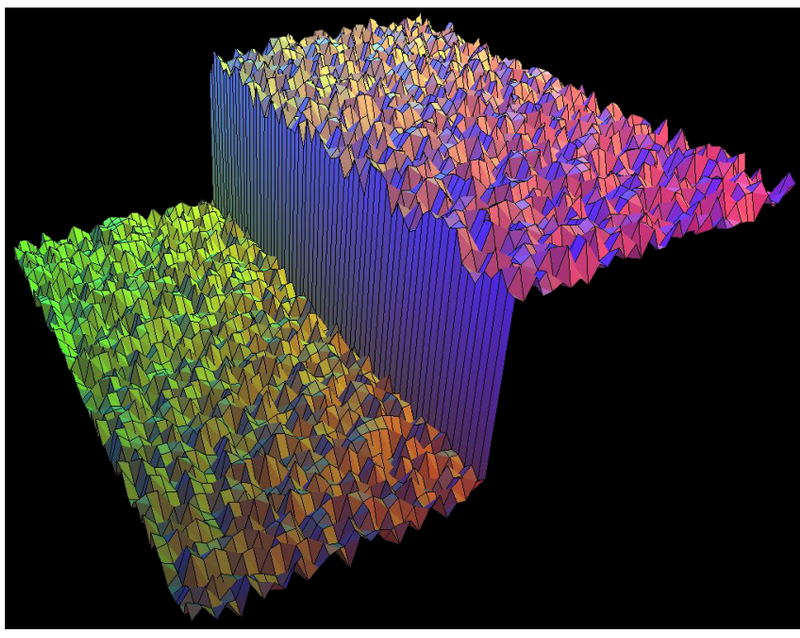
Start with Gaussian filtering

- **Here, input is a step function + noise**

$$J = f \otimes I$$





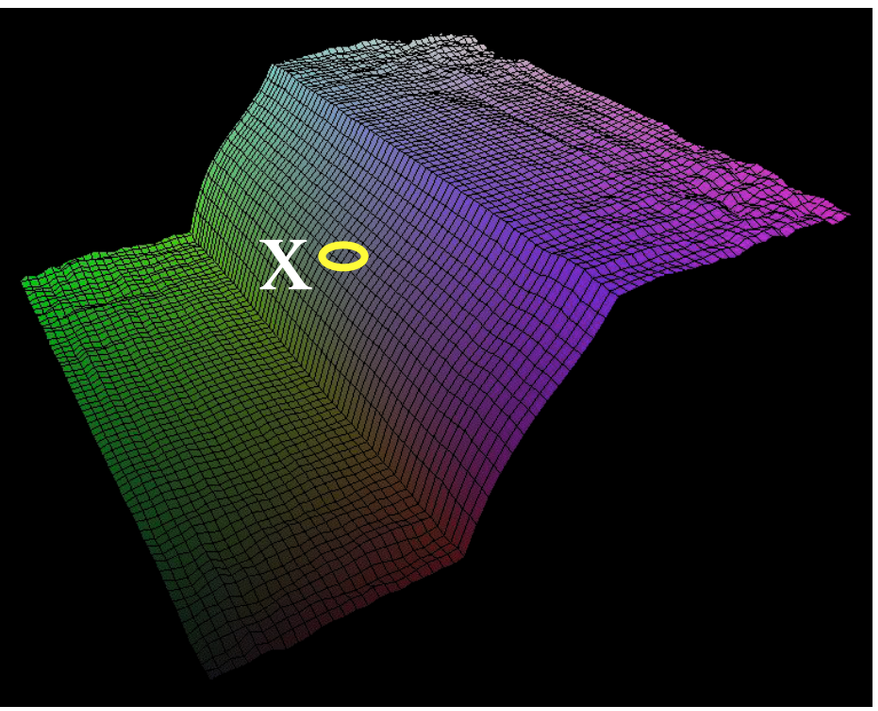


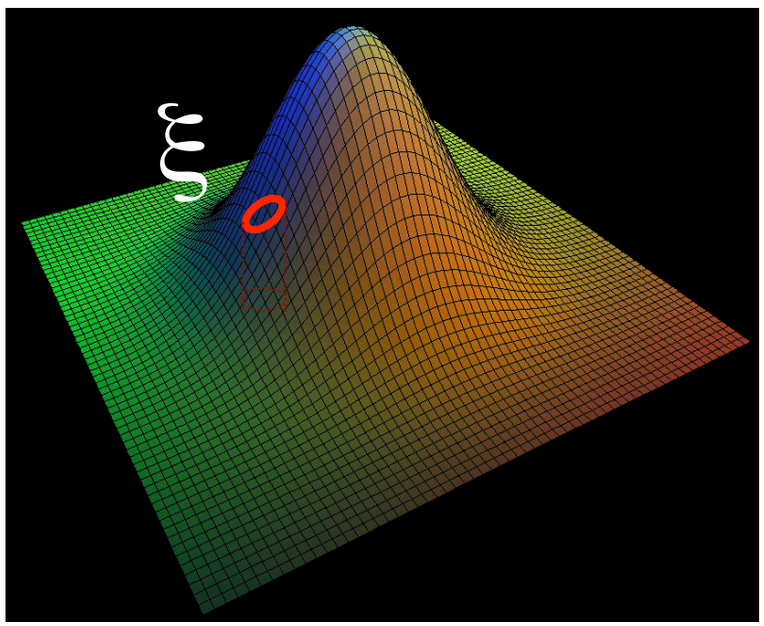
output ⟵ input

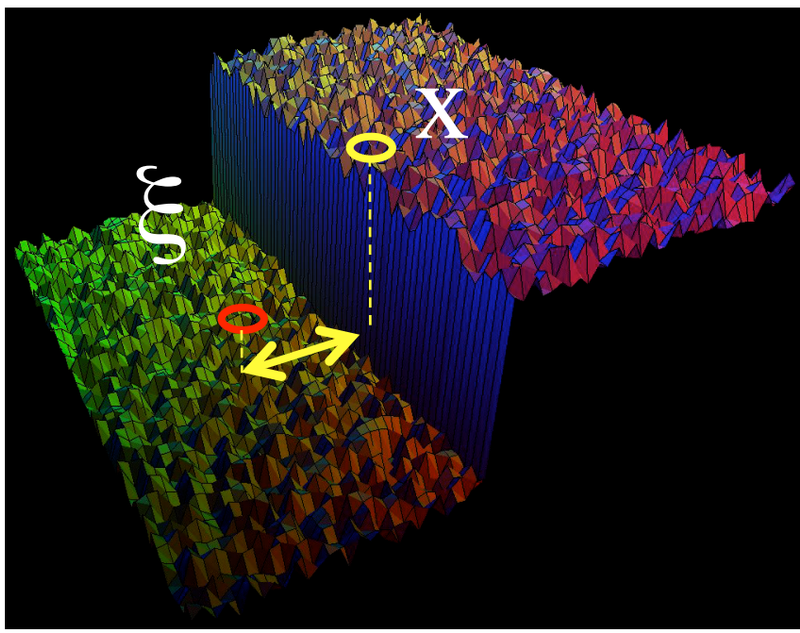
Gaussian filter as weighted average

- **Weight for ξ depends on distance to x**

$$J(x) =$$





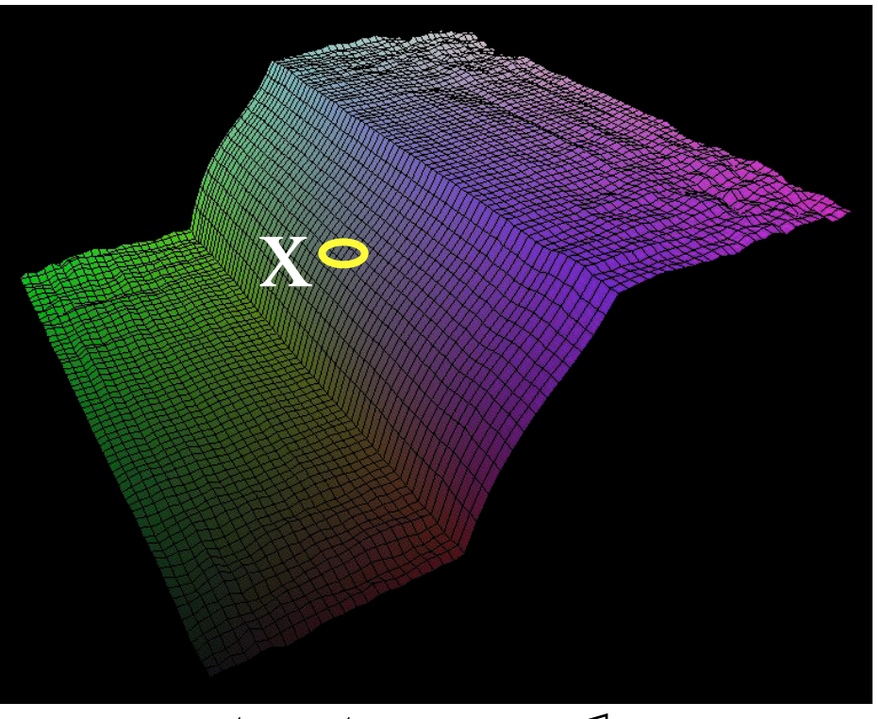


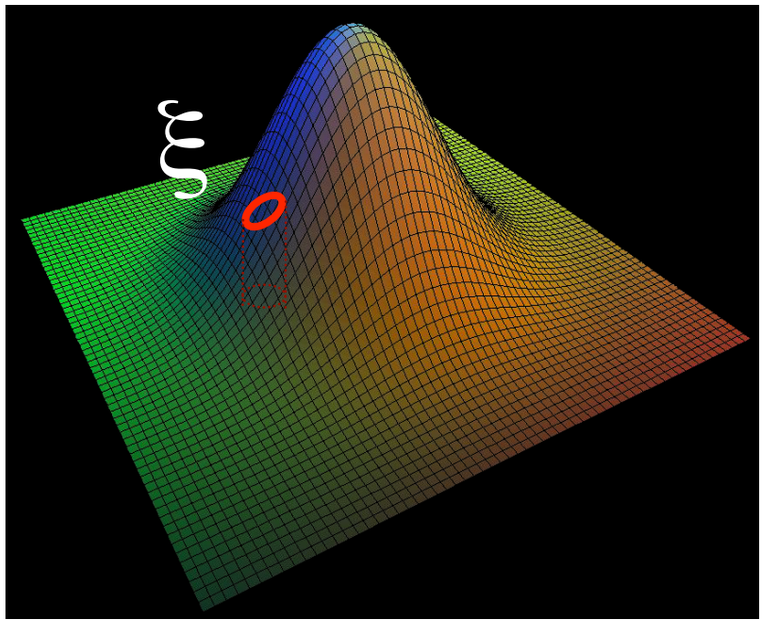
output ⟵ input

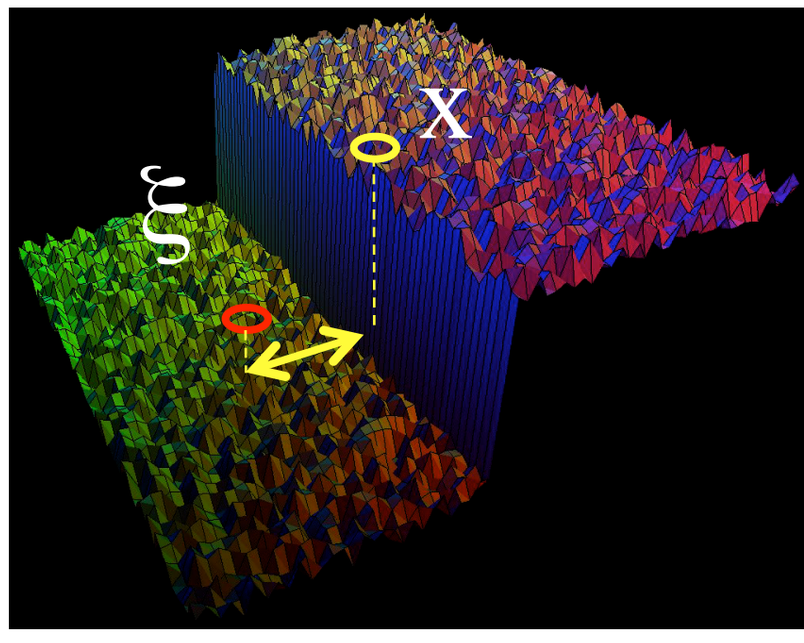
Gaussian filter as weighted average

- **Weight for ξ depends on distance to x**

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





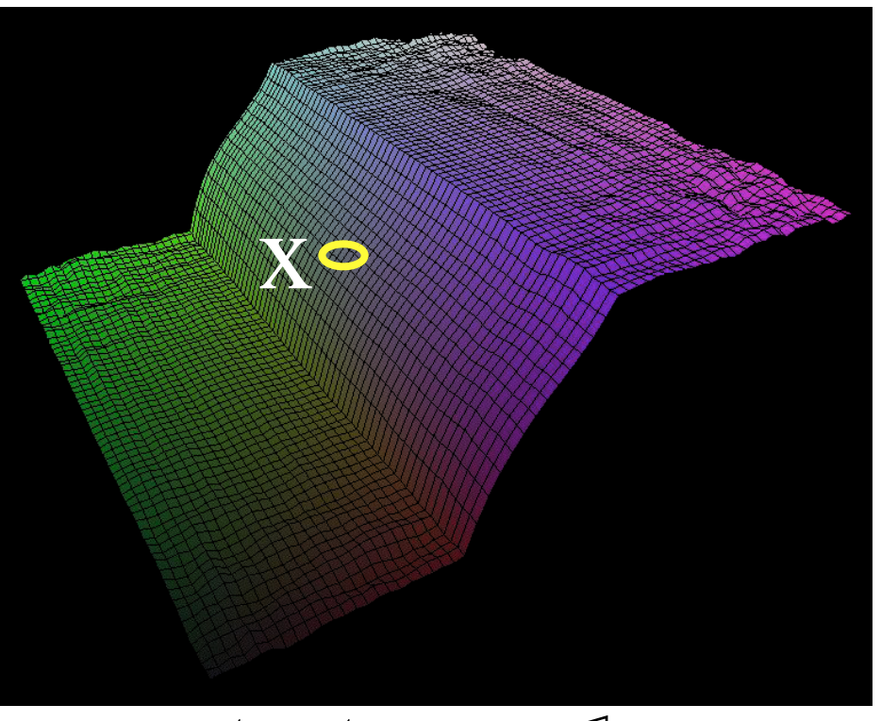


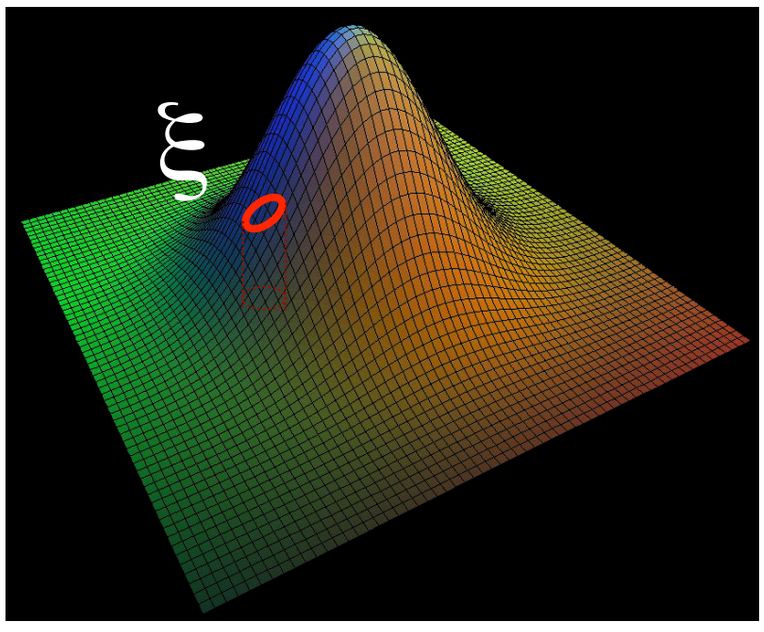
output ⟵ input

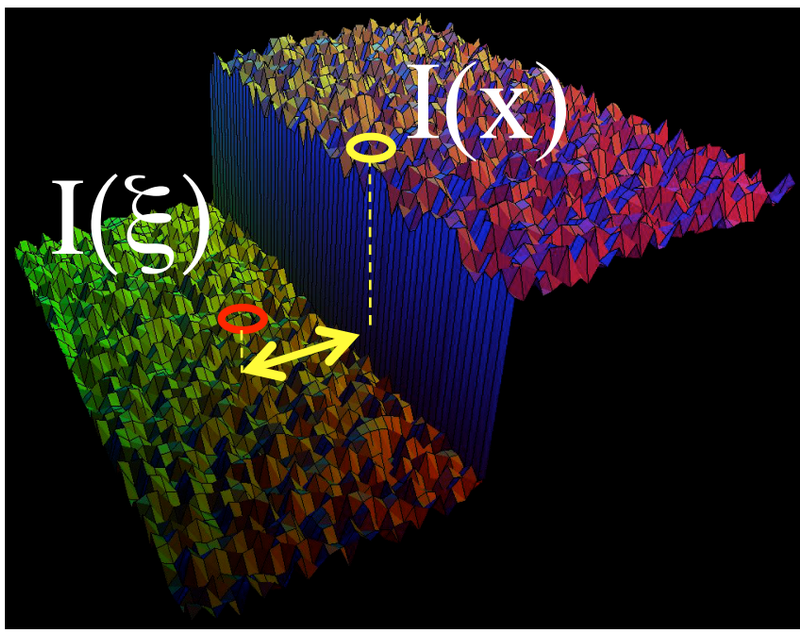
The problem of edges

- **Does $I(\xi)$ give us useful information about the value at x**

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





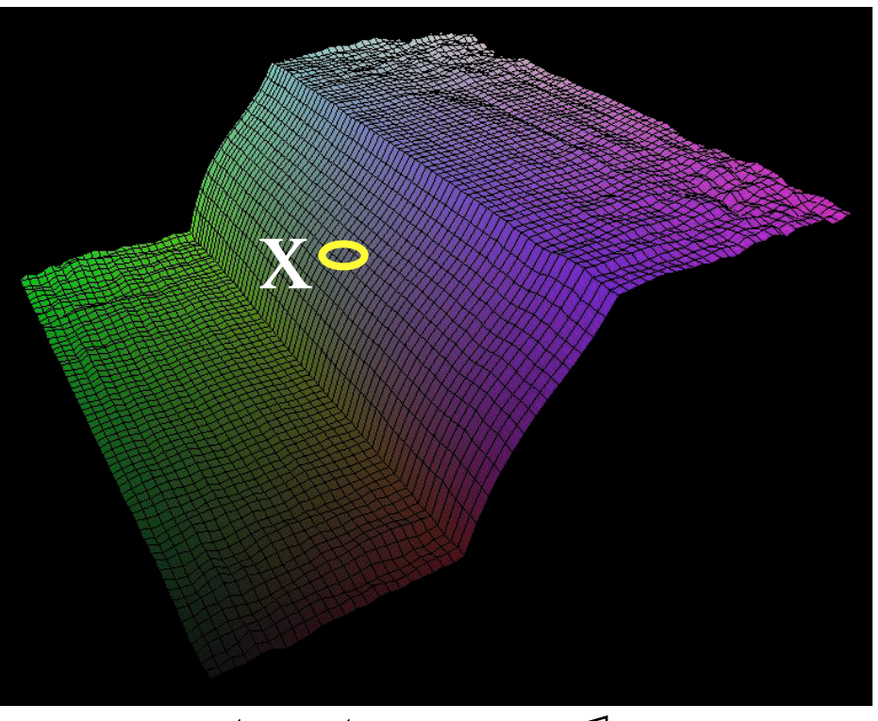


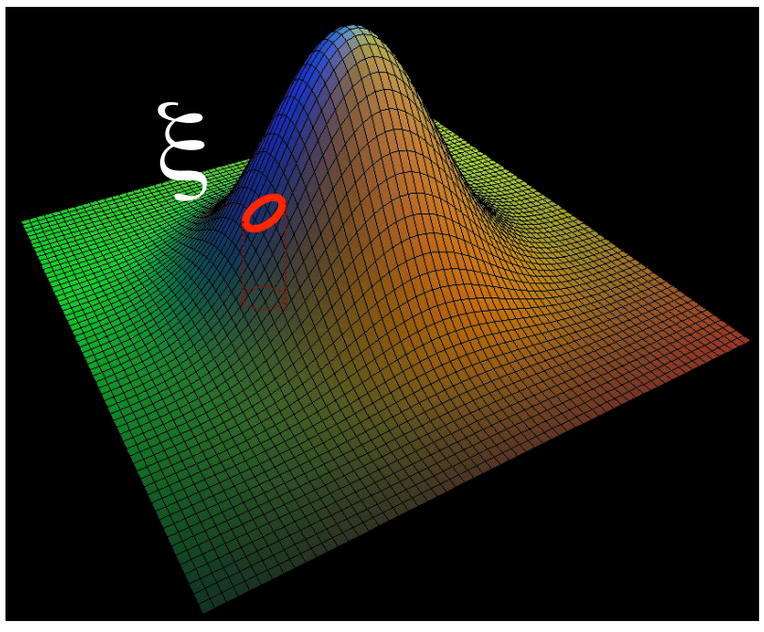
output ⟵ input

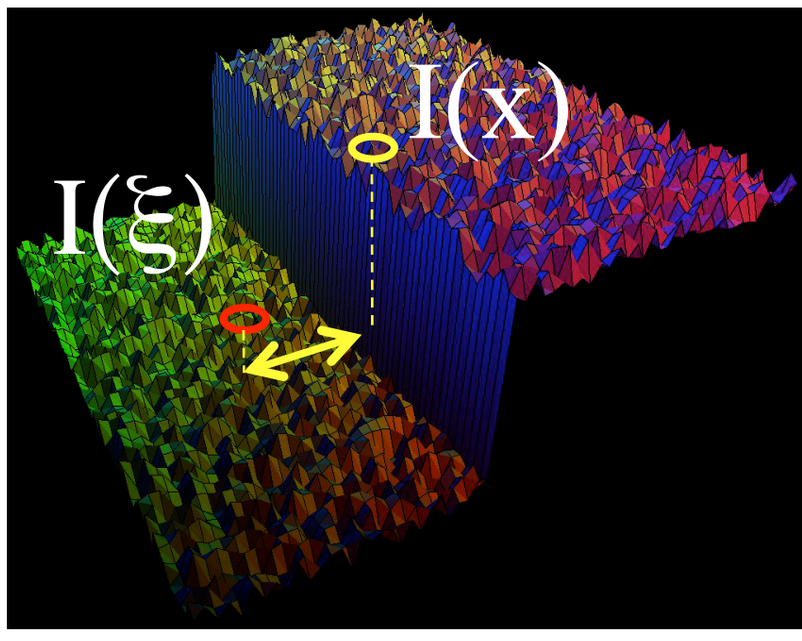
The problem of edges

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

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





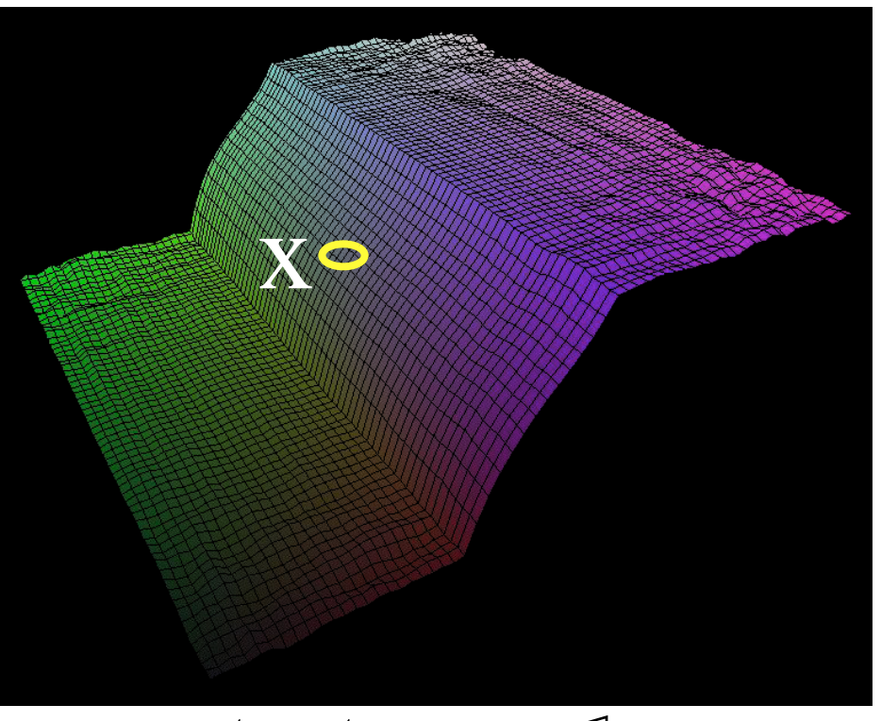


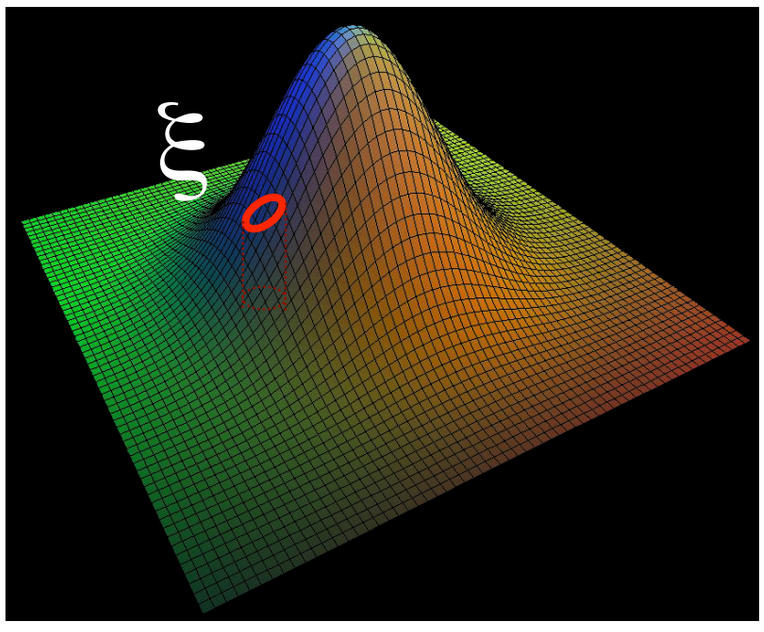
output ⟵ input

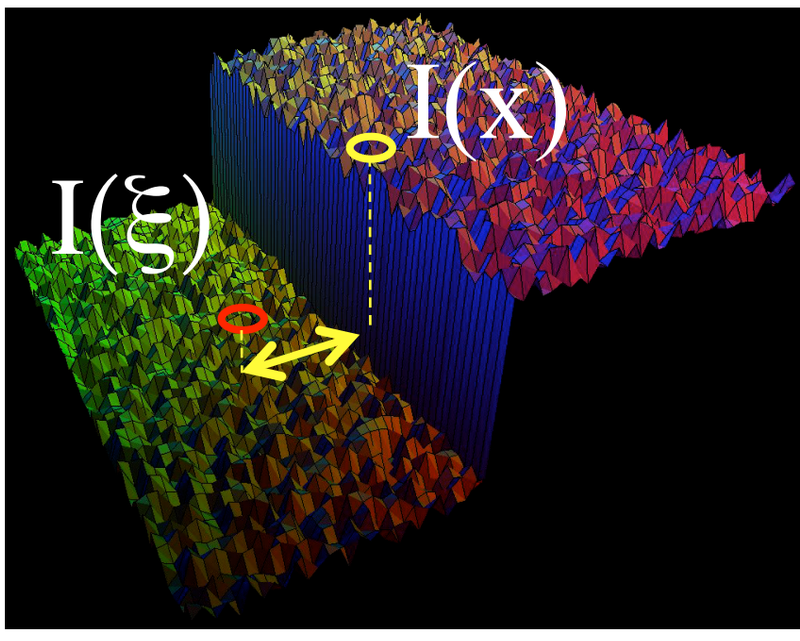
The problem of edges

- **Here, $I(\xi)$ "pollutes" our estimate J(x)**
- **Basic idea: reject $I(\xi)$ or at least give it a low weight**

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





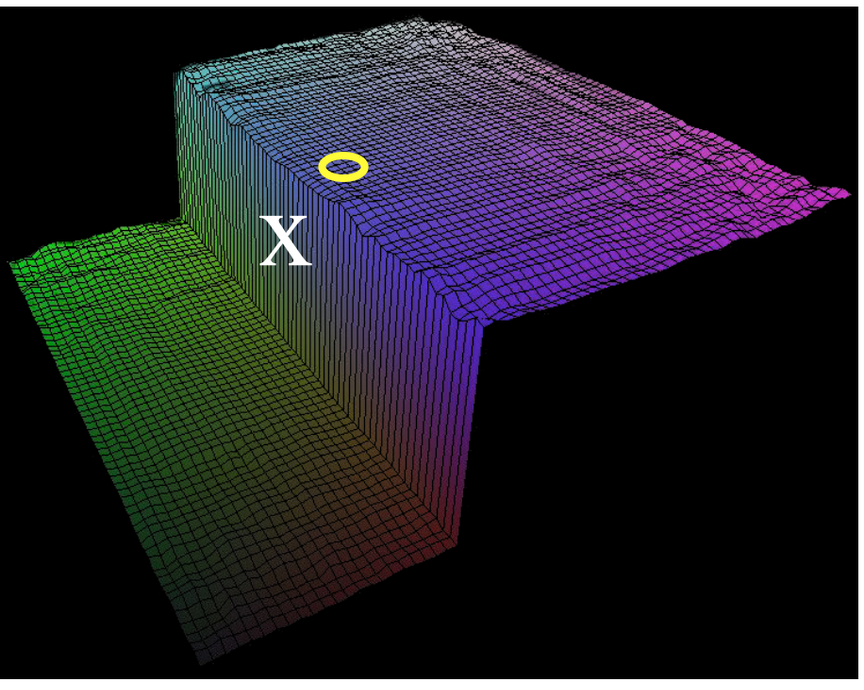


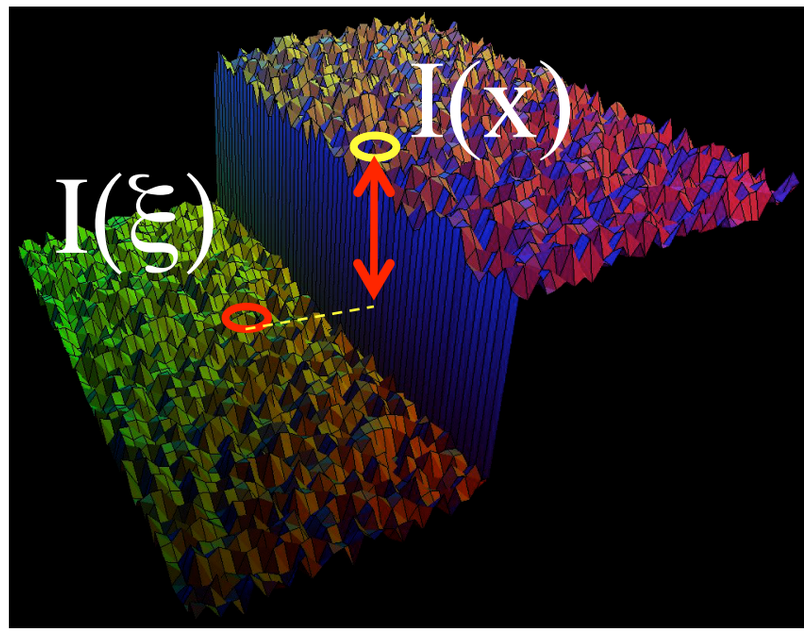
output ⟵ input

Principle of Bilateral filtering

[Tomasi and Manduchi 1998]

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





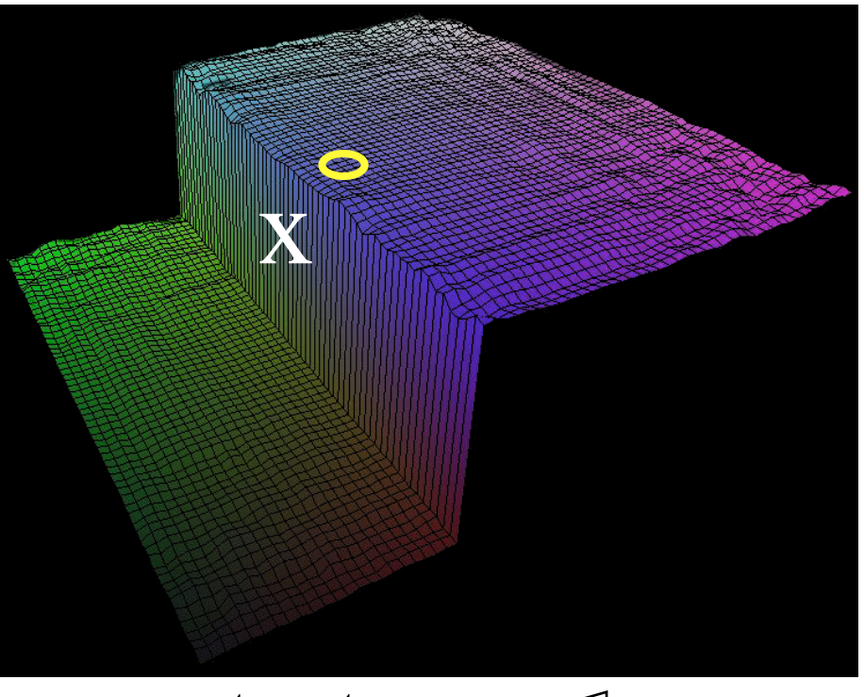
output ⟵ input

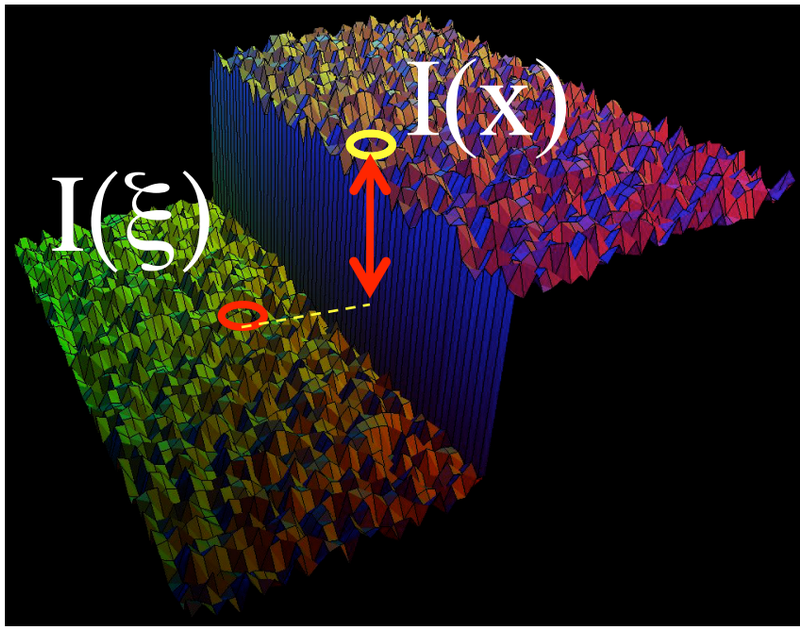
Principle of Bilateral filtering

[Tomasi and Manduchi 1998]

- **Penalty g on the intensity difference**
  - g is small when intensity difference is high

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





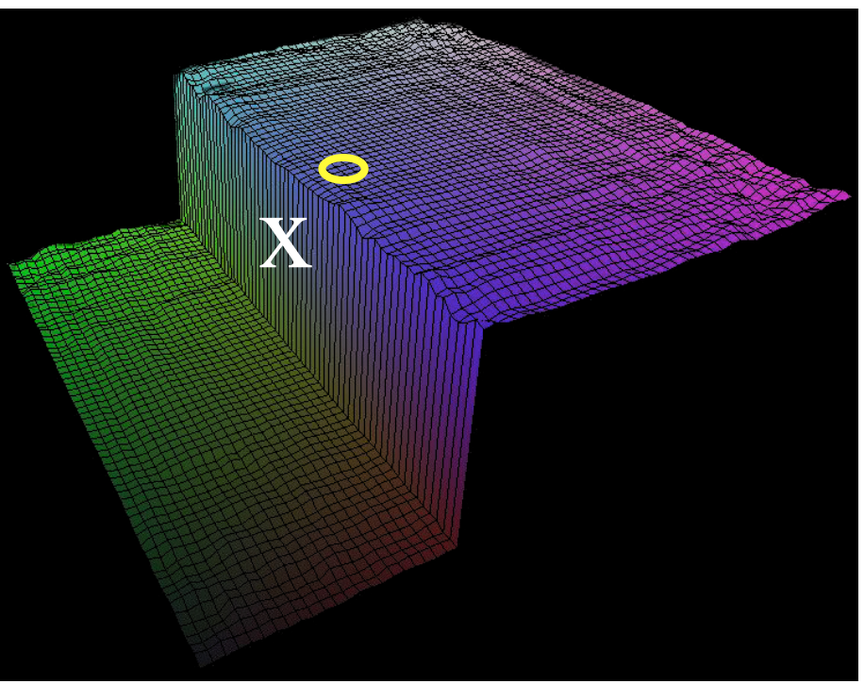
output ⟵ input

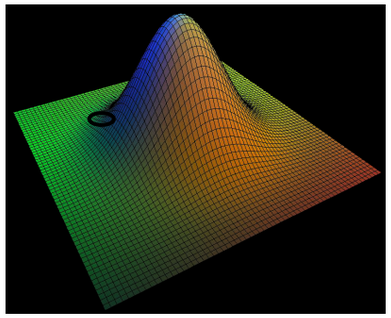
Bilateral filtering

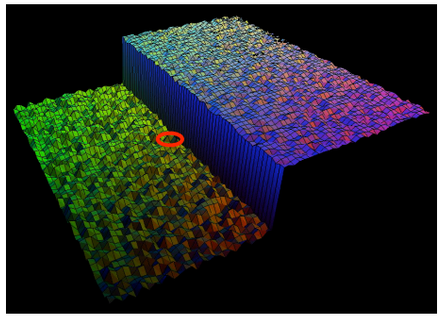
[Tomasi and Manduchi 1998]

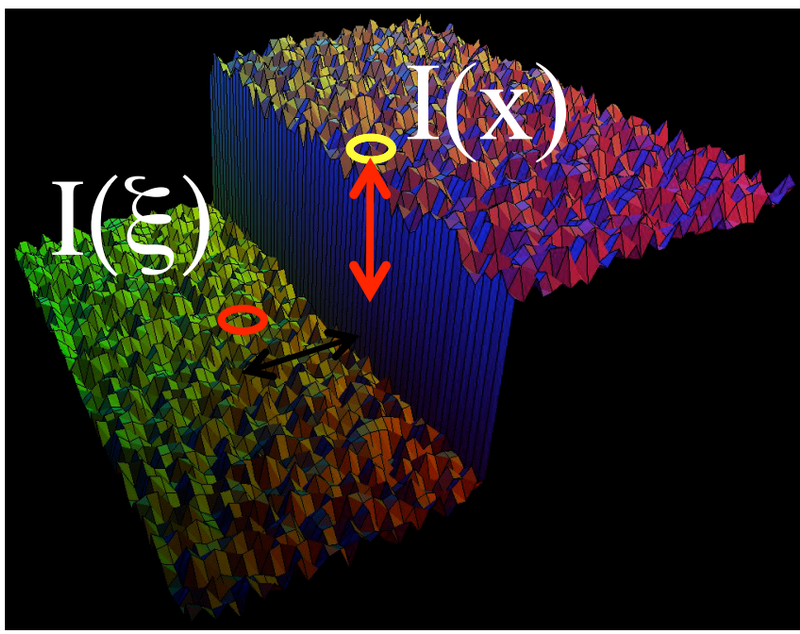
- **Spatial Gaussian f**
- **Gaussian g on the intensity difference**

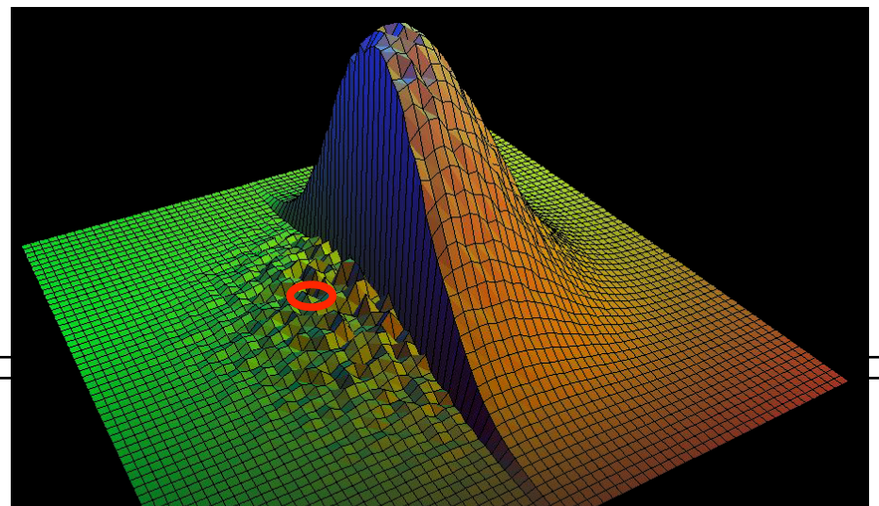
$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi)\, g(I(\xi) - I(x))I(\xi)$$











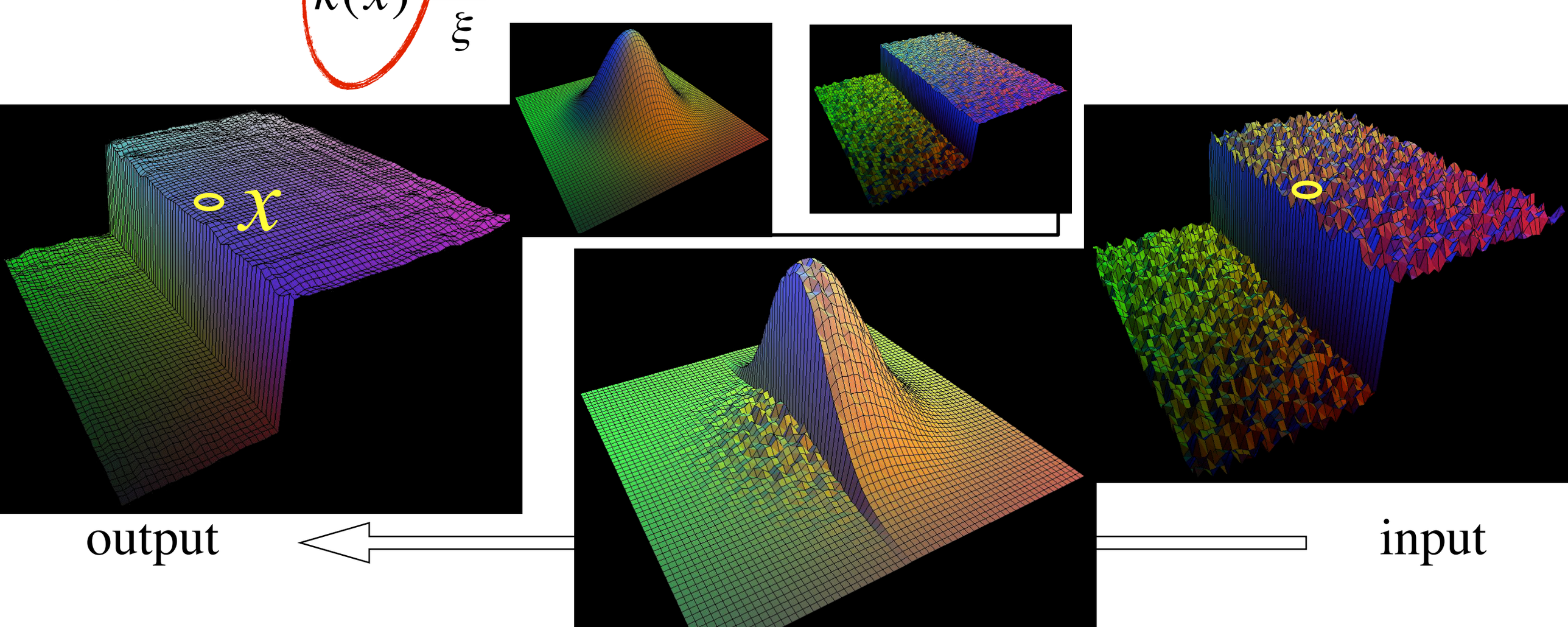
output

input

Normalization factor k

- For convolution, we can normalize the kernel
- But here, the weights change for each pixel
- needs a per-pixel normalization 1/k(x)

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

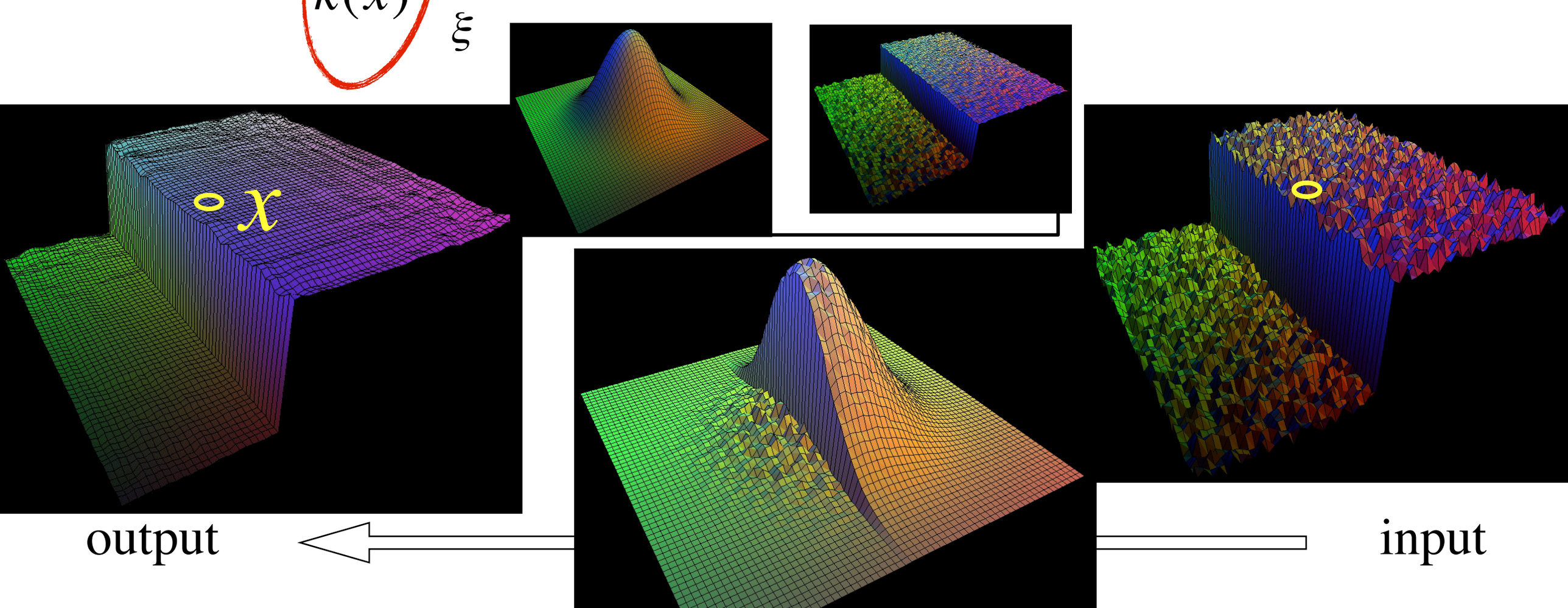


Normalization factor k

sum of the weights (just like for any weighted average)
different for each pixel!

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

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

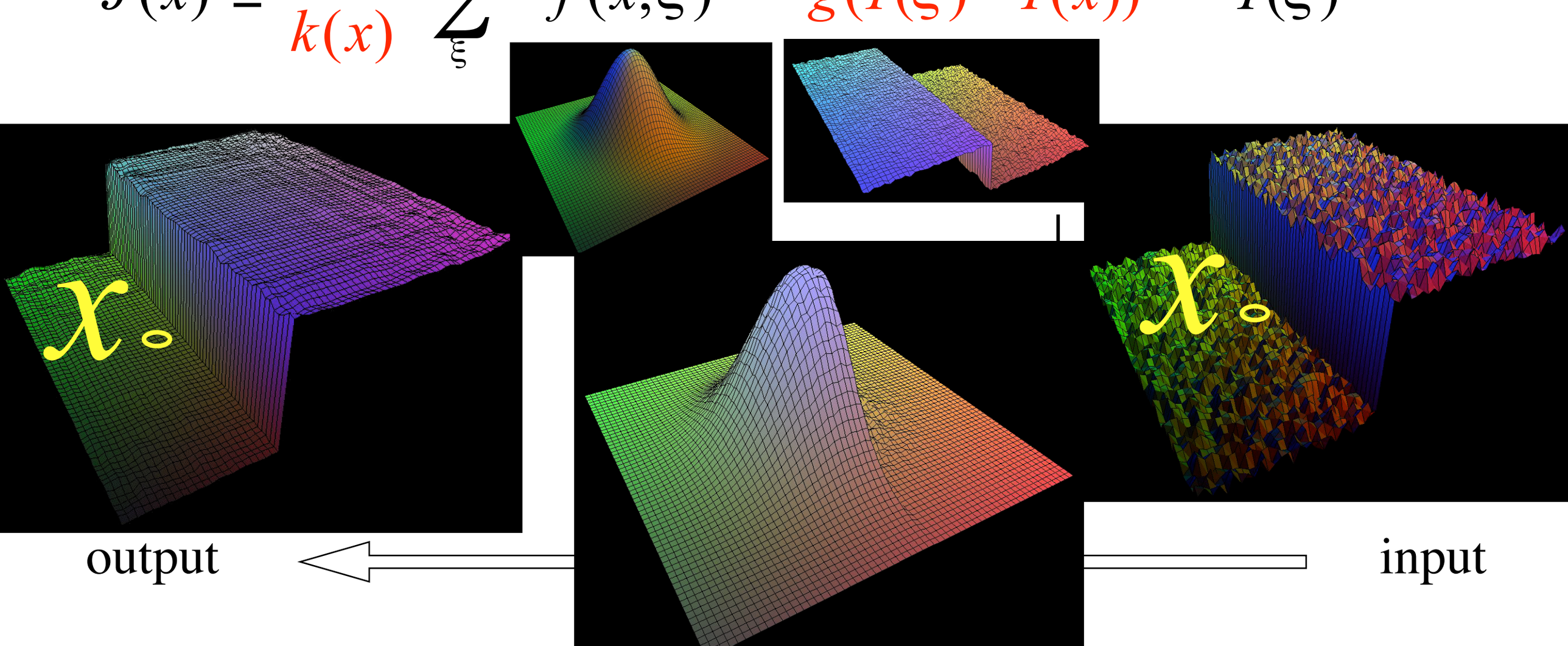


Bilateral filtering is non-linear

[Tomasi and Manduchi 1998]

- **The weights are different for each output pixel!**
  - Not a shift-invariant linear convolution

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

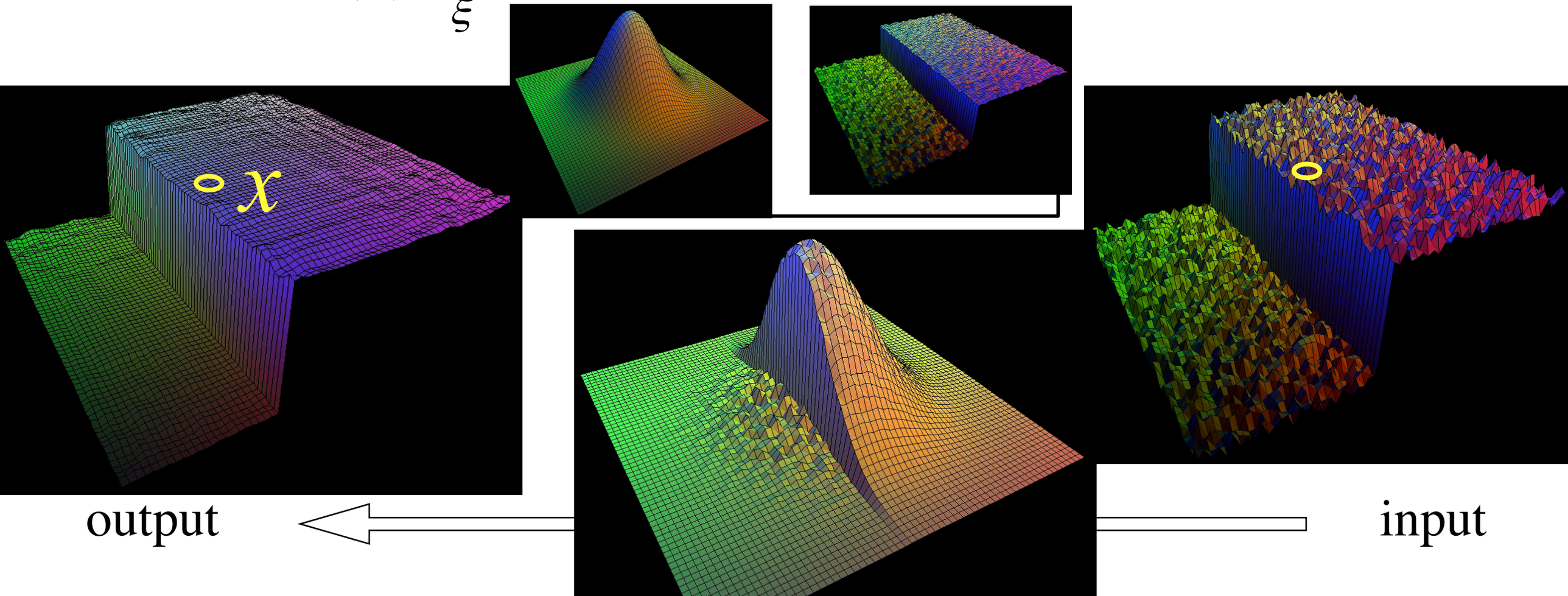


Normalization factor

[Tomasi and Manduchi 1998]

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

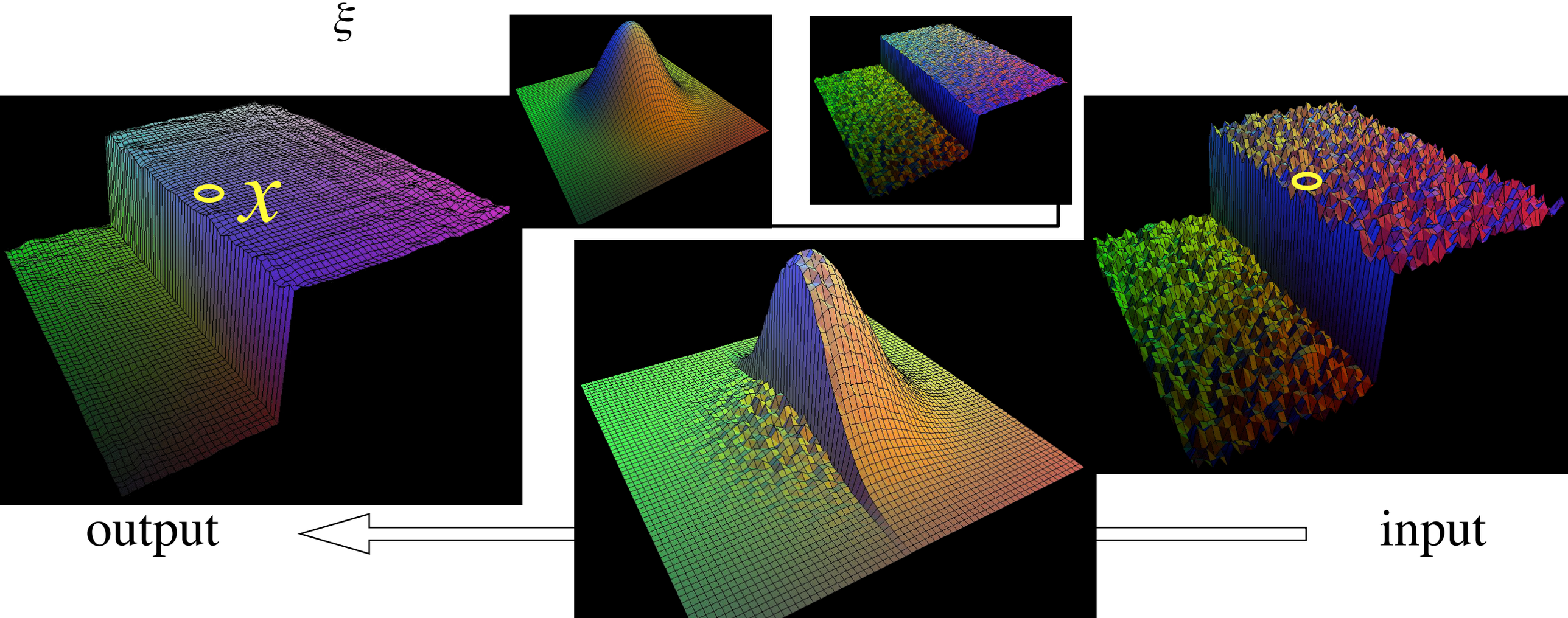
$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi) - I(x)) \quad I(\xi)$$



Let's dissect the equation

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

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



Pseudocode

- **precompute spatial weight f**
- **For each output pixel p**
  - initialize sums to zero
  - for each input pixel q in neighborhood
    - compute range weight g(Ip-Iq)
    - add f(p-q)* g(Ip-Iq)*I(q) to sum
    - add f(p-q)* g(Ip-Iq) to sum of weights
  - output pixel = sum/sum of weights

Range distance

- **For RGB images:**
  use 3D distance in RGB to compute Gaussian

IMPORTANT

- **Unlike the spatial Gaussian, you should not precompute weights for the range Gaussian**
- **Just recompute the Gaussian function for each pixel intensity difference.**
- **There are ways to tabulate, just don't worry about them**
  - (we will cover accelerations for the bilateral filter later)
- **Do not try to reuse your convolution code.**

Bilateral filtering debug

- Large range variance should yield Gaussian blur
- Tiny range variance should leave edges unaffected
- **Try half white - half black image.**
  - maybe add noise

Truncation IN SPACE

- **Same as for Gaussian**
  - –can use 2sigma instead of 3sigma if performance is an issue.
  - –Only truncate spatial Gaussian!!!
    - Range Gaussian needs to be recomputed for each pair of pixels

Color/intensity distance

Bilateral filter



Noisy input



After bilateral filter

Recap: bilateral denoising

- Spatial averaging
- Don’t go across strong edges
  - non-linear term based on intensity difference

Chrominance vs. luminance

Can we do better?



Noisy input



After bilateral filter

Chroma noise

- Our visual system has different spatial frequency response to chrominance vs. luminance
- Perform Bilateral filtering in YUV
- Bigger spatial filter in U & V

Photoshop demo!

Normal RGB Bilateral filter



Noisy input



After bilateral filter

YUV bilateral filter



Noisy input



After YUV bilateral filter

Comparison



Noisy input



Bilateral filter



YUV bilateral filter

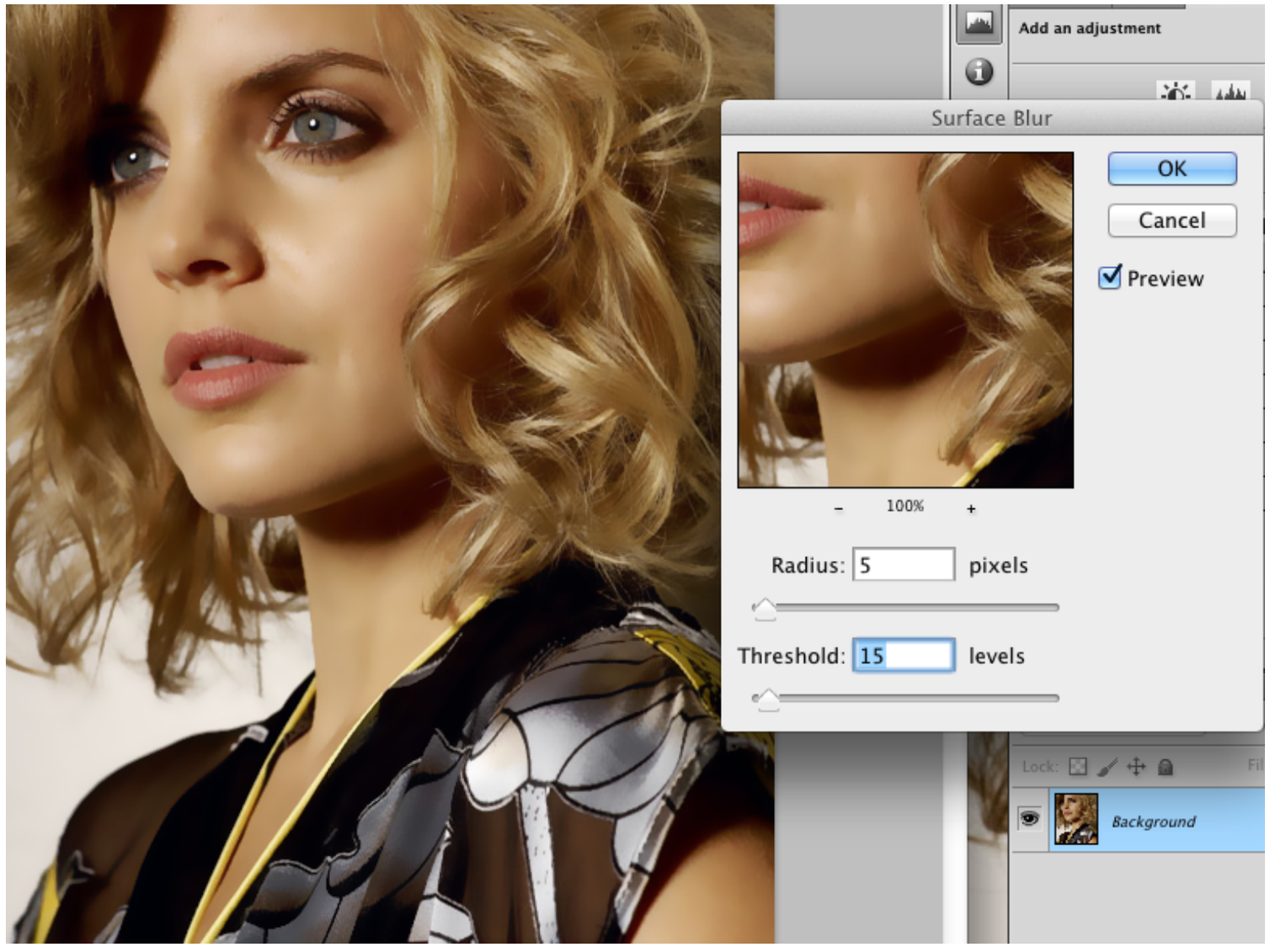
Bilateral filtering

- **Also used to remove skin blemishes in portraits**
  - Surface blur in photoshop
    - Although box spatial kernel instead of Gaussian
- **Useful for lots of other things**
  - More in future lectures
  - In particular, tone mapping for contrast reduction and high-dynamic-range imaging

Photoshop surface blur

- Note the radius and threshold controls
  - same as sigma_domain and sigma_range





Pset 3

Pset 3

- Convolution
- Separable
- Unsharp mask
- Gradient
- Denoising
- YUV denoising (6.865)

Other approaches to denoising

Denoising

- **Shrinkage/Bayesian coring in the wavelet/pyramid domain**
  - Donoho et al. ; Simoncelli & Adelson
  - Still one of the most popular in practice
- **NL means, BM3D**
  - Buades et al. : Bilateral in the space of patches
- **Deep learning**
- **Limits of denoising**
  - http://www.wisdom.weizmann.ac.il/~levina/

References

- http://www.cambridgeincolour.com/tutorials/image-noise.htm
- http://www.cambridgeincolour.com/tutorials/image-noise-2.htm
- http://www.clarkvision.com/imagedetail/does.pixel.size.matter2/
- http://www.clarkvision.com/articles/digital.sensor.performance.summary/
- http://books.google.com/books?id=OYFYt5C4N94C&pg=PA405&dq=binomial+film+grain+noise#v=onepage&q=binomial%20film%20grain%20noise&f=false
- http://en.wikipedia.org/wiki/Image_noise
- http://www.picturecode.com/noise.htm
- http://www.instructables.com/id/Avoiding-Camera-Noise-Signatures/
- http://www.photoxels.com/tutorial_noise.html
- http://people.csail.mit.edu/hasinoff/hdrnoise/hasinoff-sensornoise-tutorial-iccp10.pptx
- http://theory.uchicago.edu/~ejm/pix/20d/tests/noise/
- http://www.imatest.com/docs/noise/
- http://www.cambridgeincolour.com/tutorials/image-averaging-noise.htm
- http://www.petapixel.com/2012/02/21/a-simple-explanation-of-how-iso-works-in-digital-photography/