CS 89.15/189.5, Fall 2015

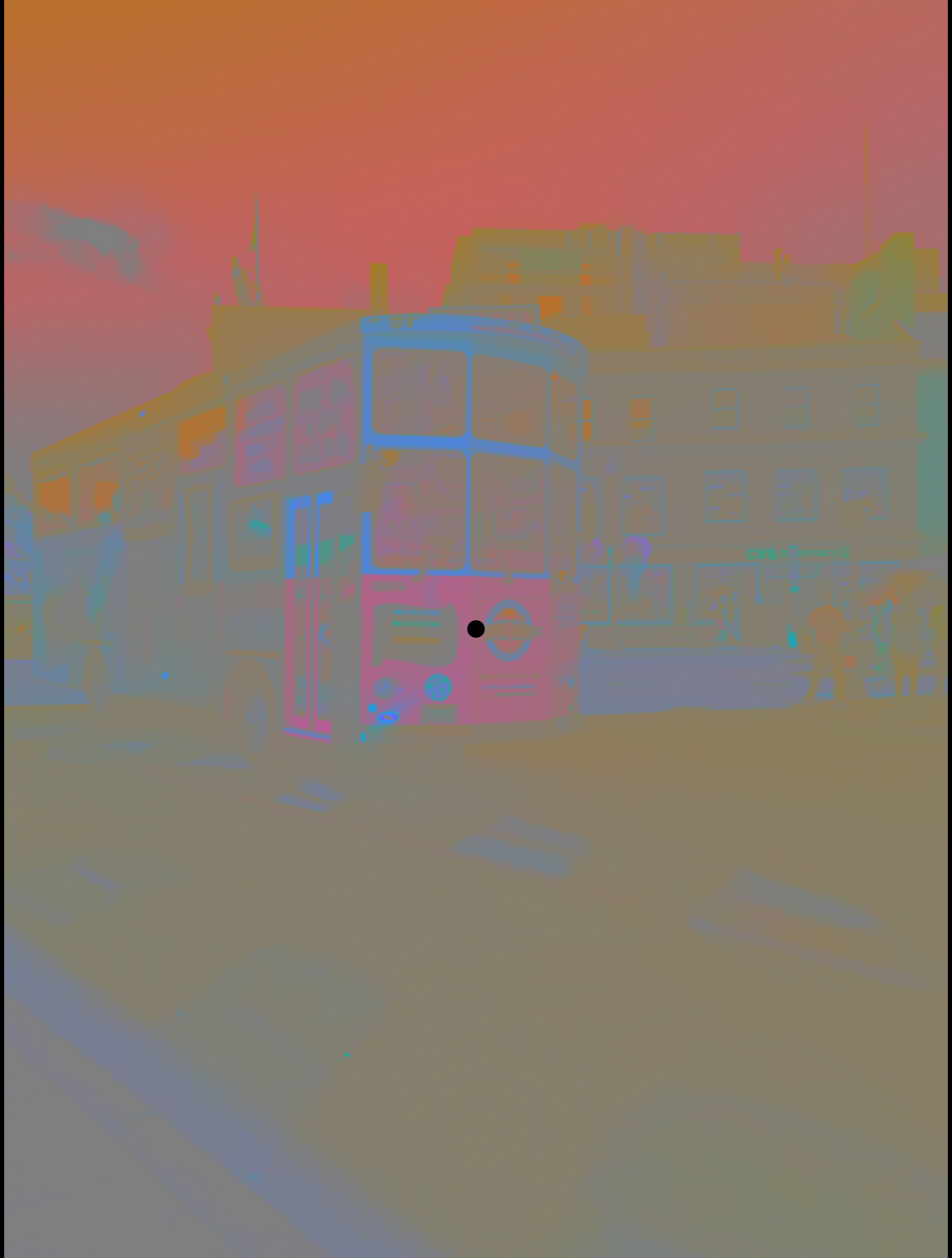
Computational Aspects of Digital Photography

Filtering & convolution

Wojciech Jarosz
wojciech.k.jarosz@dartmouth.edu

Dartmouth

Your Spanish castle illusions



Yuan Tian


13'4"
HOP ON & OFF ALL DAY
UD914
aerie
SUPER TOURS
HOP ON & OFF ALL DAY
$37
19731
CVS/pharmacy

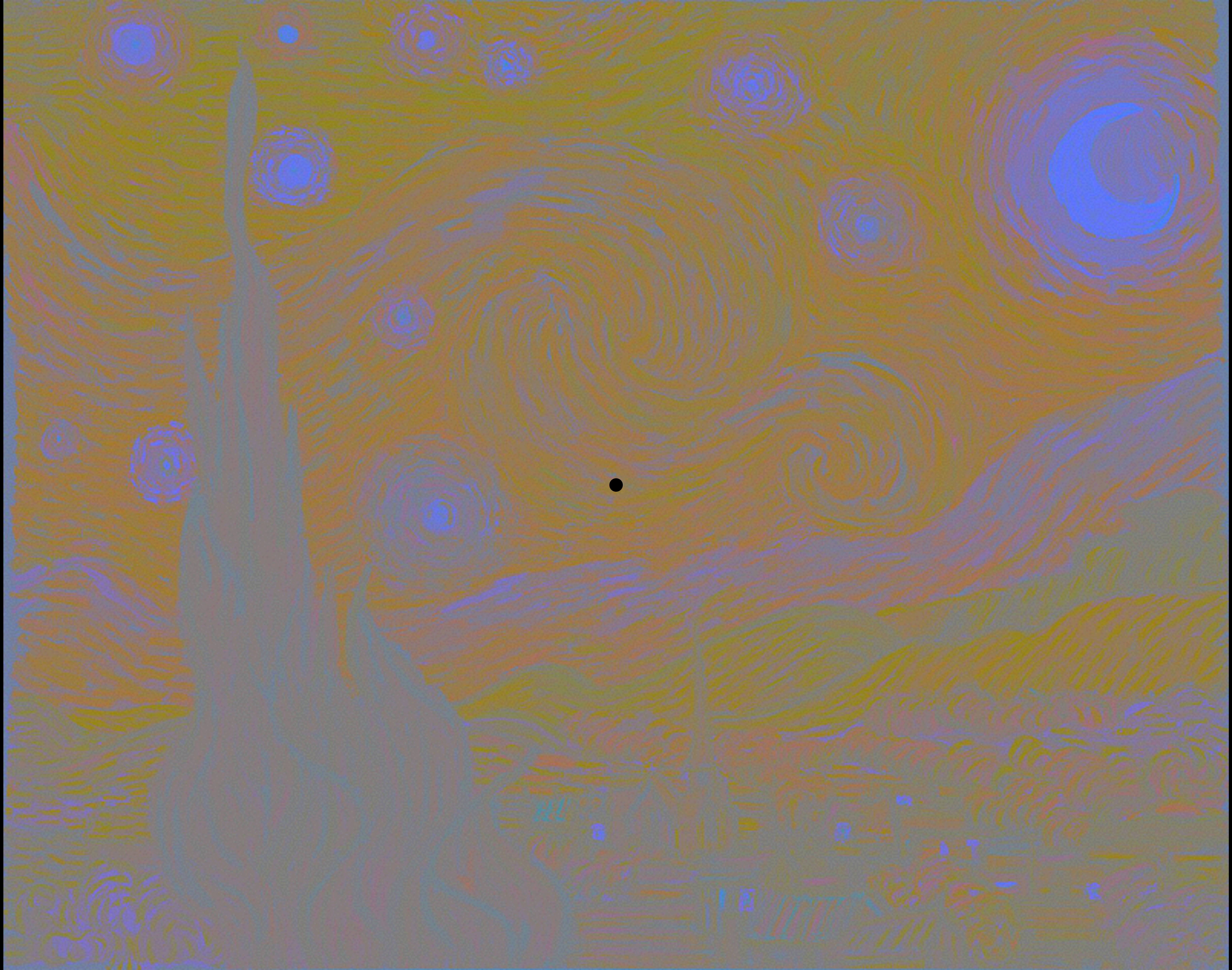
Yuan Tian

Tianyuan Zhang

Tianyuan Zhang

Weichen

Weichen



Maxim

Maxim

Timelapse photography in the news

https://youtu.be/MPql1VHbYl4

Today's agenda

Linear filtering & convolution

- blurring
- sharpening

Complexity analysis

- Optimizations

Denoising from a single image

- Bilateral filtering

Blur, sharpen

Image processing motivation

Sharpen images

Downsample images

Fake depth of field

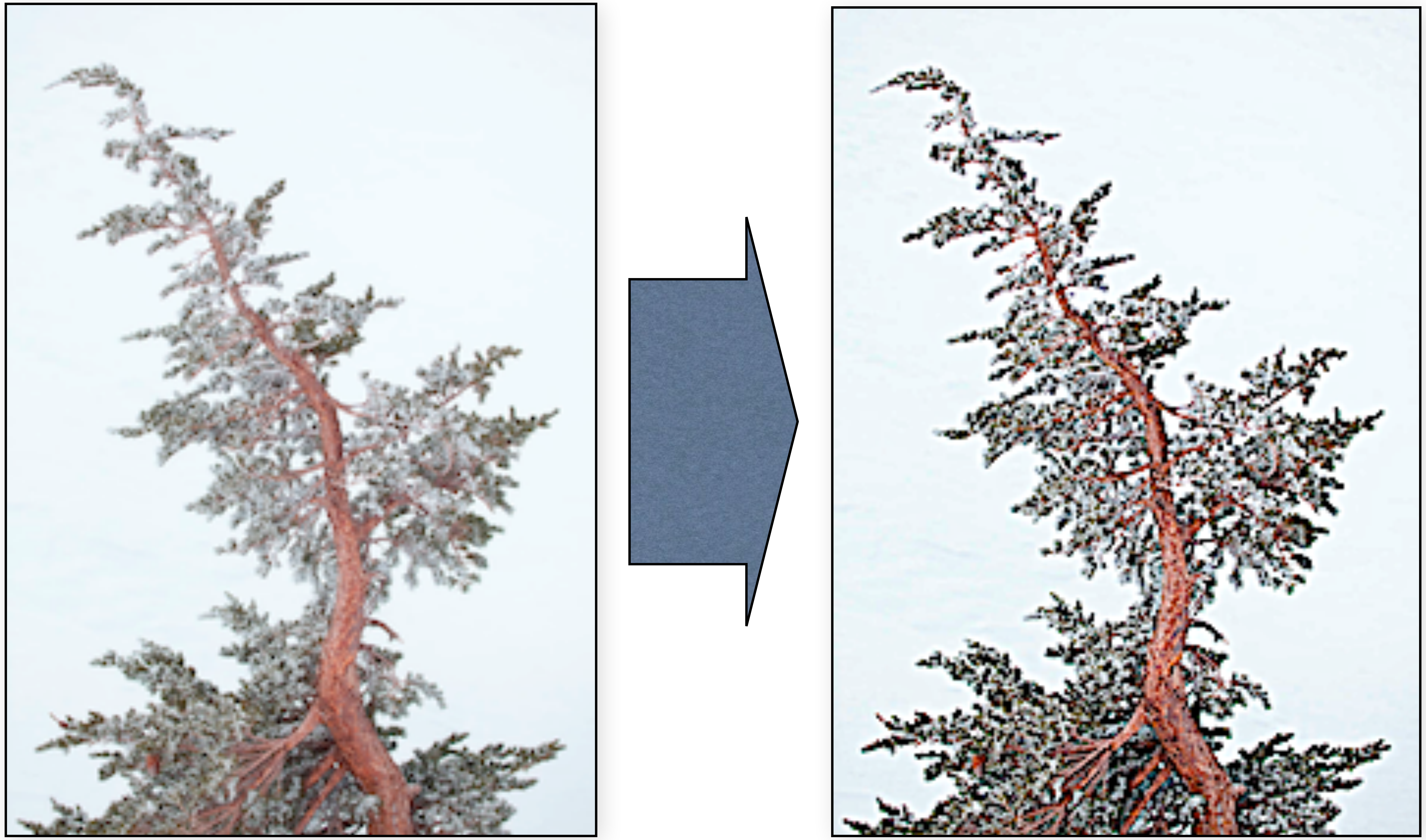
Smooth out noise, skin blemishes

…

We must understand convolution!

After a slide by Frédo Durand

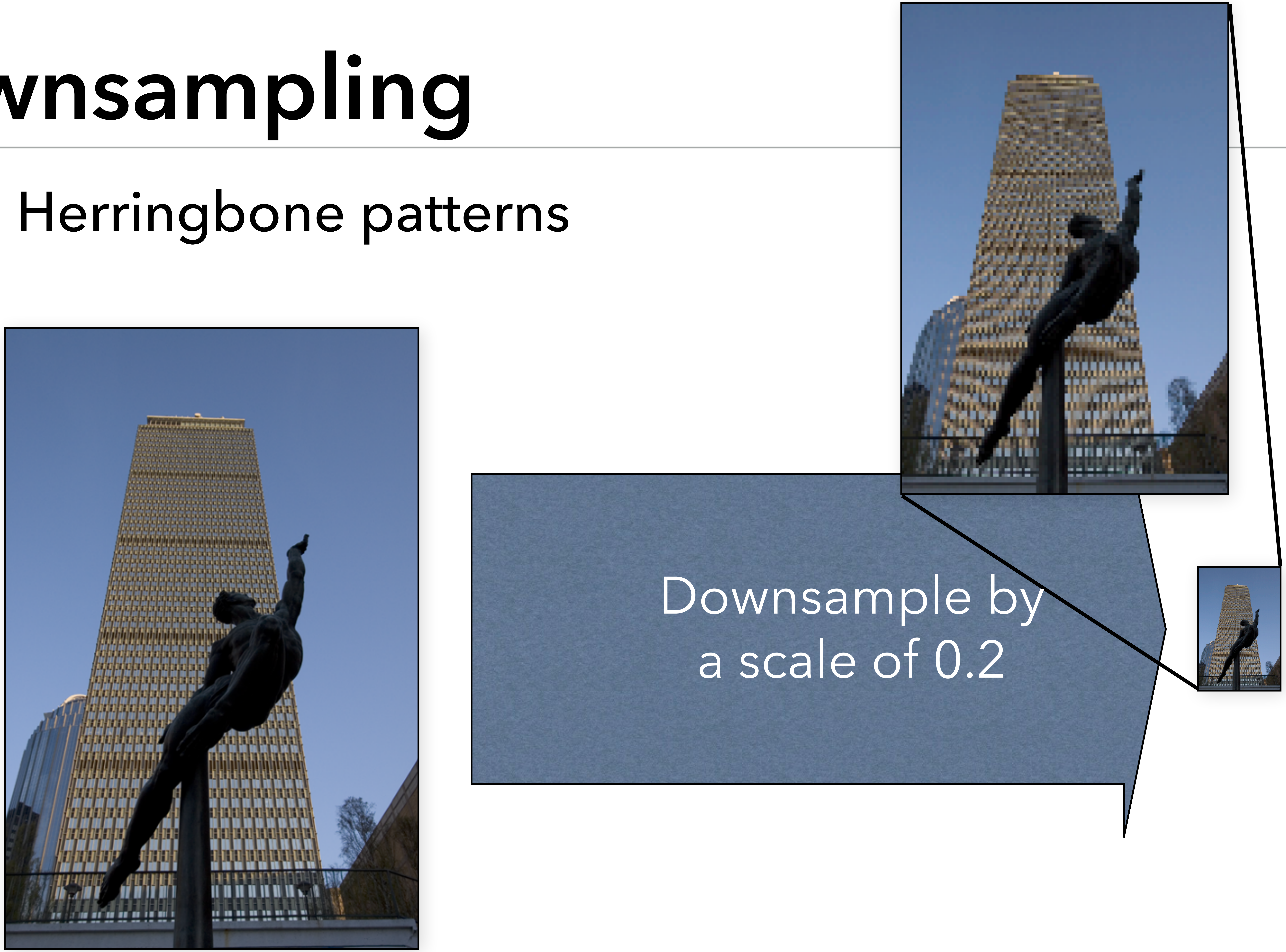
Sharpening



After a slide by Frédo Durand

Downsampling

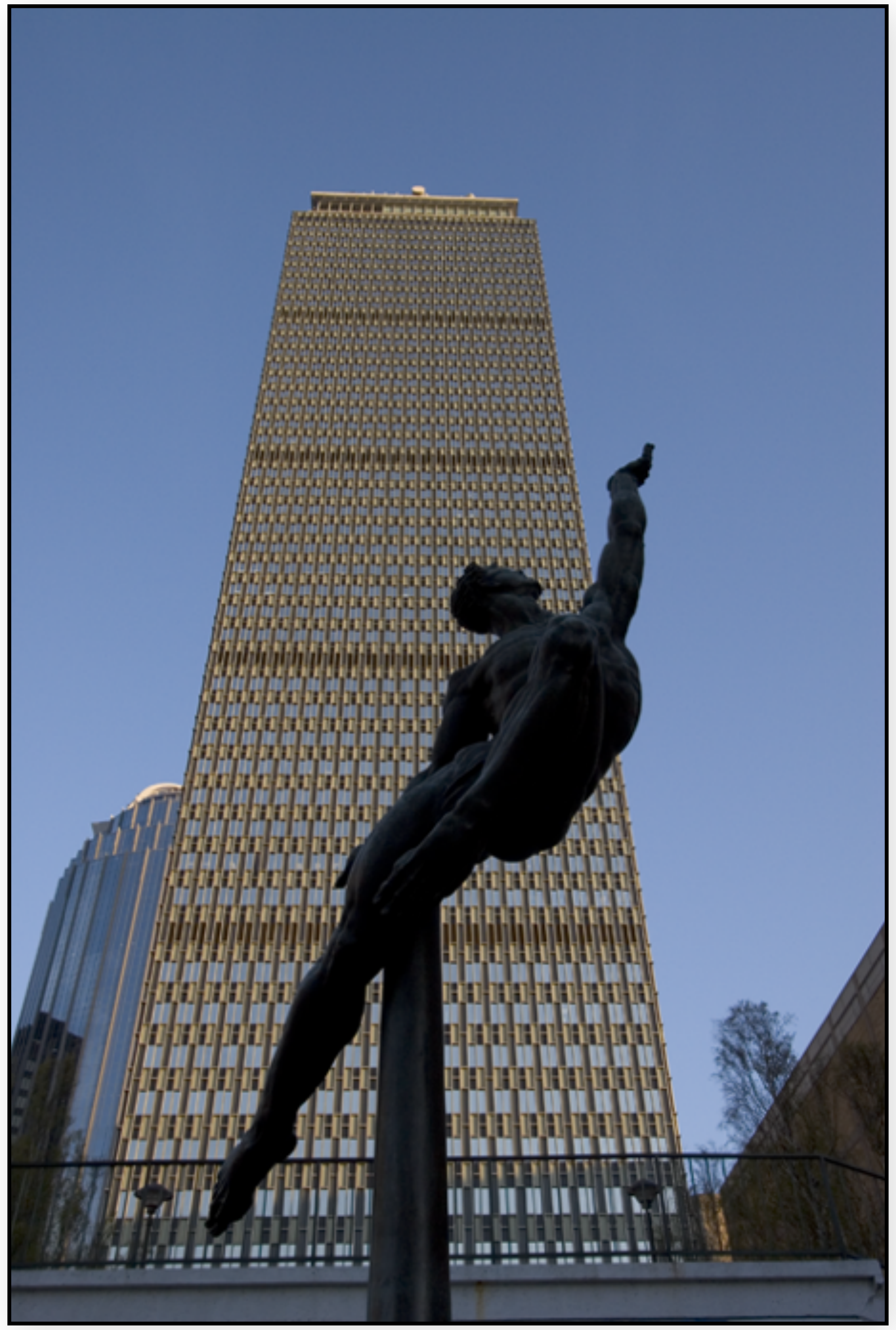
Yikes! Herringbone patterns



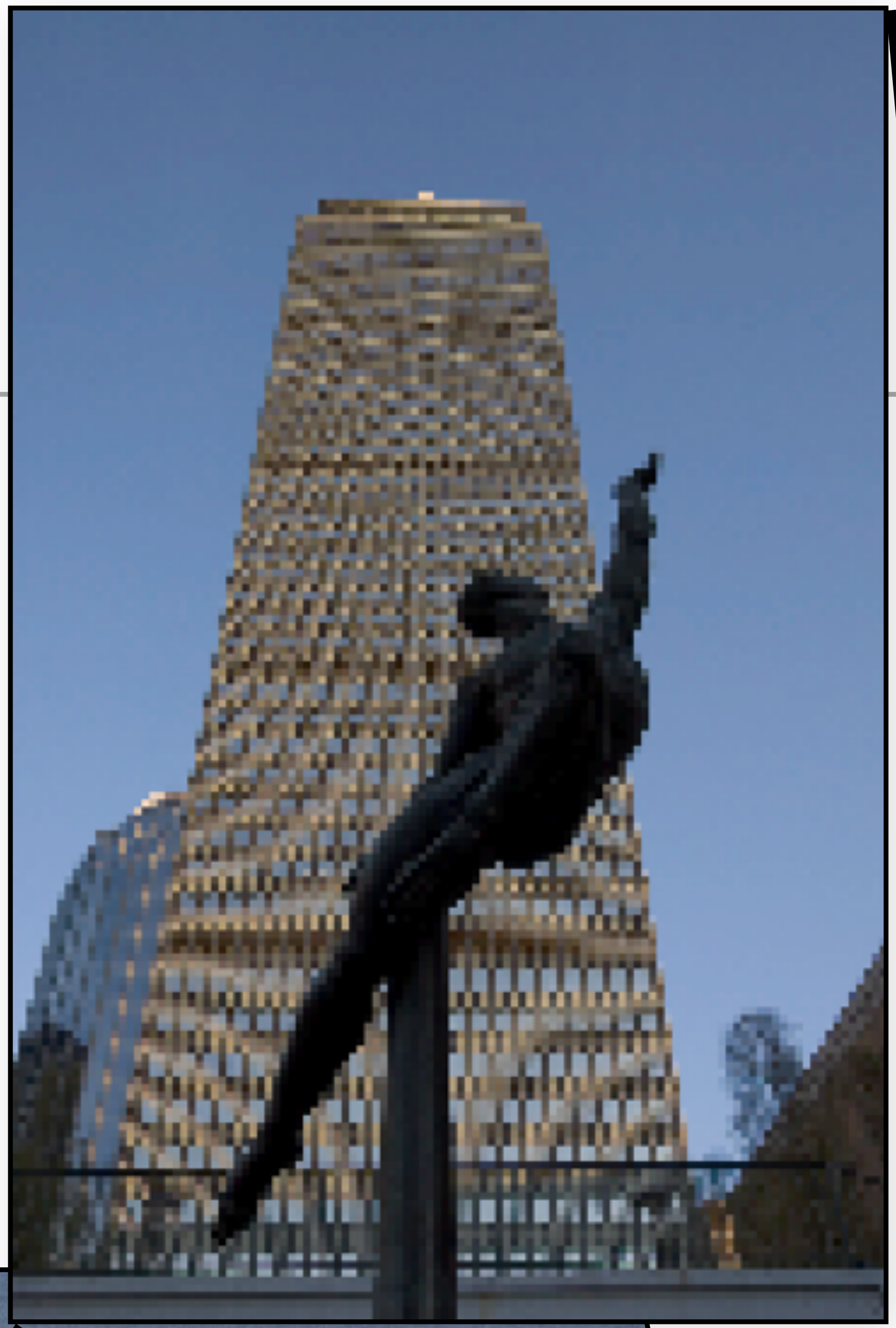
After a slide by Frédo Durand

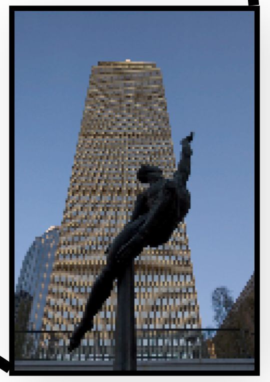
Downsampling

We "randomly" pick a color in the high frequency pattern



Downsample by a scale of 0.2

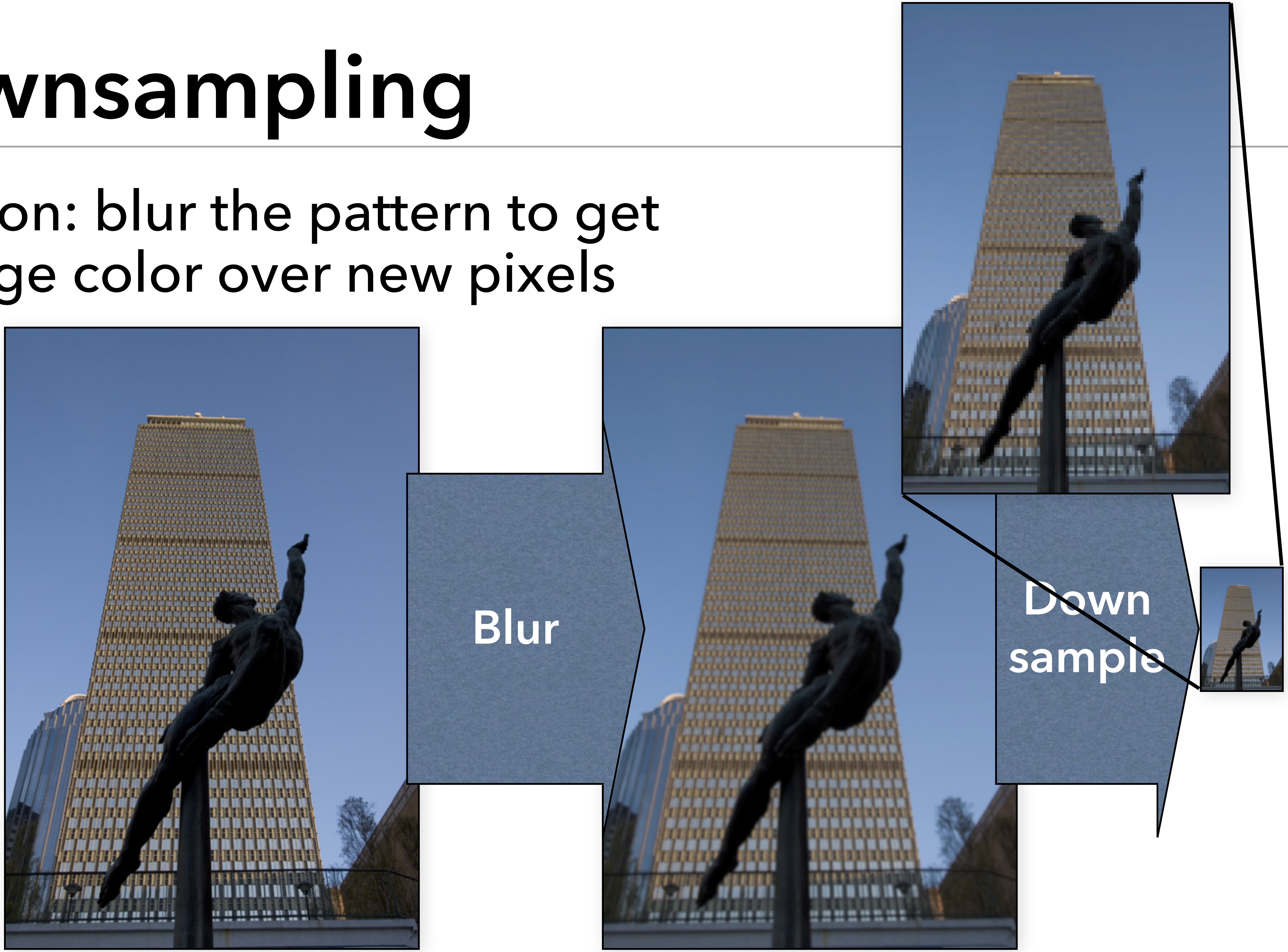




After a slide by Frédo Durand

Downsampling

Solution: blur the pattern to get average color over new pixels



After a slide by Frédo Durand

Fake tilt shift

http://www.tiltshiftphotography.net/photoshop-tutorial.php

After a slide by Frédo Durand

Blur in optics

Diffraction

Lens aberrations

Object movement

Camera shake

Can we remove blur computationally?

- invert the blur equation
- deconvolution

After a slide by Frédo Durand

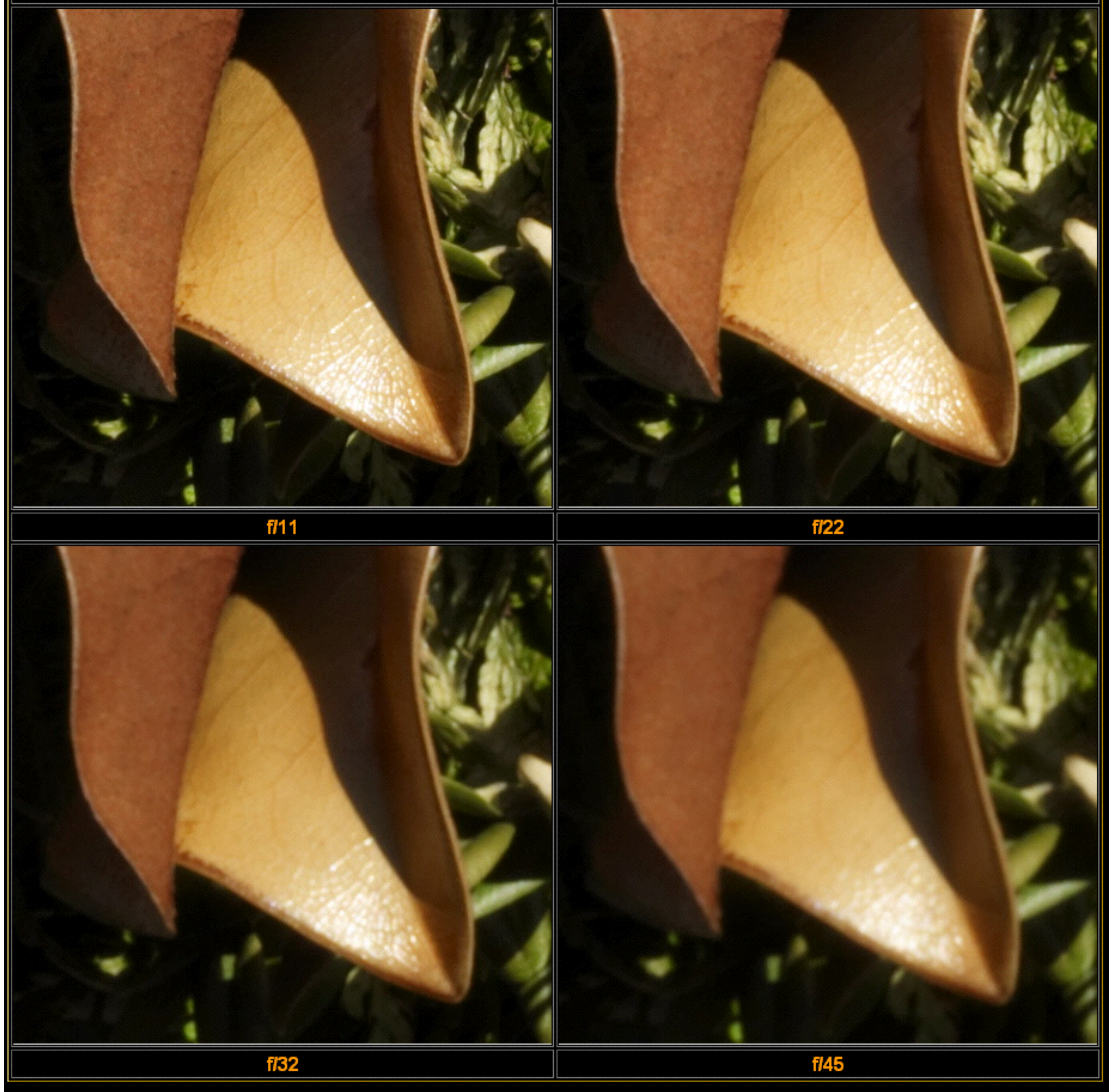
Lens diffraction

http://luminous-landscape.com/tutorials/understanding-series/u-diffraction.shtml

(heavily cropped)

See also

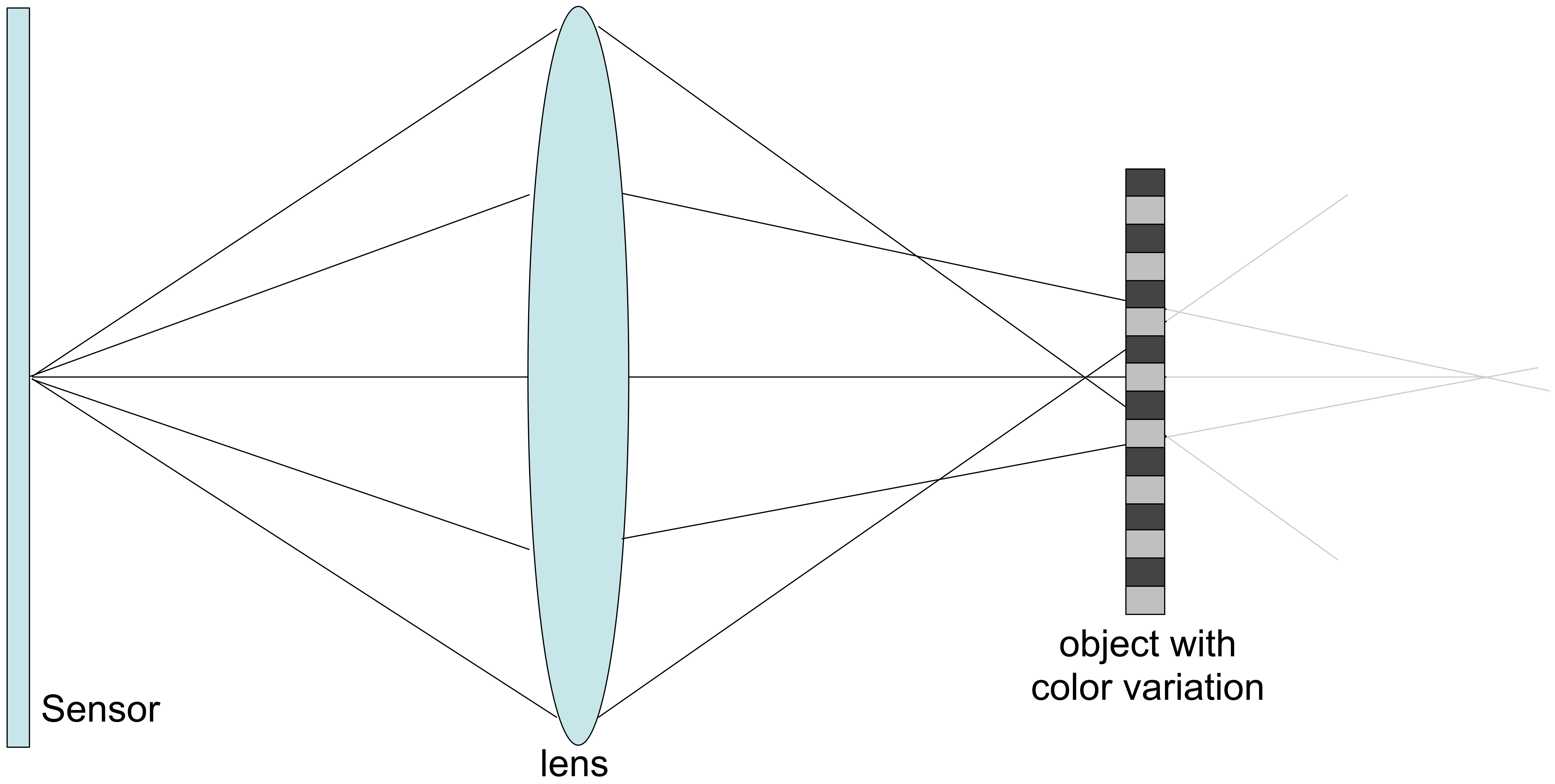
http://www.cambridgeincolour.com/tutorials/diffraction-photography.htm



After a slide by Frédo Durand

Blur example: spherical aberration

Pixel value: weighted average of local color



After a slide by Frédo Durand

Remove optical artifacts

Calibrate lenses and remove blur

e.g. DXO

After a slide by Frédo Durand

Removing camera shake

Original

Naïve Sharpening

Fergus et al's algorithm

After a slide by Frédo Durand

Convolution 101

Blur as convolution

Replace each pixel by a linear combination of its neighbors.

- only depends on relative position of neighbors

The prescription for the linear combination is called the "convolution kernel".

local image data

10	5	3
4	5	1
1	1	7

kernel

0	0	0
0	0.5	0
0	1	0.5

modified image

	7	

After a slide by Frédo Durand

Linear shift-invariant filtering

Replace each pixel by a linear combination of its neighbors.

- only depends on relative position of neighbors

The prescription for the linear combination is called the "convolution kernel".

- **same kernel for all pixels**

local image data

10	5	3
4	5	1
1	1	7

kernel

0	0	0
0	0.5	0
0	1	0.5

modified image

	7	

After a slide by Frédo Durand

Example of linear NON-shift invariant transformation?

e.g. neutral-density graduated filter (darken high y):

- $J(x,y) = I(x,y)*(1-y/ymax)$

Formally, what does linear mean?

- For two scalars a & b and two inputs x & y: $F(ax+by) = aF(x)+bF(y)$

What does shift invariant mean?

- For a translation T: $F(T(x)) = T(F(x))$
- If I blur a translated image, I get a translated blurred image

After a slide by Frédo Durand

Questions?

Convolution algorithm

```
set output image to zero
for all pixels (x,y) in output image
   for all (x',y') in kernel
      out(x,y) += input(x+x',y+y')*kernel(x',y')
```

(this assumes the kernel coordinates are centered)

After a slide by Frédo Durand

local image data

10	5	3
4	5	1
1	1	7

kernel

0	0	0
0	0.5	0
0	1	0.5

modified image

	7	

Questions?

```
s
for all pixels (x,y) in
    for all (x',y')
        out(x,y) += input(x+x',y+y')*kernel(x',y')
```

(this assumes the kernel coordinates are centered)

After a slide by Frédo Durand

local image data

10	5	3
4	5	1
1	1	7

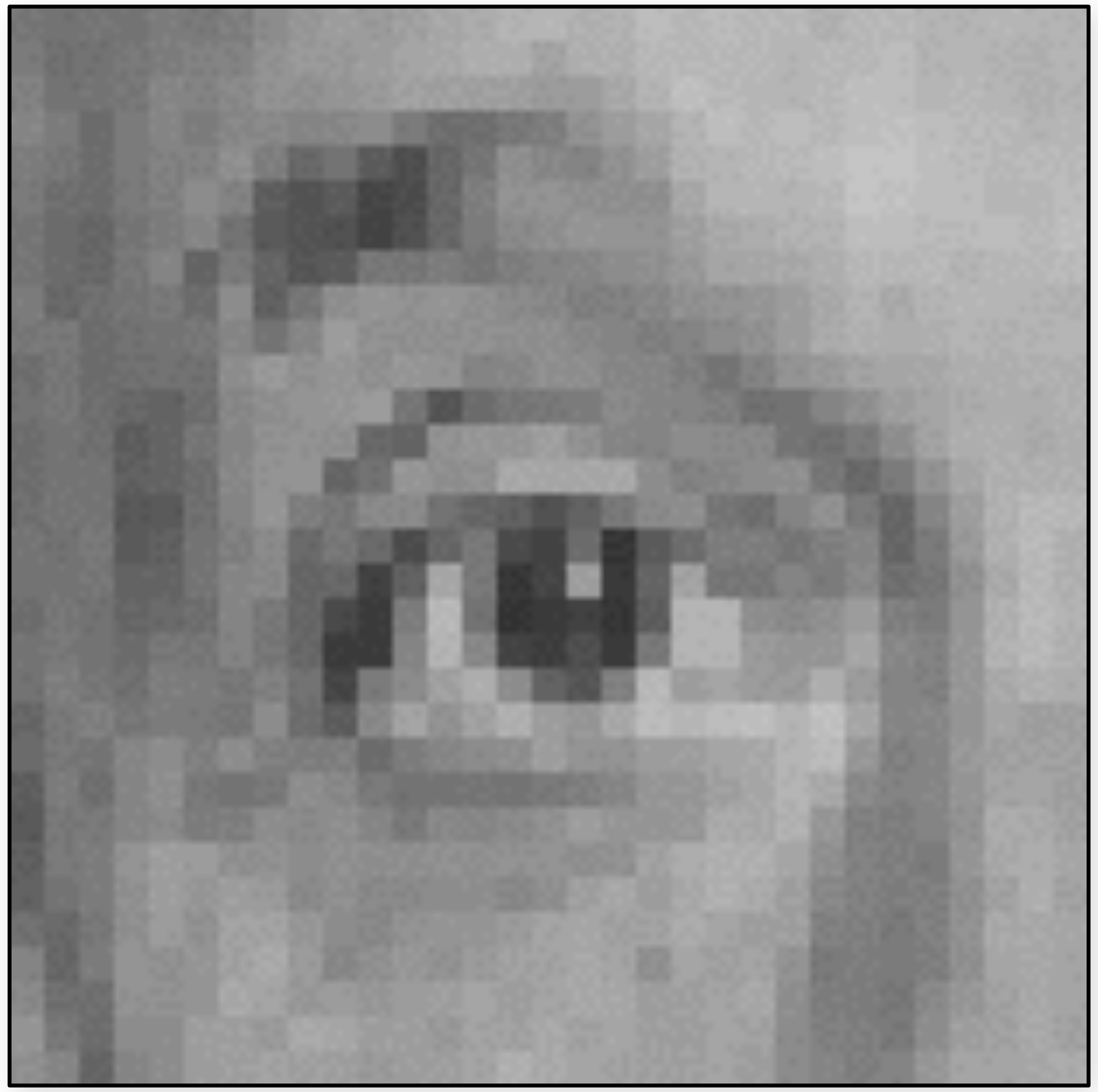
kernel

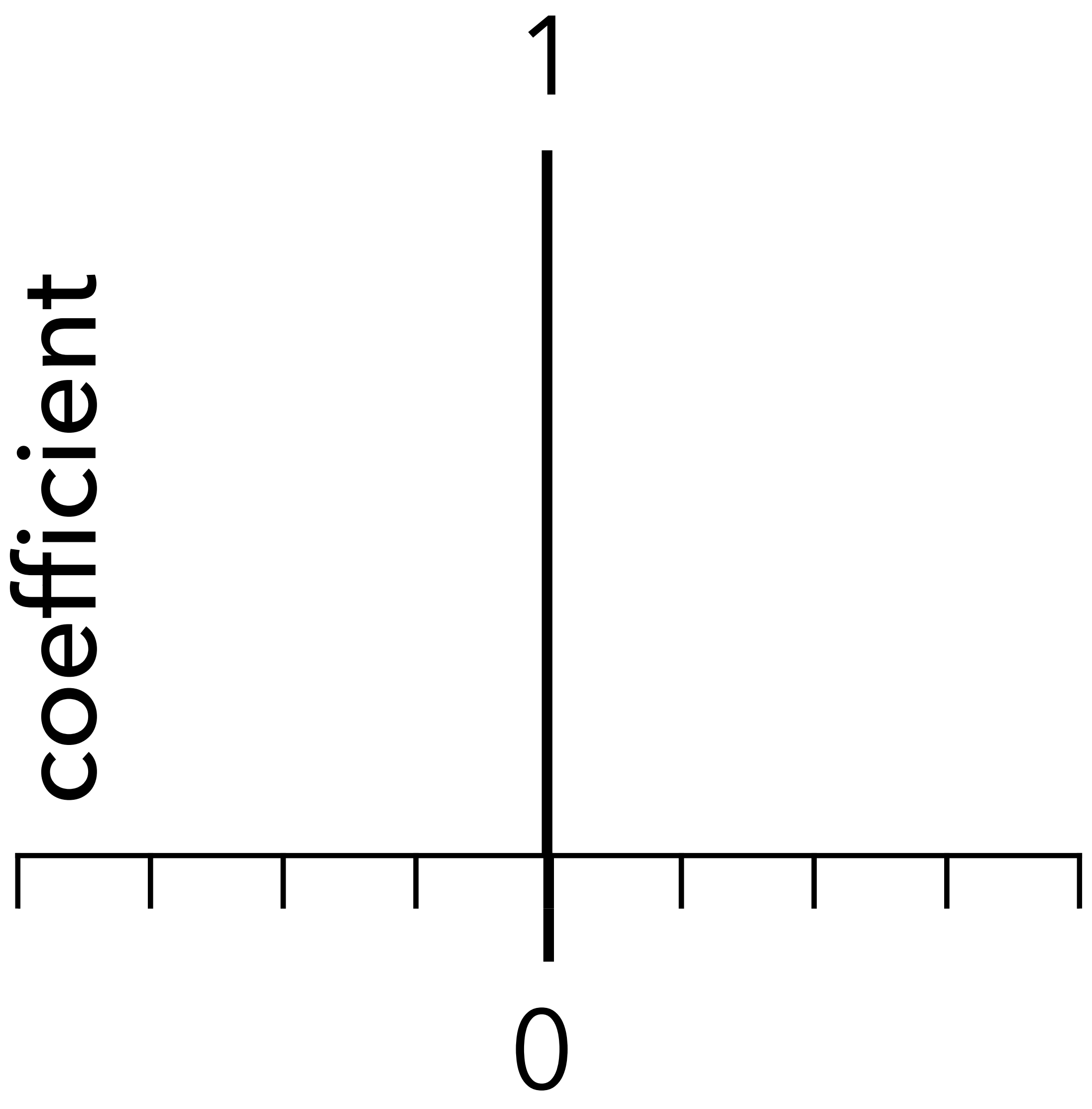
0	0	0
0	0.5	0
0	1	0.5

modified image

	7	

Convolution (warm-up slide)

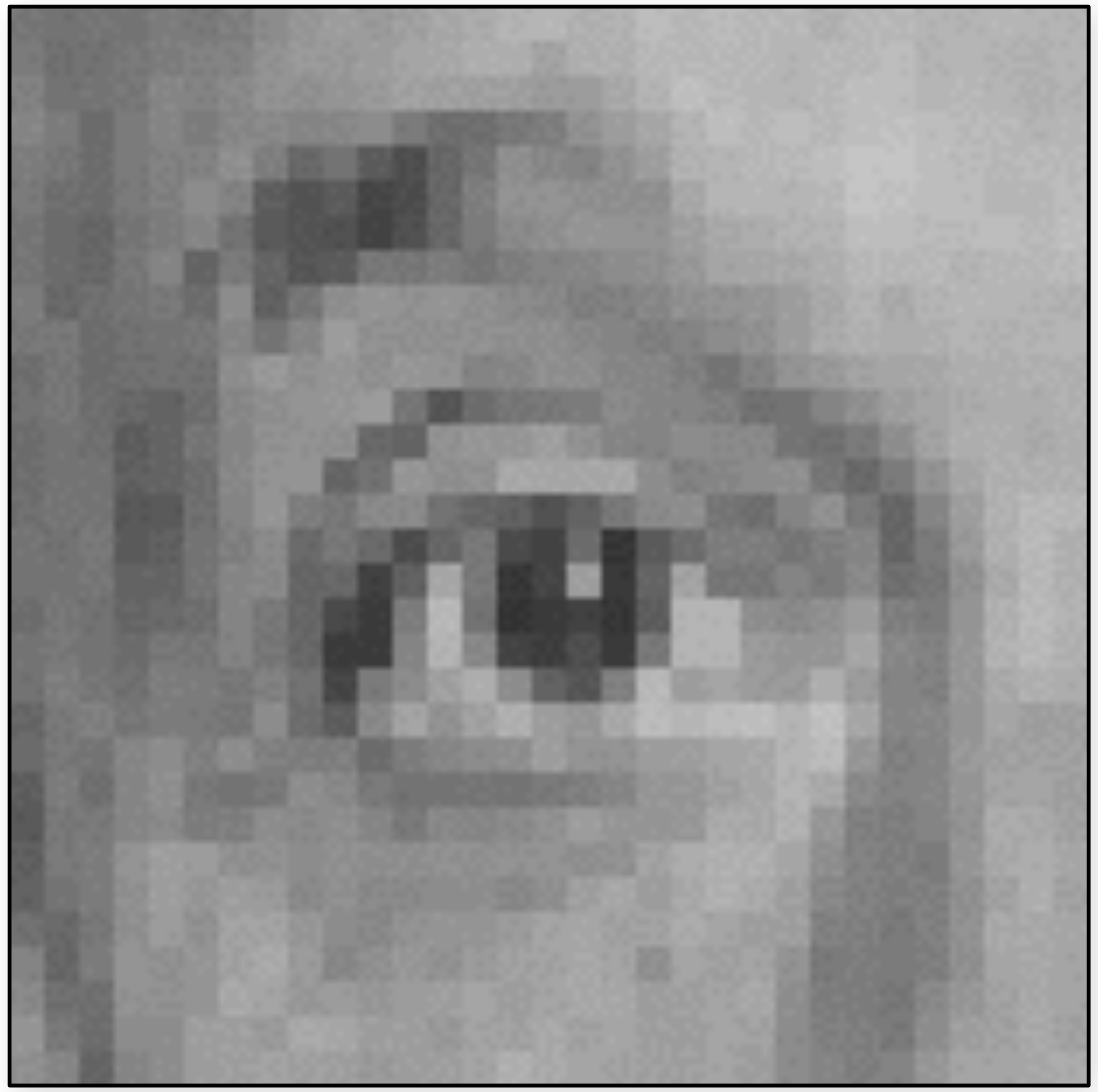

original



$\otimes$ … $=$?

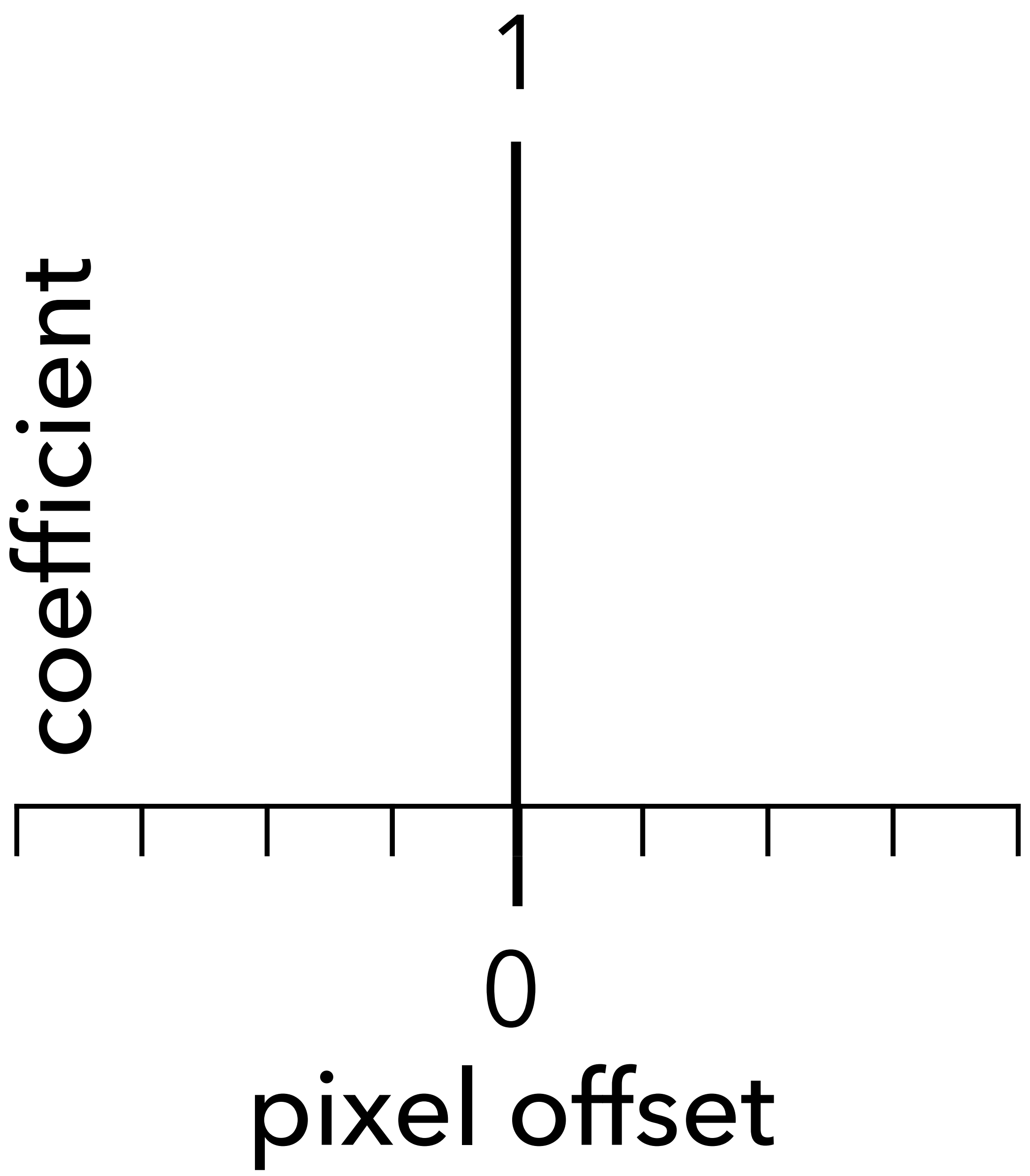
After a slide by Frédo Durand

Convolution (warm-up slide)

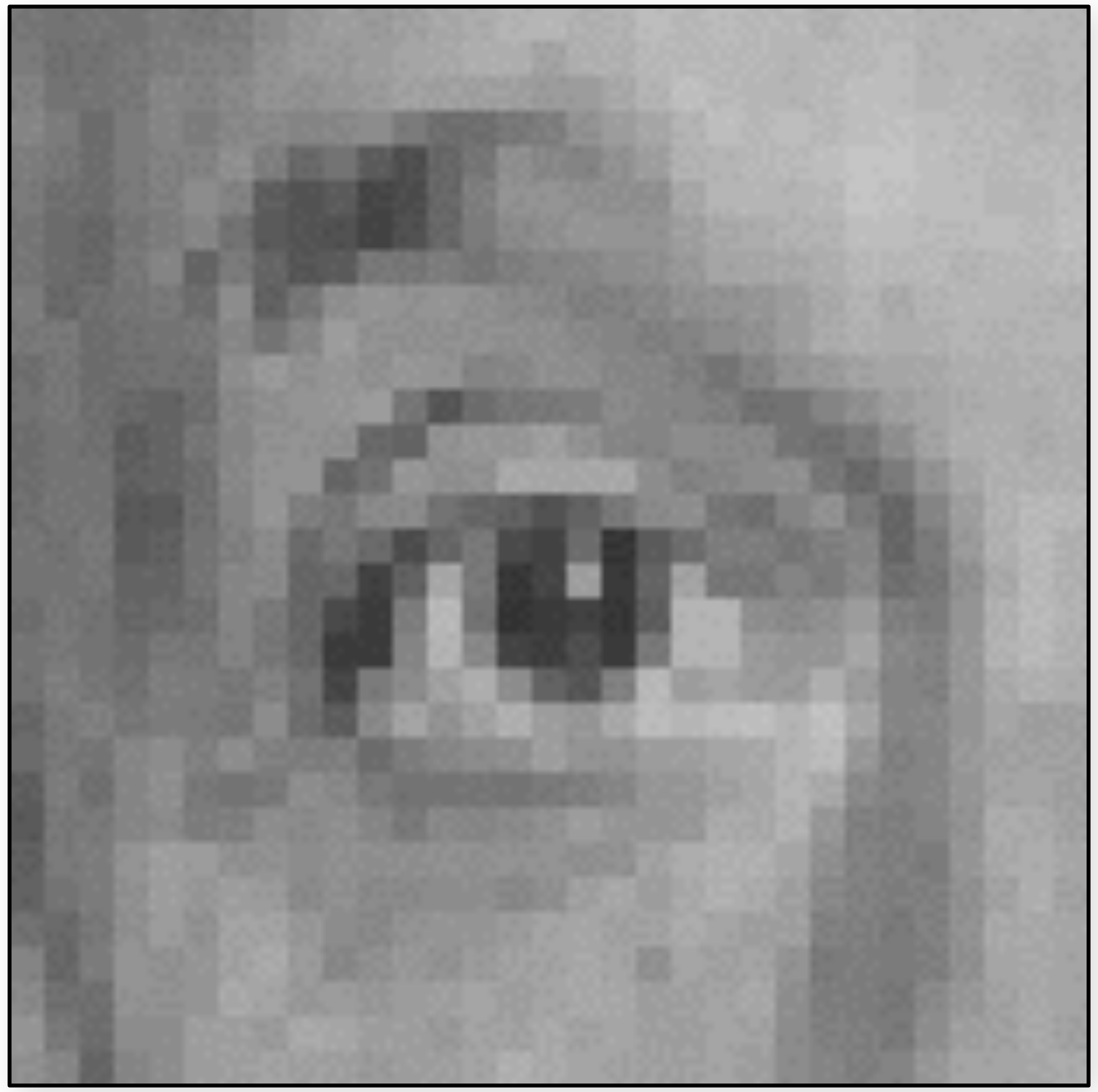


original

$\otimes$



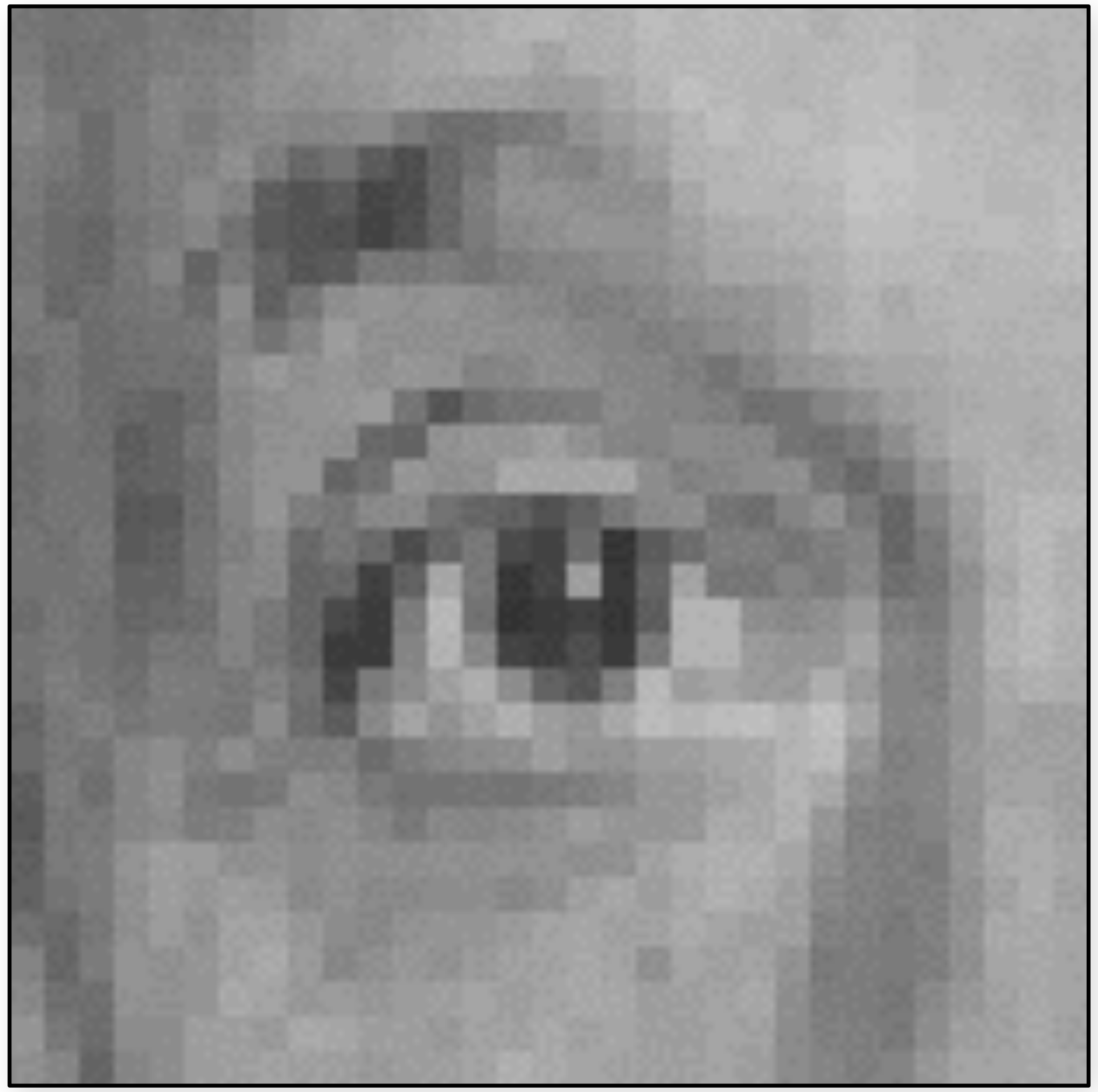
=



filtered
(no change)

After a slide by Frédo Durand

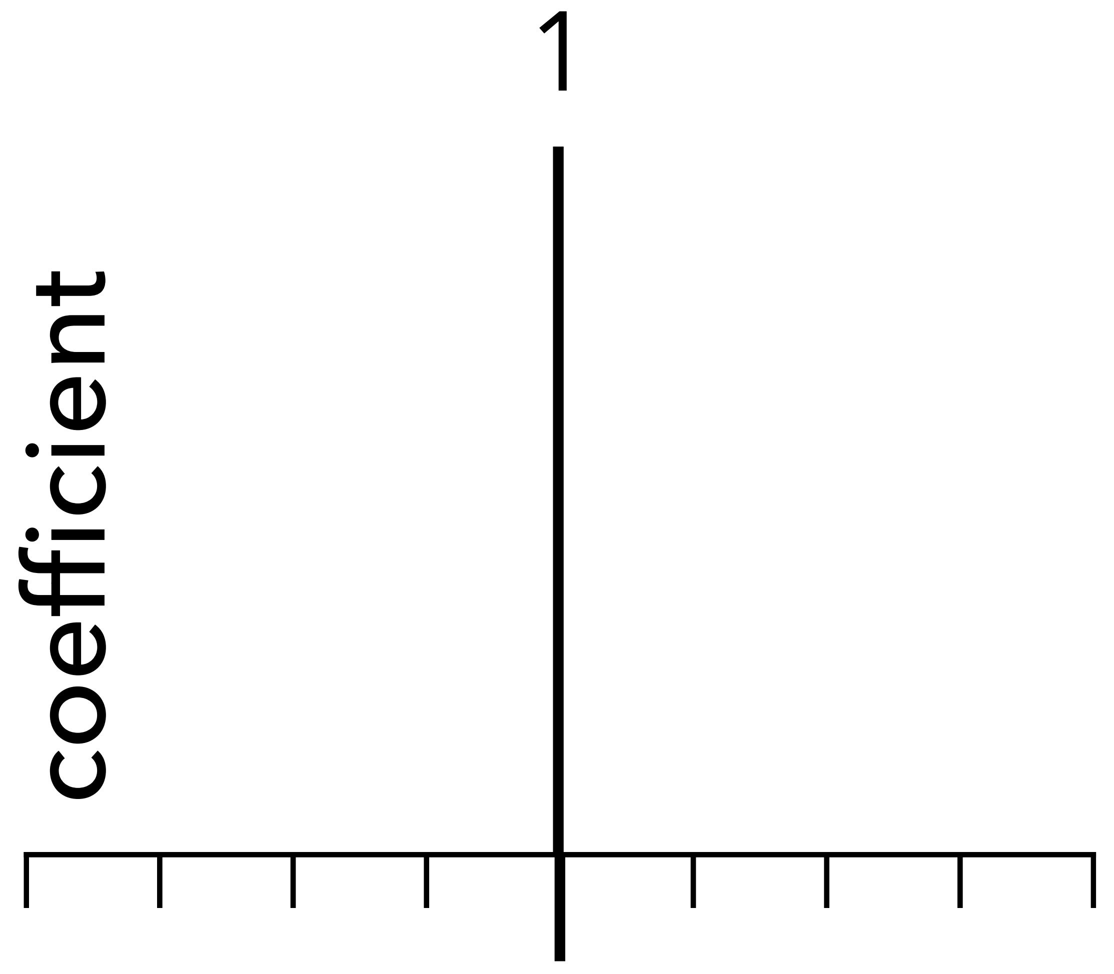
Convolution (warm-up slide)



original

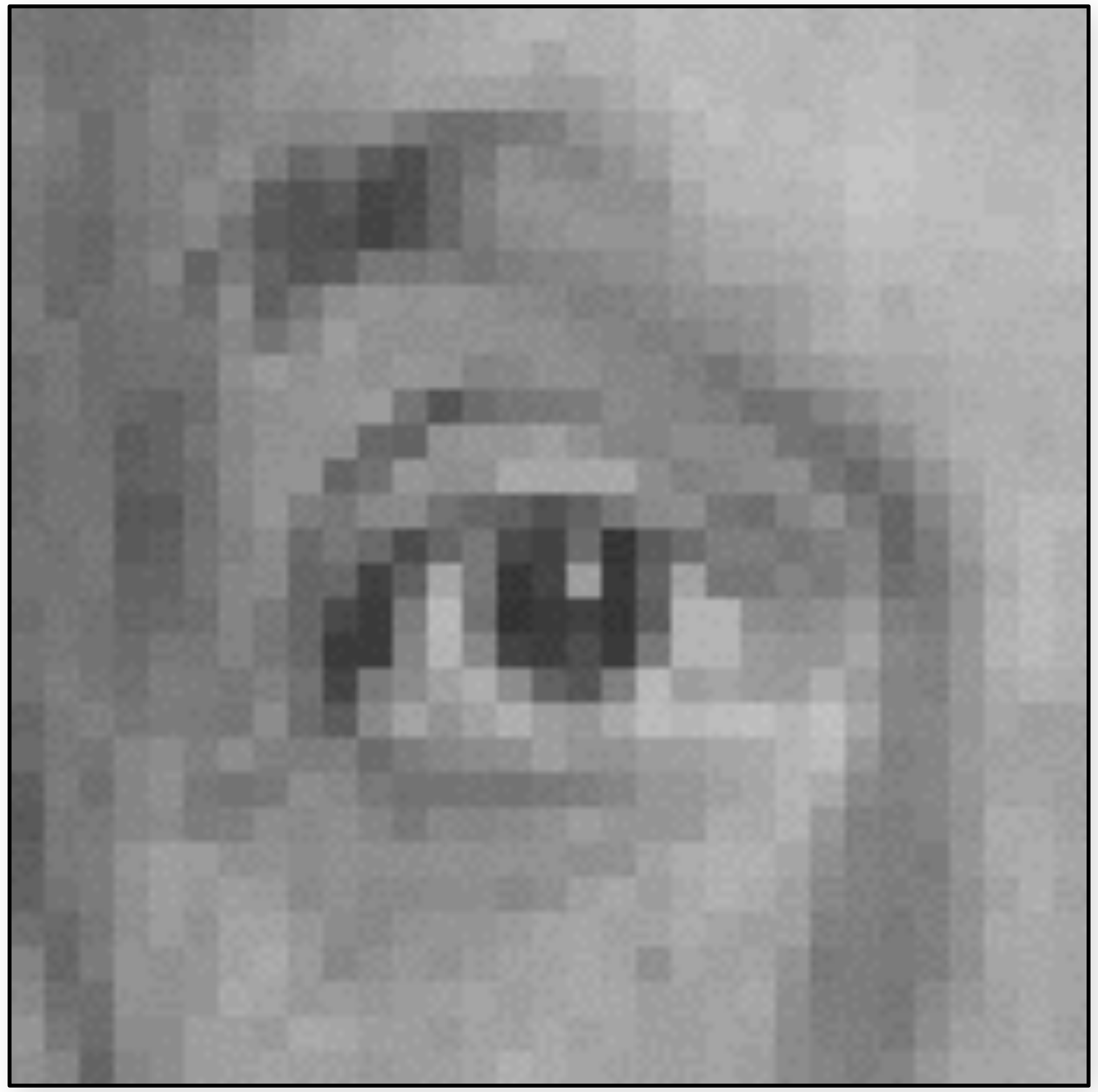
f

$\otimes$



δ

=

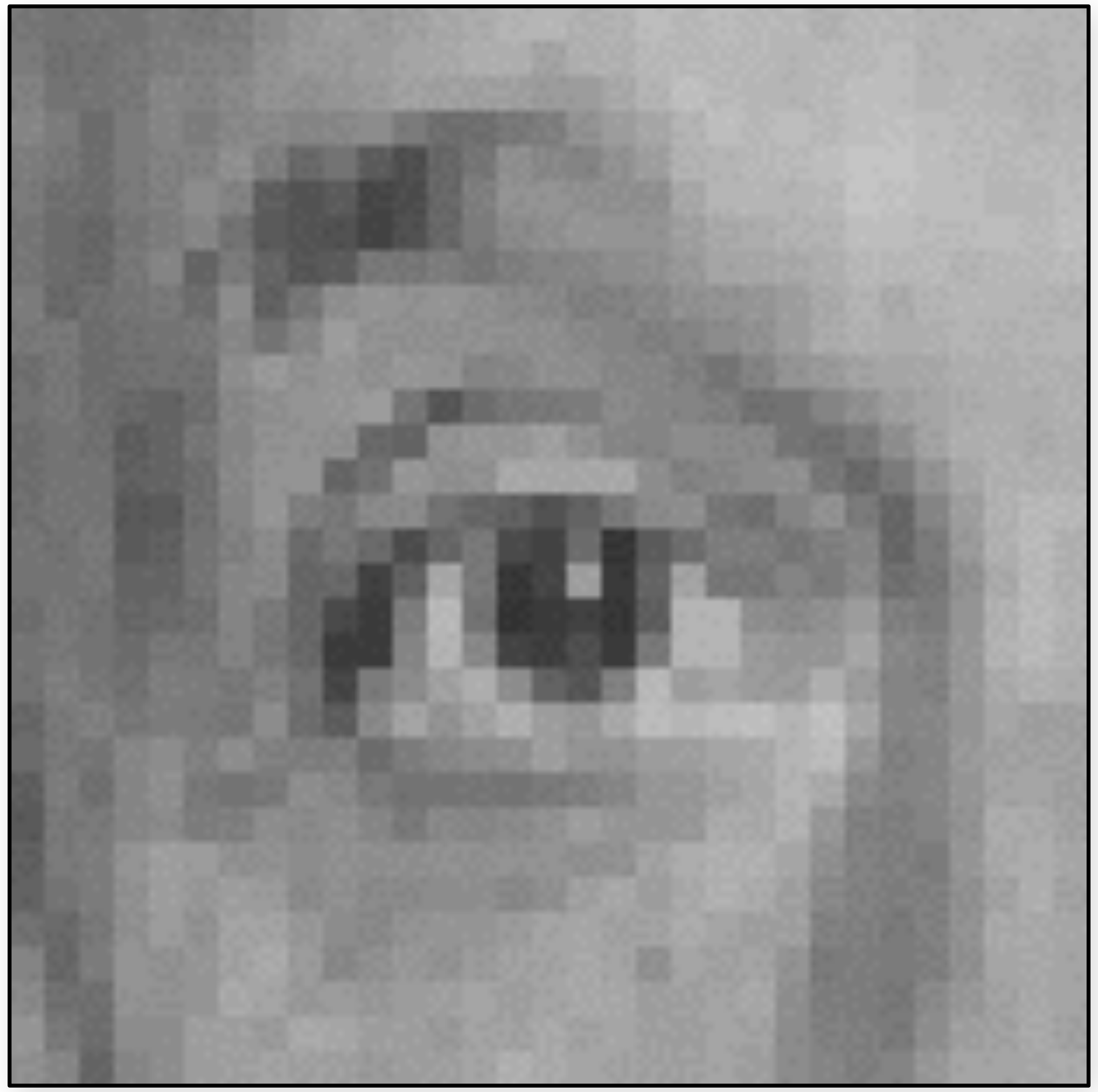


filtered
(no change)

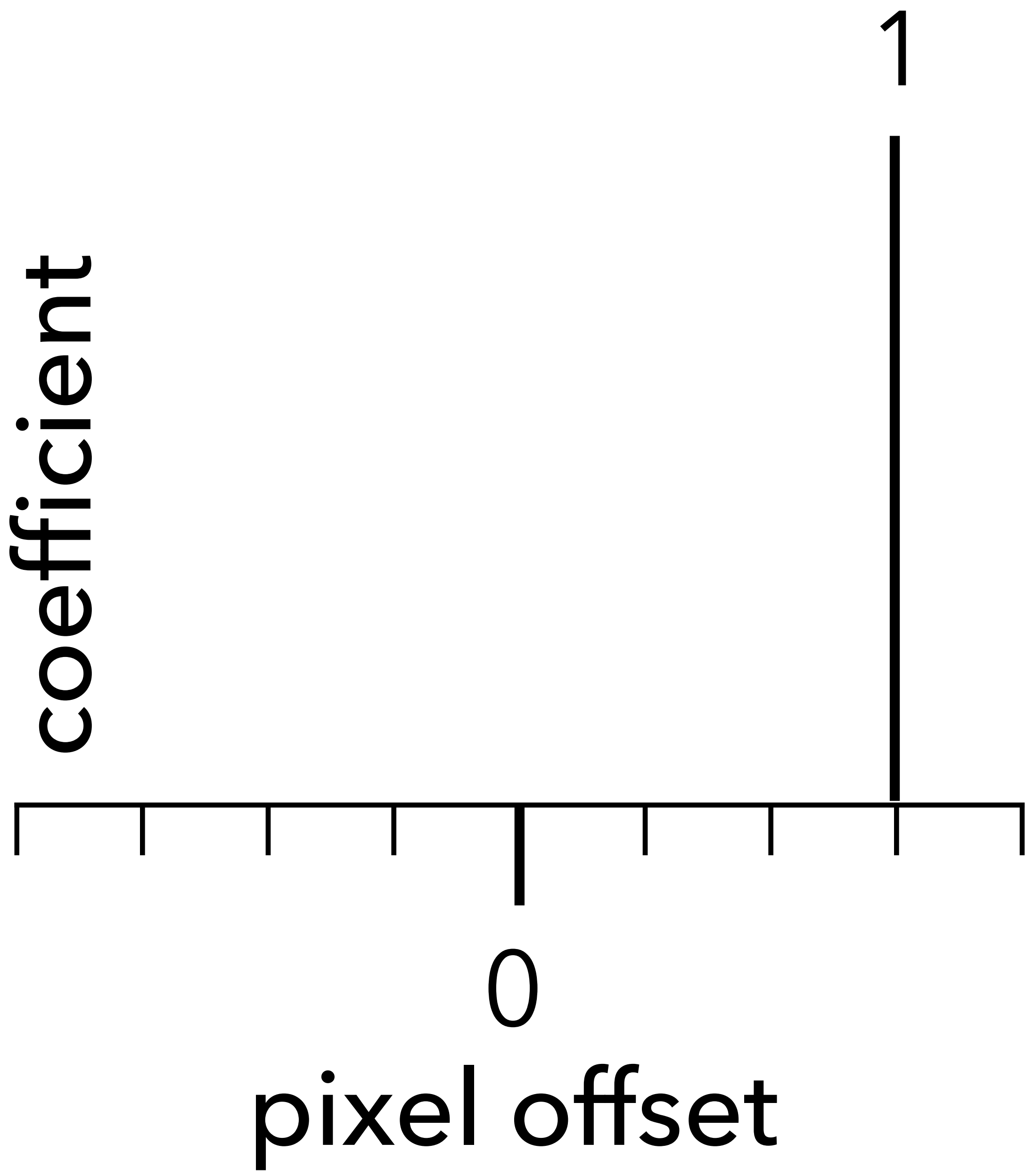
$f = f \otimes \delta$

After a slide by Frédo Durand

Convolution



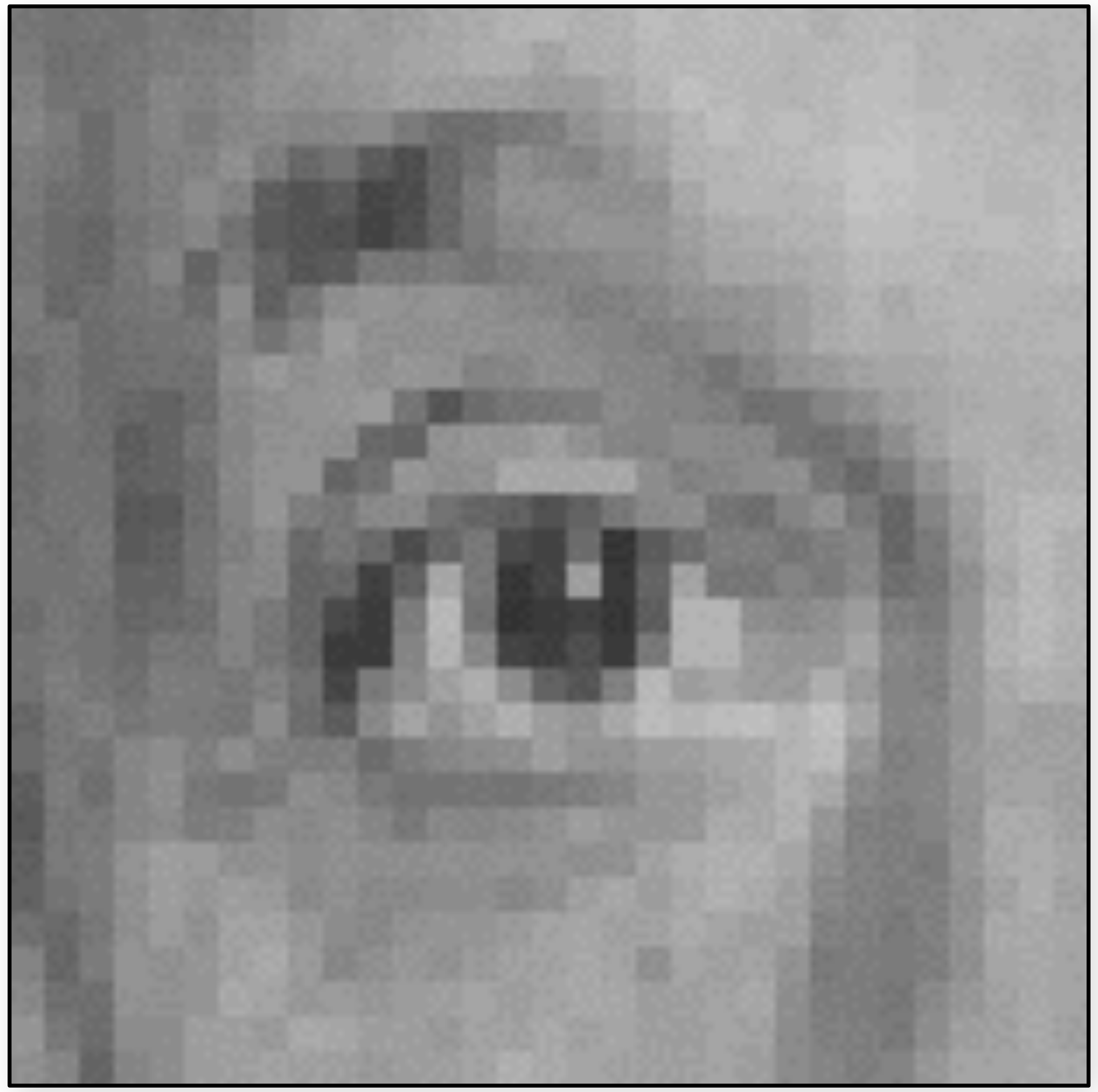
original



⊗ … = ?

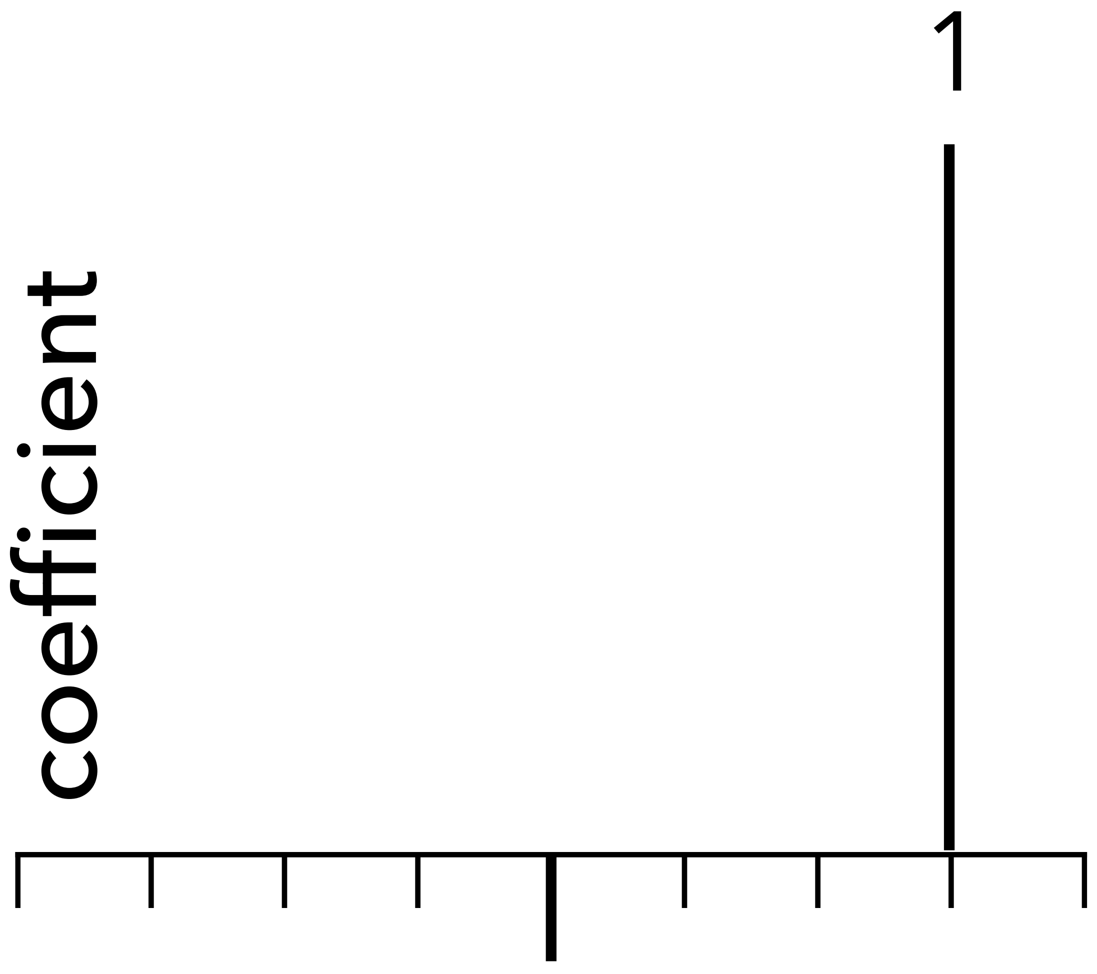
After a slide by Frédo Durand

Convolution



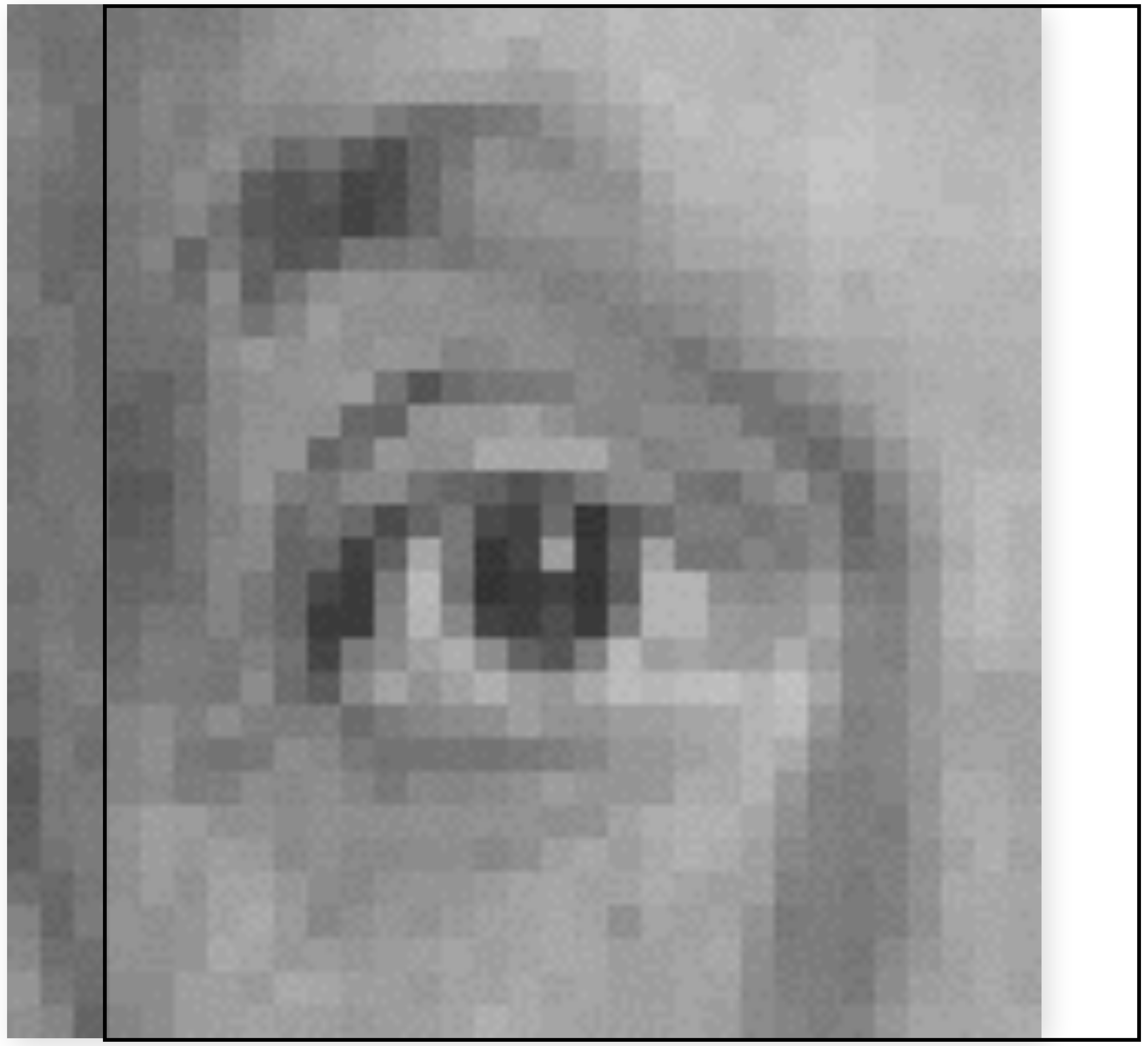
original

⊗



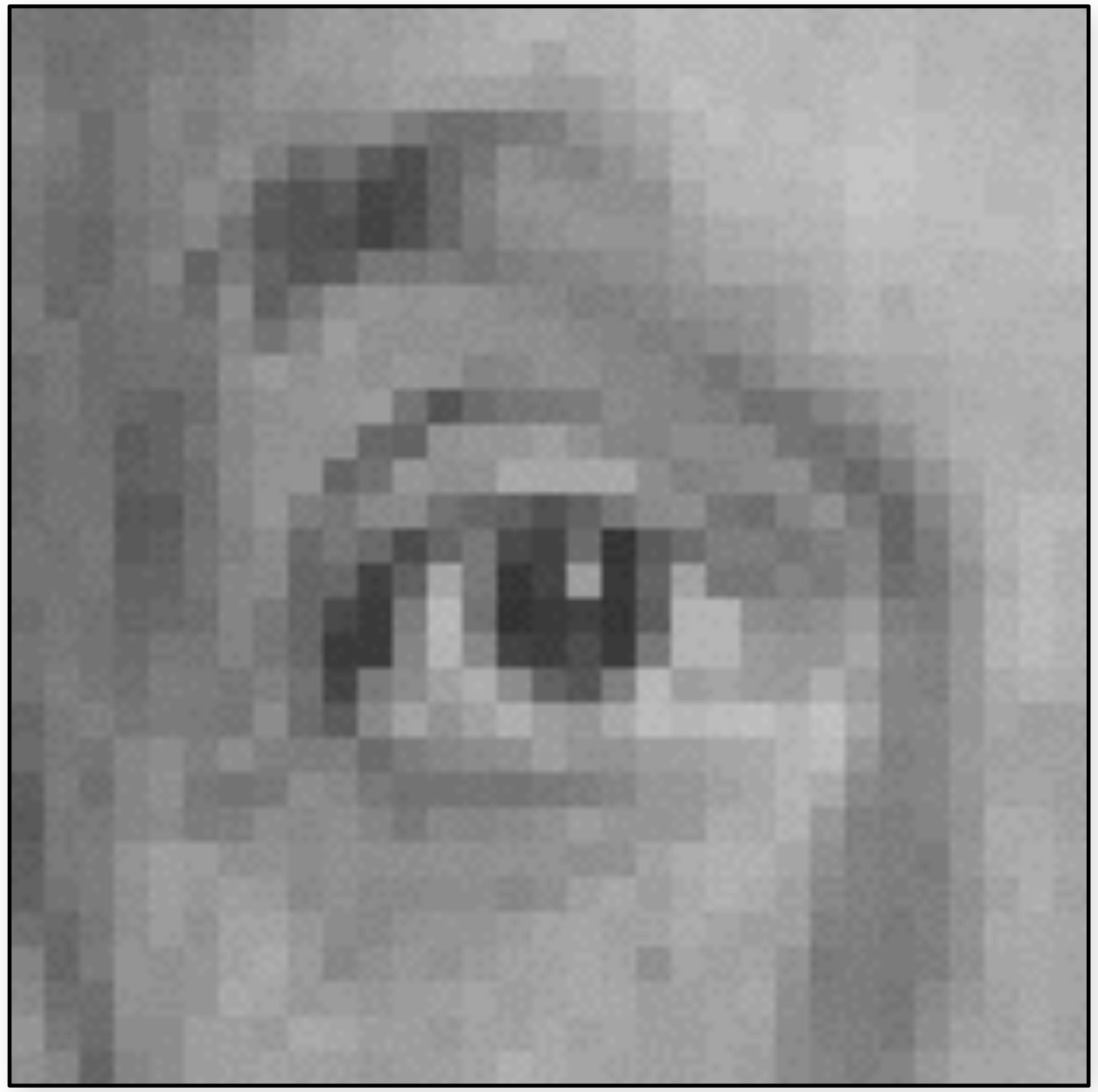
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	1
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

=

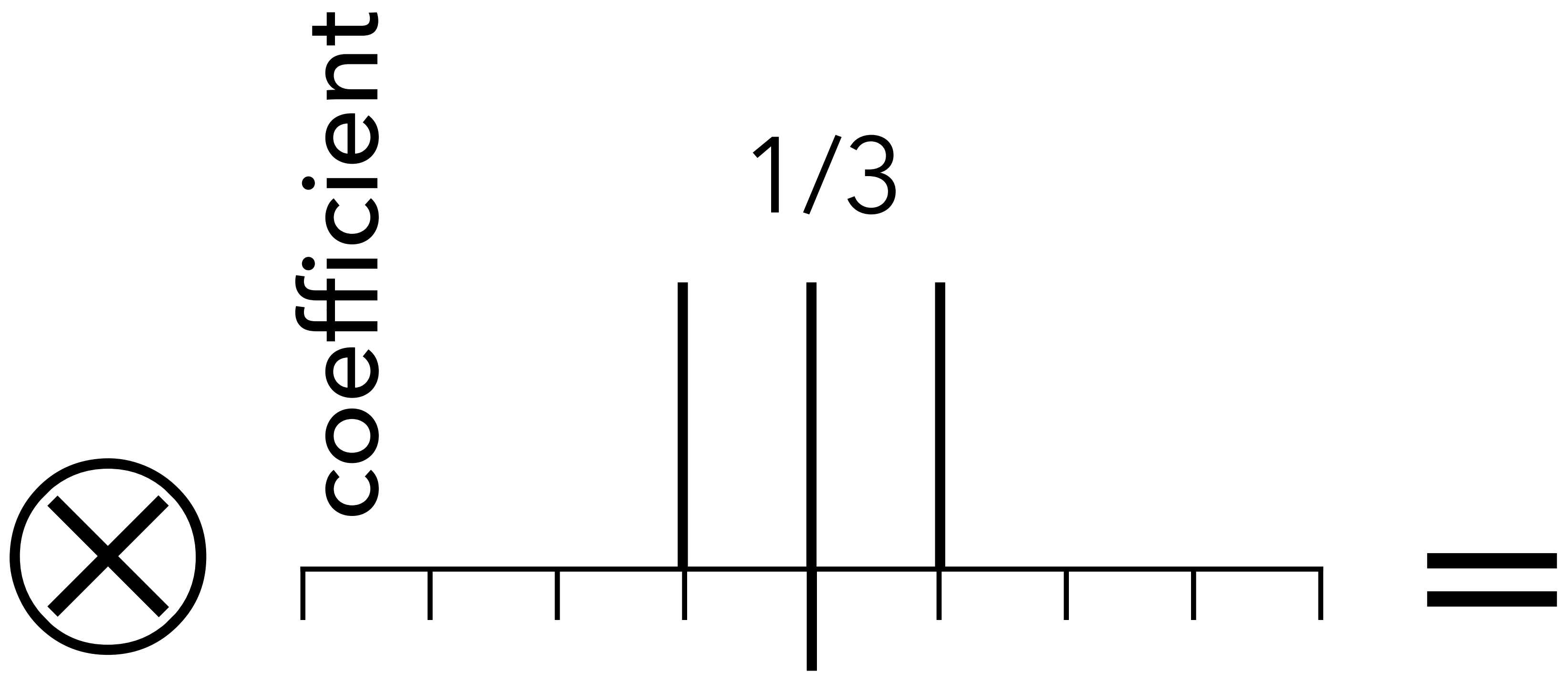


After a slide by Frédo Durand

Convolution

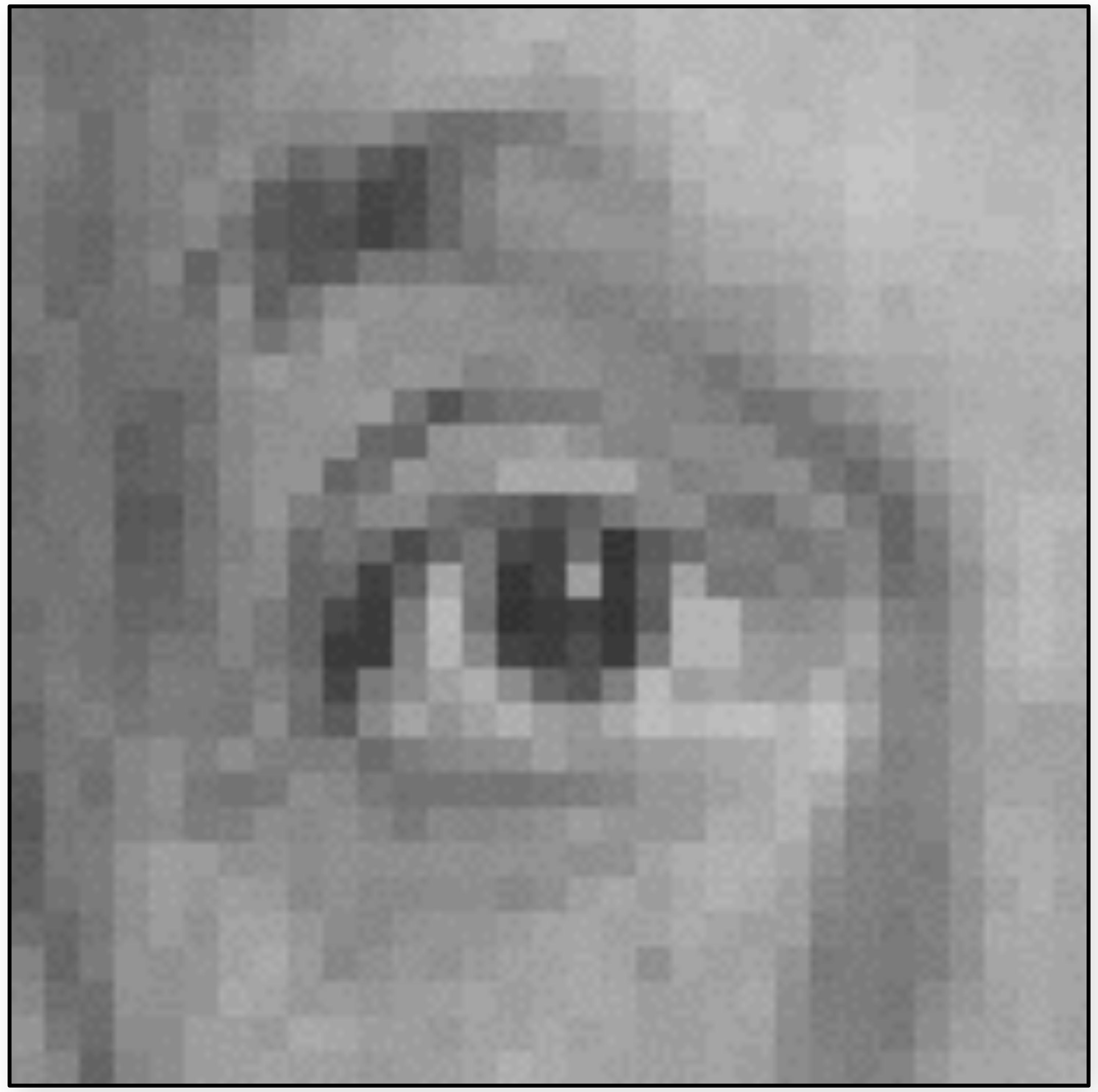


original

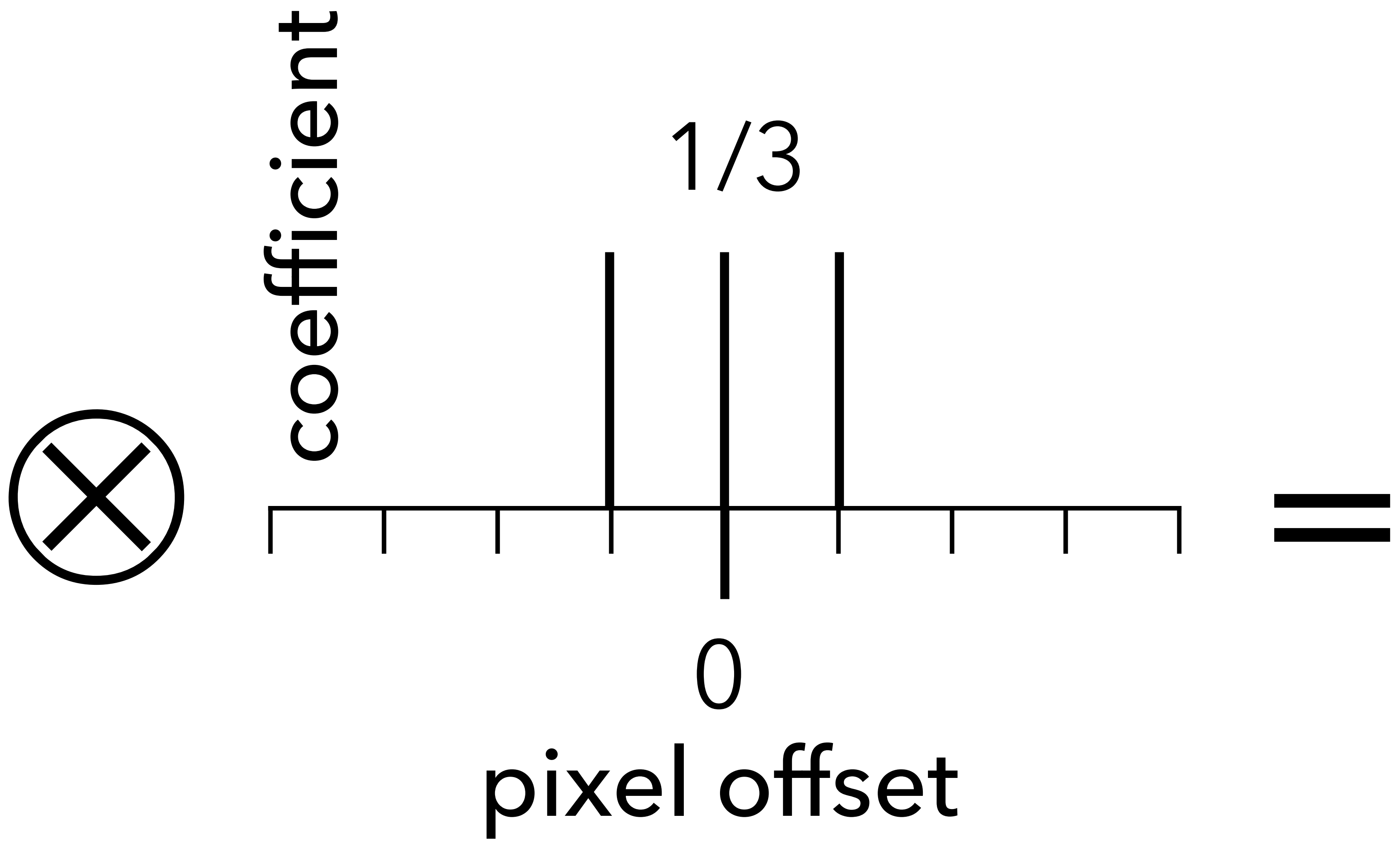


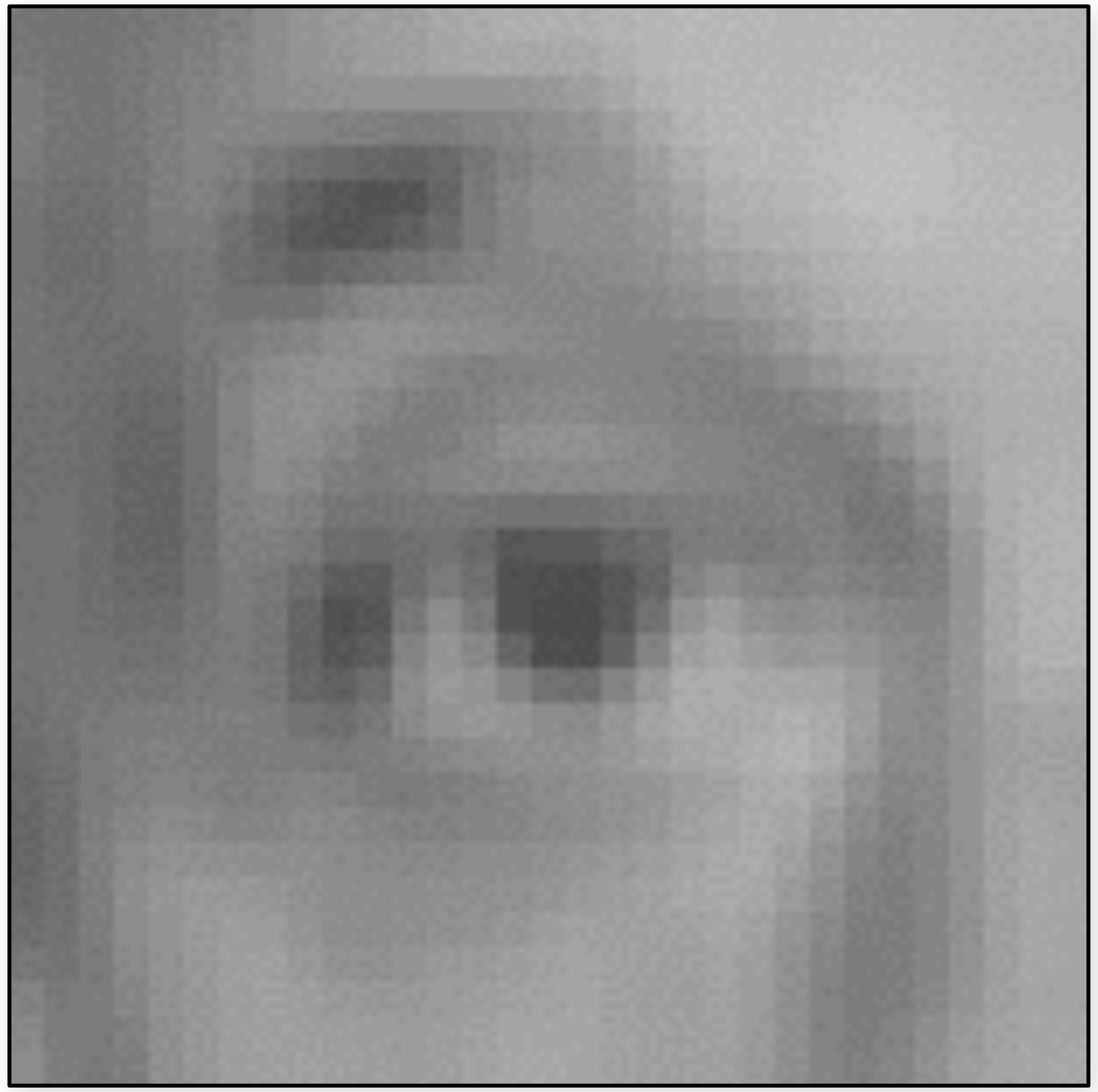
?

Blurring



original

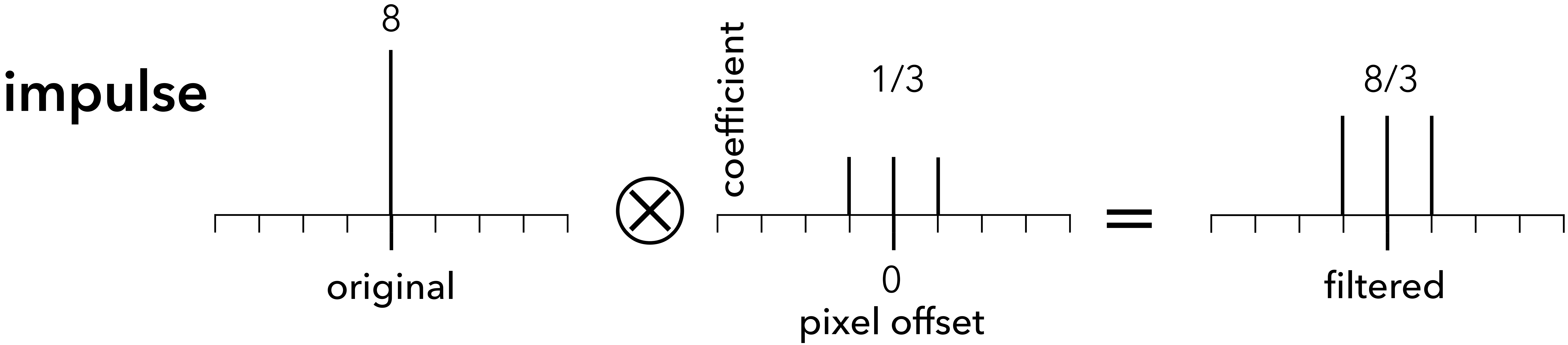




blurred
(applied in both dimensions)

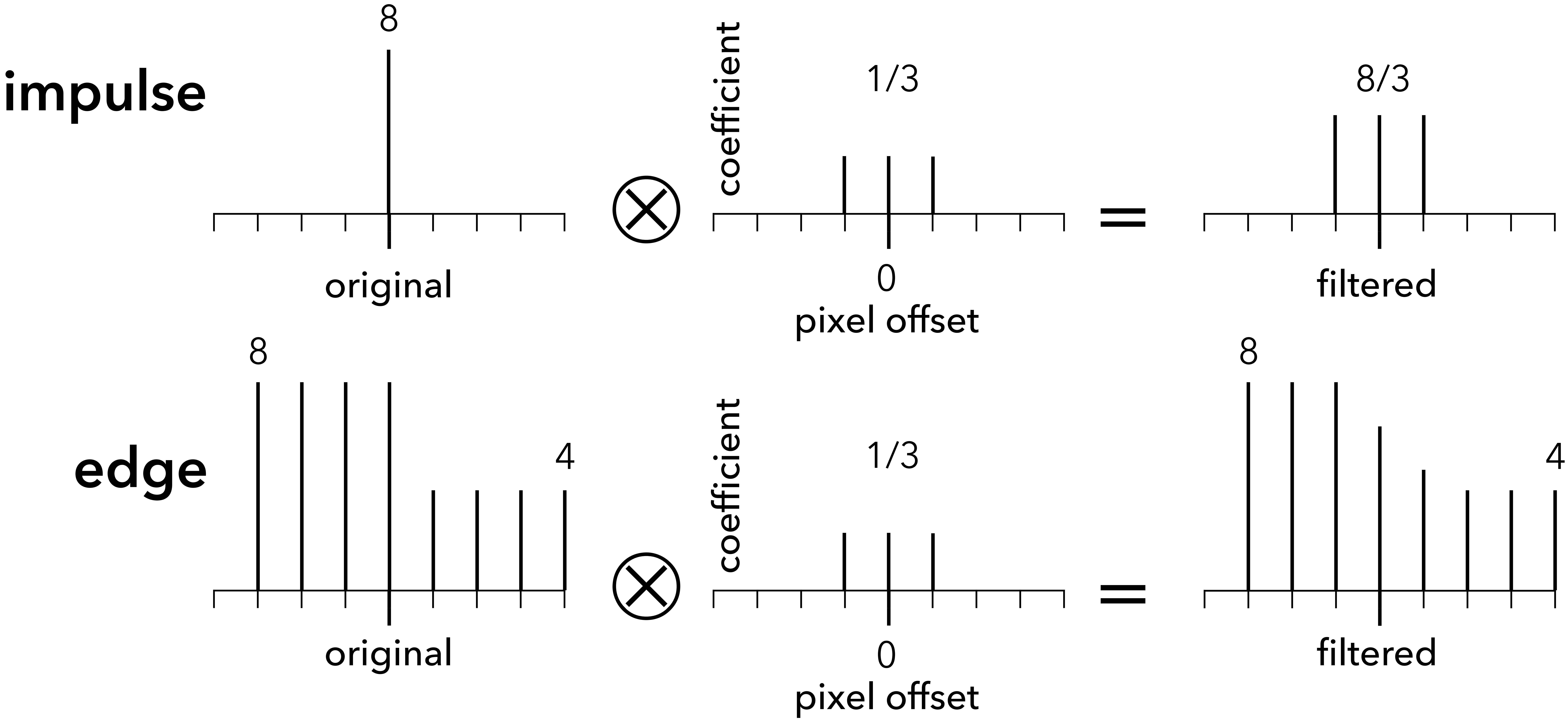
After a slide by Frédo Durand

Blur examples



After a slide by Frédo Durand

Blur examples



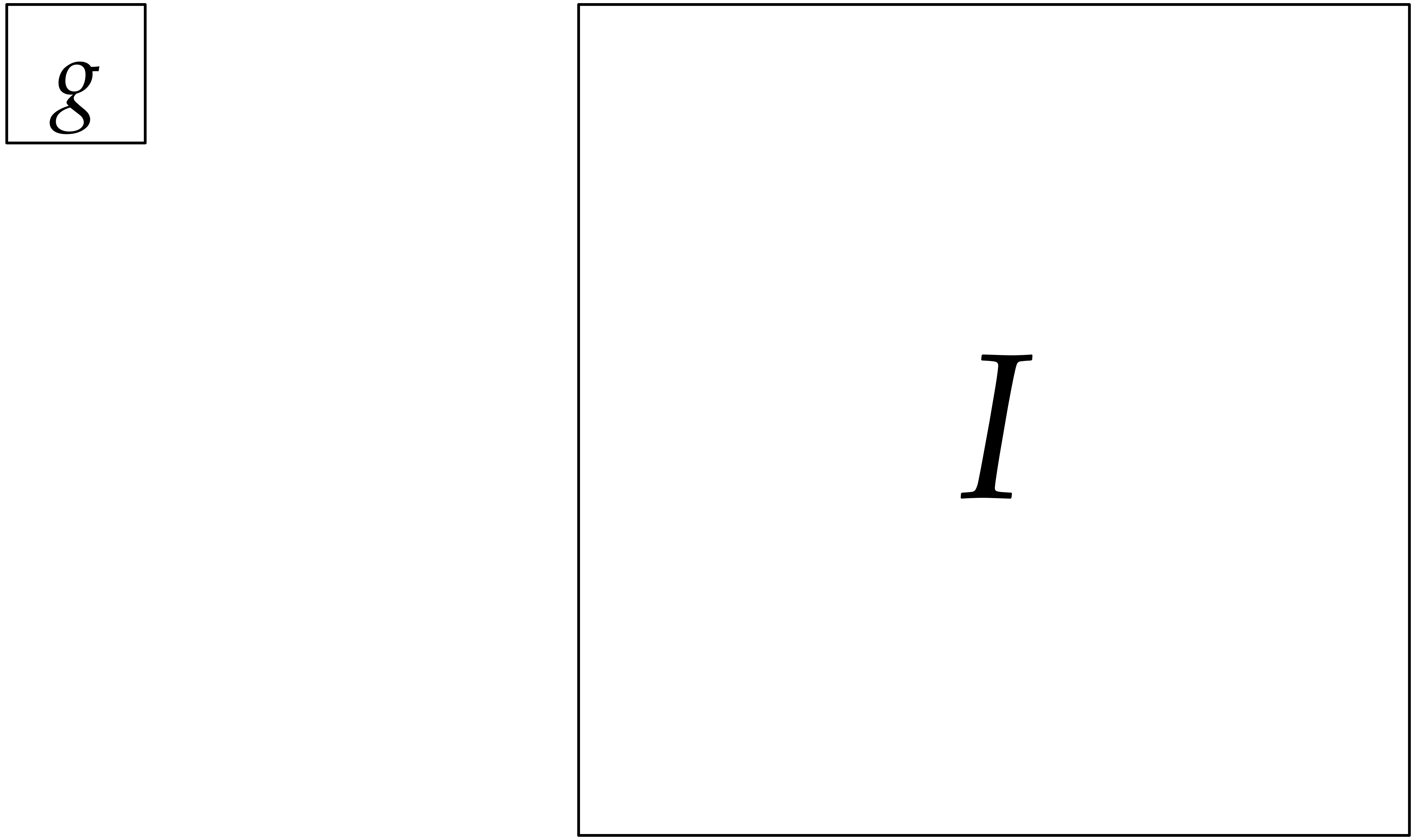
After a slide by Frédo Durand

Questions?

Formally

More formally: Convolution

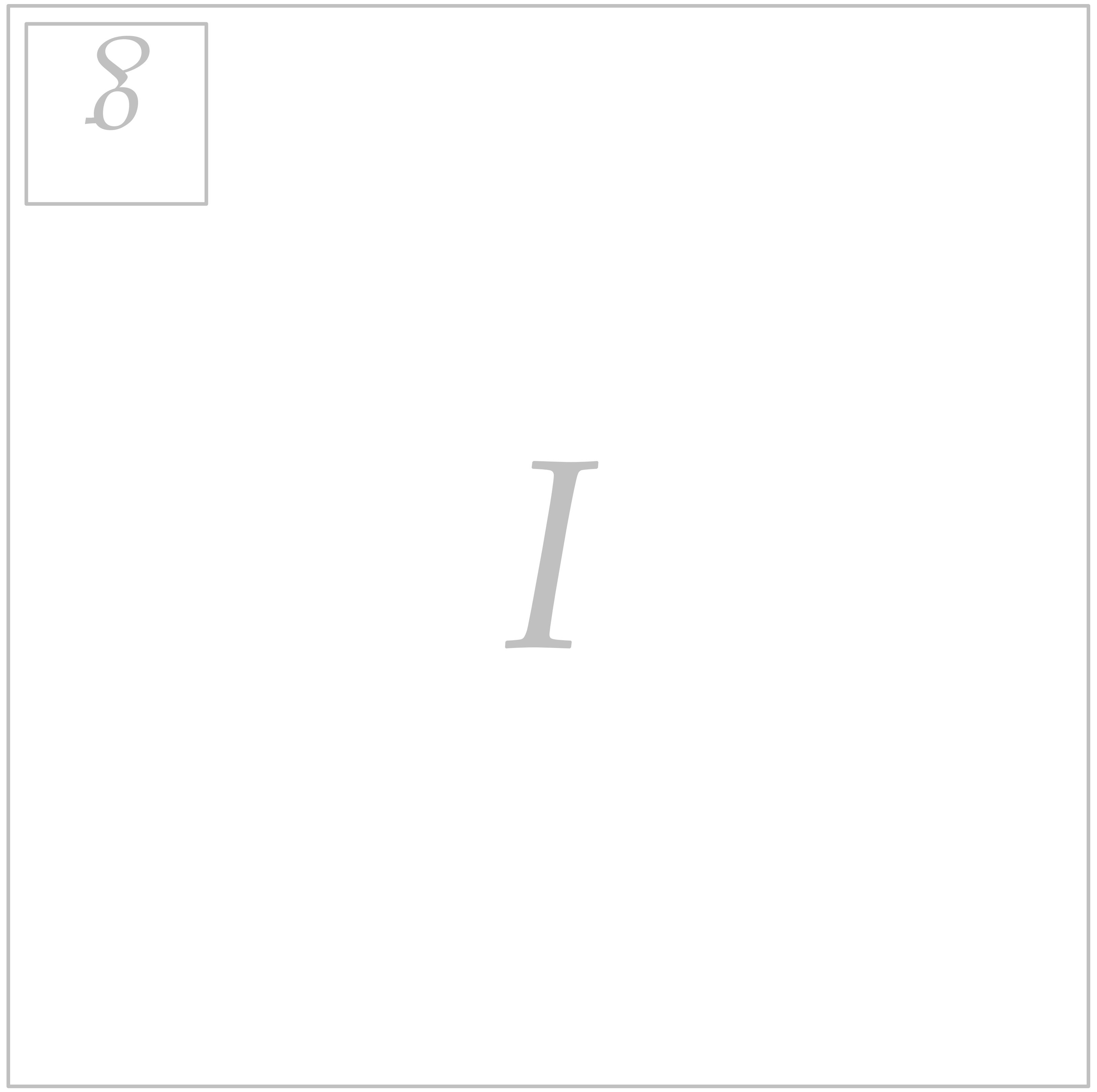
$$(I \otimes g)(x) = \int_{x'} I(x')\, g(x - x')\, \mathrm{d}x'$$



After a slide by Frédo Durand

Questions?

$$(I \otimes g)(x) = \int_{x'} I(x')\, g(x - x')\, \mathrm{d}x'$$



After a slide by Frédo Durand

What’s up with the flipping?

Convolution & probability

Convolution was first used by Laplace to study the probability of the sum of two random variables

After a slide by Frédo Durand

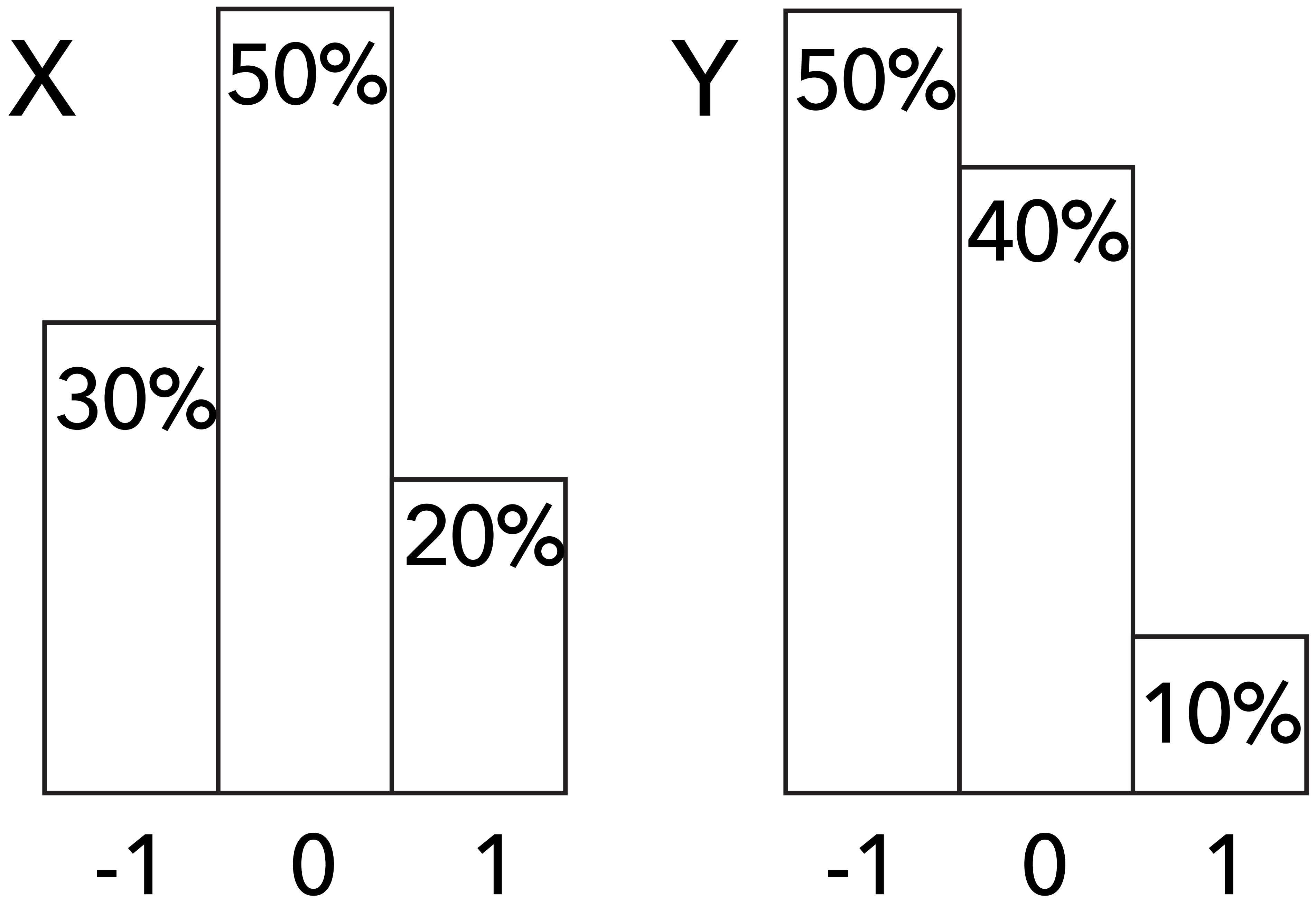
Random variables

How can X+Y=0?

- X=-1, Y=1
- X=0, Y=0
- X=1, Y=-1

Probability?

- P(X=-1)*P(Y=1)
- P(X=0)*P(Y=0)
- P(X=1)*P(Y=-1)



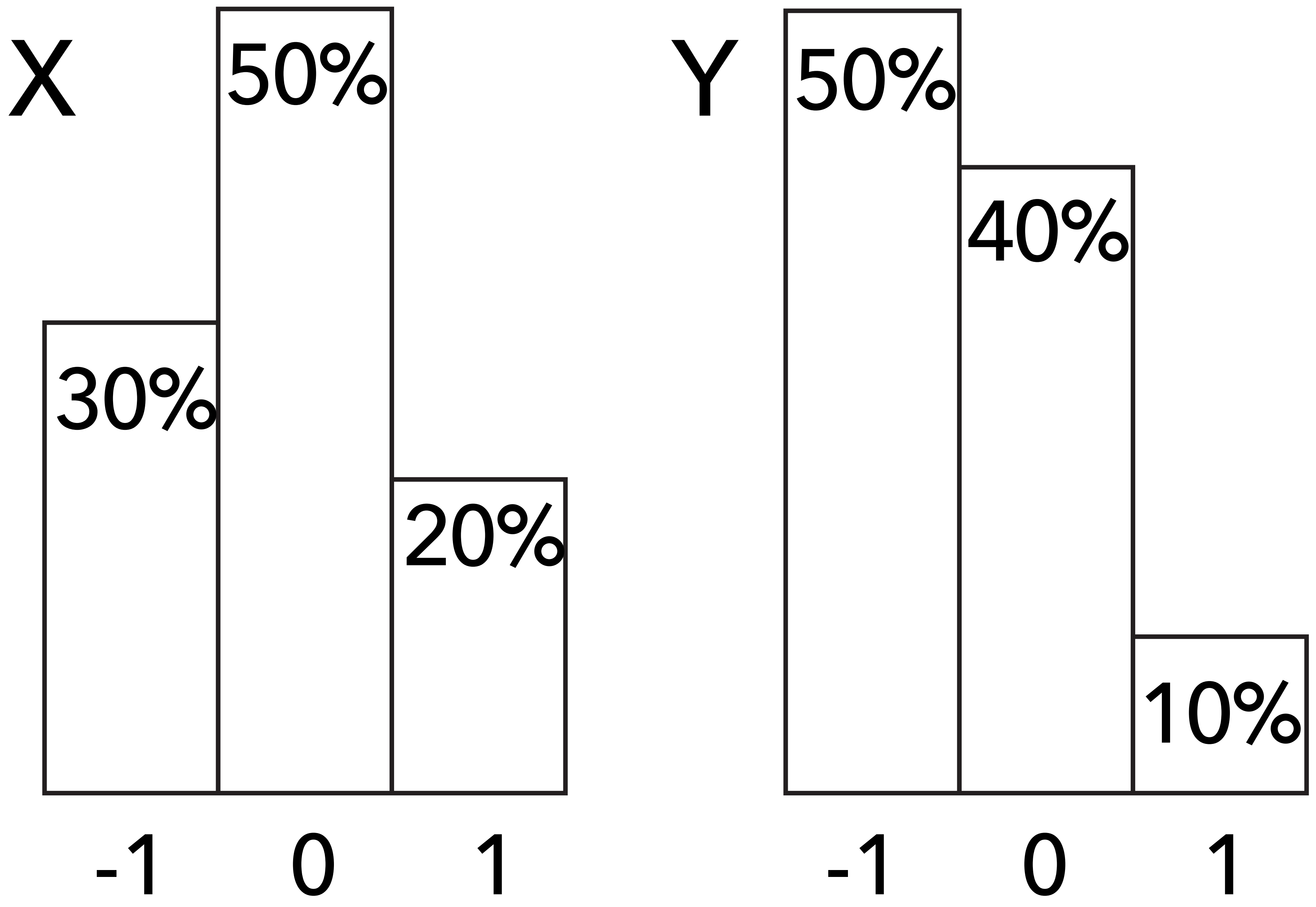
After a slide by Frédo Durand

Sum of random variables

$$P(X+Y=k)=\sum_{k'}P(X=k')P(Y=k-k')$$

How can X+Y=0?

- X=-1, Y=1
- X=0, Y=0
- X=1, Y=-1



Probability?

- P(X=-1)*P(Y=1)
- P(X=0)*P(Y=0)
- P(X=1)*P(Y=-1)

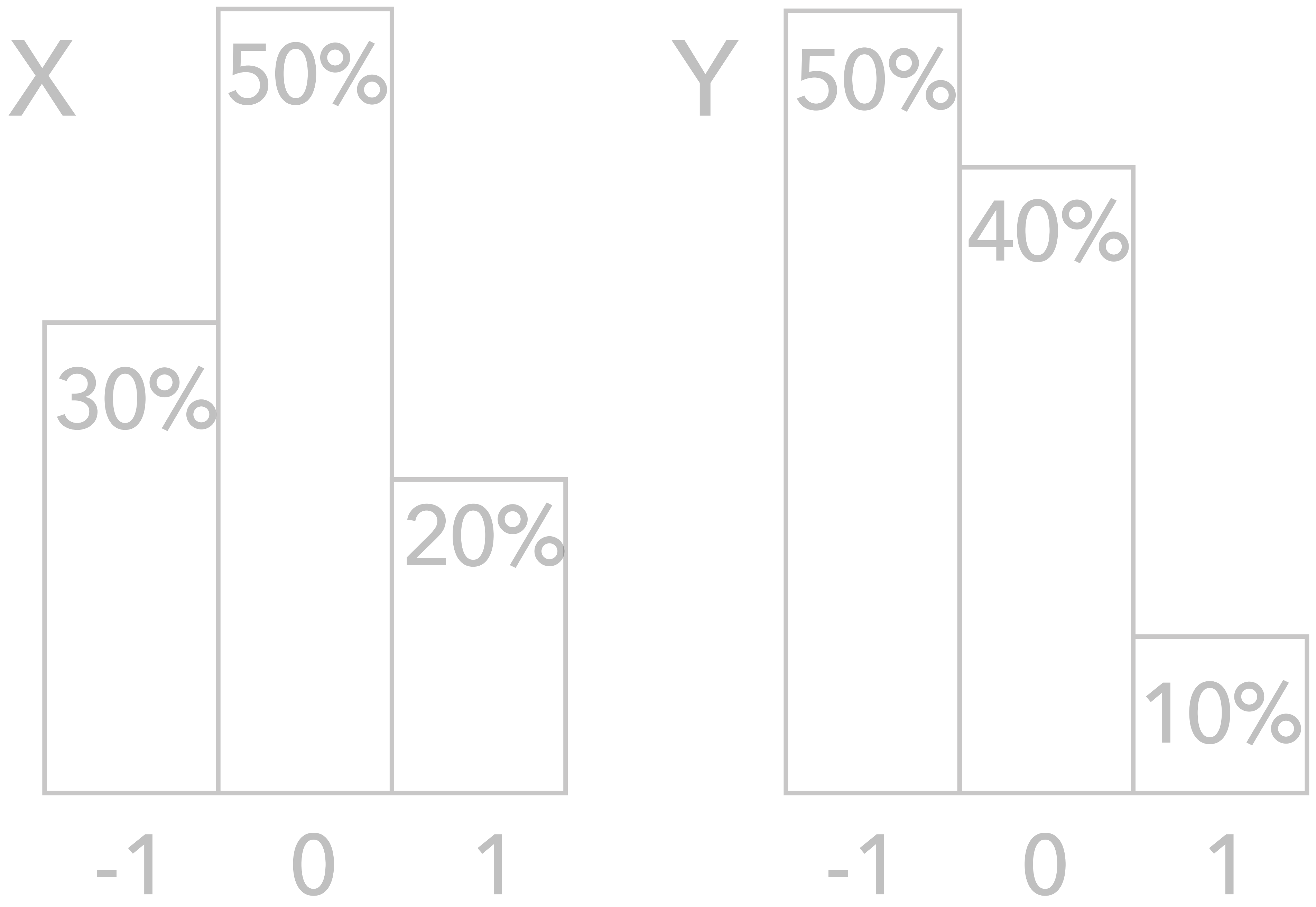
After a slide by Frédo Durand

Questions?

$$P(X+Y=k) = \sum_{k'} P(X=k')P(Y=k-k')$$

How can X+Y=0?

- X=-1, Y=1
- X=0, Y=0
- X=1, Y=-1



Probability?

- P(X=-1)*P(Y=1)
- P(X=0)*P(Y=0)
- P(X=1)*P(Y=-1)

After a slide by Frédo Durand

Compare

$$P(X+Y=k) = \sum_{k'} P(X=k')P(Y=k-k')$$

$$(I \otimes g)(x) = \int_{x'} I(x')\, g(x-x')\, \mathrm{d}x'$$

After a slide by Frédo Durand

Forward model: light goes from x to x+x′

Backward model: light at x comes from x-x′

Image processing

I will often use the term "convolution" improperly and fail to flip the kernel

- Called correlation
- Won't matter most of the time because our kernels are symmetric

After a slide by Frédo Durand

Questions?

Movie break

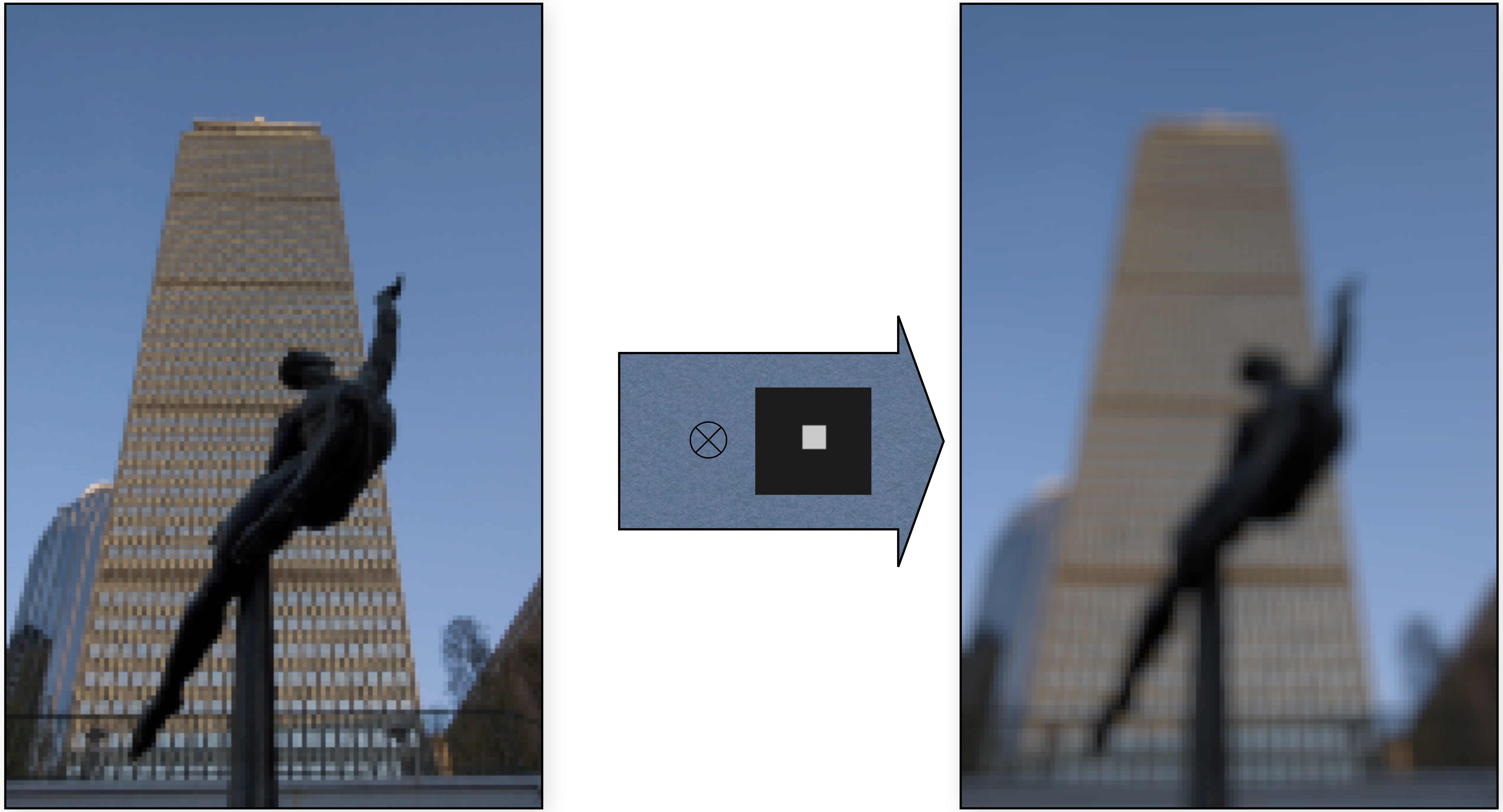
https://youtu.be/HyZfIlxwsfI

Blur zoo

http://graphics.stanford.edu/courses/cs178/applets/convolution.html

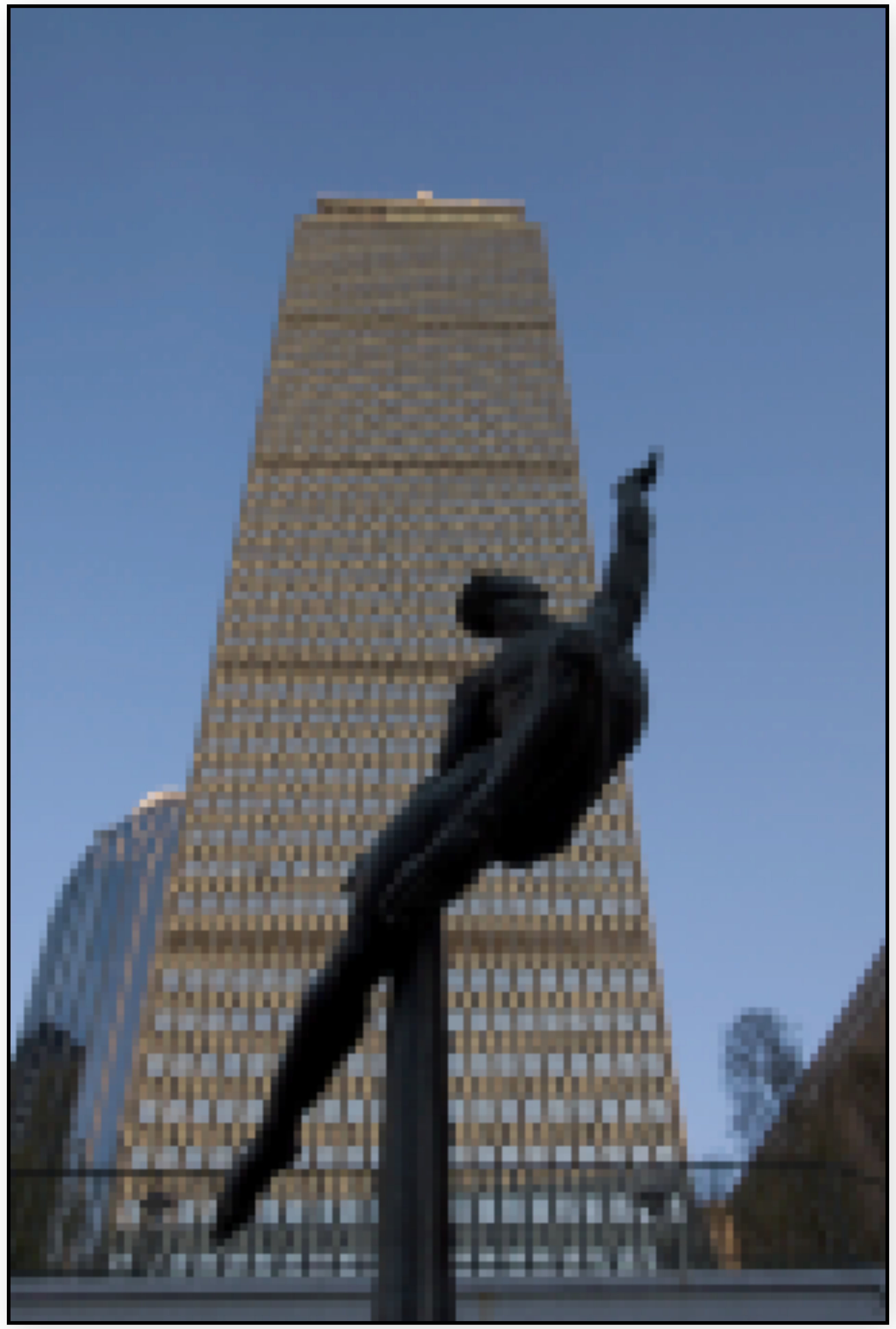
After a slide by Frédo Durand

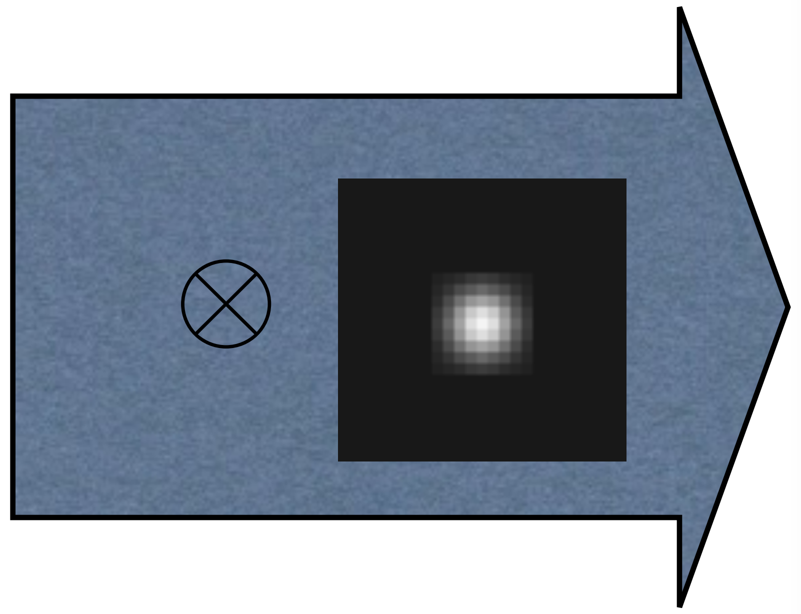
Box filter

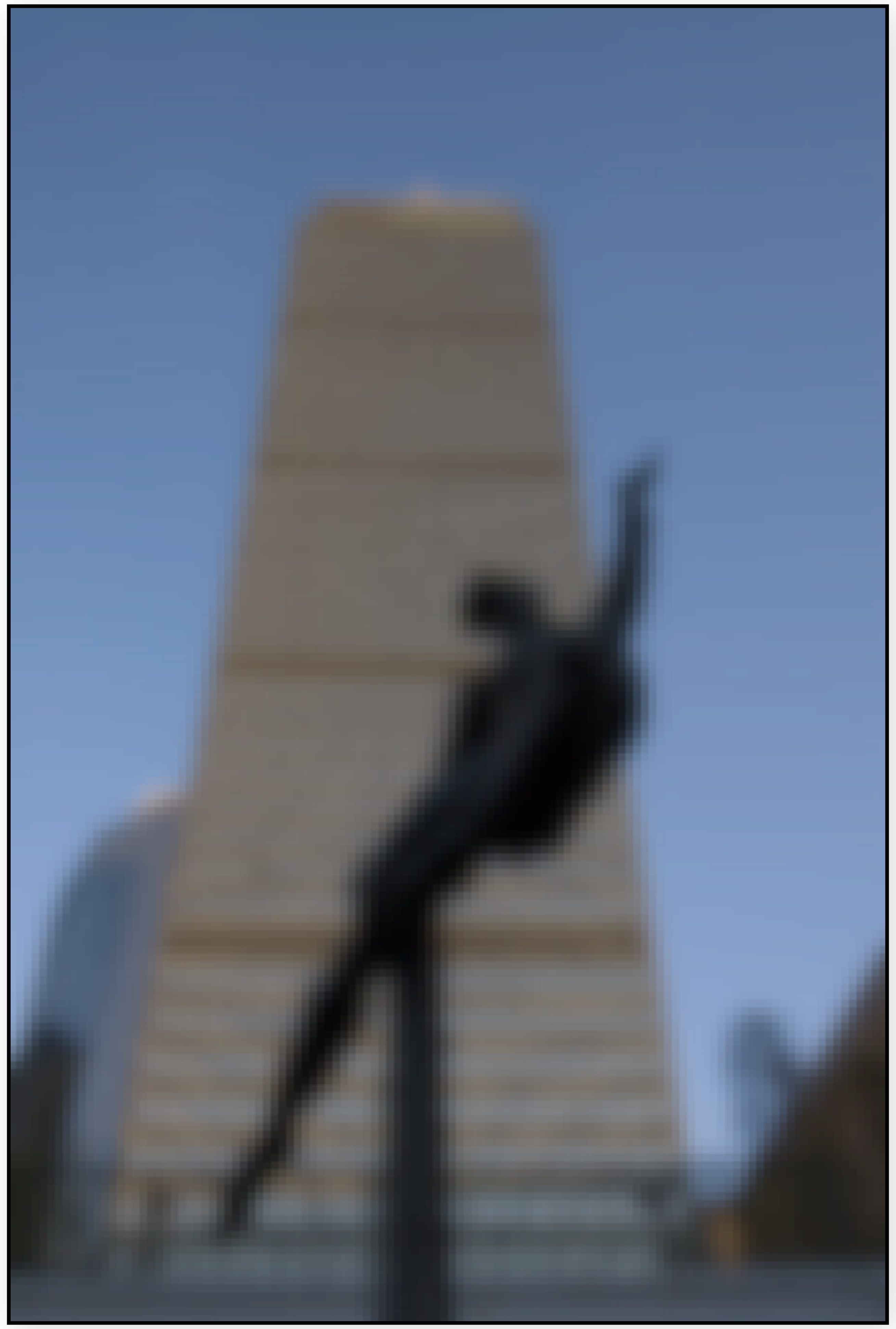


After a slide by Frédo Durand

Nice and smooth: Gaussian





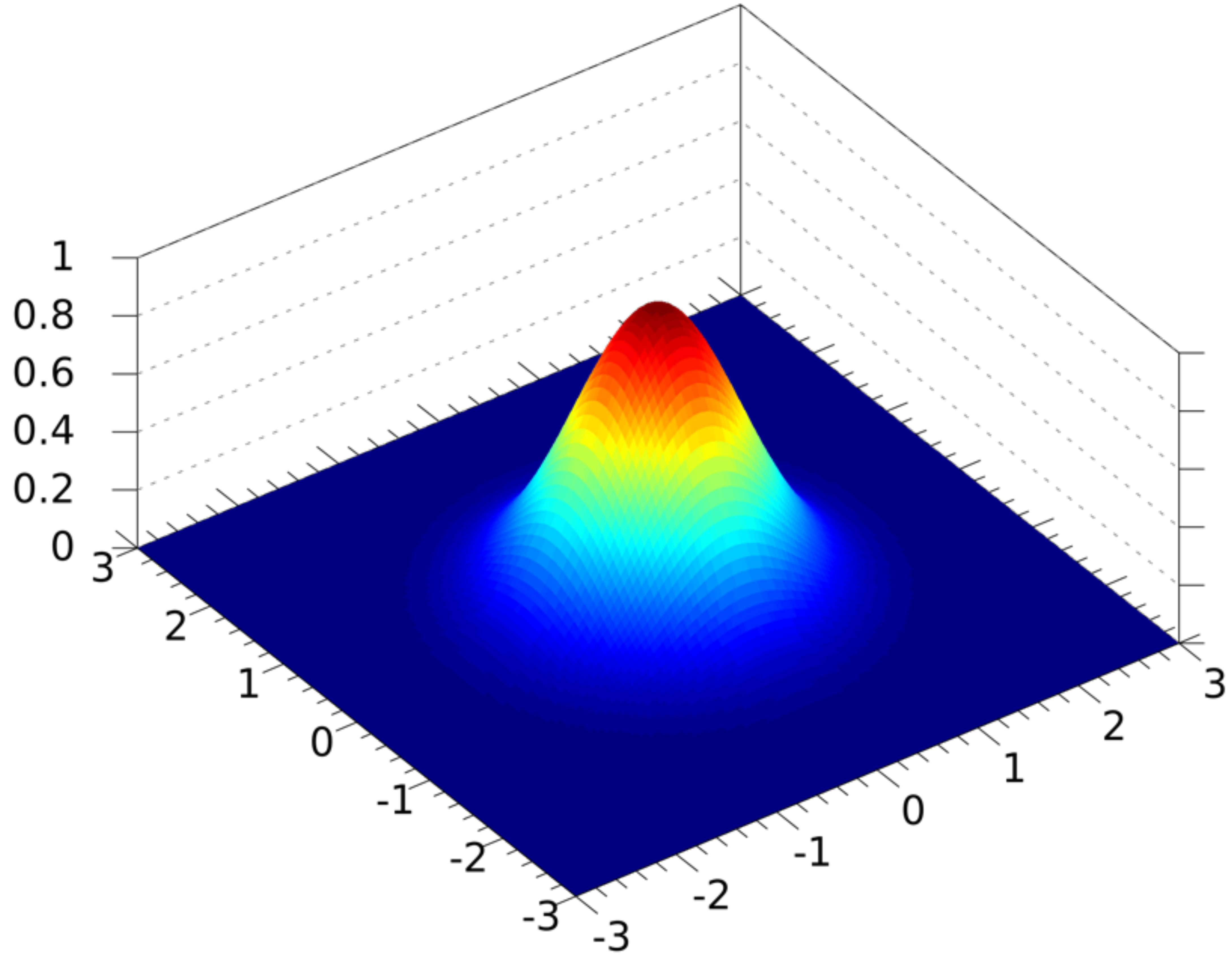


After a slide by Frédo Durand

Gaussian formula

http://en.wikipedia.org/wiki/Gaussian_function

$$ae^{-\frac{r^2}{2\sigma^2}}$$



r is the distance to the center

a is a normalization constant

- I usually just normalize my kernels after the fact

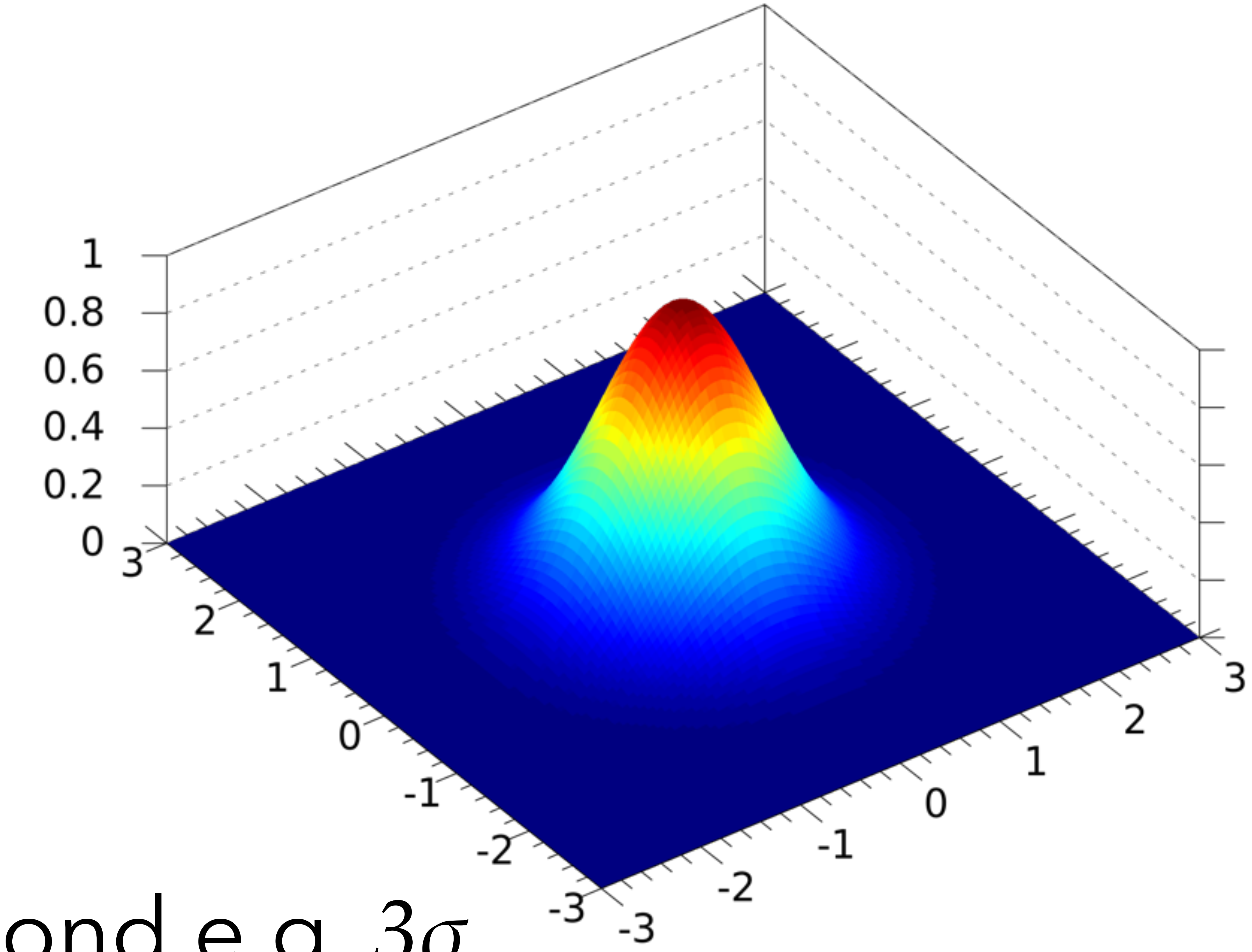
σ is the standard deviation and controls the width of the Gaussian

After a slide by Frédo Durand

Gaussian formula

http://en.wikipedia.org/wiki/Gaussian_function

$$ae^{-\frac{r^2}{2\sigma^2}}$$



Gaussians have infinite support

- >0 everywhere

but are often truncated

- consider Gaussian to be zero beyond e.g. 3σ
- for computational tractability/efficiency

After a slide by Frédo Durand

Sharpening

How can we sharpen?

Blurring was easy

Sharpening is not as obvious

After a slide by Frédo Durand

How can we sharpen?

Blurring was easy

Sharpening is not as obvious

Idea: amplify the stuff not in the blurry image

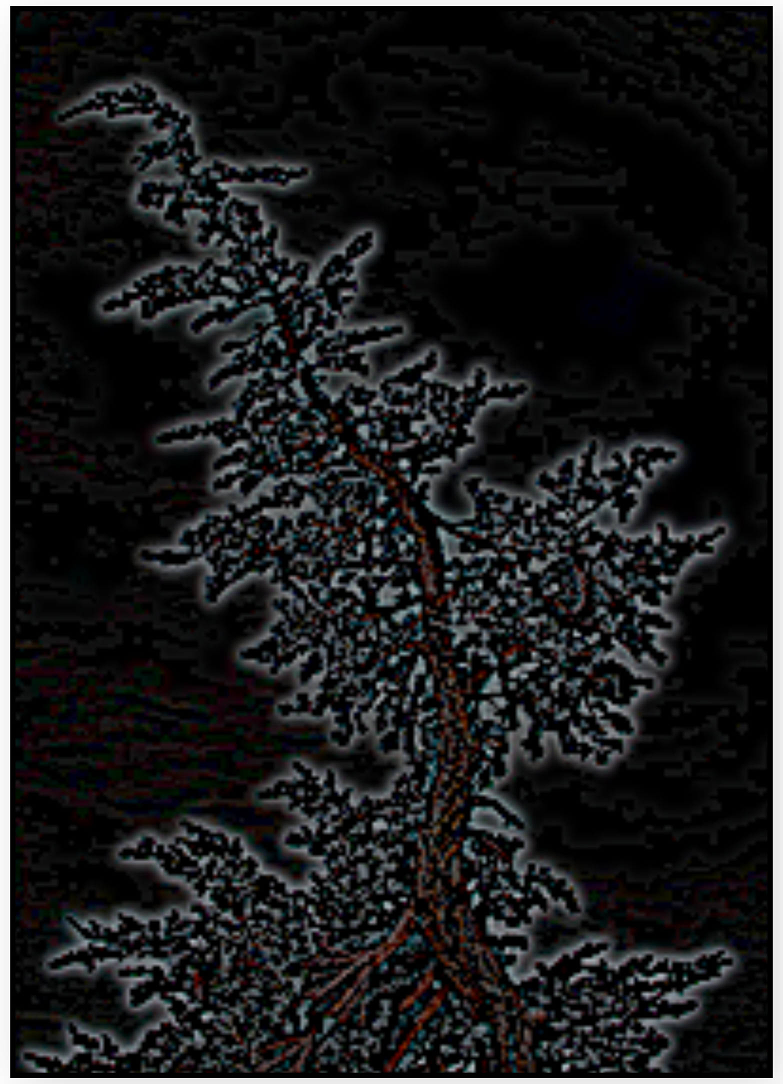
output = input + k*(input-blur(input))

After a slide by Frédo Durand

Sharpening

-

=



high pass

+k*

=

sharpened image

After a slide by Frédo Durand

Sharpening: kernel view

Recall

$$f' = f + k * (f - f \otimes g)$$

f is the input

f' is a sharpened image

g is a blurring kernel

k is a scalar controlling the strength of sharpening

After a slide by Frédo Durand

Sharpening: kernel view

Recall

$$f' = f + k * (f - f \otimes g)$$

Denote δ the Dirac kernel (pure impulse)

$$f = f \otimes \delta$$

After a slide by Frédo Durand

Sharpening: kernel view

Recall

$$f' = f + k * (f - f \otimes g)$$

$$f' = f \otimes \delta + k * (f \otimes \delta - f \otimes g)$$

$$f' = f \otimes ((k+1)\delta - g)$$

Sharpening is also a convolution

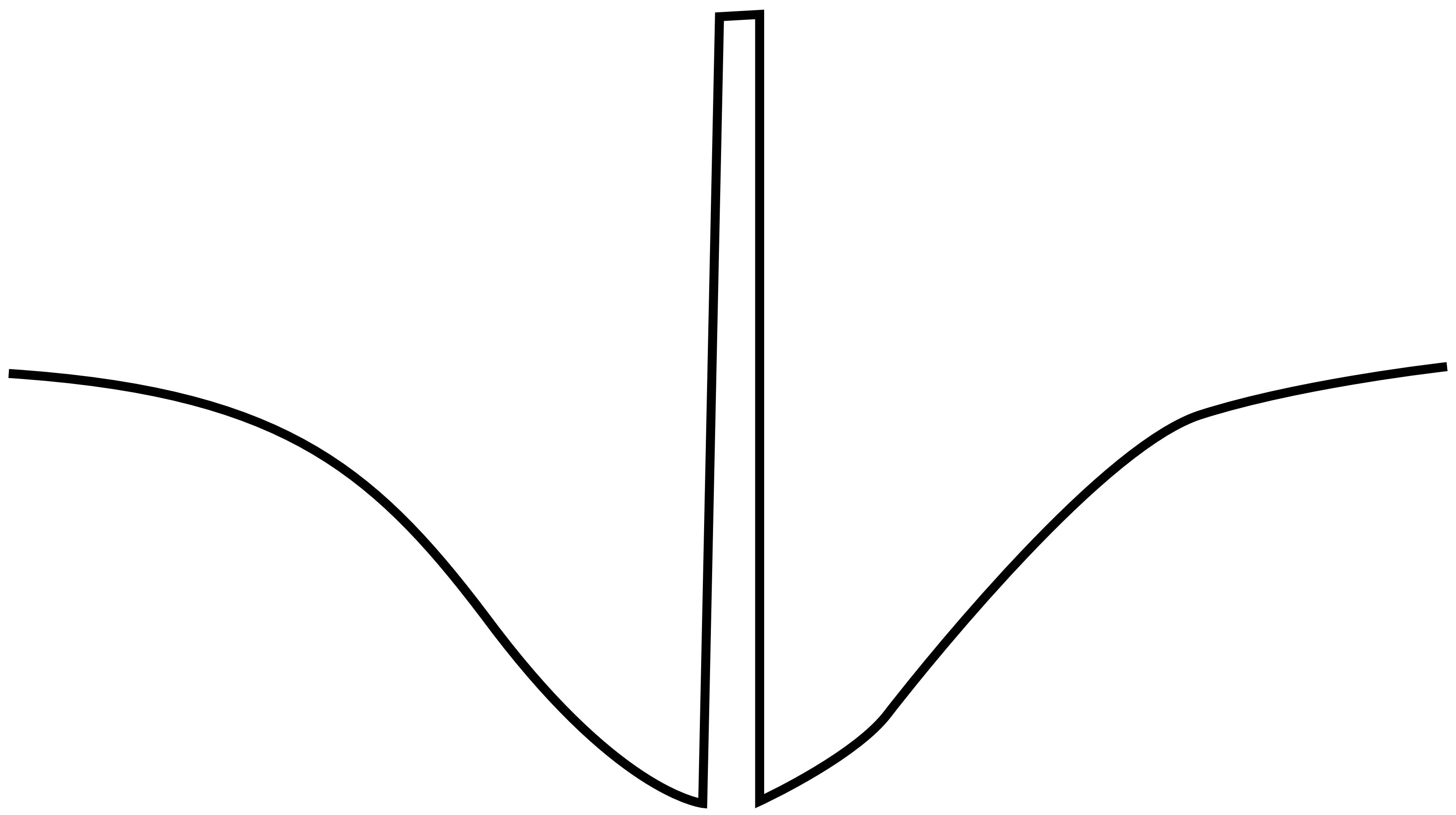
After a slide by Frédo Durand

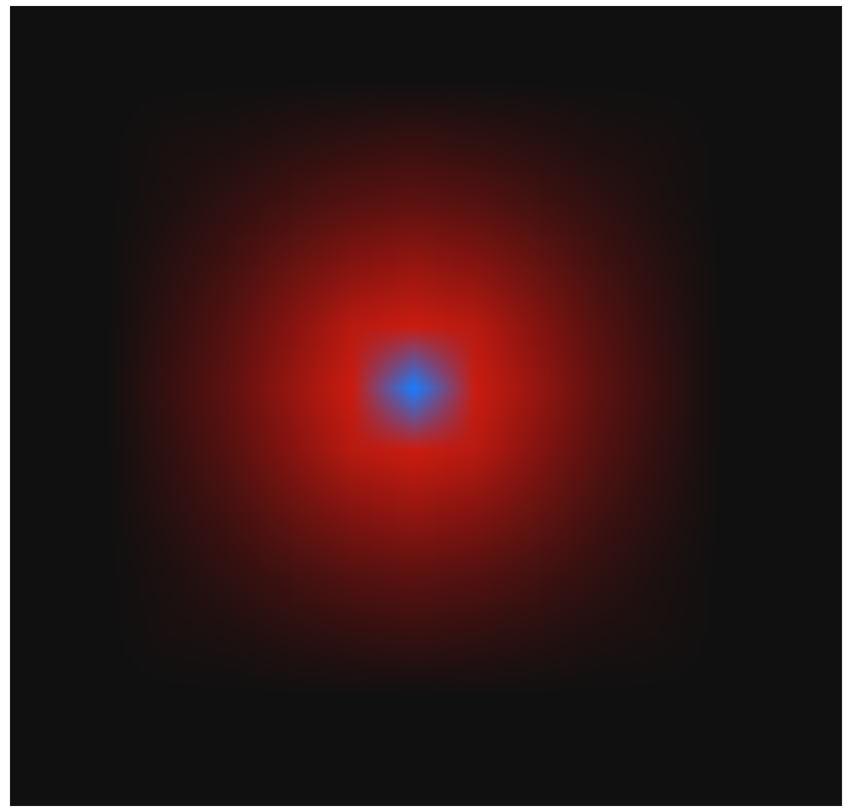
Sharpening kernel

Note: many other sharpening kernels exist
(just like we saw multiple blurring kernels)

Amplify the difference between a pixel and its neighbors

$$f' = f \otimes ((k+1)\delta - g)$$





blue: positive
red: negative

After a slide by Frédo Durand

Alternate interpretation

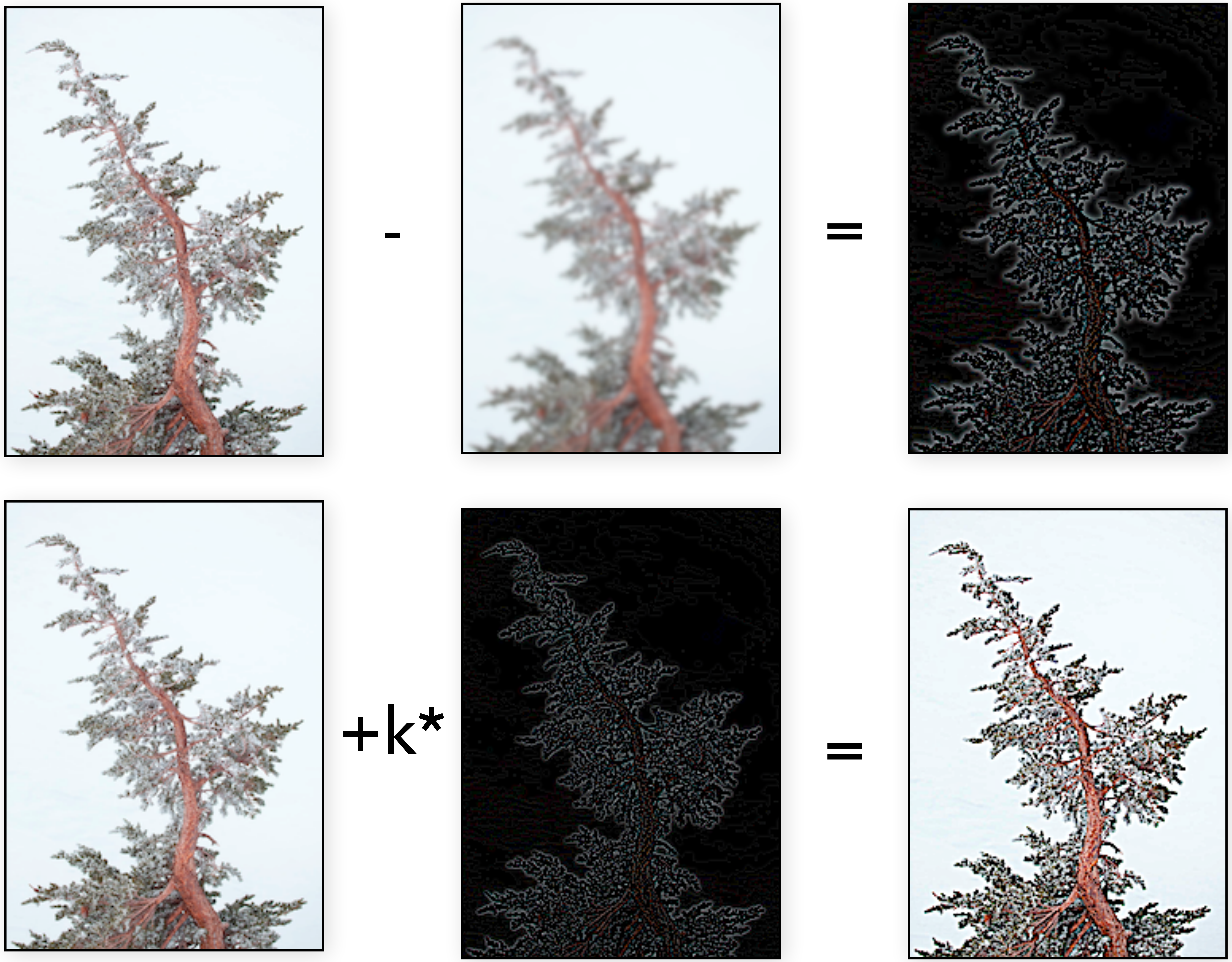
out = input + k*(input-blur(input))

out = (1 + k)*input - k*blur(input)

out = lerp(blur(input), input, 1+k)

- linearly extrapolate from the blurred image "past" the original input image

Questions?

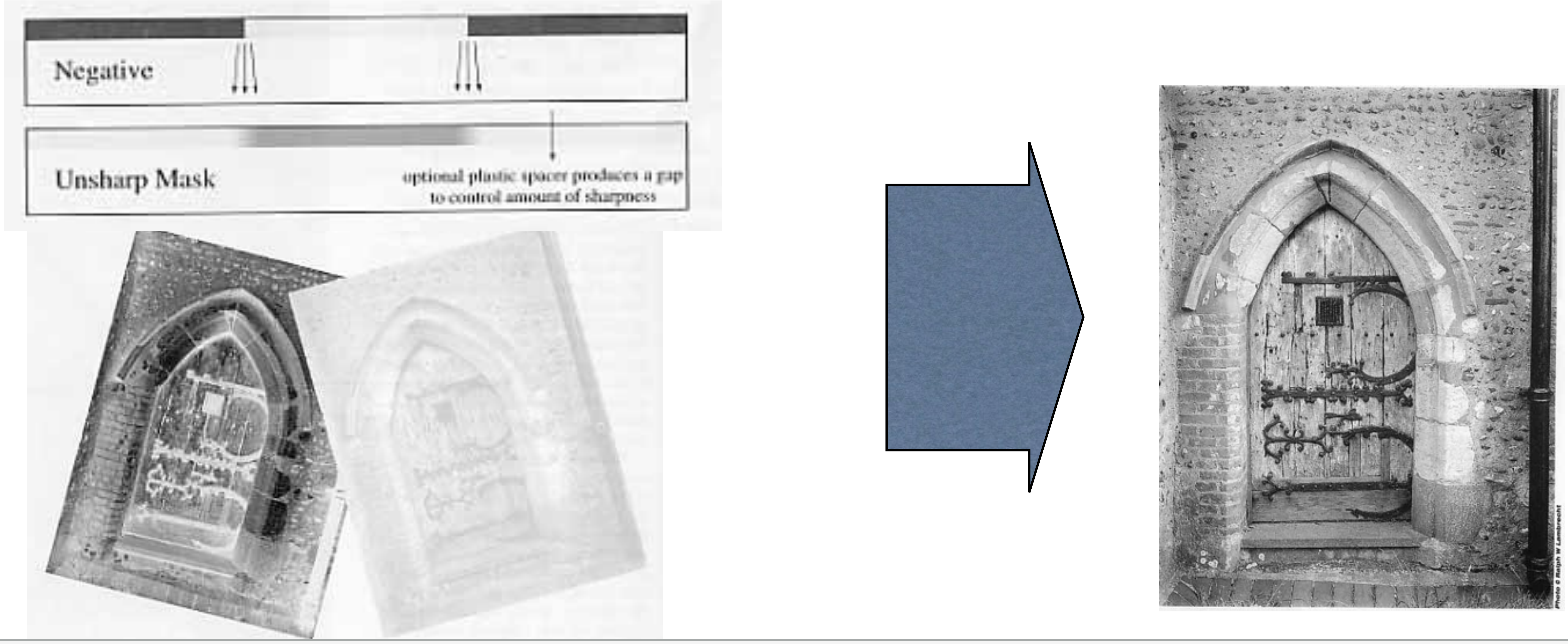


Unsharp mask

Unsharp mask

http://www.tech-diy.com/UnsharpMasks.htm

Sharpening is often called "unsharp mask" because photographers used to sandwich a negative with a blurry positive film in order to sharpen



After a slide by Frédo Durand

Fig.4: The two examples here show a detail of the brickwork to the left of the church door. The one on the left was printed with the negative alone – the one on the right was printed with both negative and mask as a sandwich. The increase in local contrast and edge sharpness is minute, but clearly visible. Grade 2.5 was used for the straight print but increased to 4.5 for the sandwiched image to compensate for the reduced contrast.

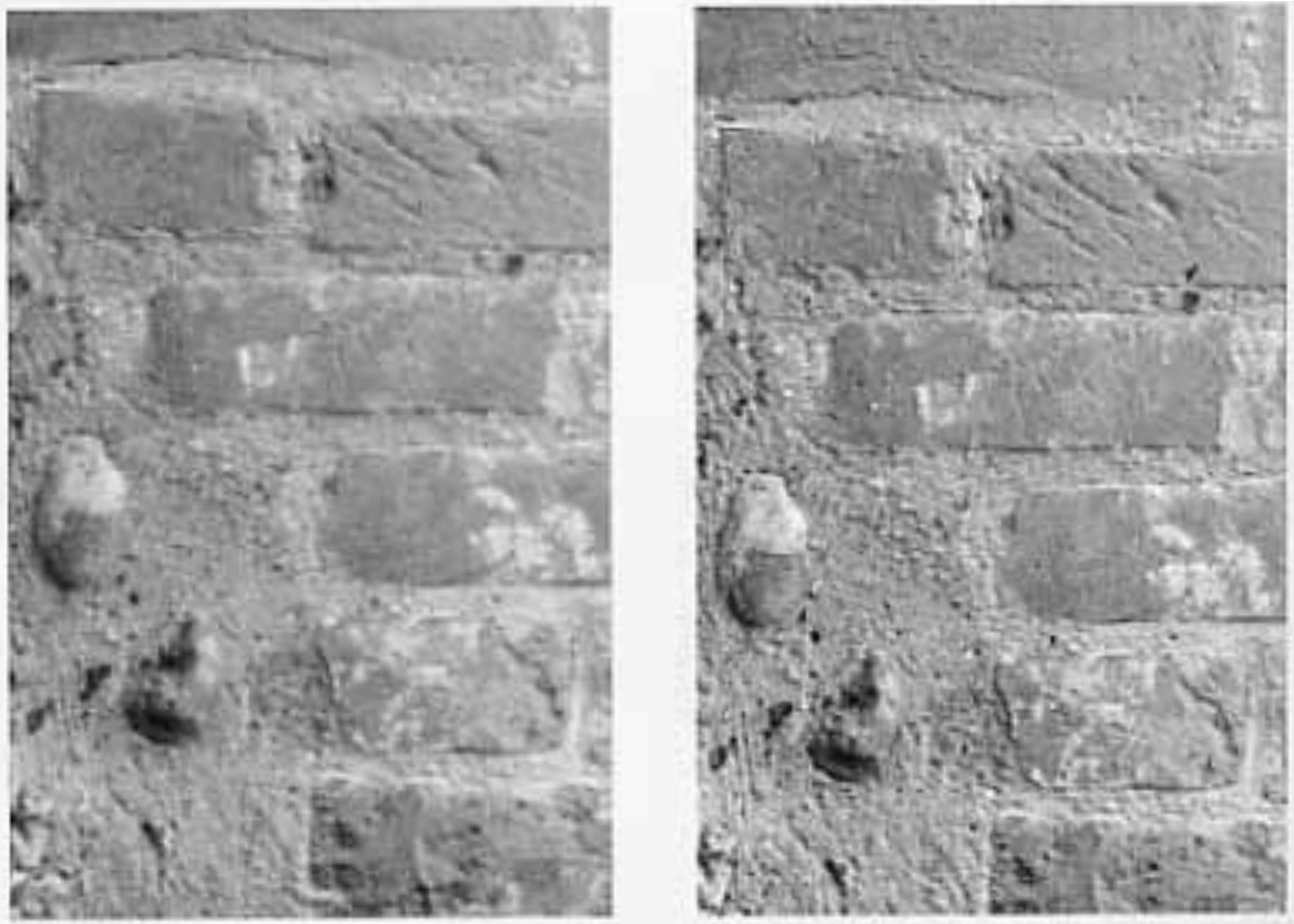
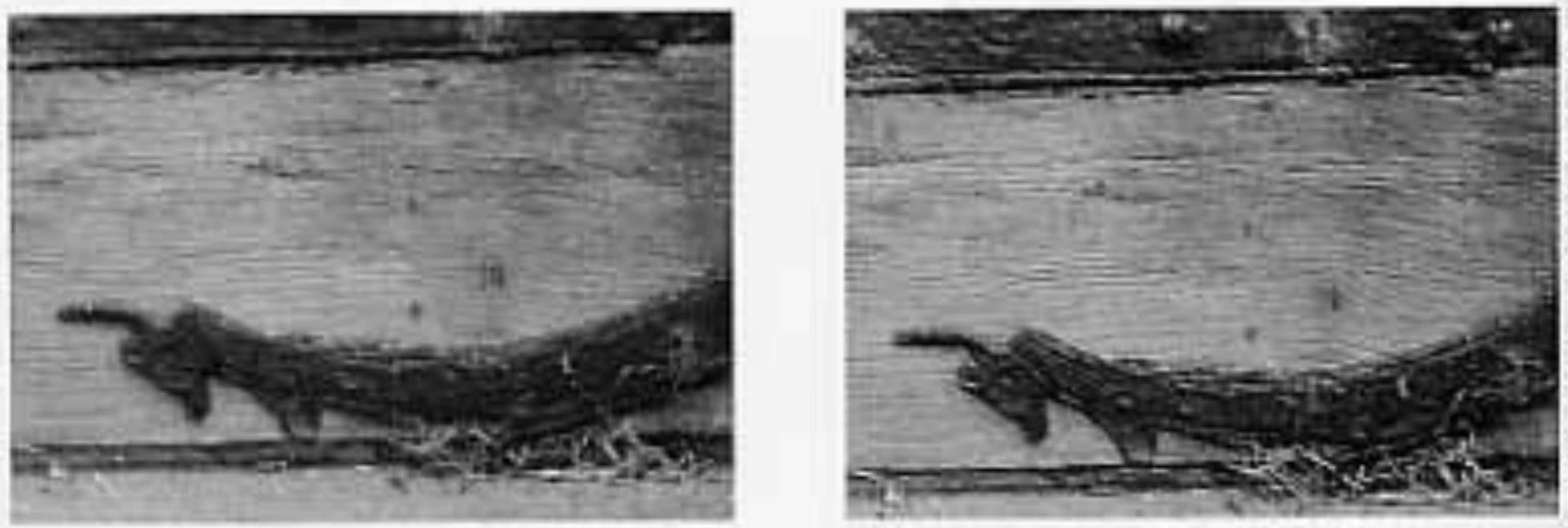


Fig.5: These two examples show a detail of the lower right hand side of the church door. Here the difference in sharpness is clearly visible between the (left) negative and (right) sandwich prints.



All photos © Ralph W Lambrecht

After a slide by Frédo Durand

http://www.tech-diy.com/images/unsharp2.jpg

Unsharp mask

http://en.wikipedia.org/wiki/Unsharp_masking

http://www.largeformatphotography.info/unsharp/

http://www.tech-diy.com/UnsharpMasks.htm

http://www.cambridgeincolour.com/tutorials/unsharp-mask.htm

After a slide by Frédo Durand

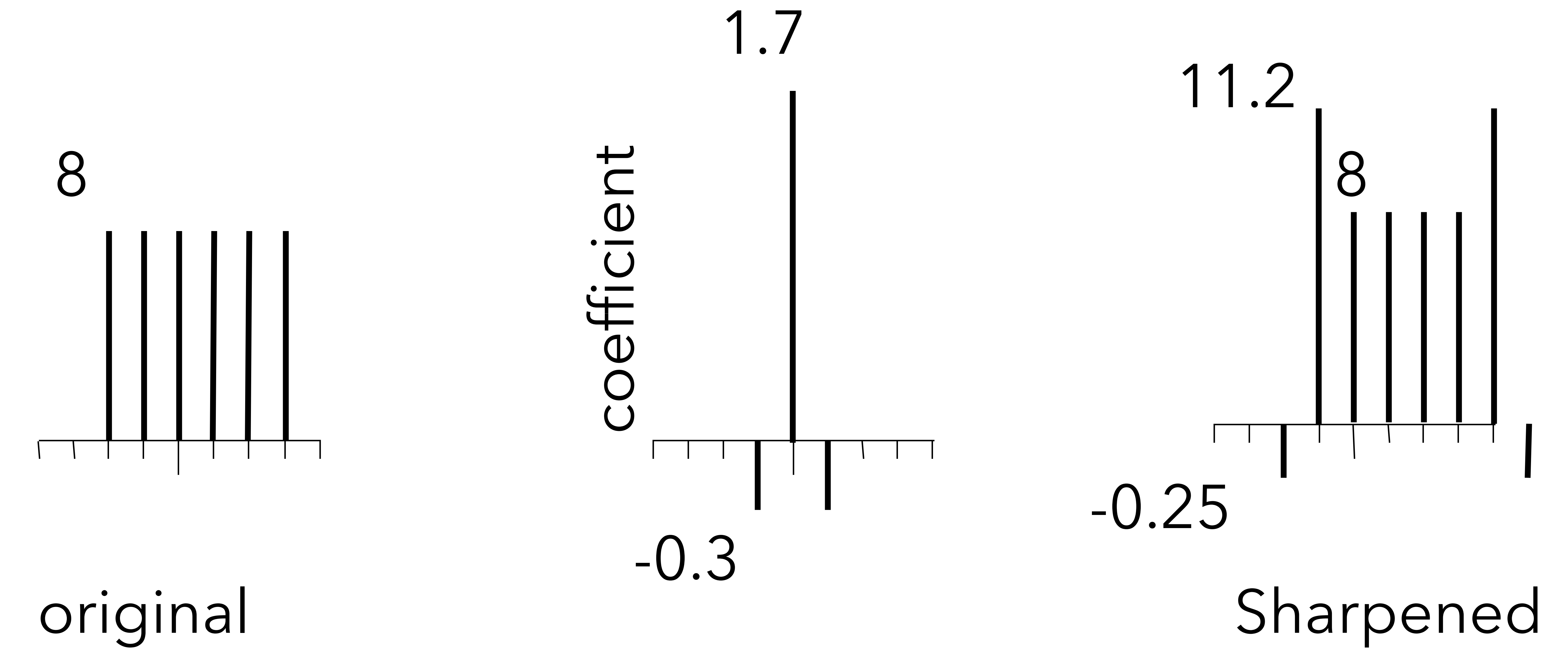
Sharpening++

Problem with excess

Haloes around strong edges

After a slide by Frédo Durand

Oversharpening



Sharpened
(differences are accentuated;
constant areas are left untouched).

After a slide by Frédo Durand

Bells and whistles

Apply mostly on luminance

Old Clarity in Lightroom/Adobe Camera Raw

- As far as I understand, apply only for mid-tones
- Avoids haloes around black and white points

Only apply at edges

- To avoid the amplification of noise

Sharpening chrominance as well

- But with very large blur

After a slide by Frédo Durand

Lightroom demo

Oriented filters

Gradient: finite difference

horizontal gradient [[-1, 1]]

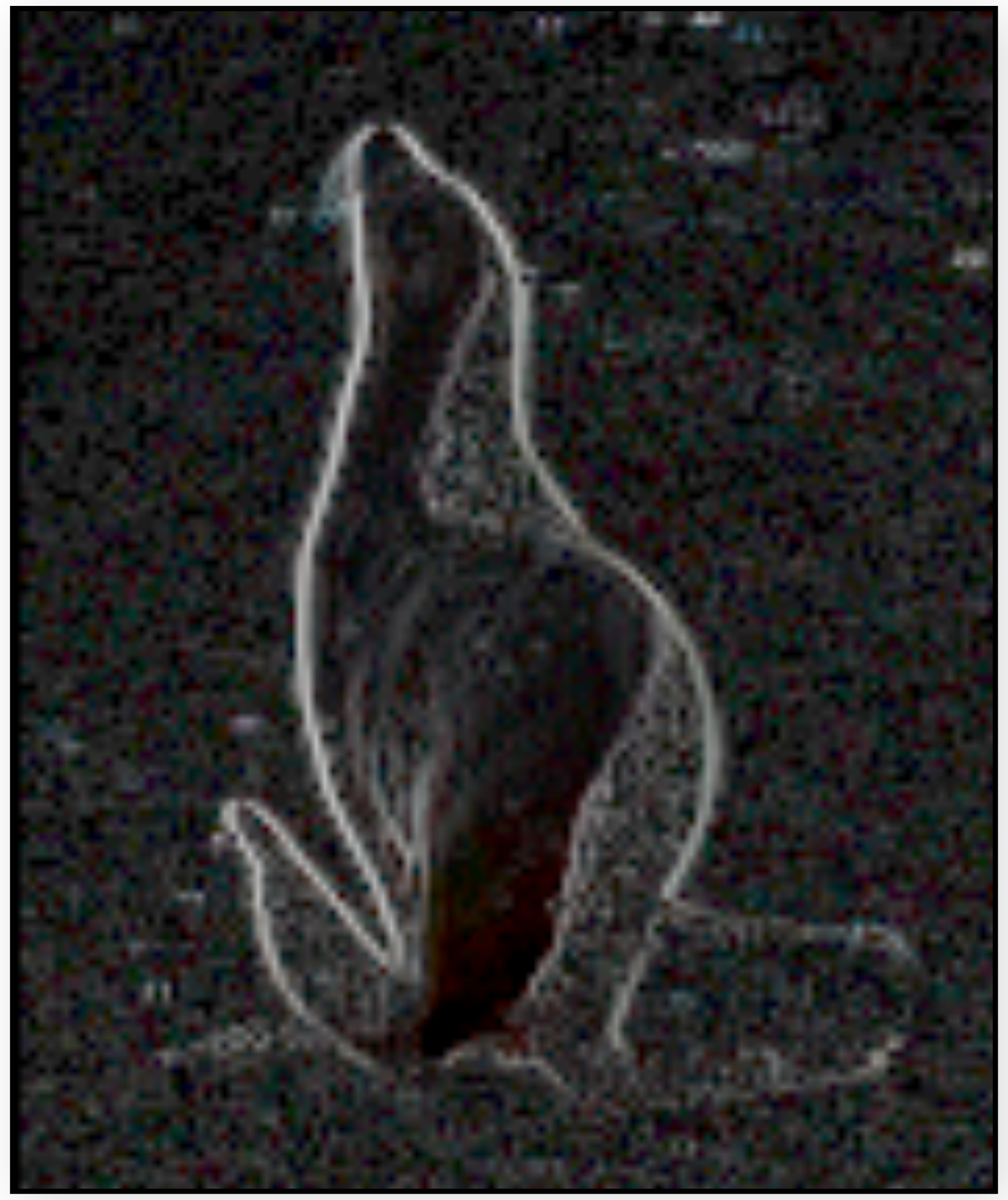
vertical gradient: [[-1], [1]]

After a slide by Frédo Durand

Gradient: finite difference

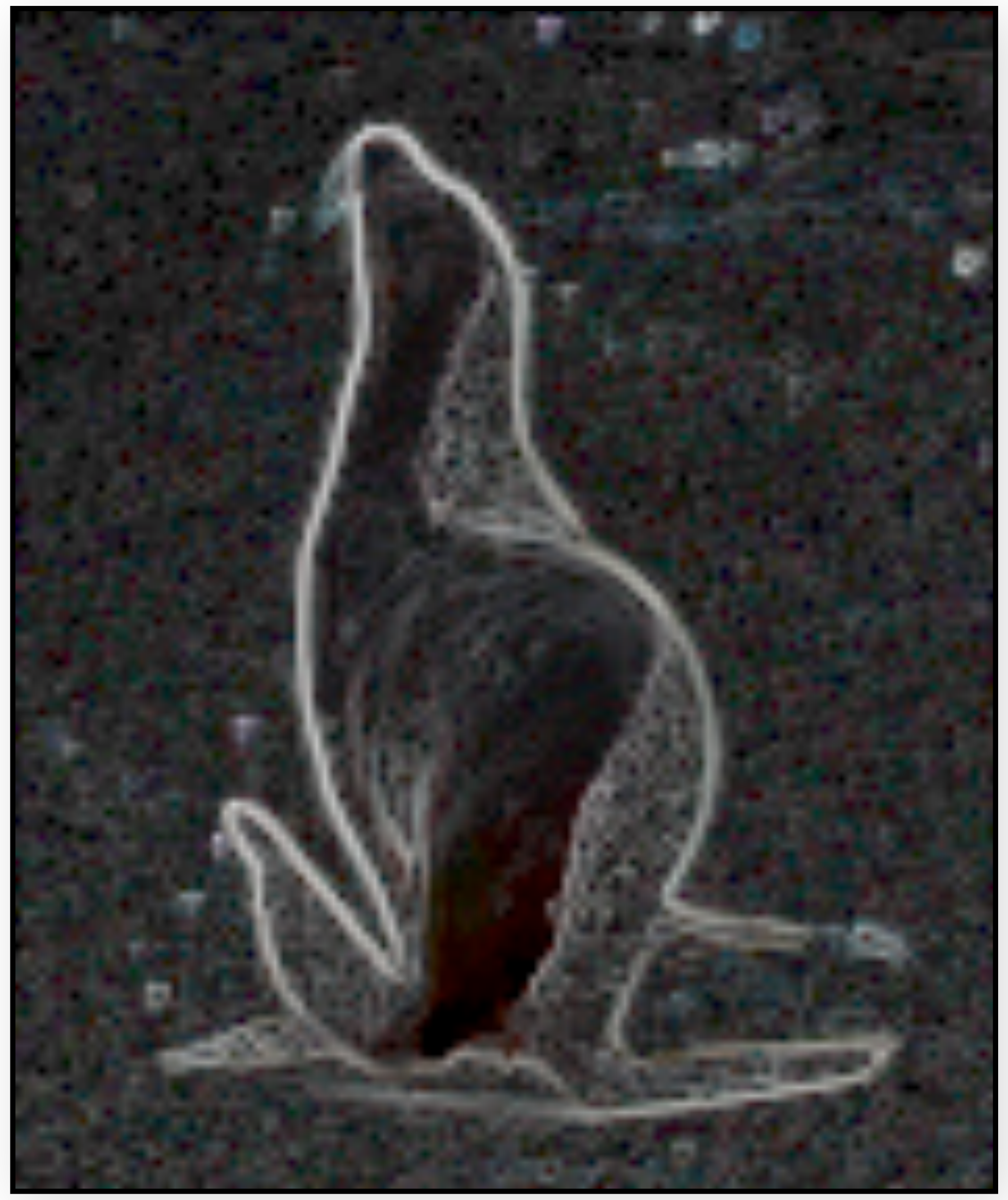
horizontal gradient [[-1, 1]]

vertical gradient: [[-1], [1]]



Horizontal gradient
(absolute value)

Vertical gradient
(absolute value)



Gradient magnitude

After a slide by Frédo Durand

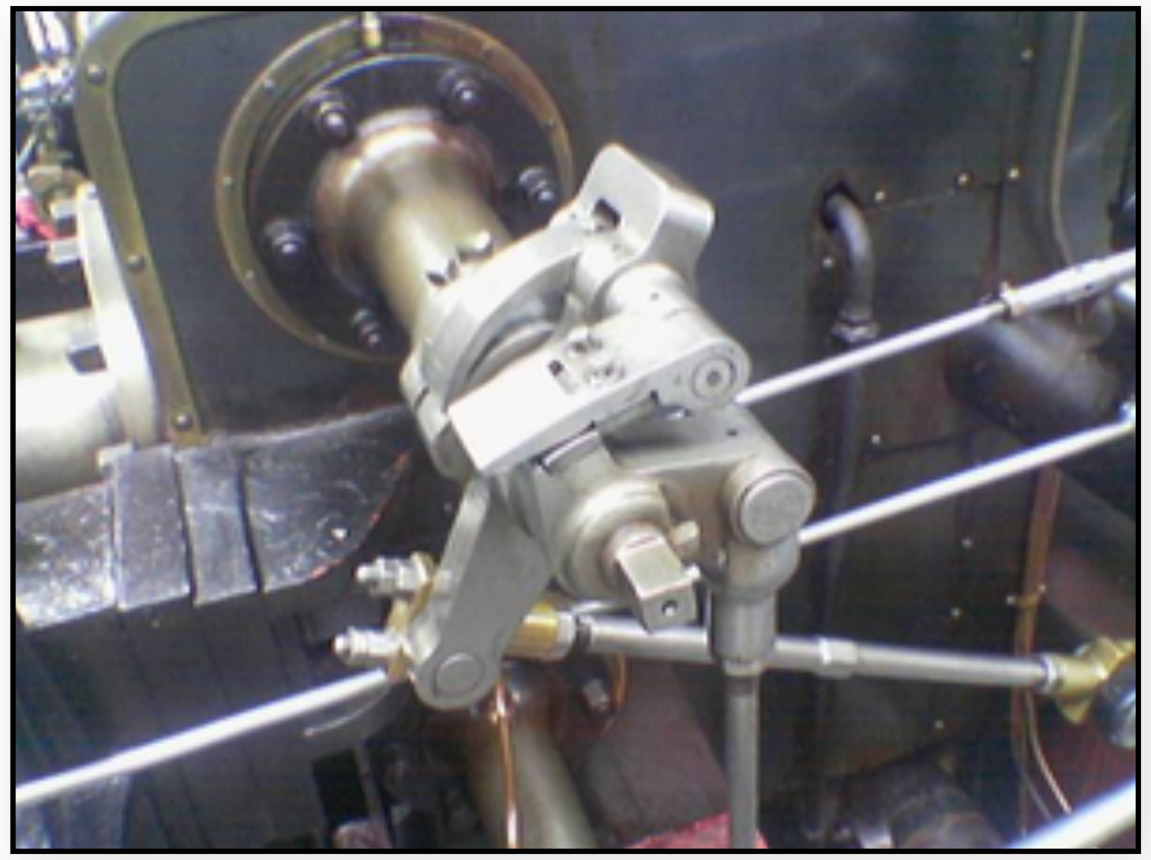
Gradient

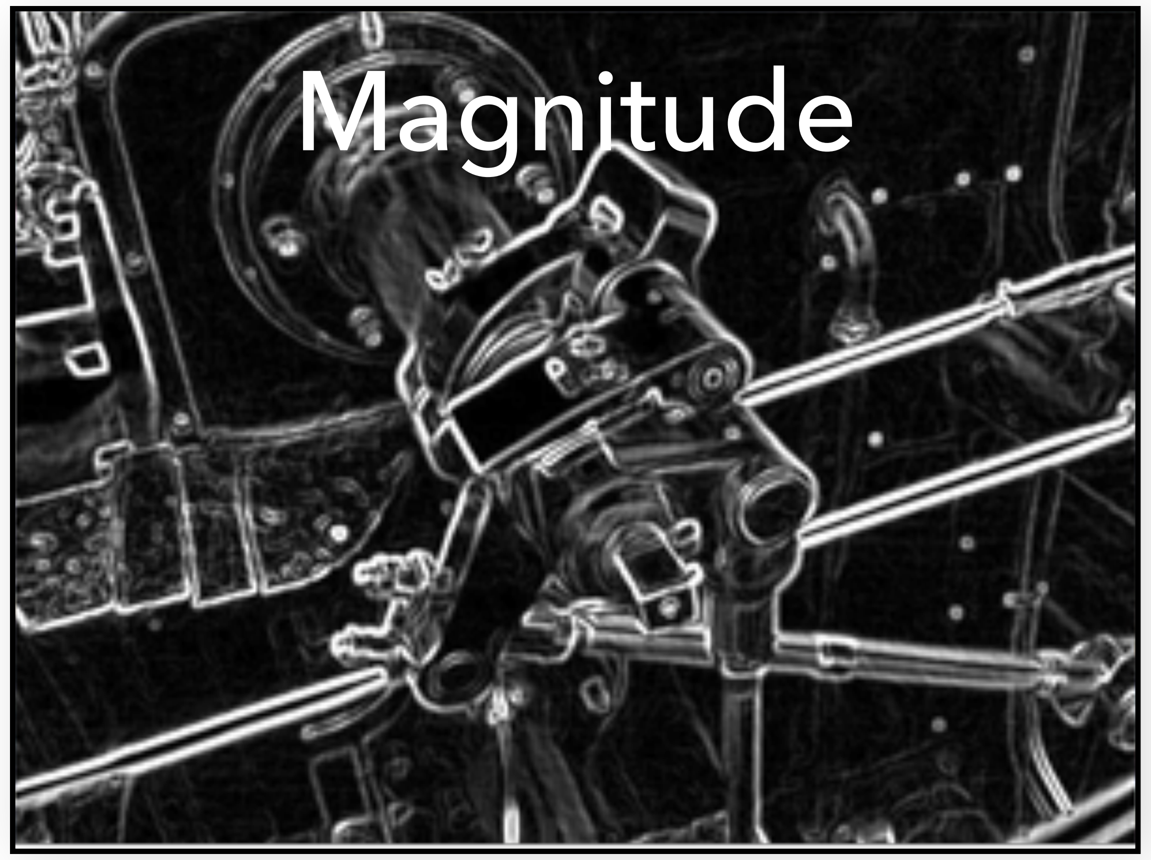
e.g. Sobel [http://en.wikipedia.org/wiki/Sobel_operator]

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \otimes \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \otimes \mathbf{A}$$

Horizontal gradient

Vertical gradient





After a slide by Frédo Durand

Cost

Convolution cost?

```
set output image to zero
for all pixels (x,y) in output image
    for all (x',y') in kernel
        out(x,y) += input(x+x',y+y')*kernel(x',y')
```

Cost?

- O(input.width * input.height * kernel.width * kernel.height)

After a slide by Frédo Durand

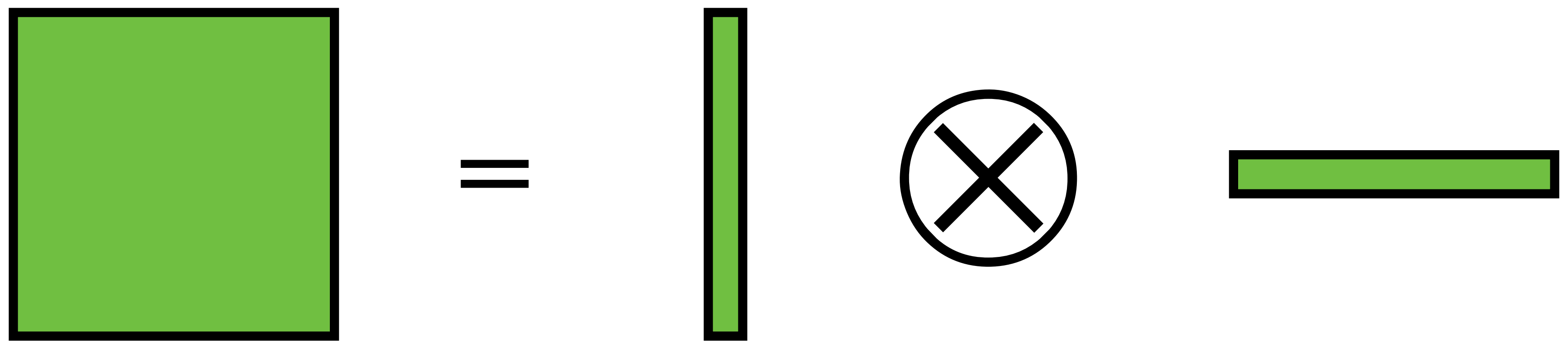
Separable filters

Separability

Sometimes the 2D kernel can be decomposed into the convolution of a horizontal and a vertical filter.

Example: box

- $g(x)$ = const if $(-k \leq x \leq k)$, 0 otherwise
- $g(x,y) = g(x) \otimes g(y)$
- (separability doesn't require the two 1D kernels to be the same, but it's the case here)

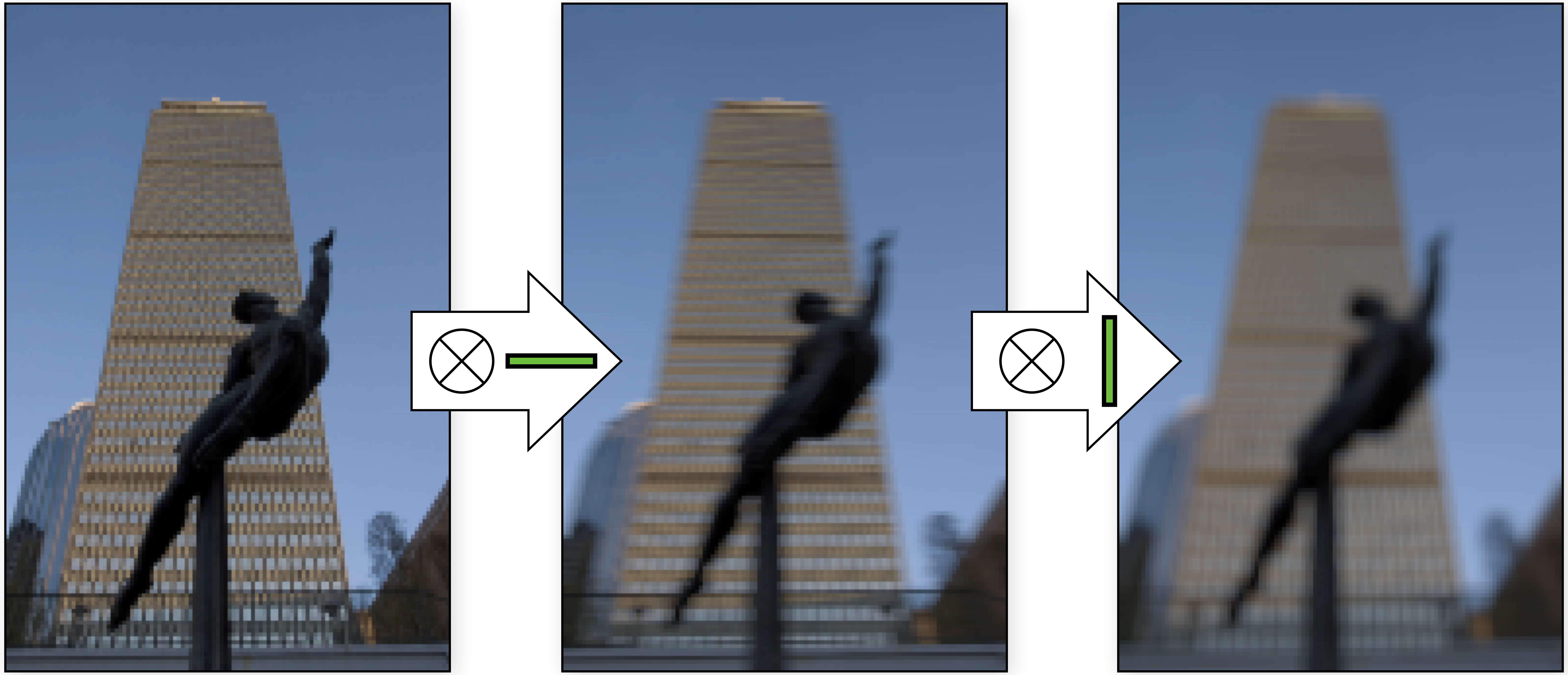


After a slide by Frédo Durand

Separable box blur

First blur horizontally using g(x)

Then blur vertically using g(y)



After a slide by Frédo Durand

Separable convolution cost?

```
for all pixels (x,y) in output image
    for all x’ in kernel
        outX(x,y) += input(x+x’,y)*kernel(x’)
for all pixels (x,y) in output image
    for all y’ in kernel
        out(x,y) += outX(x,y+y’)*kernel(y’)
```

Horizontal cost? O(input.width * input.height * kernel.width)

Vertical cost? O(input.width * input.height * kernel.height)

Total: O(input.width * input.height * (kernel.height+kernel.width))

Instead of: O(input.width * input.height * (kernel.height*kernel.width))

After a slide by Frédo Durand

Good news

Gaussians are separable too

See Assignment 4!

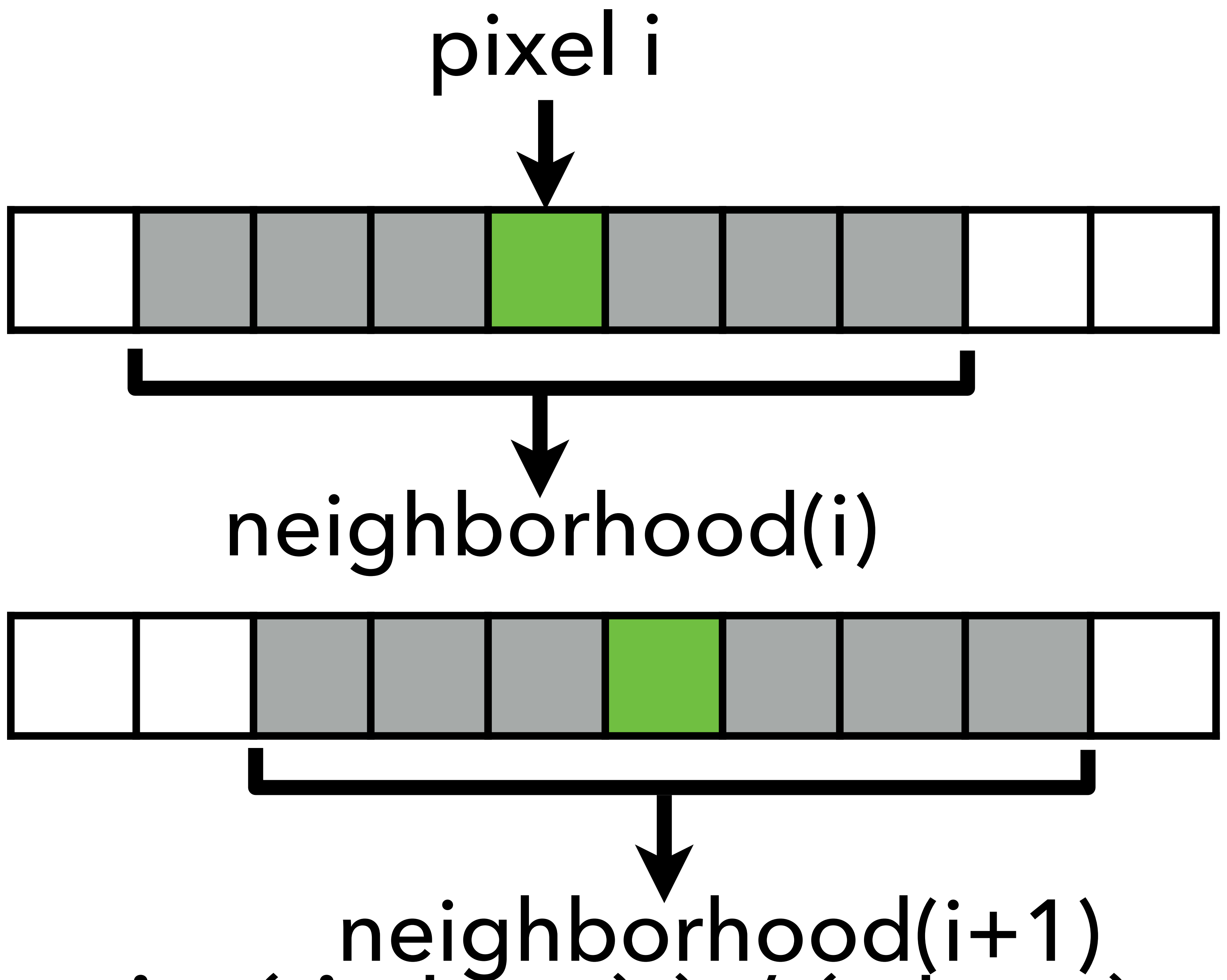
Box blur: Can we do even better?

Can we get even better asymptotic complexity?

Very large kernel sizes?

Box blur: Can we do even better?

Since 2D box is separable, let's focus on the 1D case



The neighborhoods of pixel i and pixel i+1 are very similar

In fact, they only differ by 2 pixels, so:

```
out(i+1) = out(i) + (in(i+k+1) - in(i-k+1))/(2k+1)
```

Asymptotically independent of kernel size, depends only on image size!

Box blur cost?

Naïve: O(input.width * input.height * (kernel.height*kernel.width))

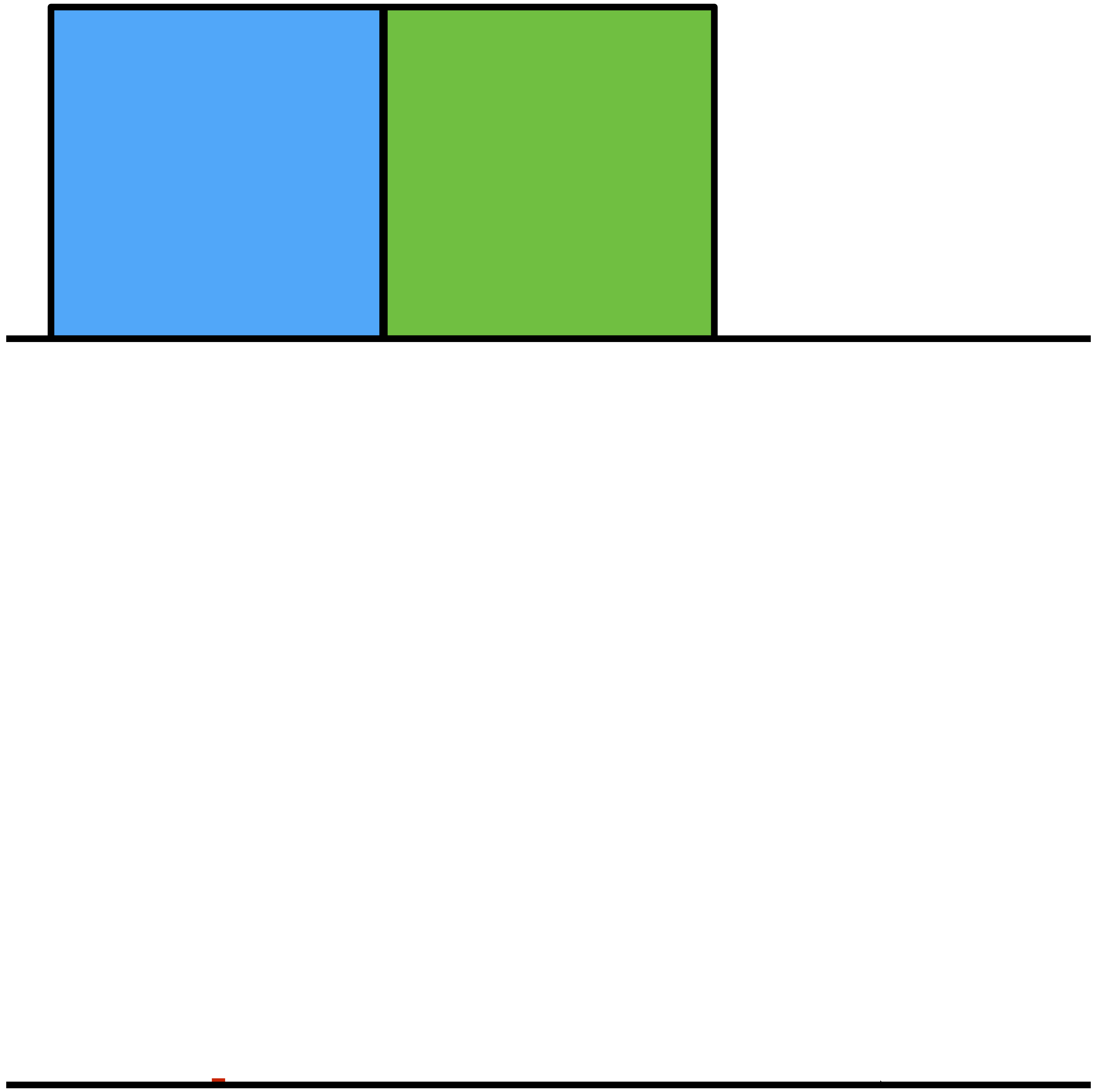
Separable: O(input.width * input.height * (kernel.height+kernel.width))

Incremental: O(input.width * input.height ~~+ (kernel.height+kernel.width)~~)

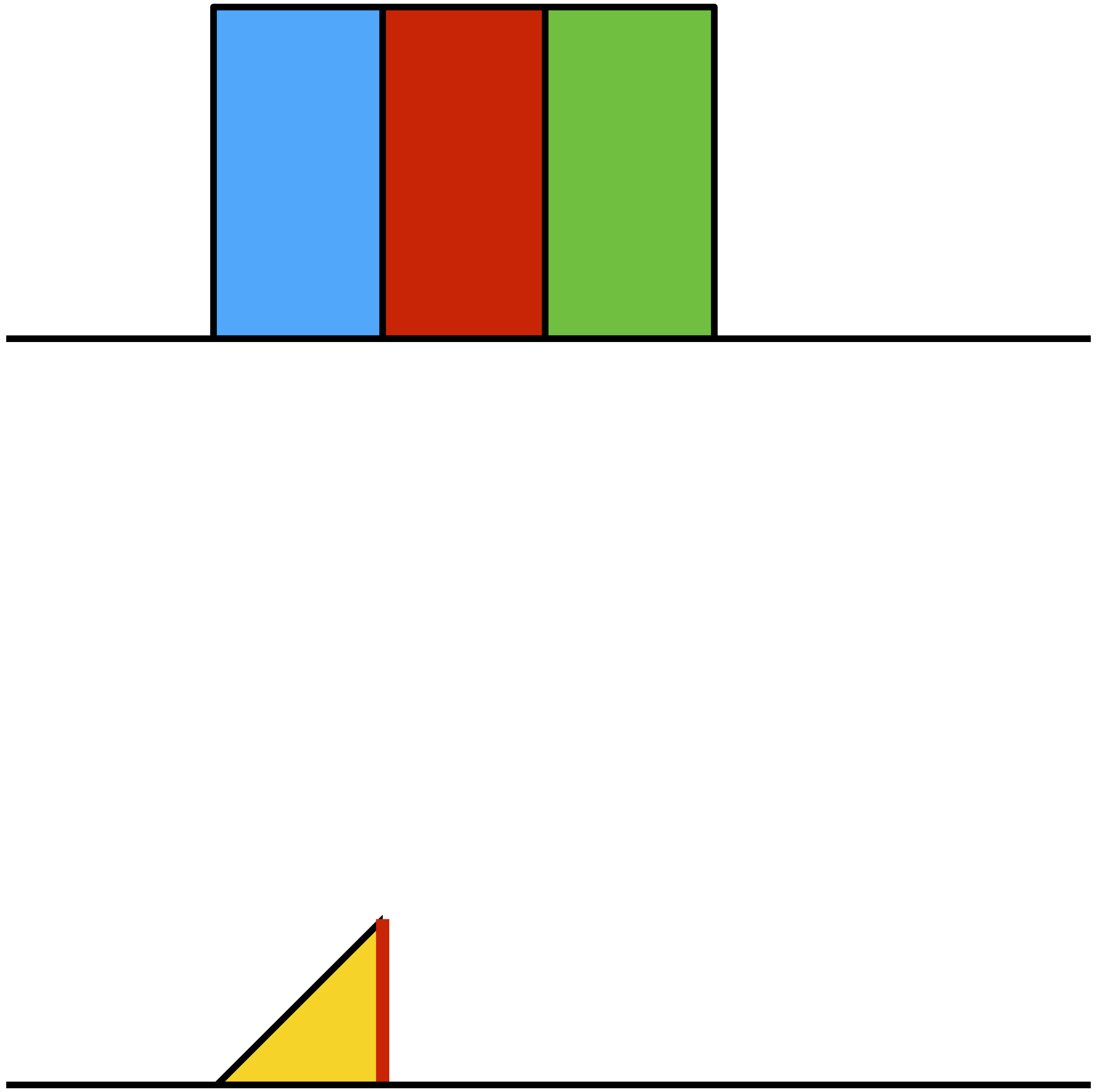
O(input.width * input.height)

After a slide by Frédo Durand

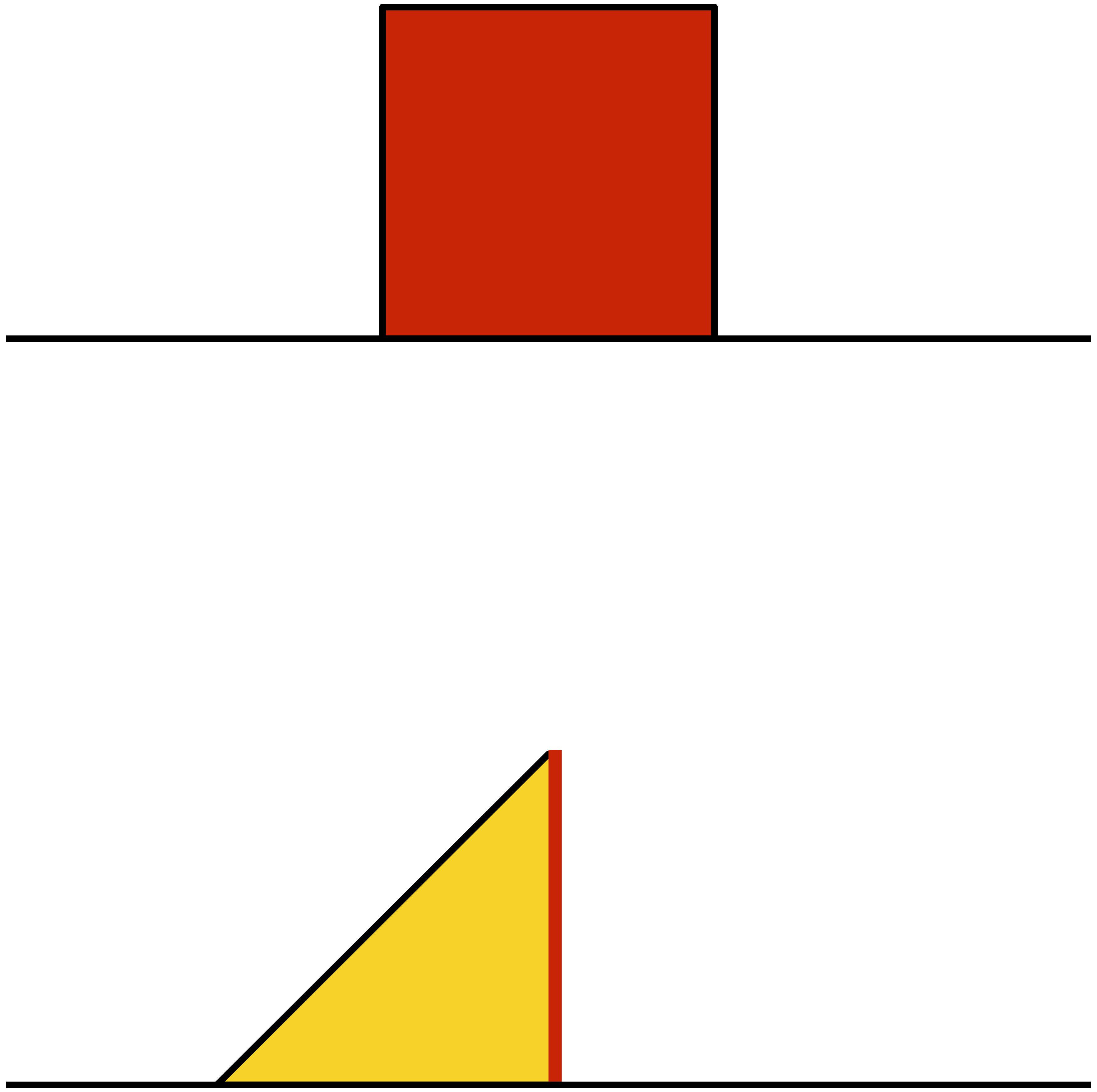
Repeated convolution



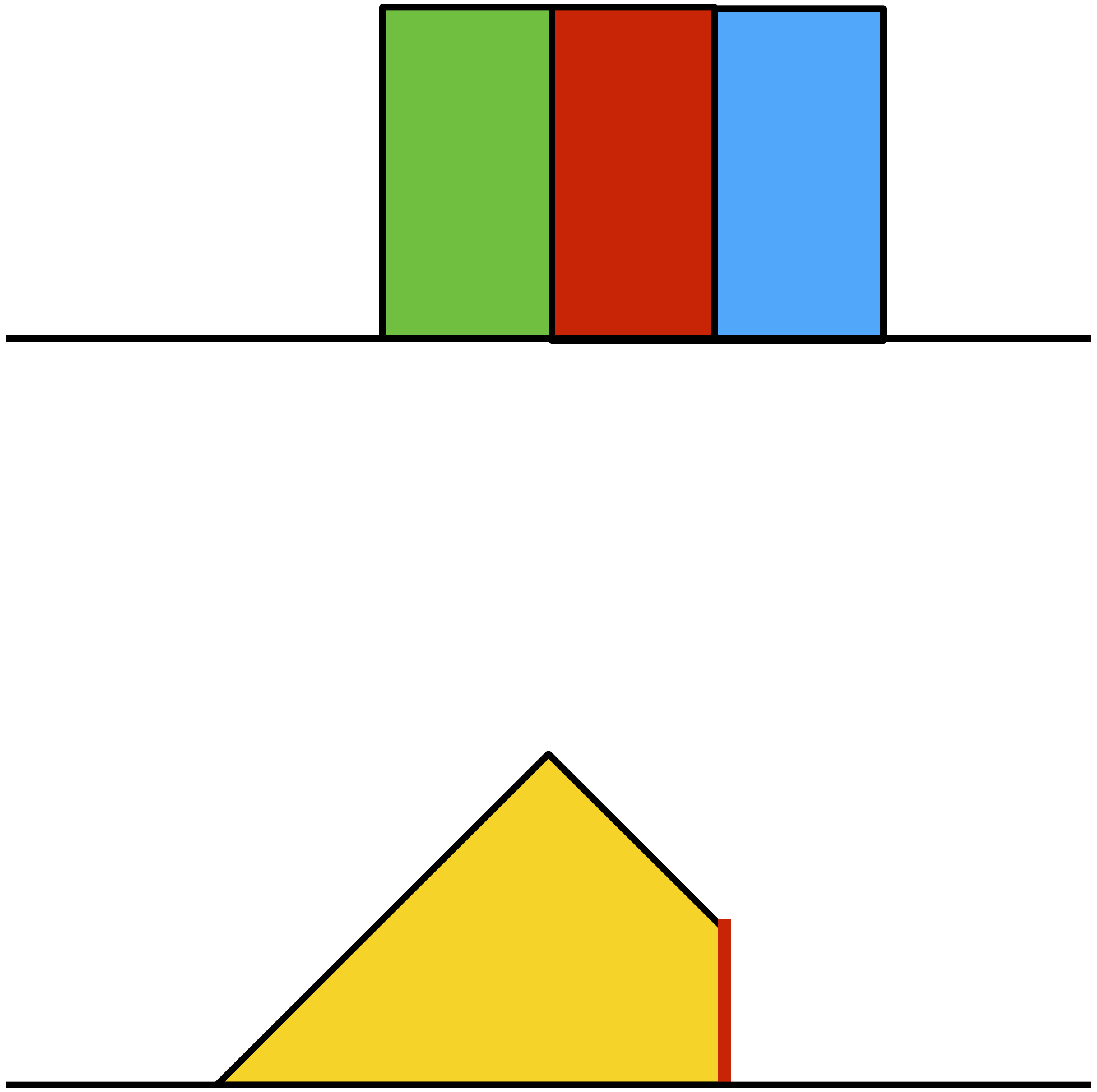
Repeated convolution



Repeated convolution

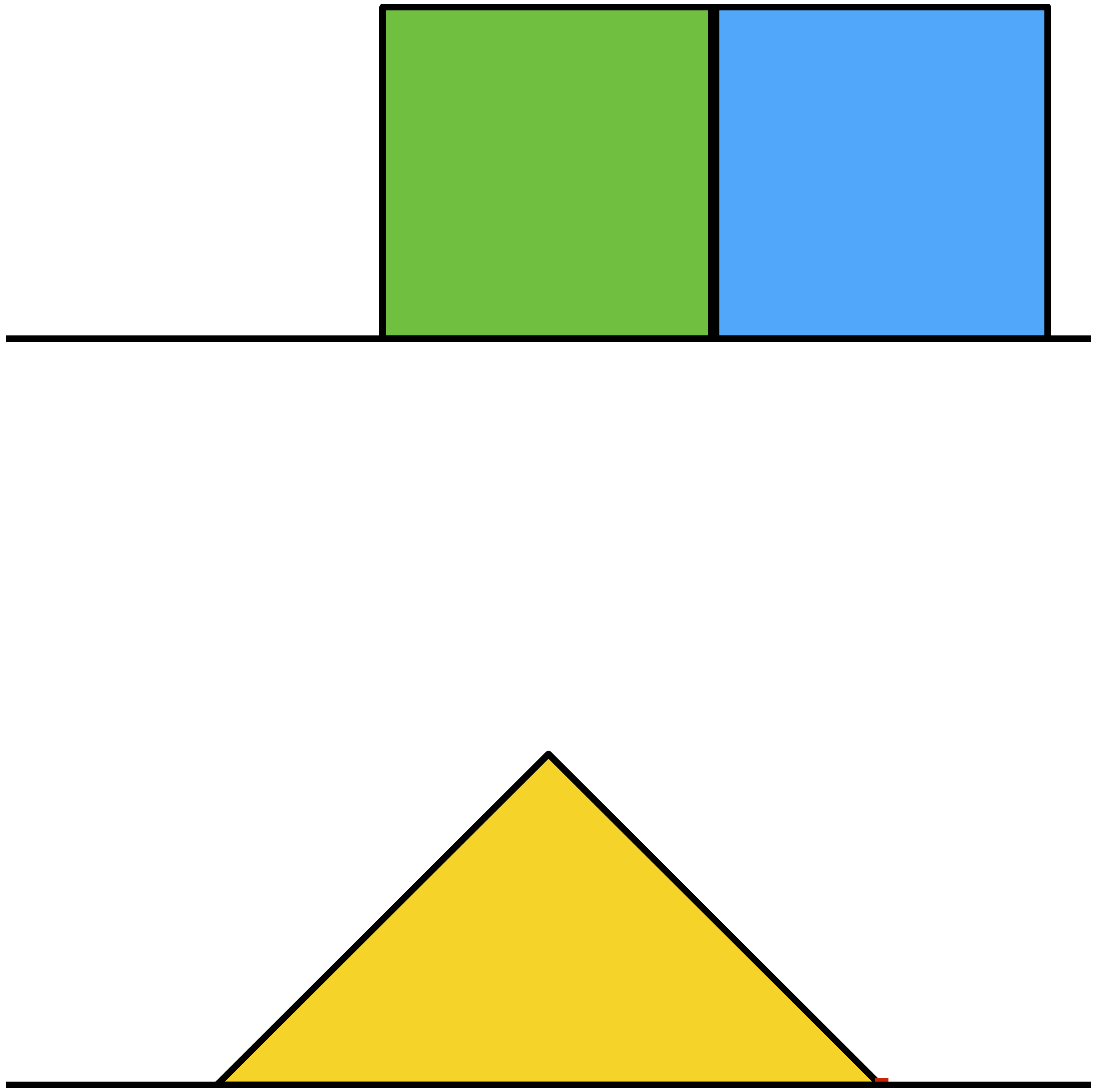


Repeated convolution



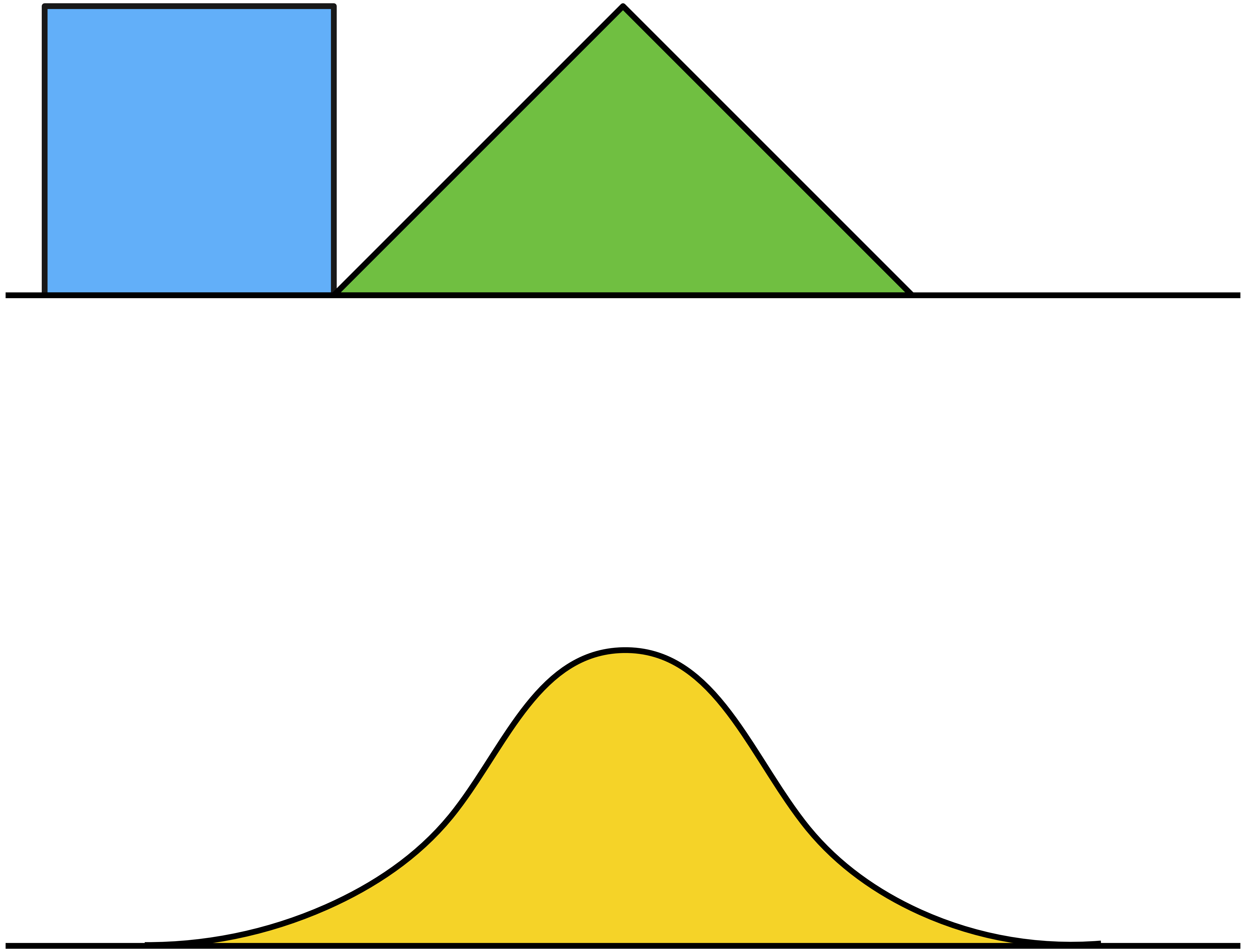
Repeated convolution

Convolution of two box kernels yields a tent kernel



Repeated convolution

Yet another convolution with a box yields piecewise quadratic



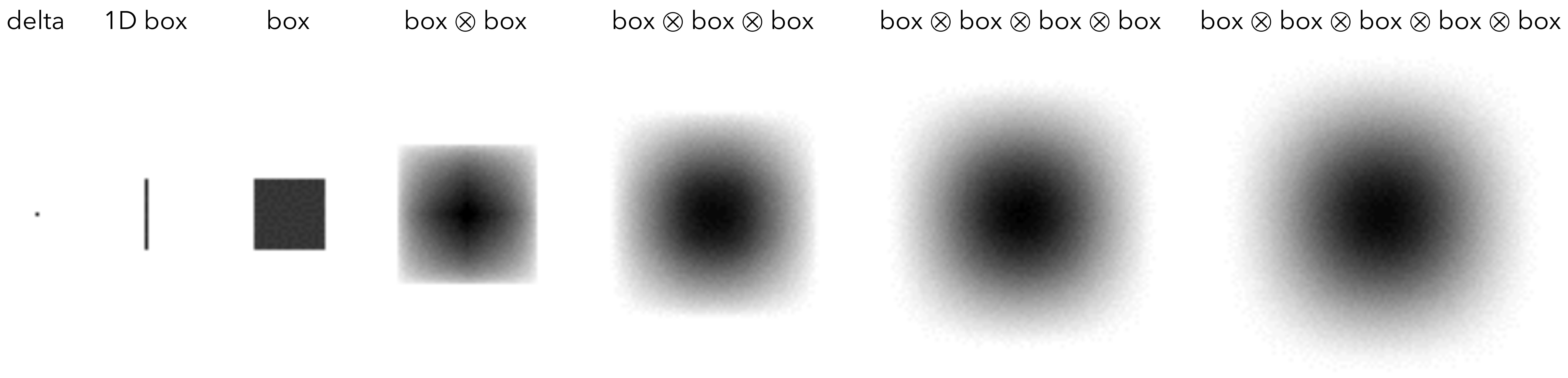
Repeated convolution

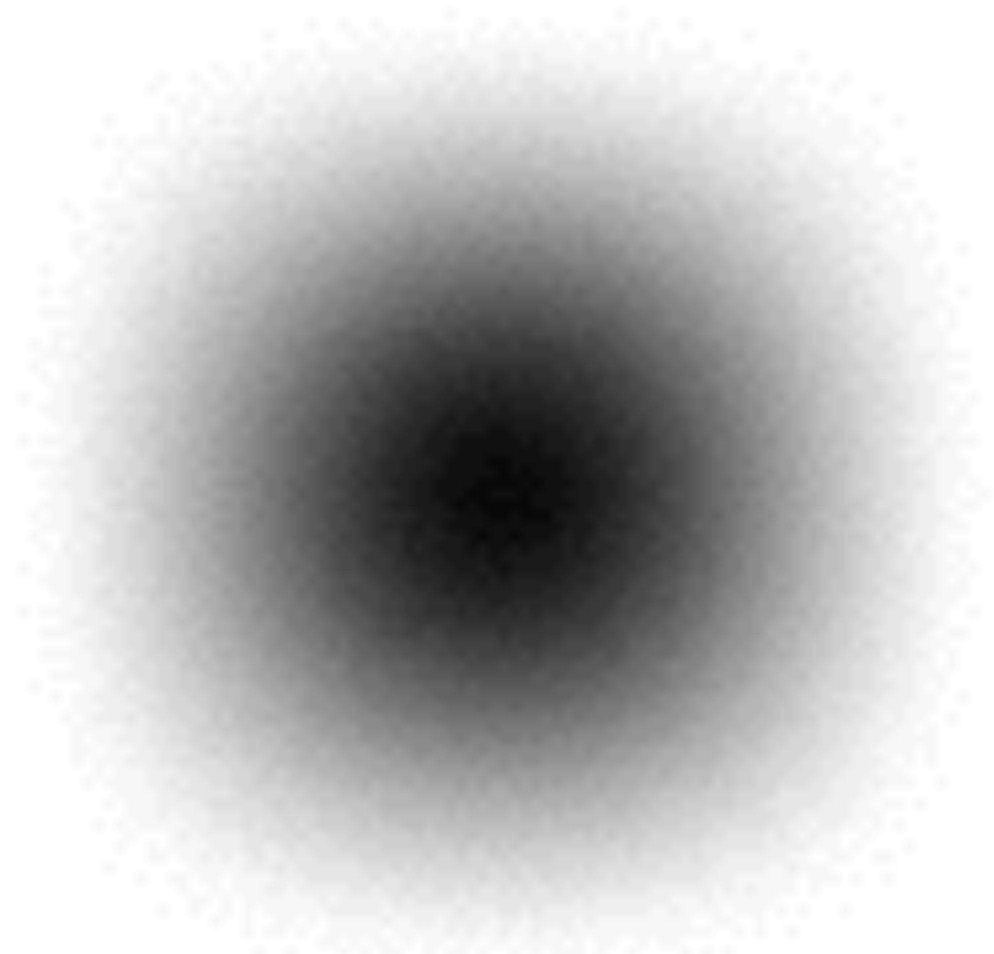
The pattern continues

- Box filtering the piecewise quadratic will yield a piecewise cubic, and so on.

Each time we make the kernel smoother

Taking this to the limit will yield a Gaussian





Photoshops' Gaussian

not a true Gaussian

Gaussian blur as multi-box blur

Can approximate Gaussian blur with several box blurs

Asymptotically independent of kernel size!

Assignment 4 extra credit

- what is Gaussian's σ for 5 box blurs?

Nitty-gritty stuff

Best input to debug convolution

Impulse

After a slide by Frédo Durand

Centering the kernel

Our images are defined with 0,0 in the upper left corner

Kernels are usually assumed to have origin at the center

After a slide by Frédo Durand

Normalization

As a rule of thumb, you want kernels to be normalized when you want the output to preserve the overall brightness of the image.

After a slide by Frédo Durand

Denoising from a single image

Denoising from 1 image

We can't take average over multiple images

After a slide by Frédo Durand

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

Here: Gaussian blur

After a slide by Frédo Durand

Gaussian blur

After a slide by Frédo Durand

Gaussian blur

Noise is mostly gone

But image is blurry

- duh!

After a slide by Frédo Durand

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 similar values

After a slide by Frédo Durand

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

After a slide by Frédo Durand

Bilateral filtering

Images are often piecewise constant with noise added

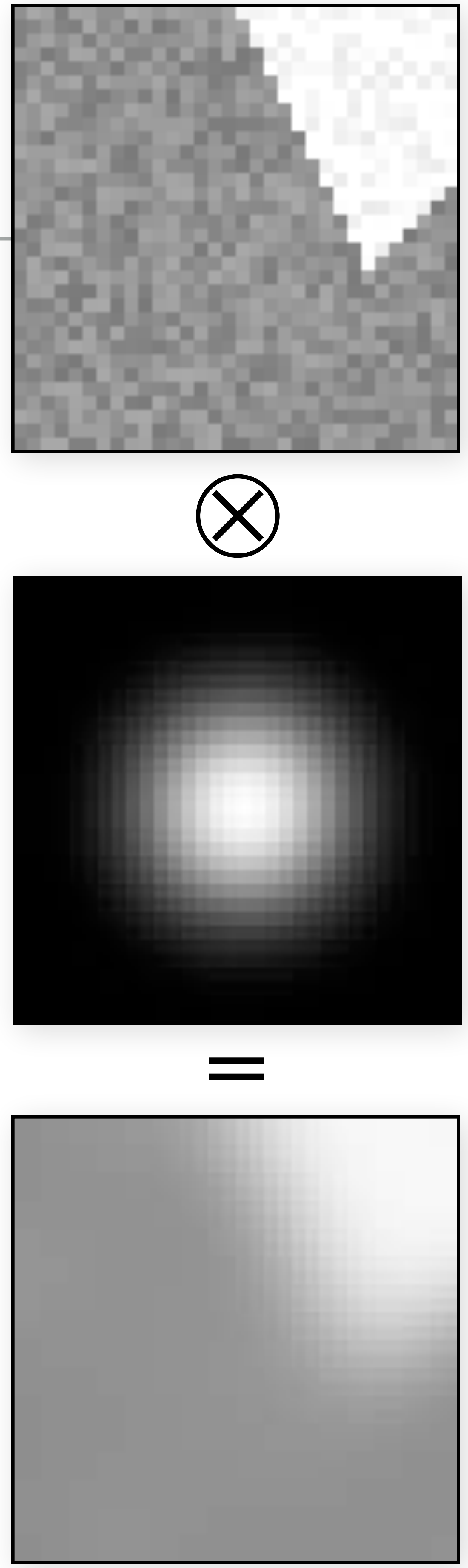
- Then nearby pixels are a different noisy measurement of the same value

Simply blurring doesn't work

- also blurs edges

We should blur only within each constant-colored region

- not across edges between regions



After a slide by Marc Levoy

[Tomasi and Manduchi 1998]

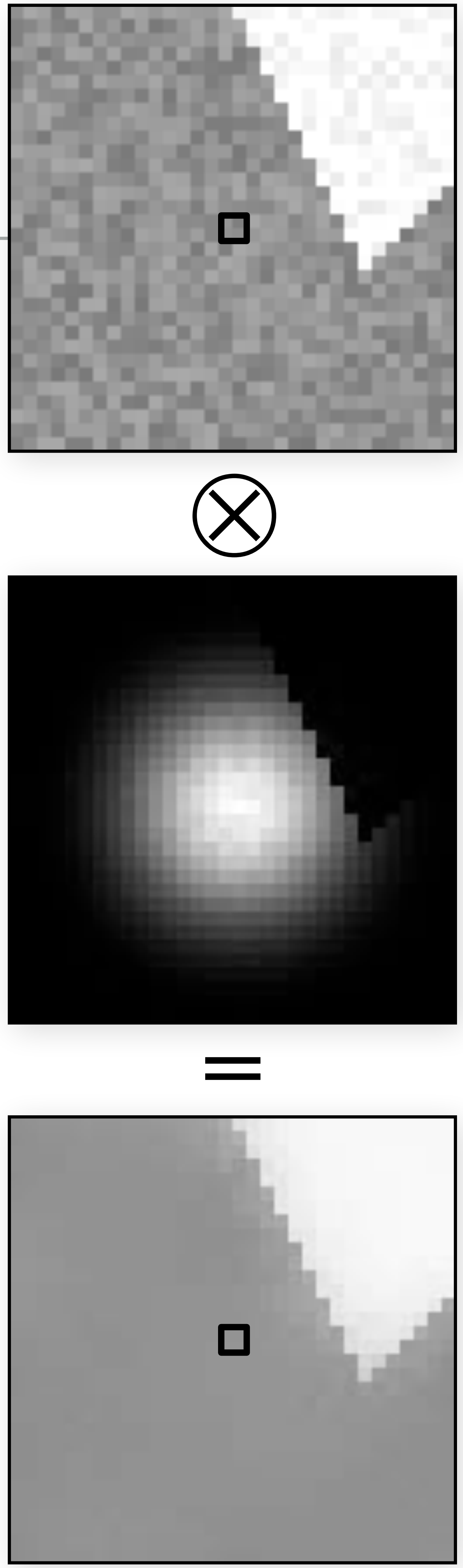
Bilateral filtering

If pixels are similar in intensity, they are probably from the same region of the scene

Perform a "convolution" where the weight applied to nearby pixels falls off with:

- increasing (x,y) distance from the pixel being blurred
- increasing intensity difference from the pixel being blurred

i.e. blur in *domain* and *range* dimensions!

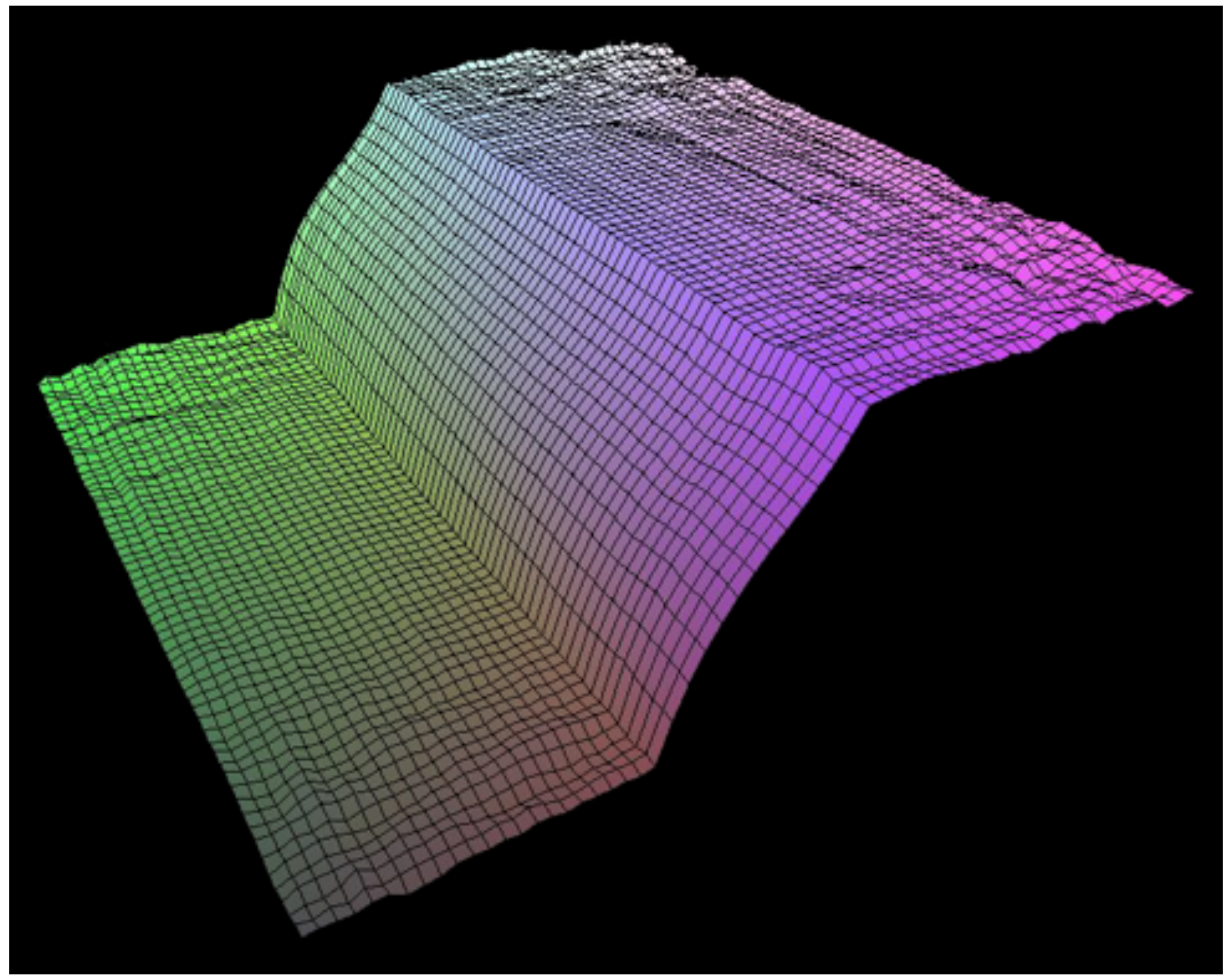


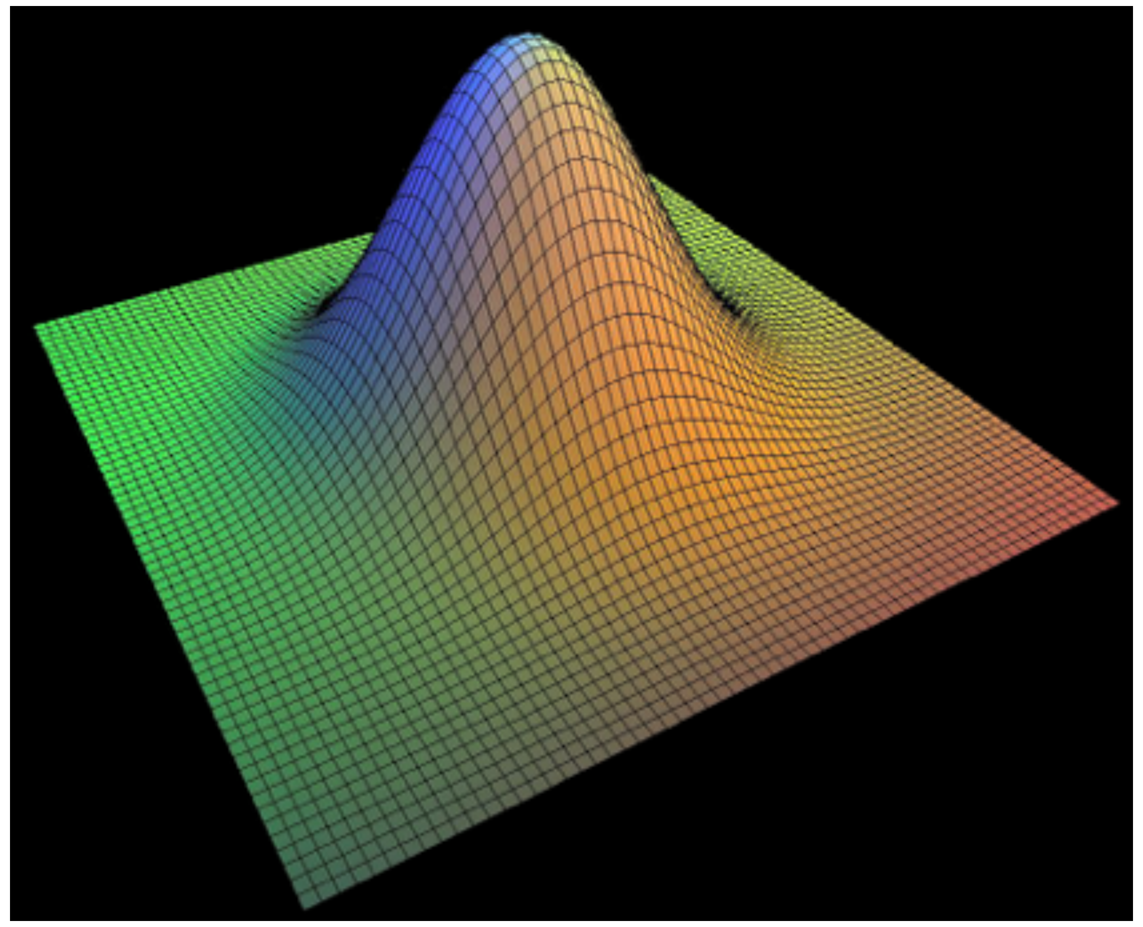
After a slide by Marc Levoy

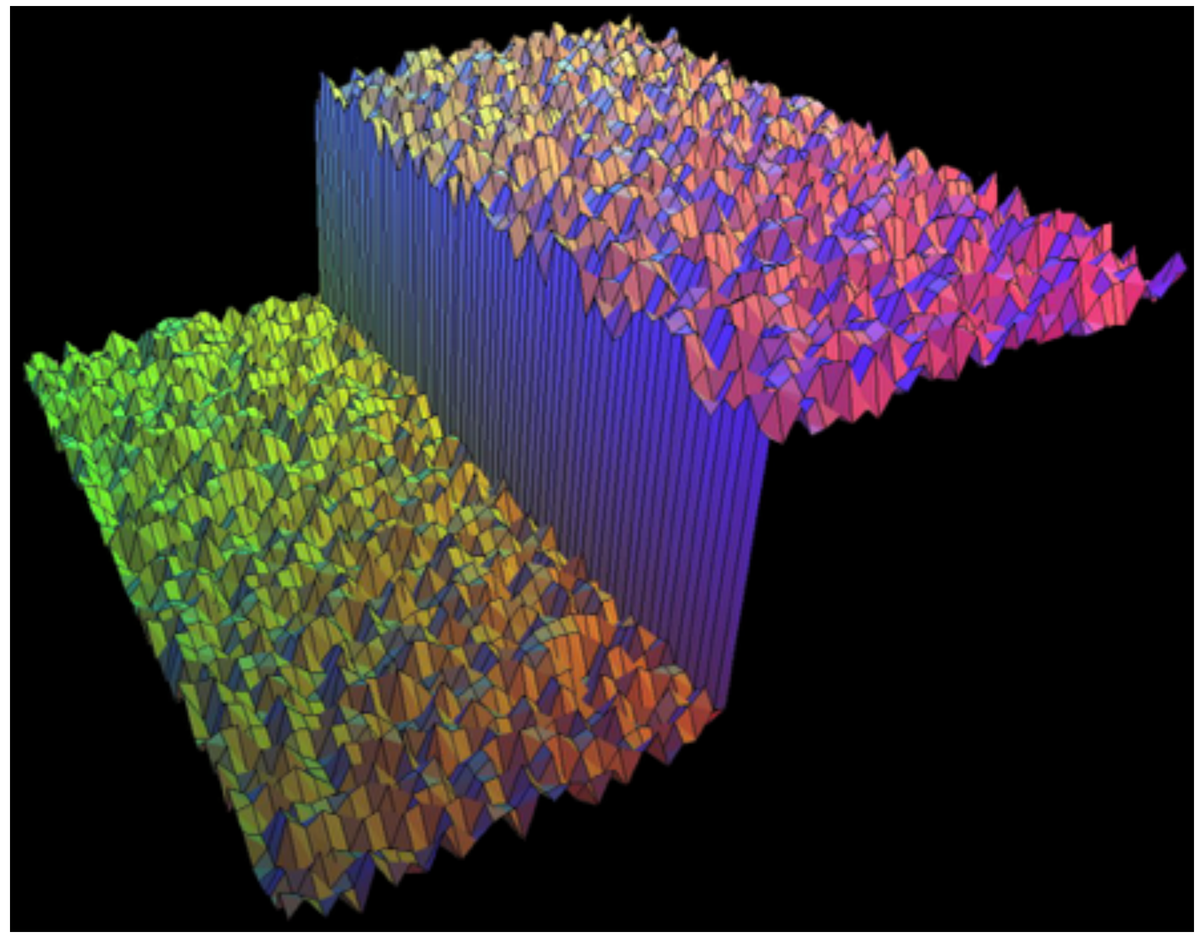
Start with Gaussian filtering

Here, input is a step function + noise

$$J = f \otimes I$$







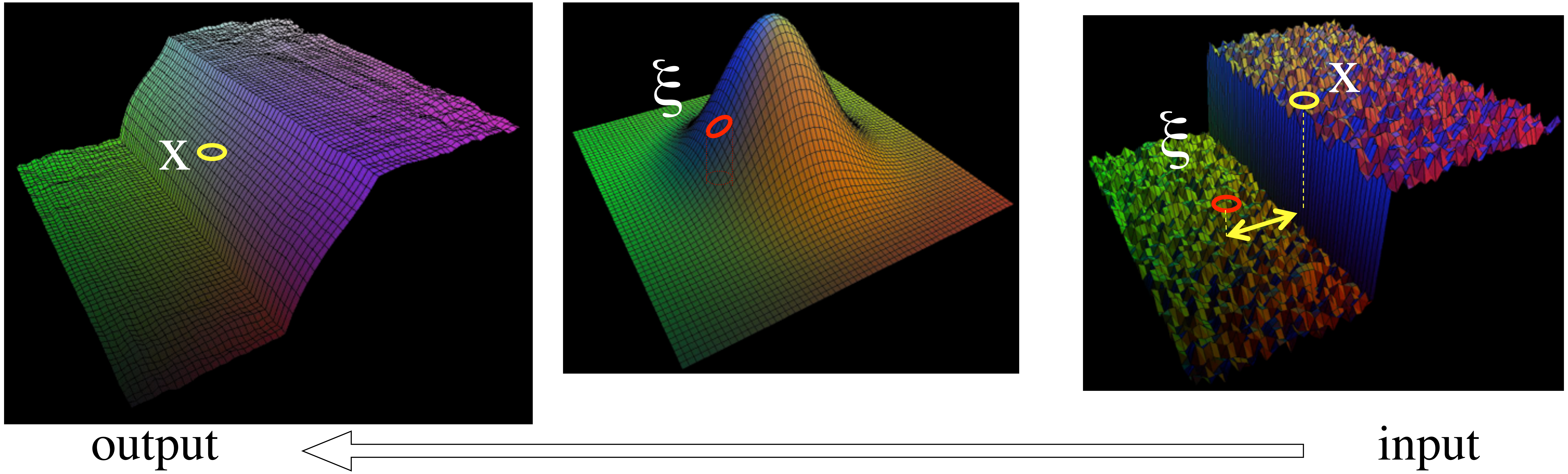
output ⟵ input

After a slide by Frédo Durand

Gaussian filter as weighted average

Weight of ξ depends on distance to x

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



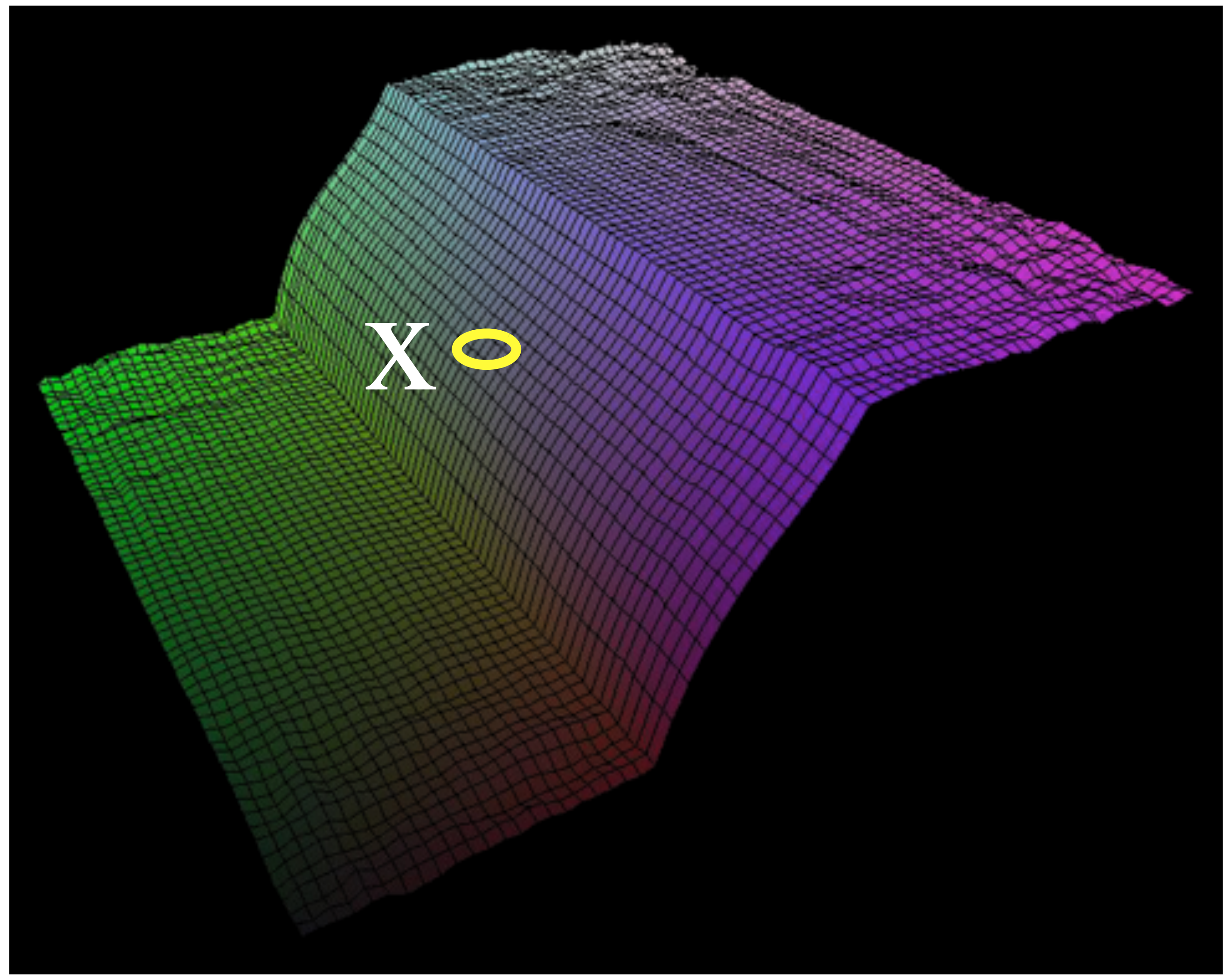
After a slide by Frédo Durand

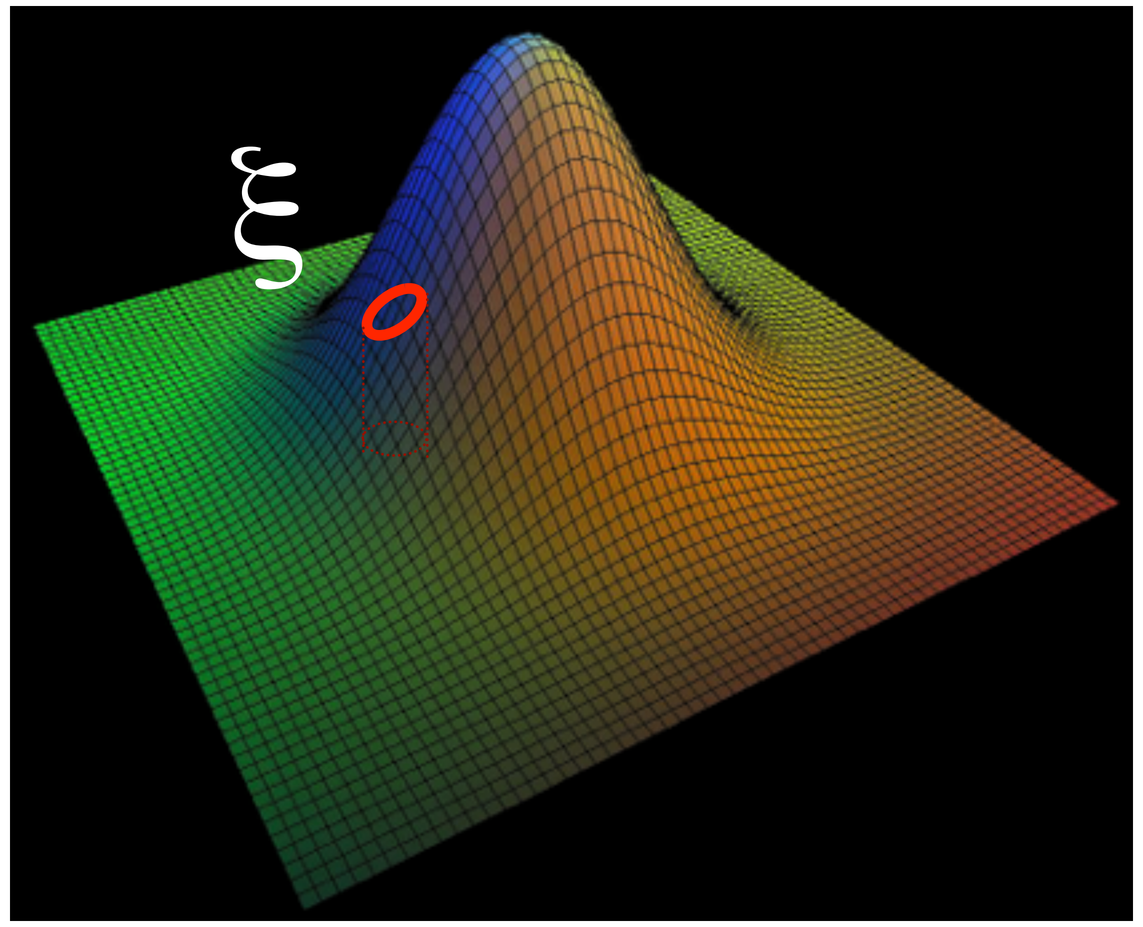
The problem of edges

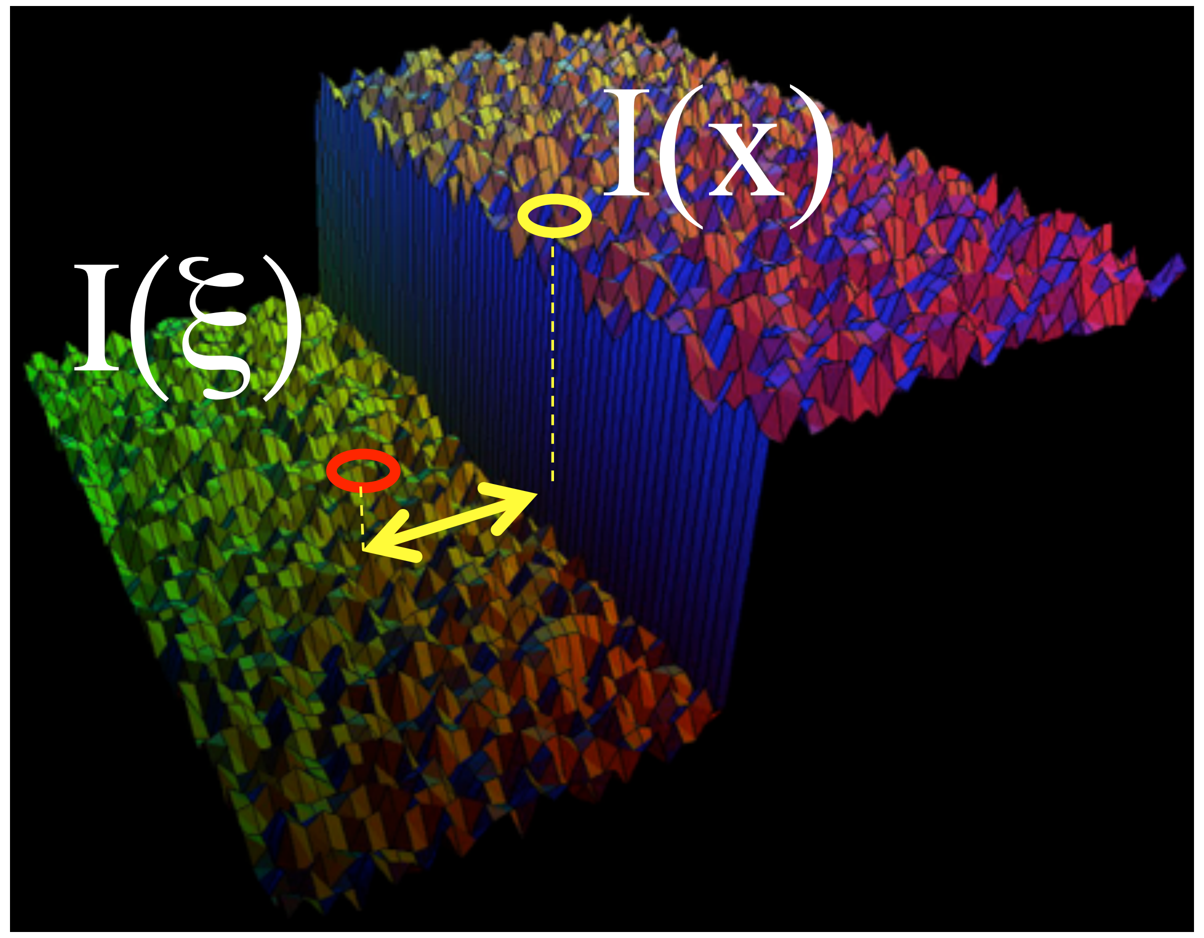
Here, $I(\xi)$ "pollutes" our estimate $J(x)$

It is too different

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







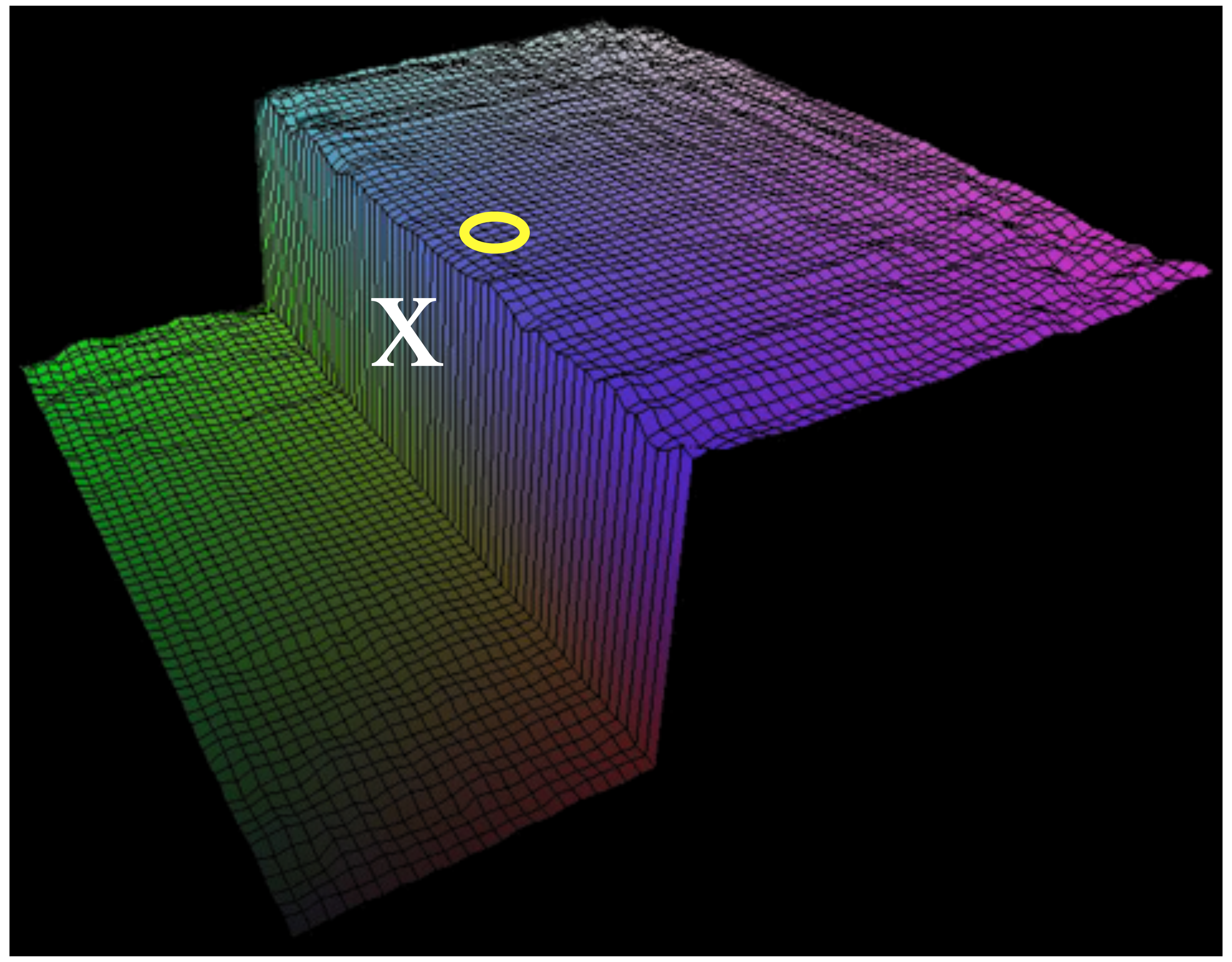
output ⟵ input

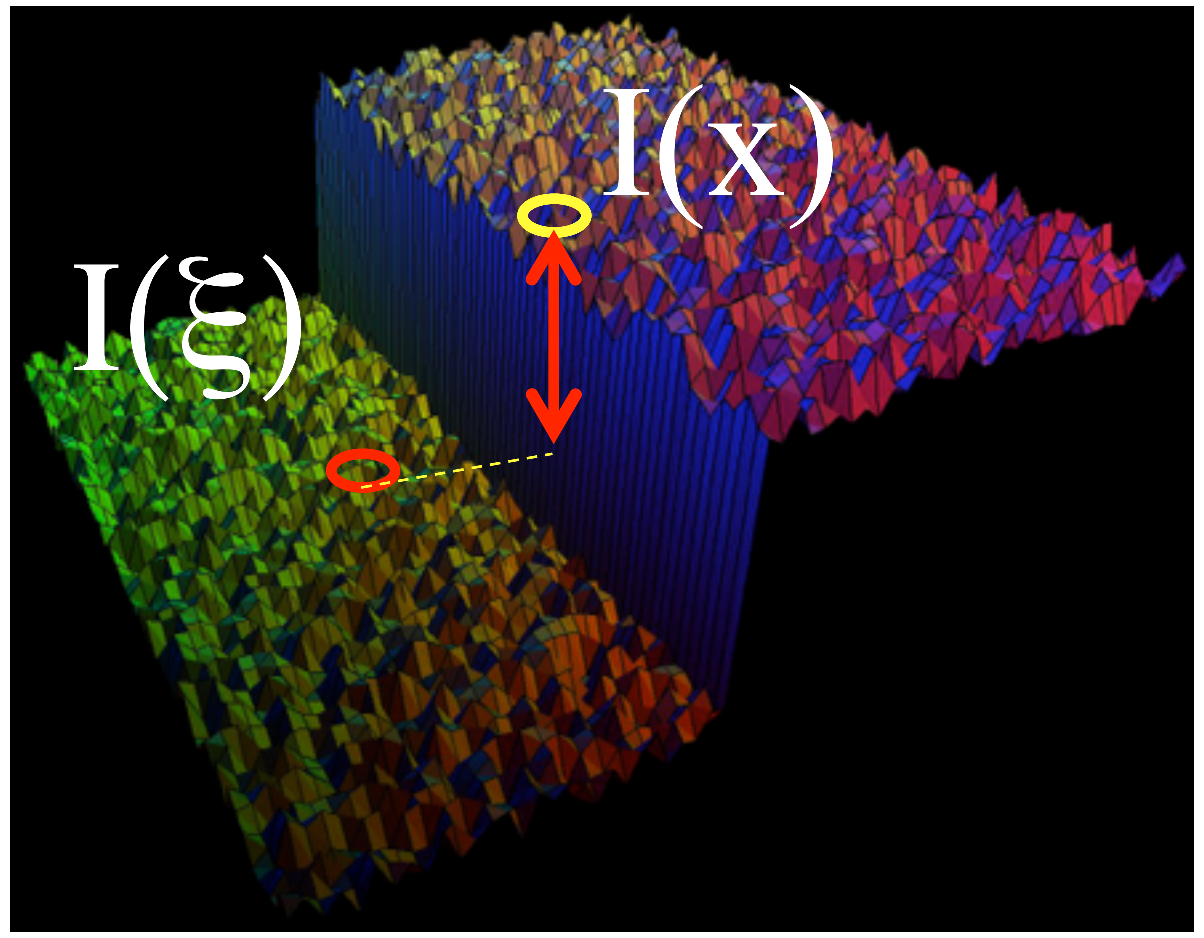
After a slide by Frédo Durand

Principle of Bilateral filtering

Penalty g on the intensity difference

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





output ⟸ input

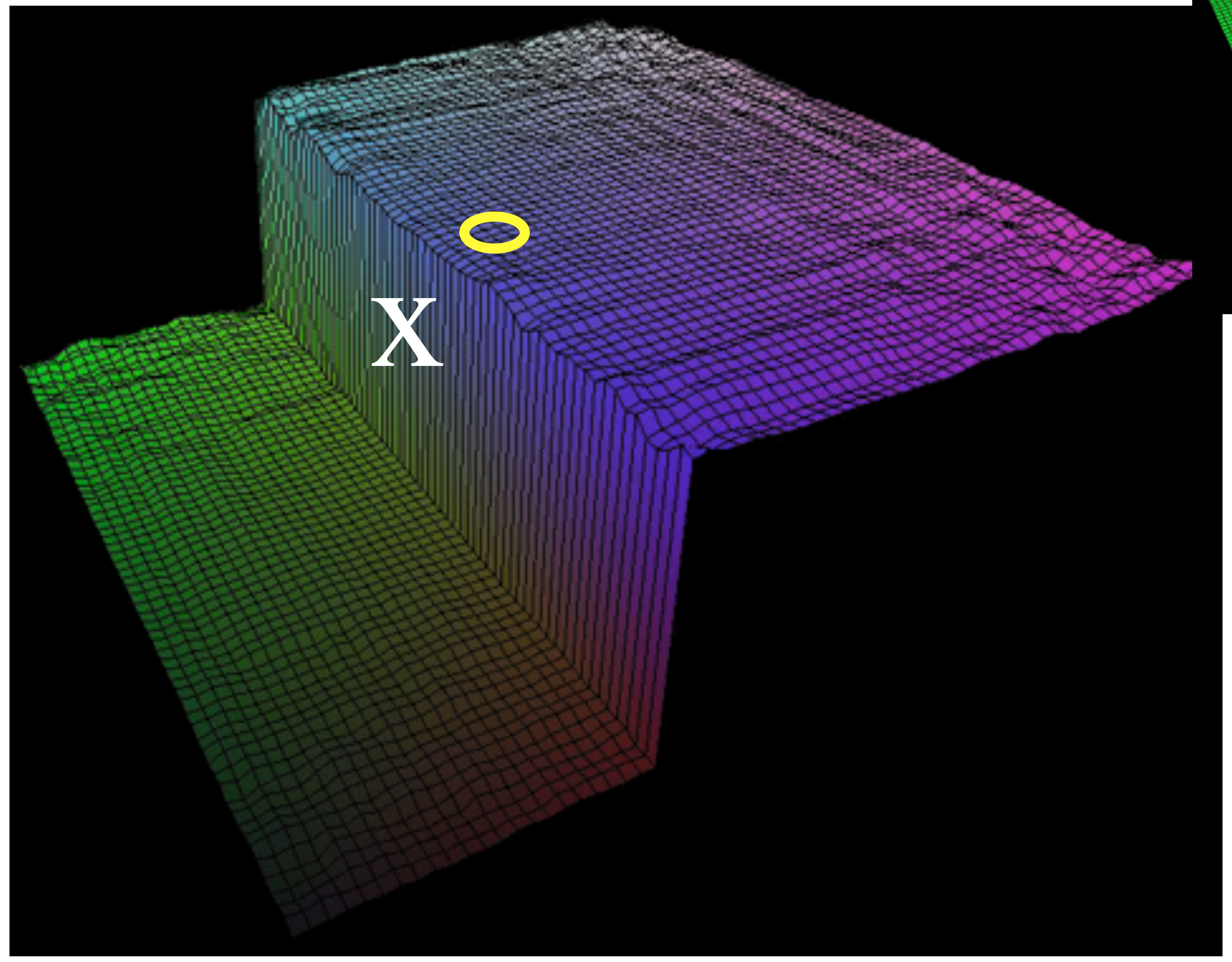
After a slide by Frédo Durand

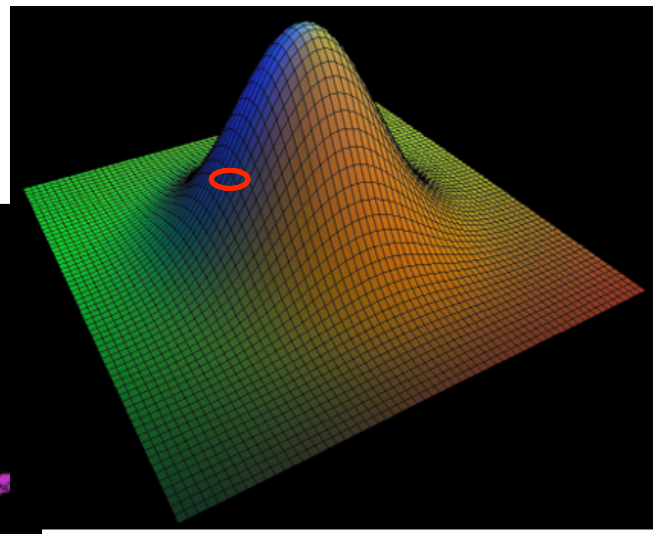
Bilateral filtering

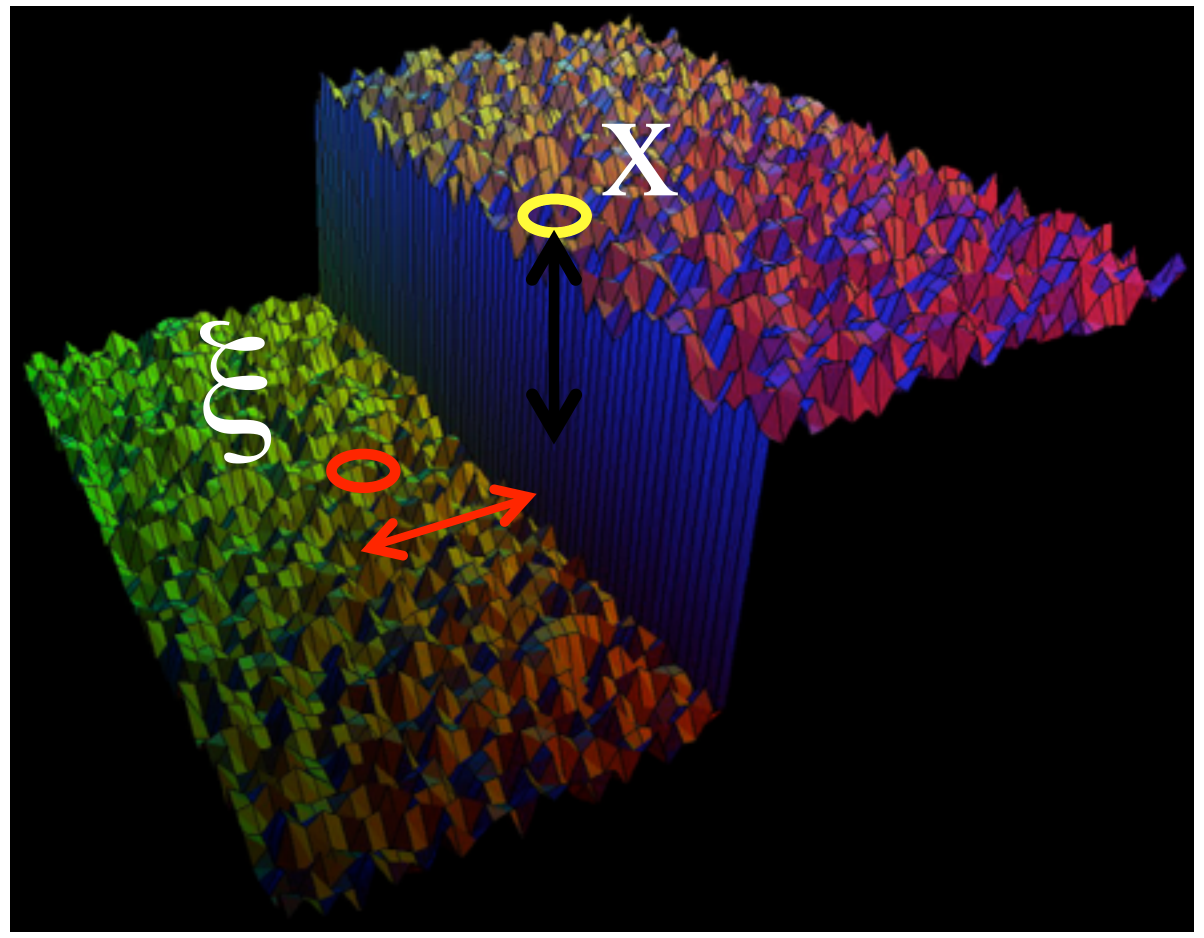
[Tomasi and Manduchi 1998]

Spatial Gaussian f

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







output ⟵ input

After a slide by Frédo Durand

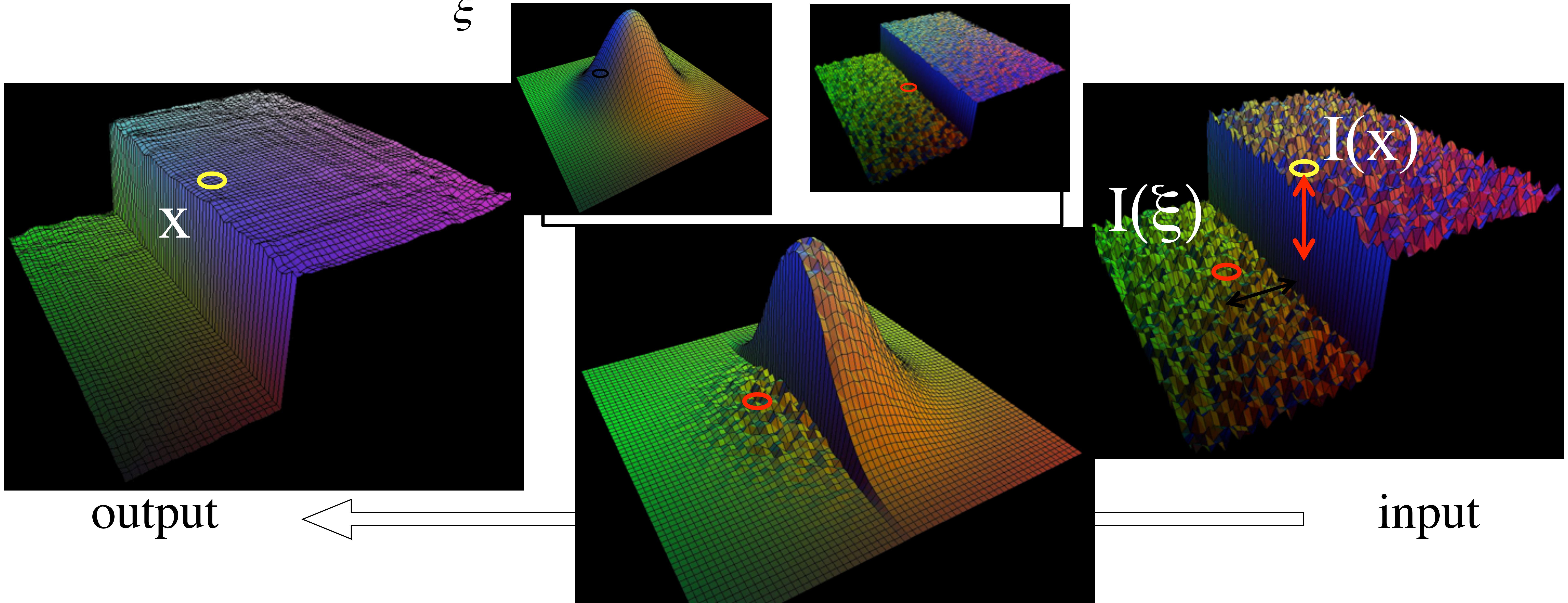
Bilateral filtering

[Tomasi and Manduchi 1998]

Spatial Gaussian f

Gaussian g on the intensity difference

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



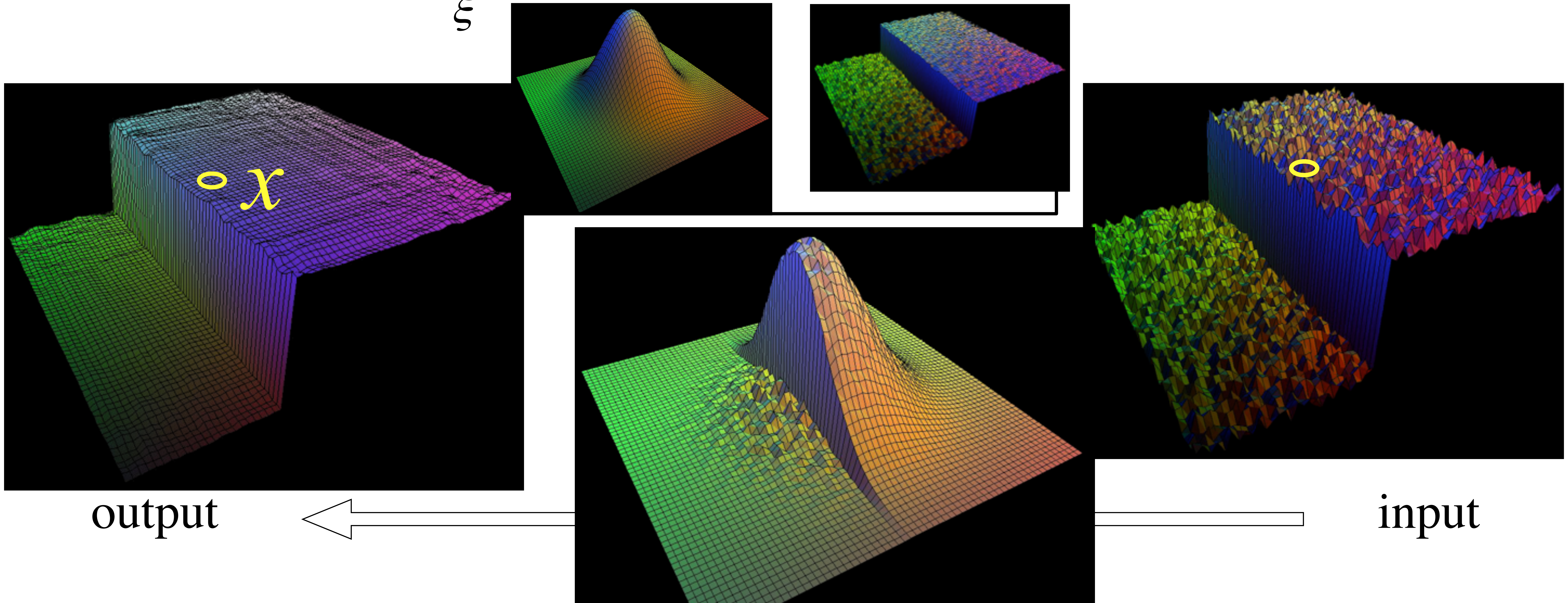
After a slide by Frédo Durand

Normalization factor

[Tomasi and Manduchi 1998]

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

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

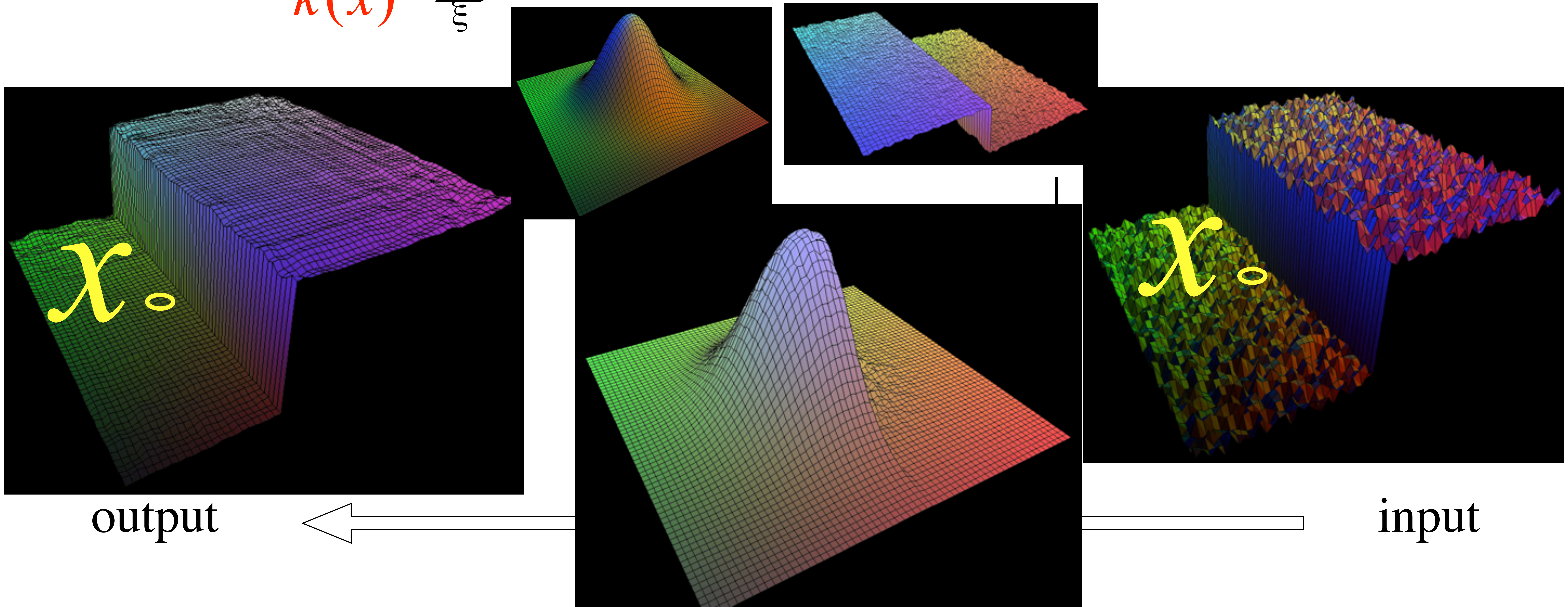


After a slide by Frédo Durand

Bilateral filtering is non-linear [Tomasi and Manduchi 1998]

The weights are different for each output pixel

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



output

input

After a slide by Frédo Durand

Bilateral filter

Noisy input

After bilateral filter

After a slide by Frédo Durand

Can we do better?

Noisy input

After bilateral filter

chroma noise

After a slide by Frédo Durand

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

After a slide by Frédo Durand

Normal RGB Bilateral filter

Noisy input

After bilateral filter

After a slide by Frédo Durand

YUV Bilateral filter

Noisy input

After YUV bilateral filter

After a slide by Frédo Durand

Comparison

Noisy input

Bilateral filter

YUV bilateral filter

After a slide by Frédo Durand

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

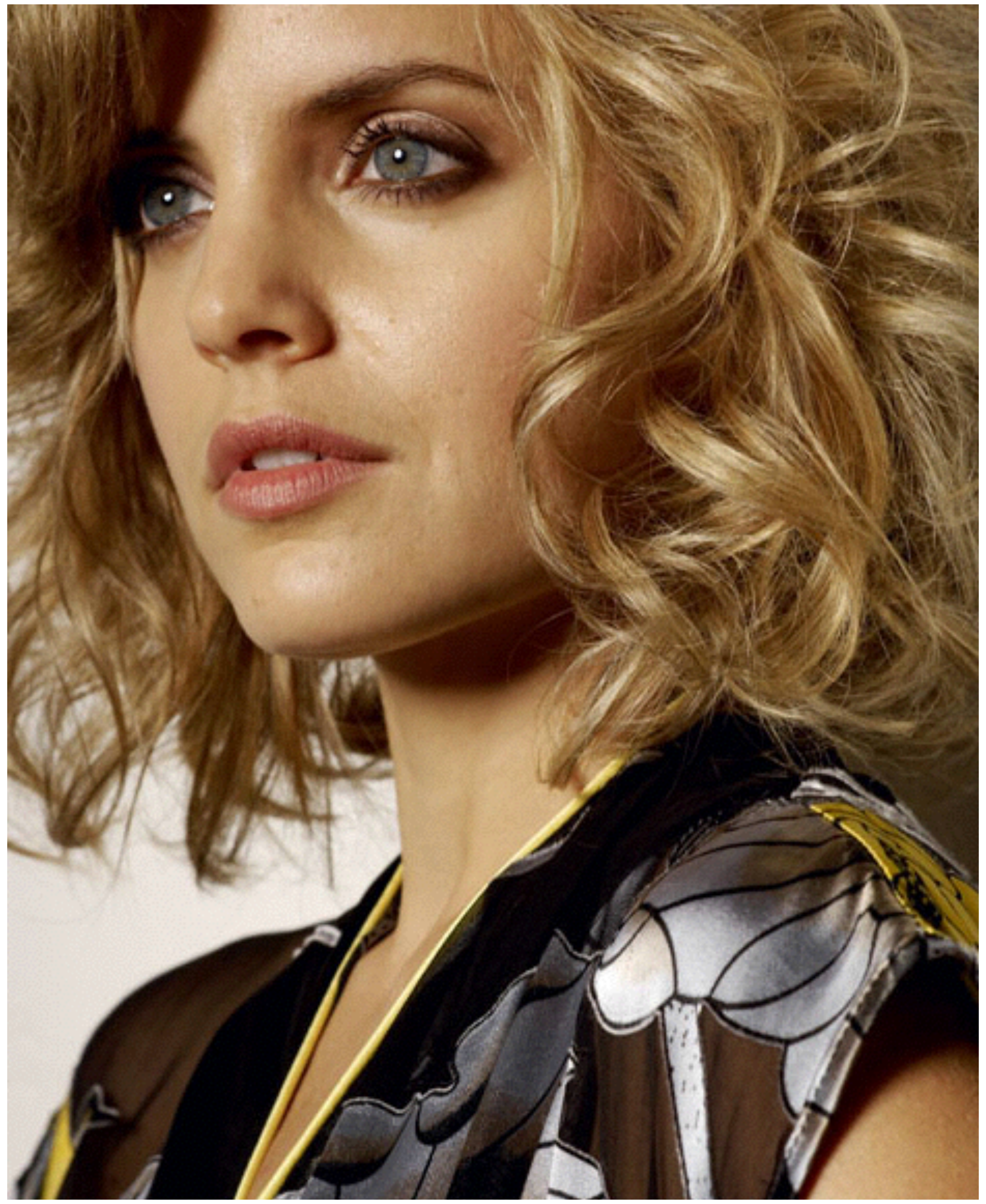
- More in future lectures
- In particular, tone mapping for contrast reduction and high-dynamic-range imaging

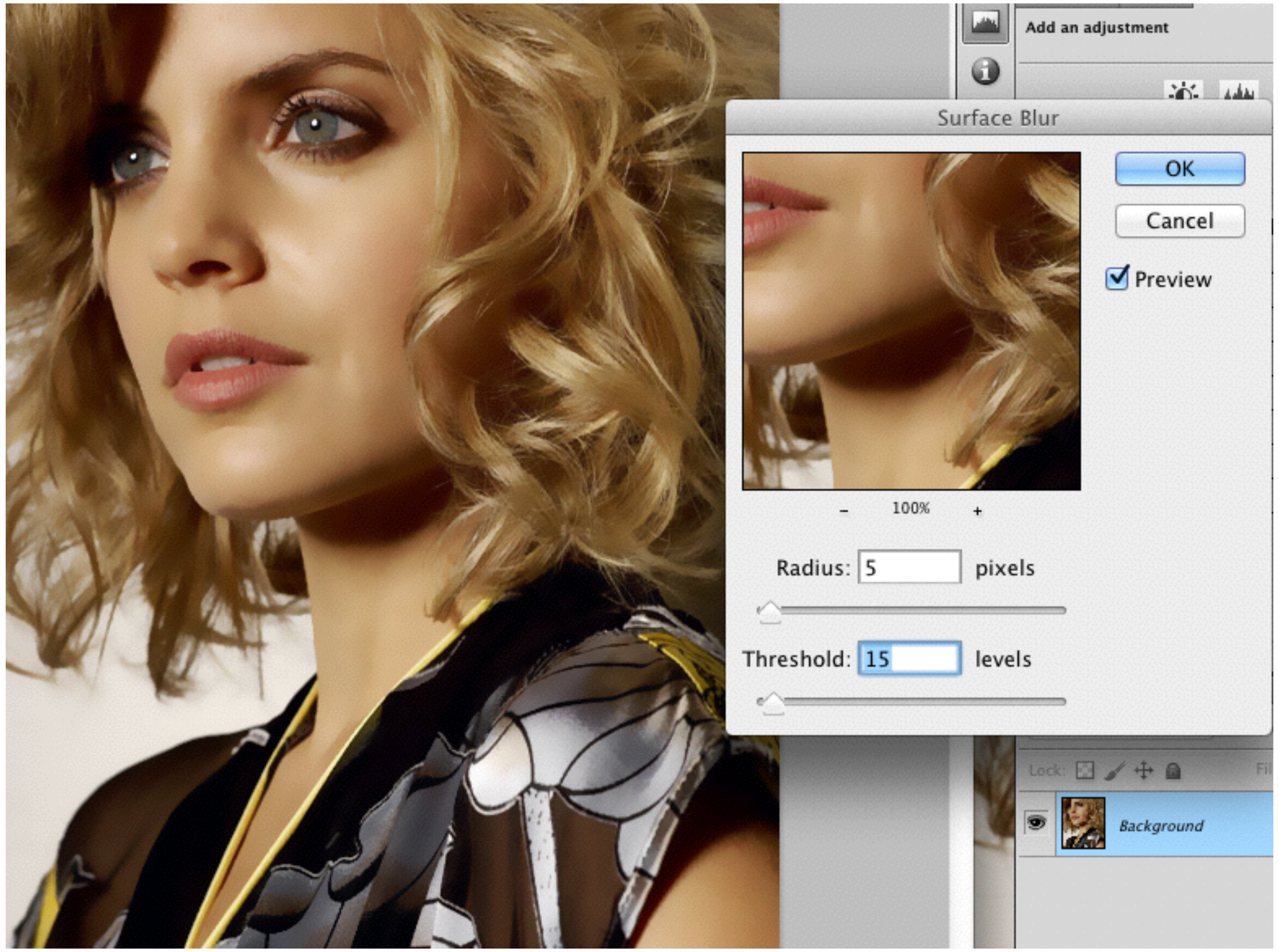
After a slide by Frédo Durand

Photoshop surface blur

Note the radius and threshold controls

- same as σ_{domain} and σ_{range}





After a slide by Frédo Durand

Assignment 4

Assignment 4

Convolution

Separable

Unsharp mask

Gradient

Denoising

YUV denoising

Other approaches to denoising

Denoising

Bayesian coring in the wavelet domain

- Simoncelli & Adelson

Big heuristics

- BM3D

NL means

- Buades et al.
- Bilateral in the space of patches

Statistics of natural images

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/

Slide credits

Frédo Durand