Lecture 1: Introduction

EECS 442: Computer Vision

UNIVERSITY OF MICHIGAN

Course staff

Sarah Jabbour

GSI

Yiming Dou

GSI

Interacting with us

- In person office hours (we’ll have one over Zoom)
- If you have questions during lecture, either raise your hand or post it as a Zoom message.
- Ask homework and class questions on Piazza
- Homework submission via Gradescope

Course website

EECS 442: Computer Vision

Instructor: Andrew Owens   Fall 2023

Schedule | Staff | Course info · Piazza | Canvas | Gradescope | Zoom | Recordings

Schedule

Note: Welcome to EECS 442. Unfortunately, 2 of the 5 optional discussion sections that were originally in the course calendar will not be offered, since they were listed in error (namely, the Friday 10:30am and Thursday 3:30pm sections). Please see here for the sections that are still offered. If this affects you, we apologize for the inconvenience. You are welcome to attend any of the 3 remaining sections, regardless of whether you are officially enrolled in them. Please note that discussion sections largely review course material that was covered during lecture, and thus attendance is completely optional. We will also record them, for those who are unable to attend. We apologize for any inconvenience that this caused, and hope that those who wanted to attend can still do so. Please note that the lecture will still take place at the usual time.

Lecture	Date	Topic	Materials	Assignments
Lec. 1	Mon, Aug. 28	**Introduction** About the course Neighborhood filtering Blurring Gradient filters	• Torralba et al. manuscript: Signal processing • Reminder: recording can be found in the lecture capture system	ps1 out (filtering)
Lec. 2	Wed, Aug. 30	**Filtering** Convolution and cross-correlation Edge detection Nonlinear filtering	• Torralba et al.: Signal processing • Szeliski Chapter 3.2, page 111	
Sec. 1	Fri, Sep. 1	**Linear algebra and filtering**	• Colab Notebook	
	Mon, Sep. 4	**No class - Labor Day**		

https://www.eecs.umich.edu/courses/eecs442-ahowens/fa23/

Temporary mirror: https://www.andrewowens.com/eecs442-fa23-mirror/

Grading

- Assignments (70%)
- Final project (30%)
- No exams

Assignments

- Weekly problem sets (≈ 9) with equal weight
- Due each Wednesday at midnight
    - PS1 out at midnight tonight, due on 9/13.
- You'll have **5 late days**
    - Once they're used up, 30% penalty per day
- Assignments be done independently.
    - Encouraged to discuss the problems
    - Programming/writing should all be yours

Project

- Some options:
  - Choose from a list of project topics.
  - Implement a recent paper
  - Your own idea (recommended)
- Small groups: 1-4 people suggested.
- Deliverables:
  1. Project proposal (early November)
  2. Short presentation (beginning of finals period)
  3. Writeup (middle of finals period)

Suggested background

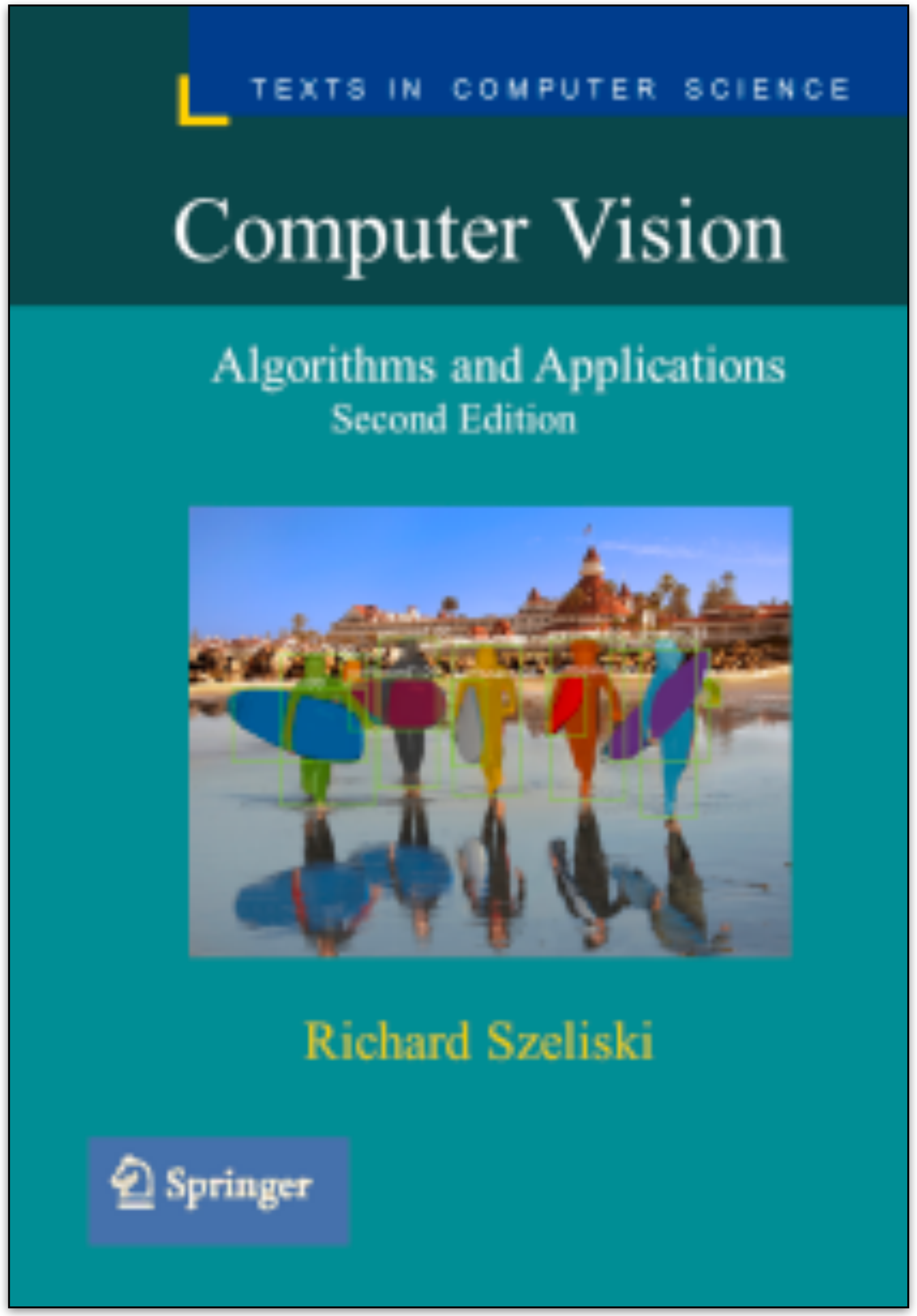
- Linear algebra + calculus (required)
- Math review during the next two sections.
- Python programming with numerical libraries like numpy

Discussion sections

- **Important:** unfortunately, two sections are cancelled.
  - Canceled sections: Thu. 3:30pm, Fri. 10:30am
- You can attend any section, and attendance is optional. We'll record it, too.
- What's covered? Mostly review material from lecture, programming tutorials, project help.

Time	Place
Thu 4:30-5:30pm	1303 EECS
Fri 12:30-1:30pm	107 GFL
Fri 1:30-2:30pm	1017 DOW

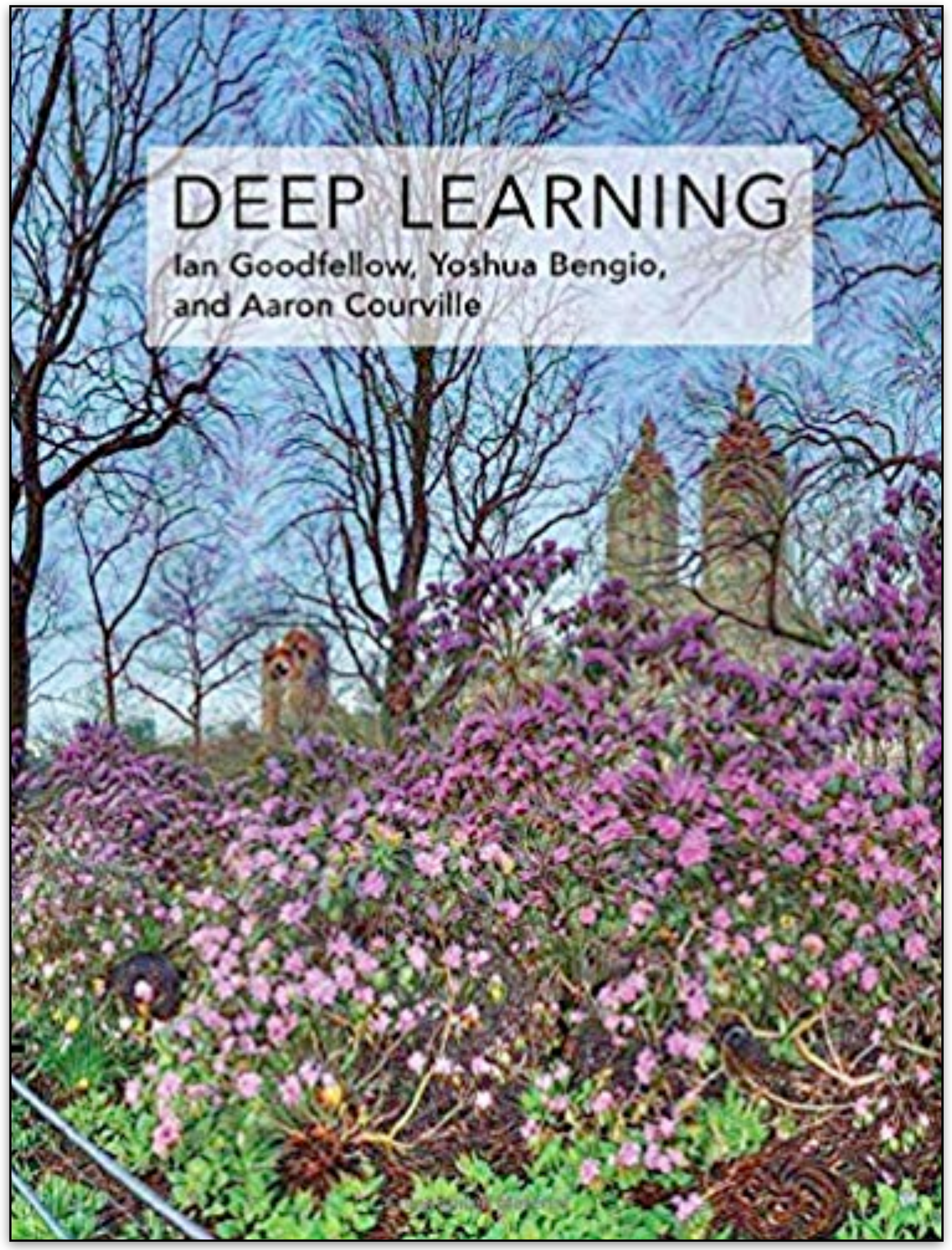
Readings



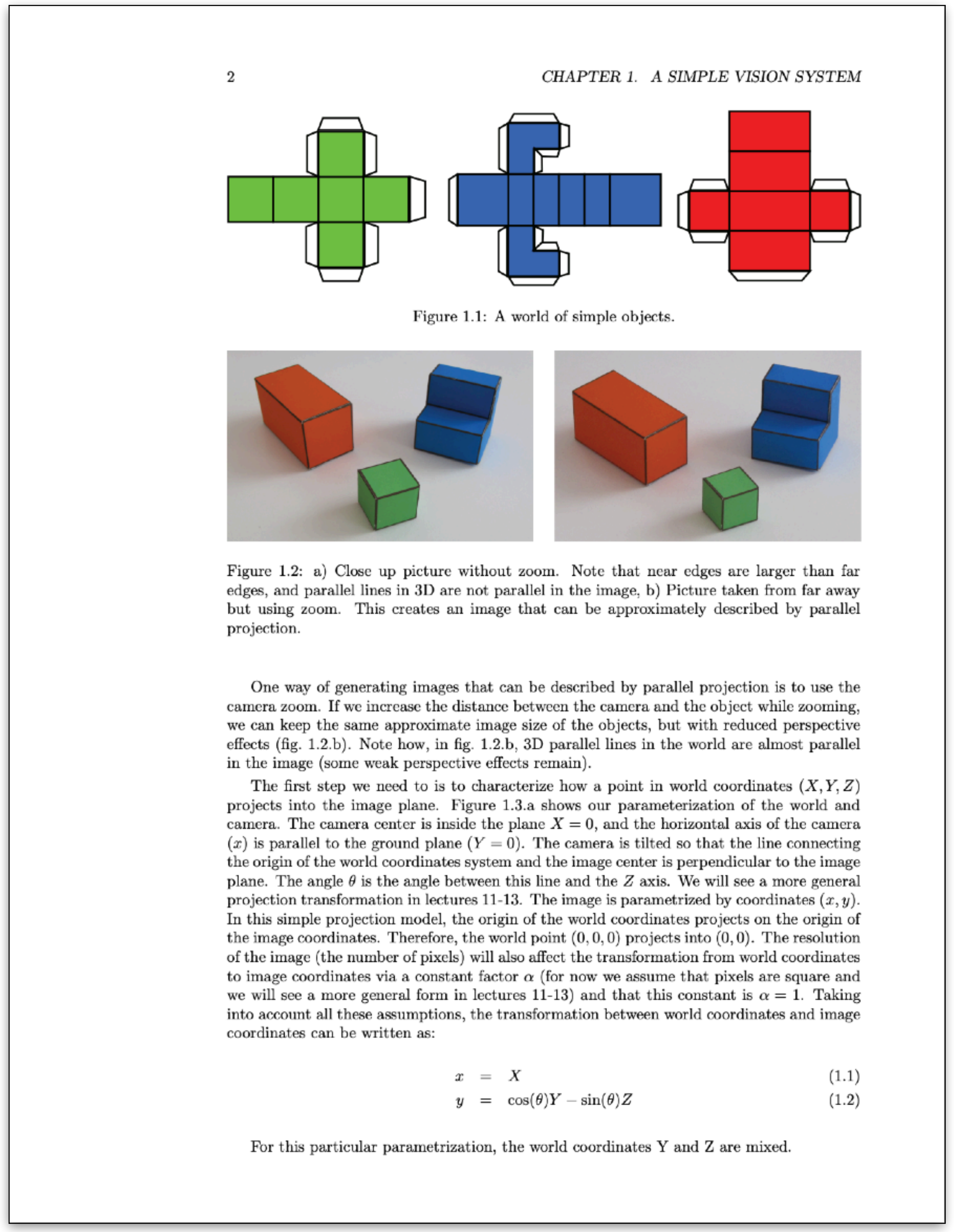
http://szeliski.org/Book

We'll use **2nd edition**

Other useful references:



http://deeplearningbook.org



2 CHAPTER 1. A SIMPLE VISION SYSTEM

Figure 1.1: A world of simple objects.

Figure 1.2: a) Close up picture without zoom. Note that near edges are larger than far edges, and parallel lines in 3D are not parallel in the image, b) Picture taken from far away but using zoom. This creates an image that can be approximately described by parallel projection.

One way of generating images that can be described by parallel projection is to use the camera zoom. If we increase the distance between the camera and the object while zooming, we can keep the same approximate image size of the objects, but with reduced perspective effects (fig. 1.2.b). Note how, in fig. 1.2.b, 3D parallel lines in the world are almost parallel in the image (some weak perspective effects remain).

The first step we need to is to characterize how a point in world coordinates (X, Y, Z) projects into the image plane. Figure 1.3.a shows our parameterization of the world and camera. The camera center is inside the plane $X = 0$, and the horizontal axis of the camera (x) is parallel to the ground plane $(Y = 0)$. The camera is tilted so that the line connecting the origin of the world coordinates system and the image center is perpendicular to the image plane. The angle θ is the angle between this line and the Z axis. We will see a more general projection transformation in lectures 11-13. The image is parametrized by coordinates (x, y). In this simple projection model, the origin of the world coordinates projects on the origin of the image coordinates. Therefore, the world point $(0, 0, 0)$ projects into $(0, 0)$. The resolution of the image (the number of pixels) will also affect the transformation from world coordinates to image coordinates via a constant factor α (for now we assume that pixels are square and we will see a more general form in lectures 11-13) and that this constant is $\alpha = 1$. Taking into account all these assumptions, the transformation between world coordinates and image coordinates can be written as:

$$x = X \quad (1.1)$$
$$y = \cos(\theta)Y - \sin(\theta)Z \quad (1.2)$$

For this particular parametrization, the world coordinates Y and Z are mixed.

Foundations of Computer Vision manuscript by Torralba, Isola, and Freeman

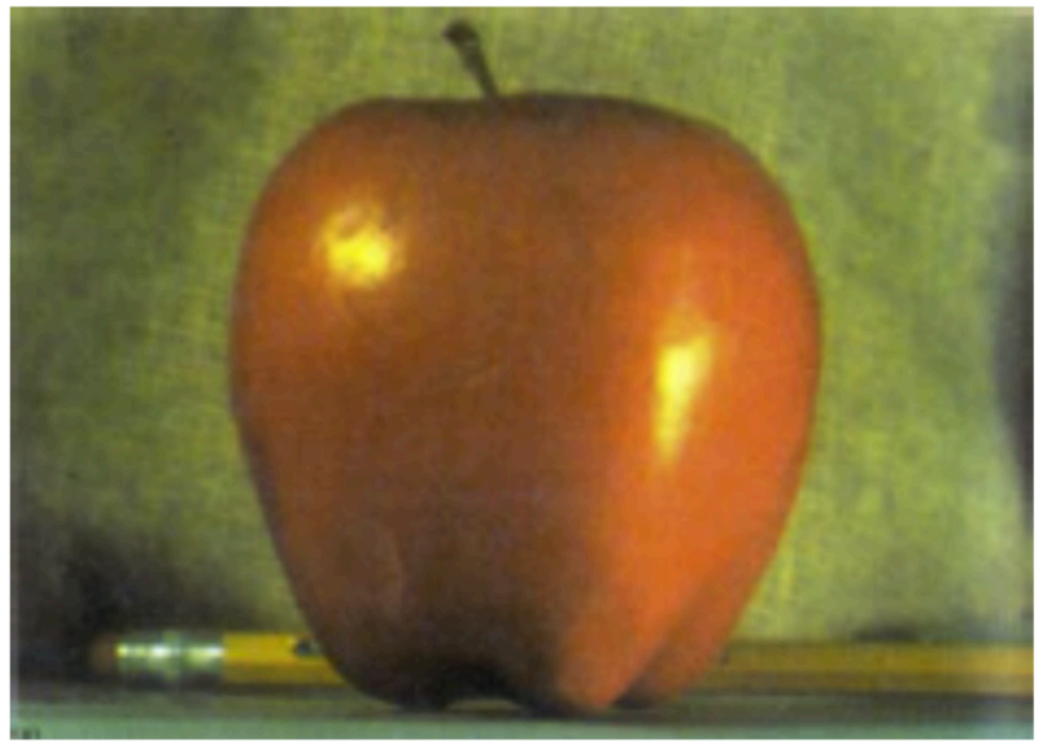
GPU computing

- Later problem sets (PS5 and onwards) use GPUs.
- GPUs are very expensive.
- Recommend using Google Colab
  - Free, but comes with usage limits (per email address)
  - You can consider purchasing Colab Pro, but it's not necessary. If you are unable to afford it and would like to use it, we have been provided with a small amount of funding from the CSE DEI office. Please send the course staff a message.
- CAEN has machines you can use in their computer labs

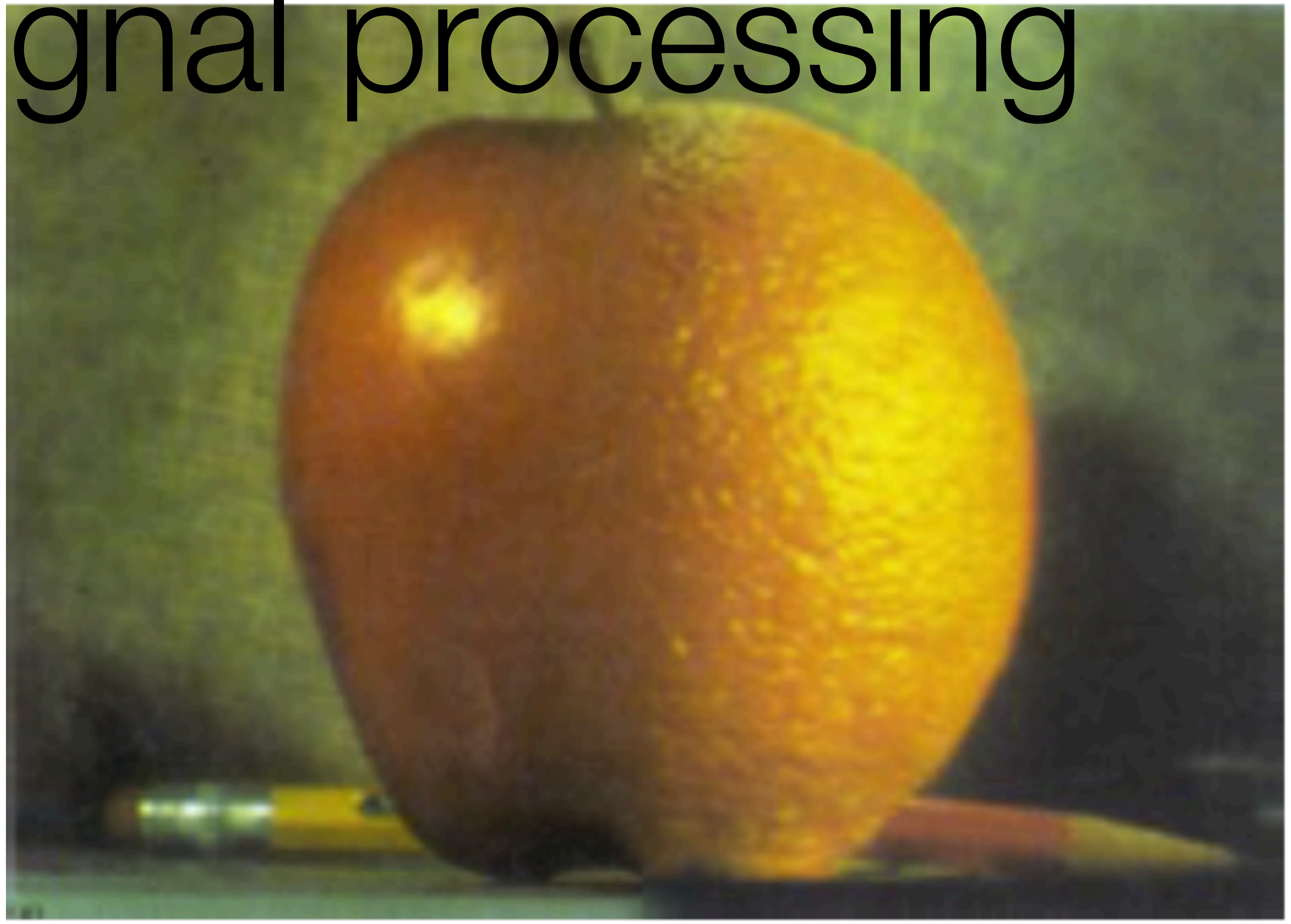
Class topics

Lec. 1	**A simple vision system** About the course Cameras Simple edge detection
Lec. 2	**Image filters** Convolution Gradient filters Blurring
Sec. 1	**Linear algebra review**
Lec. 3	**Nonlinear filtering** Template matching Edge detection Bilateral filtering
Lec. 4	**Frequencies** Amplitude and phase Fourier transform
Sec. 2	**More linear algebra**
Lec. 3	**Fourier analysis** Fourier basis Compression
Lec. 4	**Multi-scale pyramids** Gaussian and Laplacian pyramids Image blending Texture analysis
Sec. 3	**Fourier tutorial**
Lec. 5	**Statistical models for images** Image priors Denoising

Problem set #2: image blending

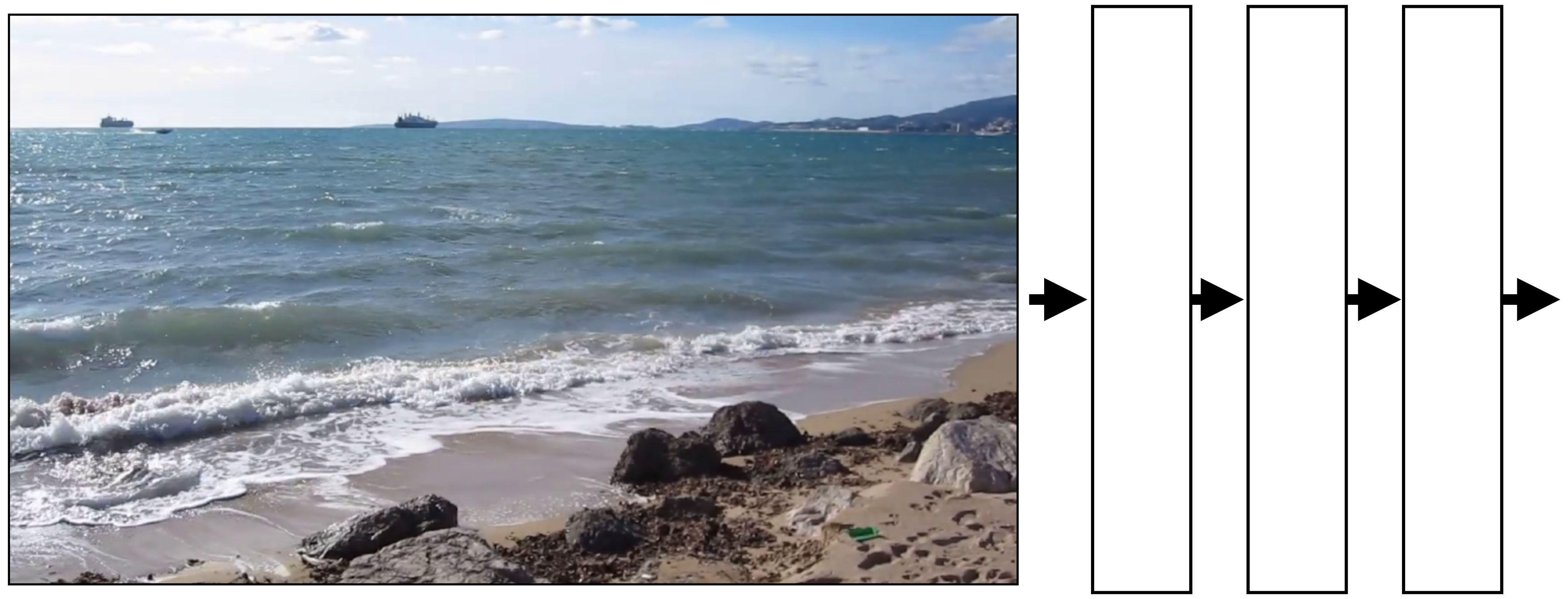



Signal processing



Sec. 3	**Fourier tutorial**	
Lec. 5	**Statistical models for images** Image priors Denoising Example-based texture synthesis	
Lec. 6	**Machine learning** Learning Datasets Linear regression	
Sec. 3	**Learning tutorial**	
Lec. 7	**Linear models** Logistic regression Gradient descent	
Lec. 7	**Neural networks** Nonlinearities Network structure Regularization	
Sec. 4	**Office hours**	
Lec. 8	**Optimization** Backpropagation SGD	
Lec. 9	**Convolutional networks** Convolution layers Pooling Normalization	
Sec. 4	**PyTorch tutorial**	

Problem #5: scene recognition



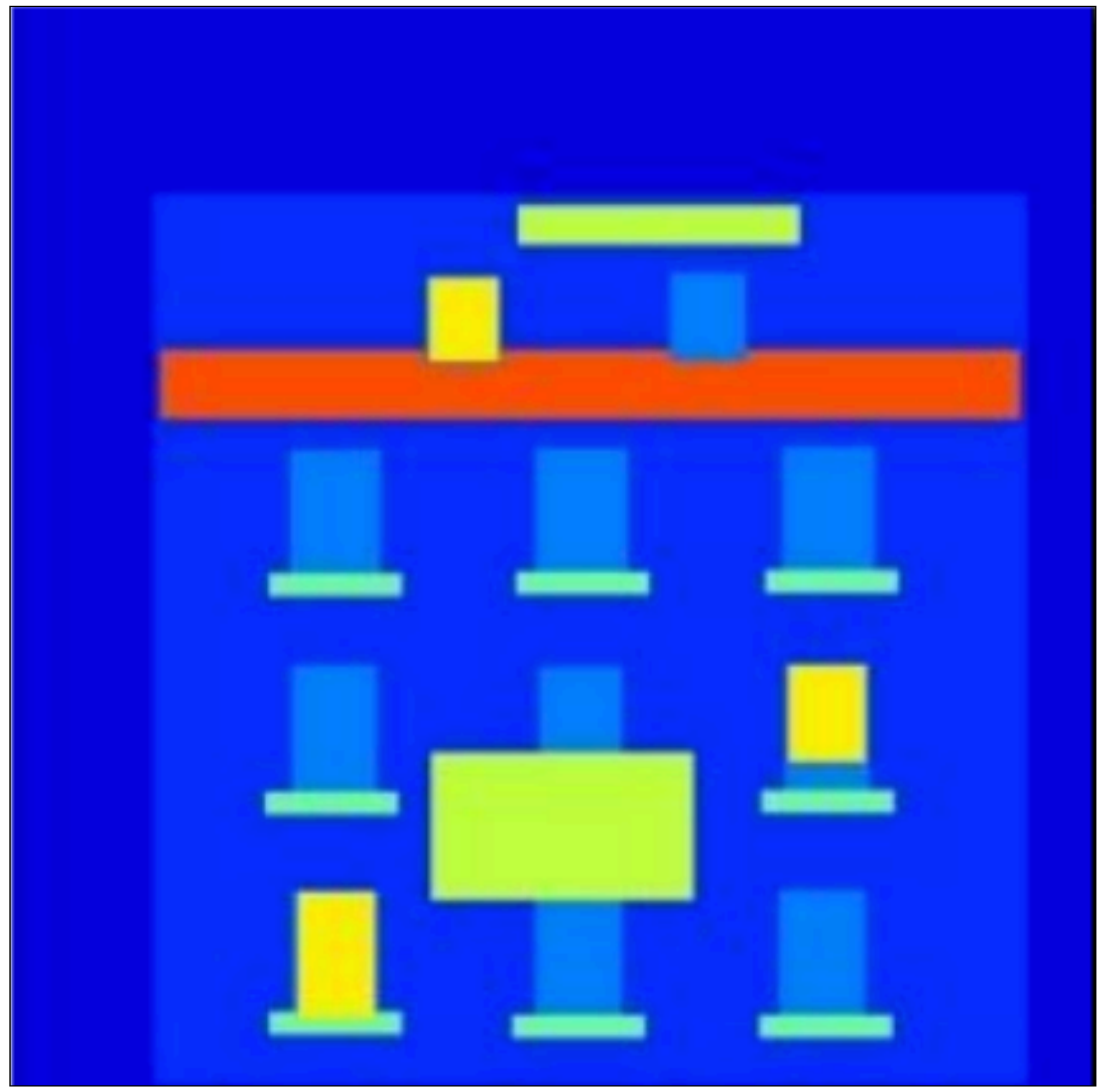
“Beach”

Intro to deep learning

Note: Guest lectures / Zoom for Sep. 20 - Oct. 4

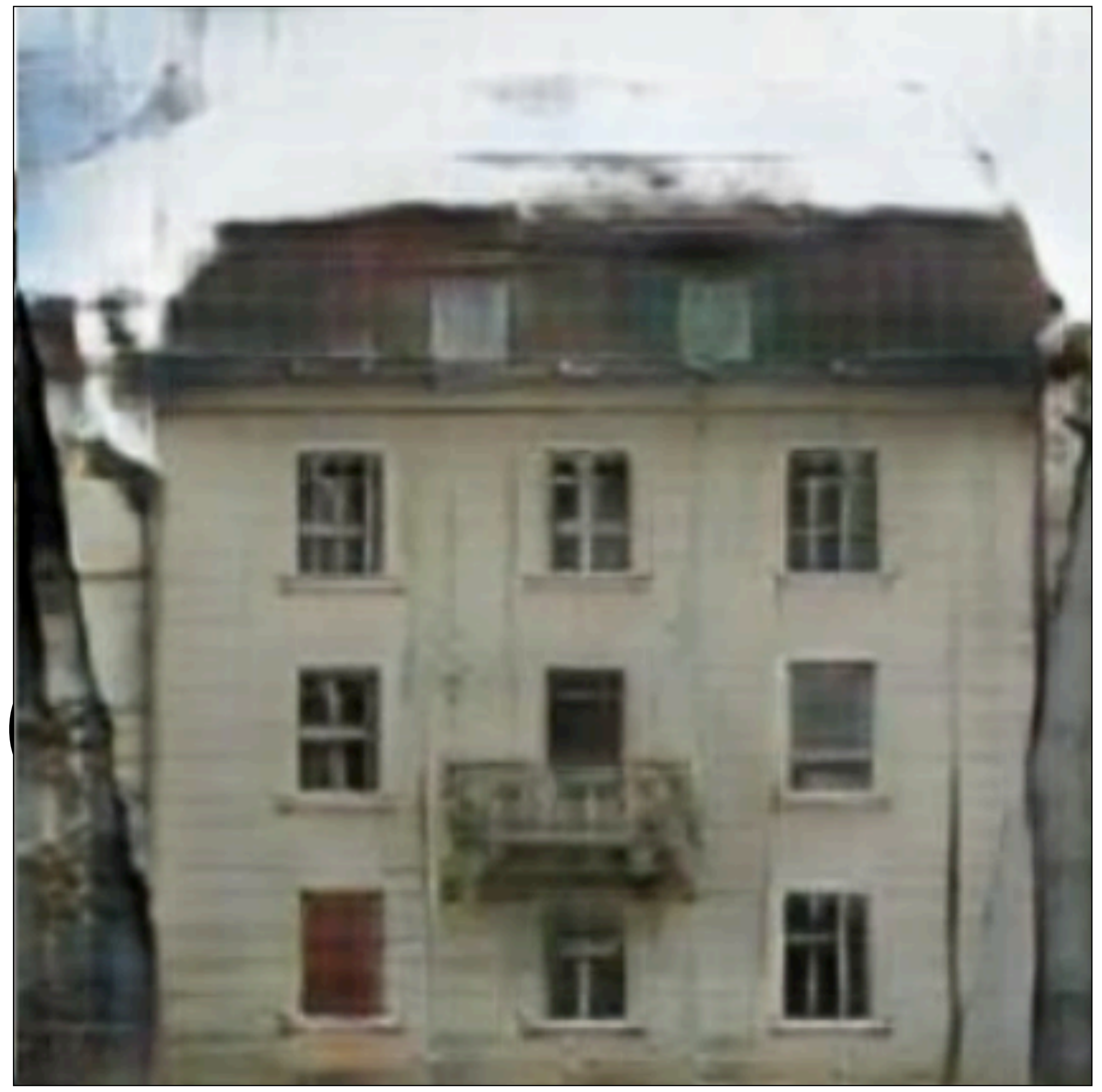
		Denoising Example-based texture synthesis
Lec. 6	**Machine learning**	Learning Datasets Linear regression
Sec. 3	**Learning tutorial**	
Lec. 7	**Linear models**	Logistic regression Gradient descent
Lec. 7	**Neural networks**	Nonlinearities Network structure Regularization
Sec. 4	**Office hours**	
Lec. 8	**Optimization**	Backpropagation SGD
Lec. 9	**Convolutional networks**	Convolution layers Pooling Normalization
Sec. 4	**PyTorch tutorial**	
Lec. 10	**Scene understanding**	Scene recognition Semantic segmentation
Lec. 11	**Object detection**	Sliding window

Problem set #6: image translation



Labels

n visi



Synthesized image

Sec. 4	**Project office hours**
Lec. 16	**Image formation** Plenoptic function Pinhole cameras Homogeneous coordinates Projection matrix
Lec. 17	**Multi-view geometry** Triangulation Epipolar lines Homographies Warping
Sec. 4	**Geometry tutorial**
Lec. 18	**Multi-view reconstruction** Feature matching RANSAC Structure from motion
Lec. 19	**Depth estimation** Stereo matching Graphical models Belief propagation
Sec. 4	**Project office hours**
Lec. 20	**Motion** Optical flow Aperture problem Multi-scale estimation
Lec. 21	**Color** Color perception Color constancy

Homework #9: image stitching

Physically-based methods

Sec. 4	**Project office hours**	
Lec. 20	**Motion** Optical flow Aperture problem Multi-scale estimation	
Lec. 21	**Color** Color perception Color constancy	
Sec. 4	**Project office hours**	
Lec. 22	**Light and shading** Shape from shading Photometric stereo Intrinsic images	
Lec. 24	**Language** Attention Captioning	
Sec. 4	**Project office hours**	
Lec. 25	**Embodied vision** Learning from demonstrations Reinforcement learning	
Lec. 27	**Bias and disinformation** Datasets Algorithmic fairness Image forensics	

Advanced topics and applications

Any questions?

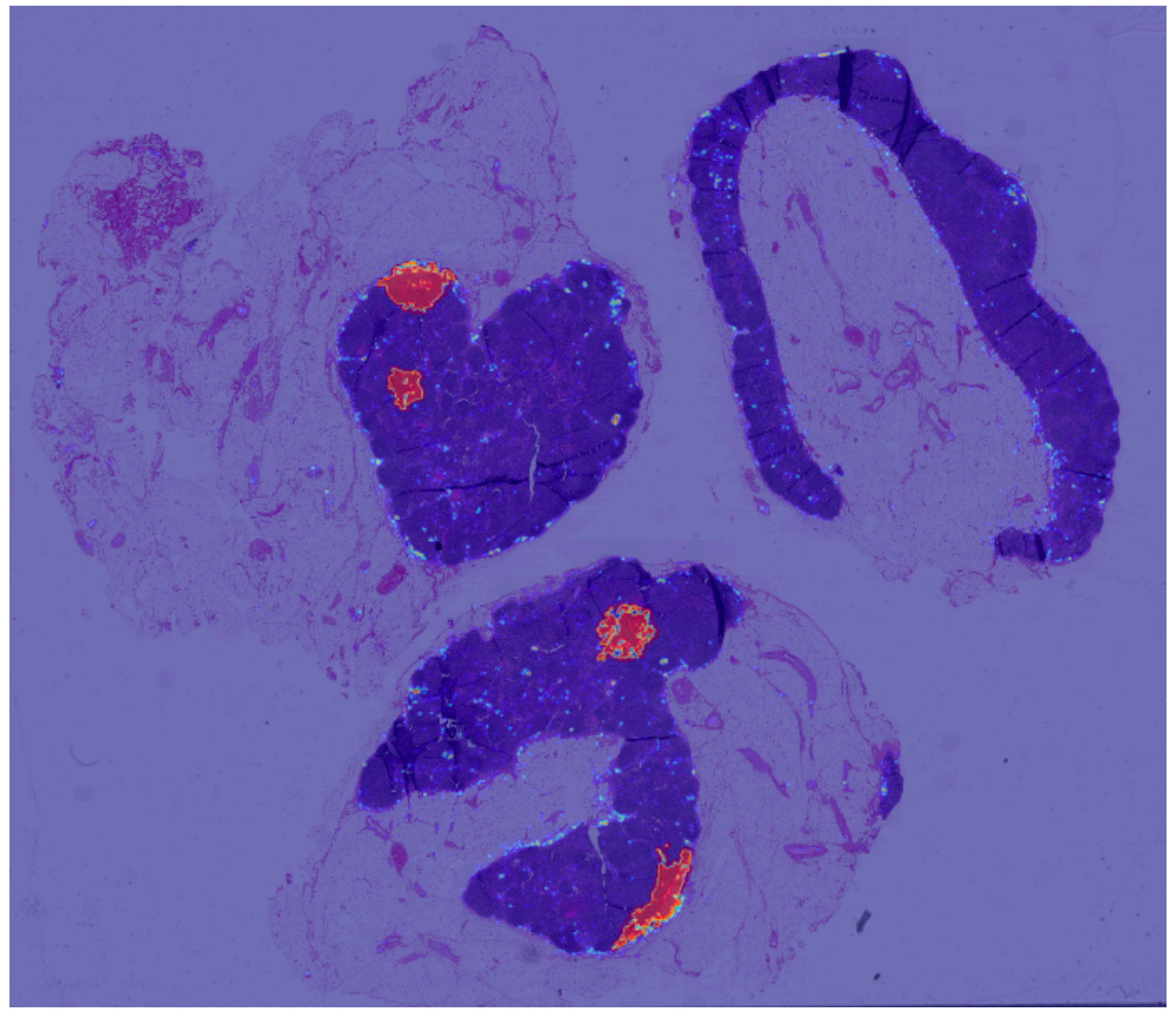
Today

1. Class logistics
2. **Computer vision today**
3. What makes vision hard?
4. Intro to image filtering

Exciting times for computer vision

Robotics

Medical imaging



3D modeling

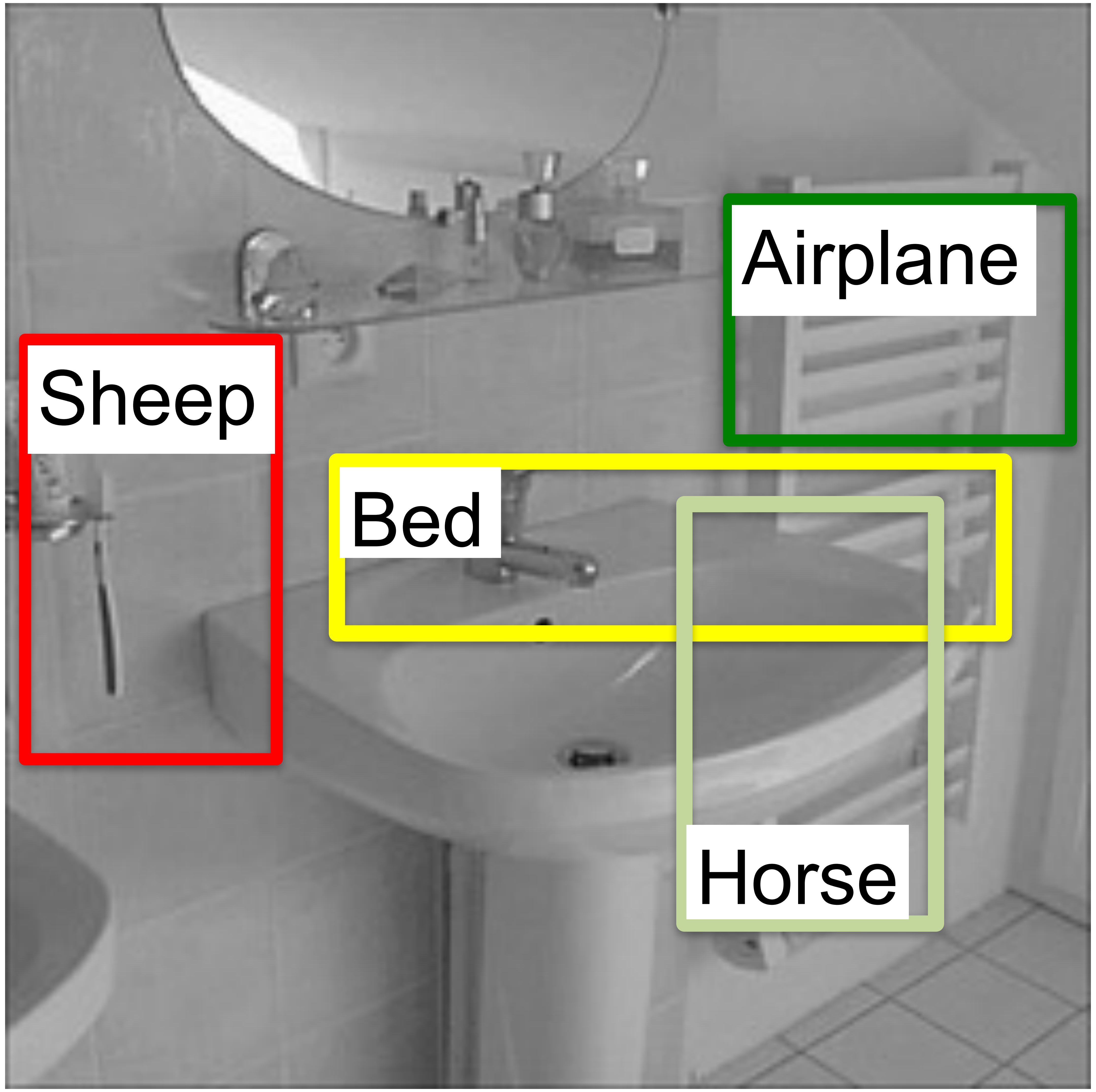
Driving

Communication

Accessibility

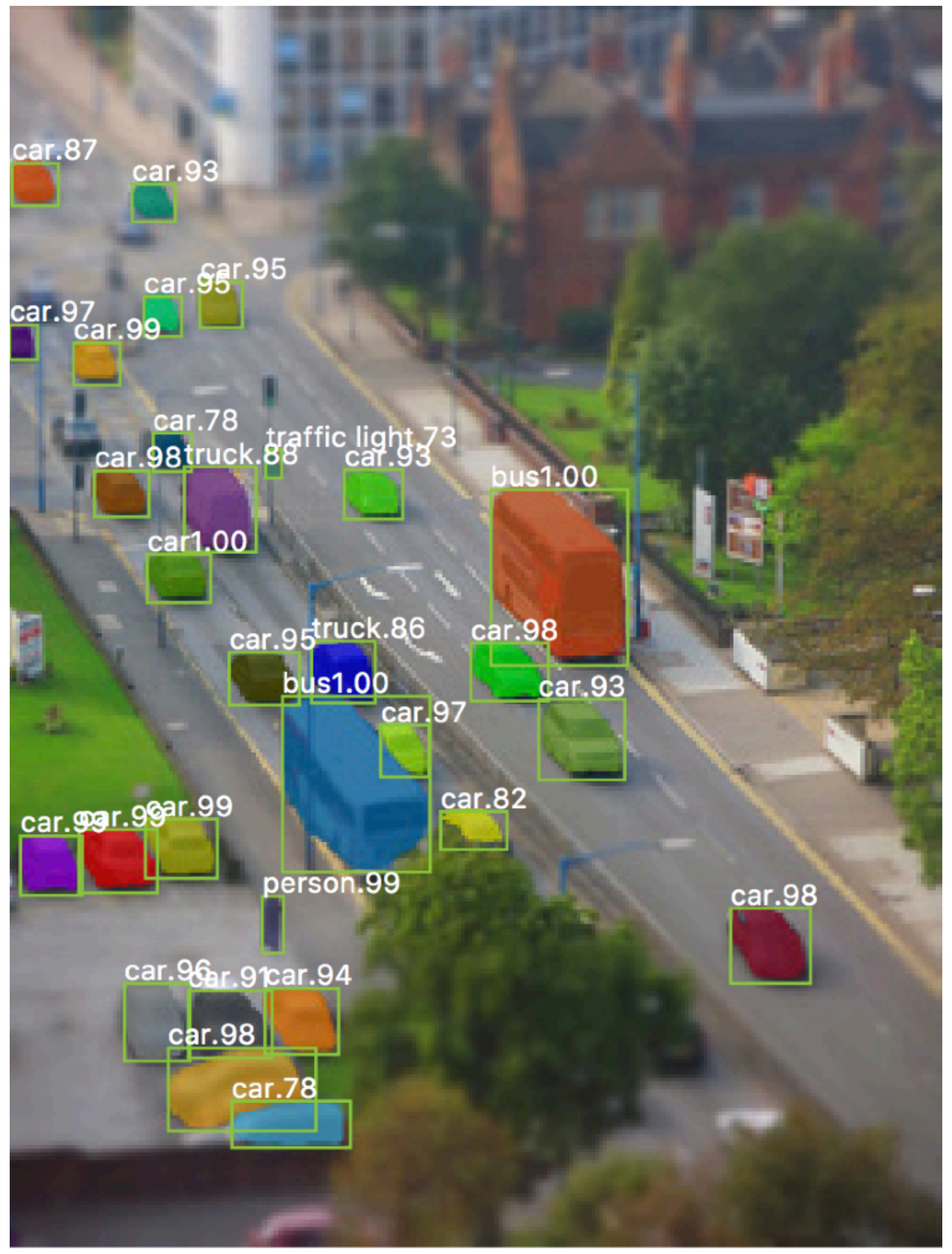
Adapted from: Torralba, Freeman, Isola

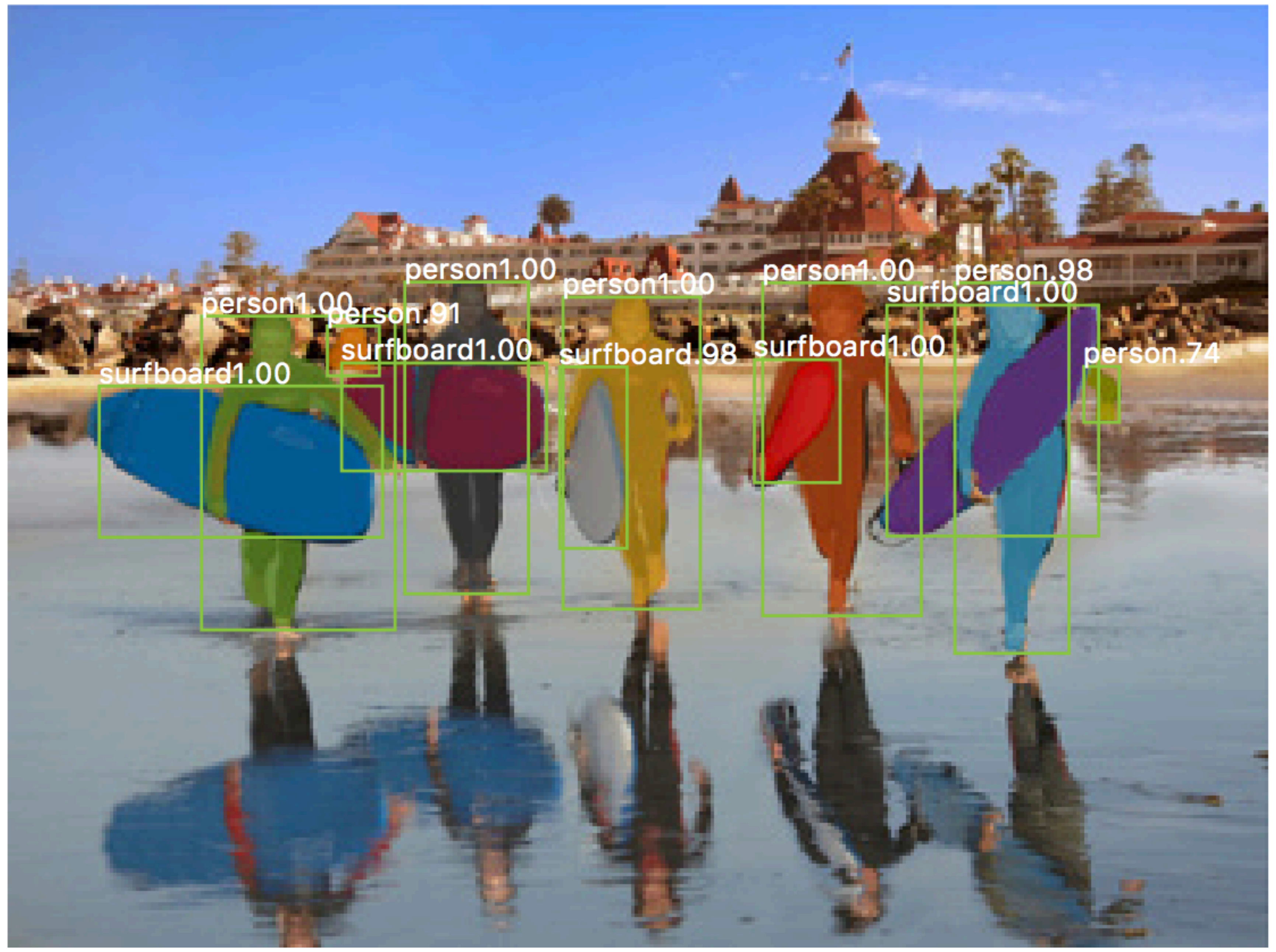
Object detection circa 2010

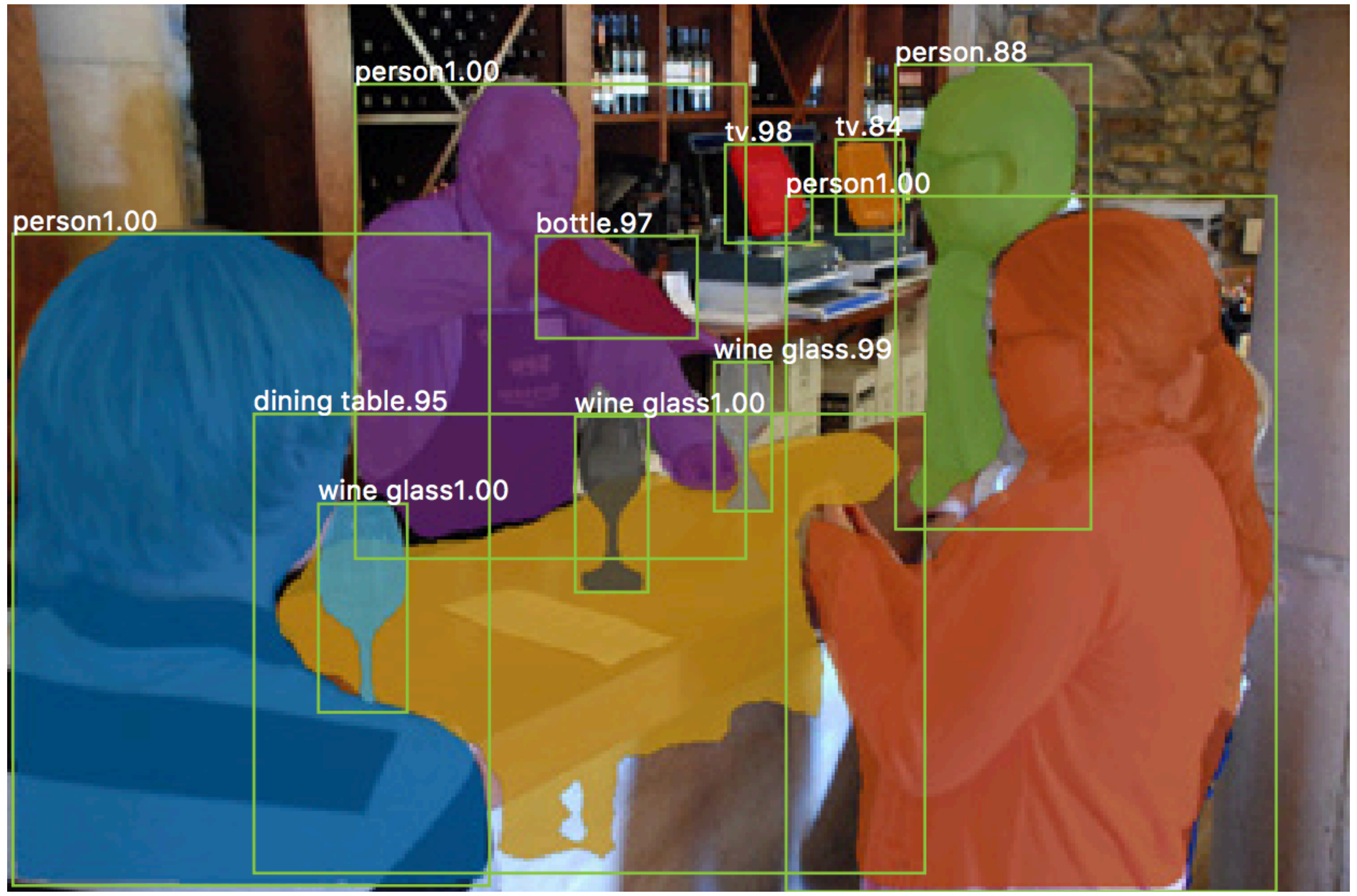


Slide credit: Torralba, Freeman, Isola

Object detection now







Slide credit: Torralba, Freeman, Isola

[“Mask RCNN”, He et al., ICCV 2017]

Generative models

A photo of a frog reading the newspaper named “Toaday” written on it. There is a frog printed on the newspaper too.

[Yu et al., “Parti”, 2022]

“a pig wearing a backpack”

[Poole et al., “DreamFusion”, 2022]

Robot vision

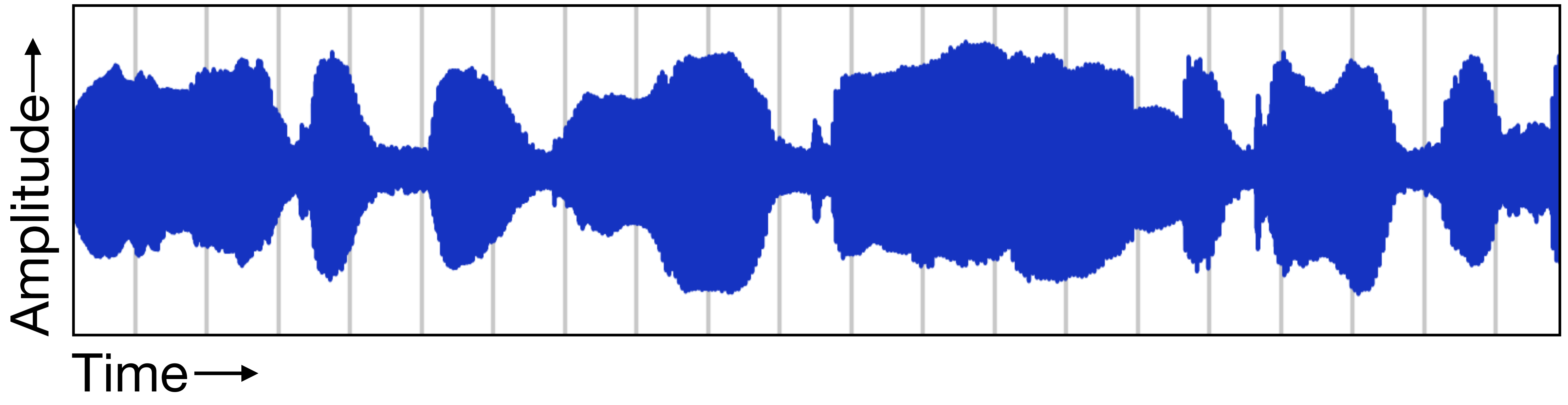
[Agarwal, Kumar, et al., 2022]

Reconstructing 3D scenes

[Barron et al., "Zip-NeRF", 2023]

Different signals, but really similar methods!

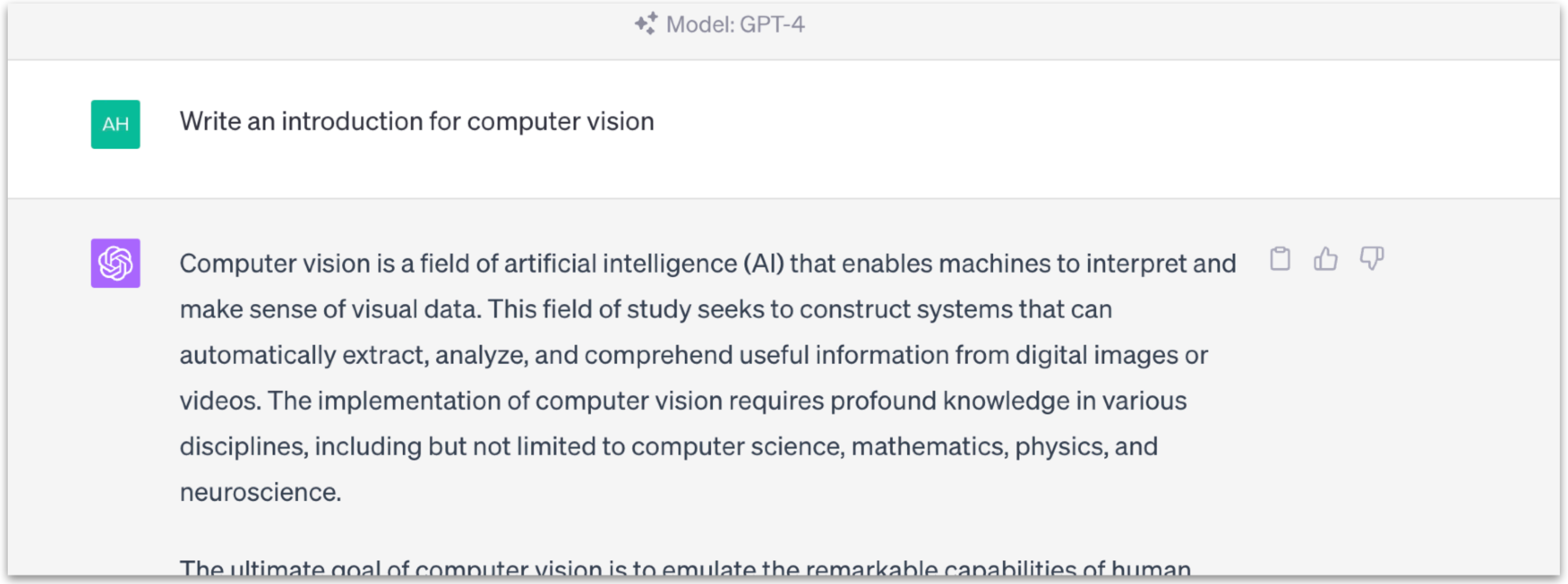
Sound



WiFi

[Zhao et al. 2019]

Language



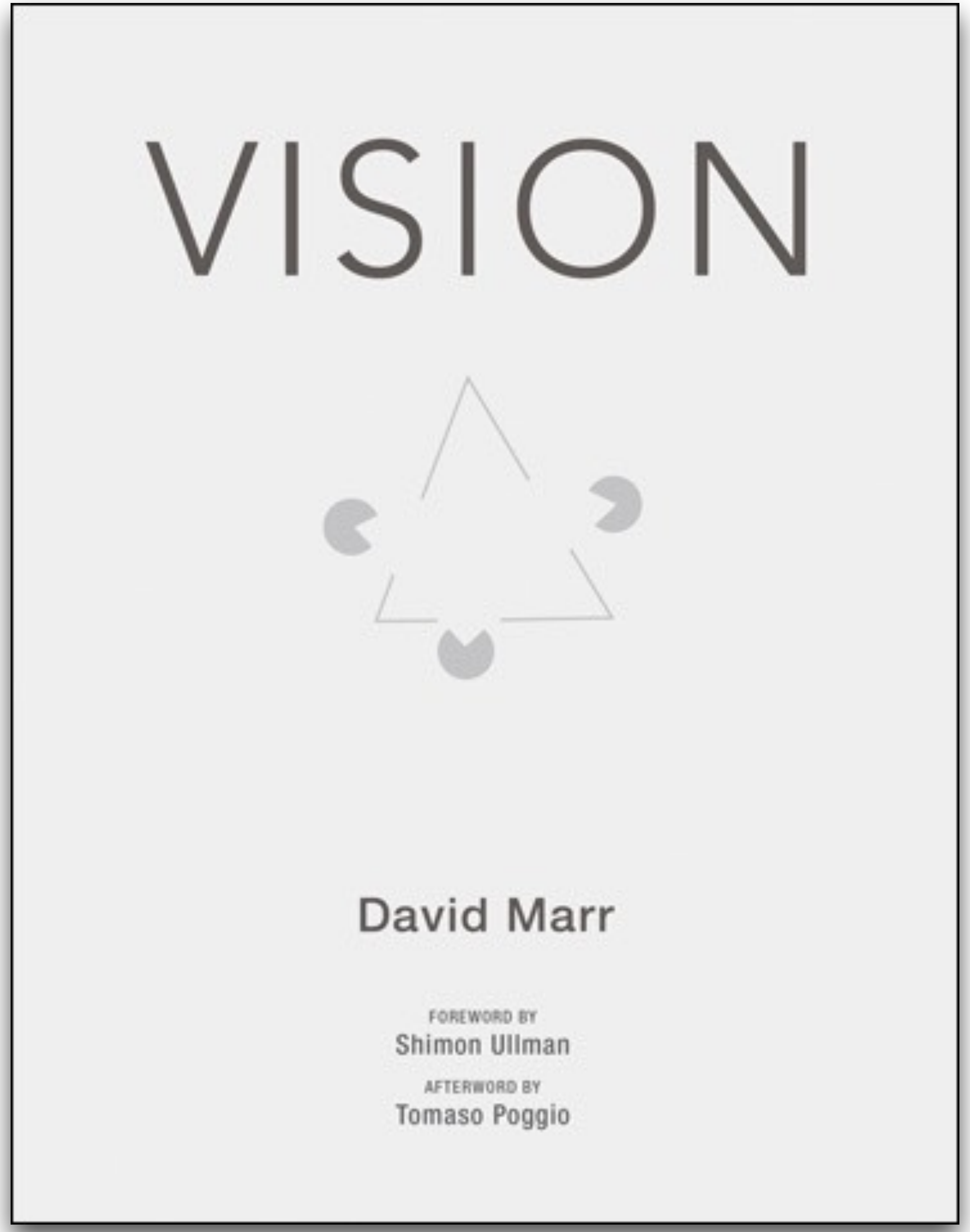
Today

1. Class logistics
2. Computer vision today
3. **What makes vision hard?**
4. Intro to image filtering

To see

“What does it mean, to see? The plain man's answer (and Aristotle's, too) would be, to know what is where by looking.”

To discover from images what is present in the world, where things are, what actions are taking place, to predict and anticipate events in the world.



Source: Torralba, Freeman, and Isola

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

Artificial Intelligence Group July 7, 1966

Vision Memo. No. 100.

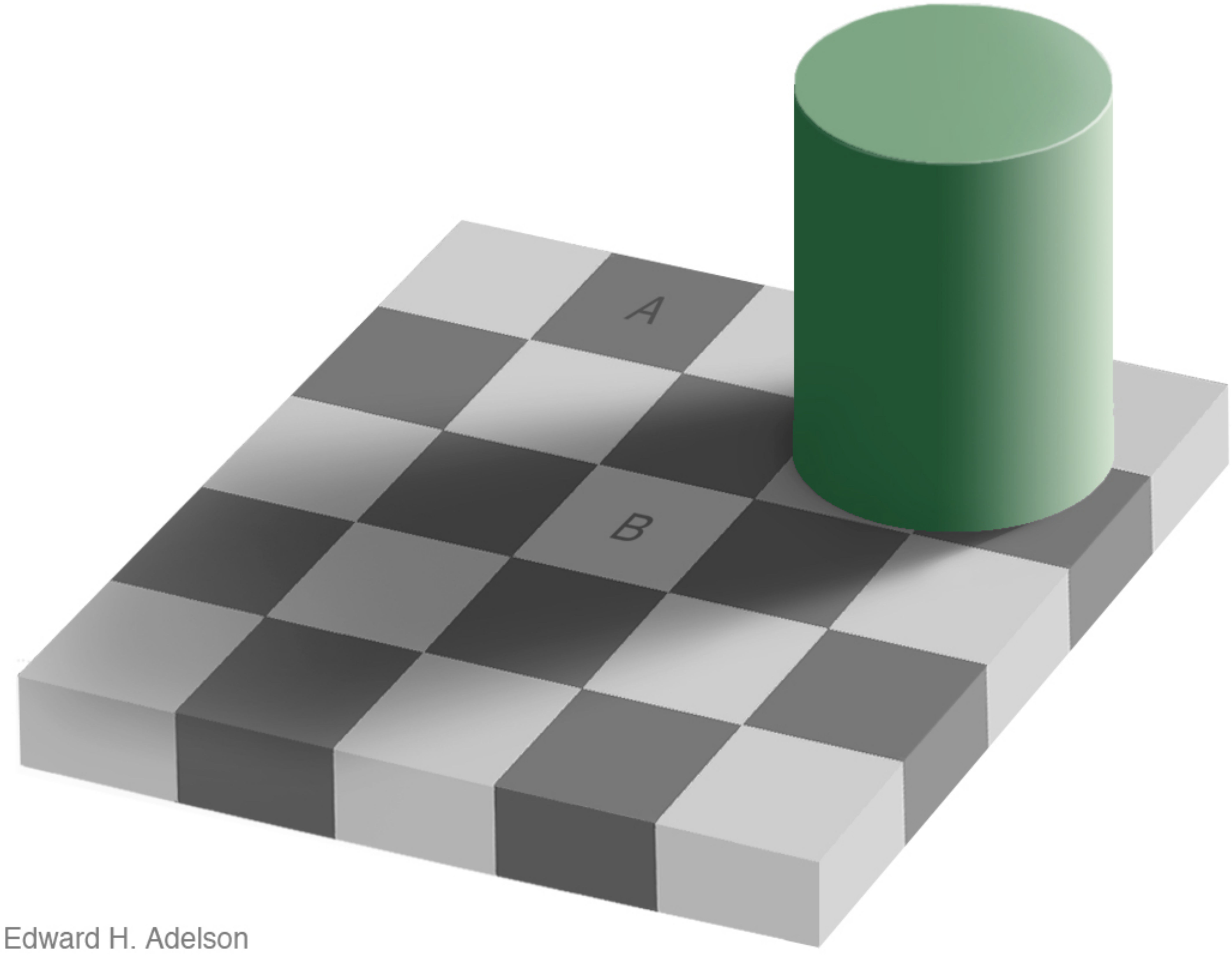
THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

Slide credit: Torralba, Freeman, Isola

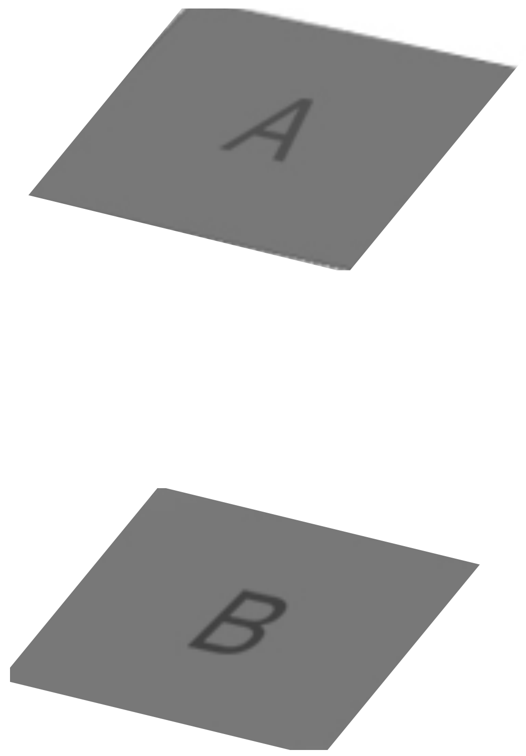
To see: perception vs. measurement



[Adelson, 1995]

Source: Torralba, Freeman, Isola

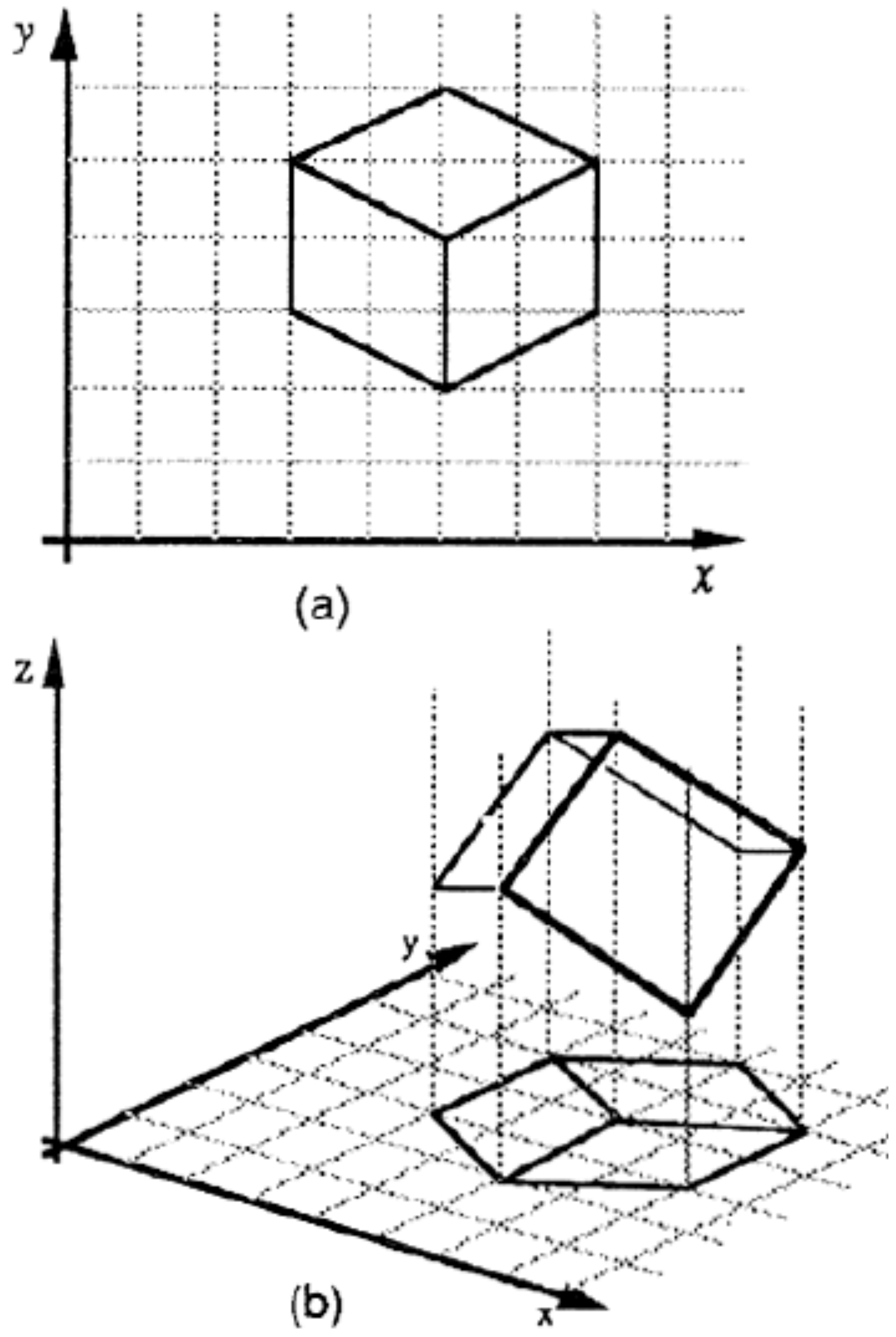
To see: perception vs. measurement



[Adelson, 1995]

Source: Torralba, Freeman, Isola

Fundamental ambiguities



[Sinha & Adelson, 1993]

Source: A. Torralba

Fundamental ambiguities

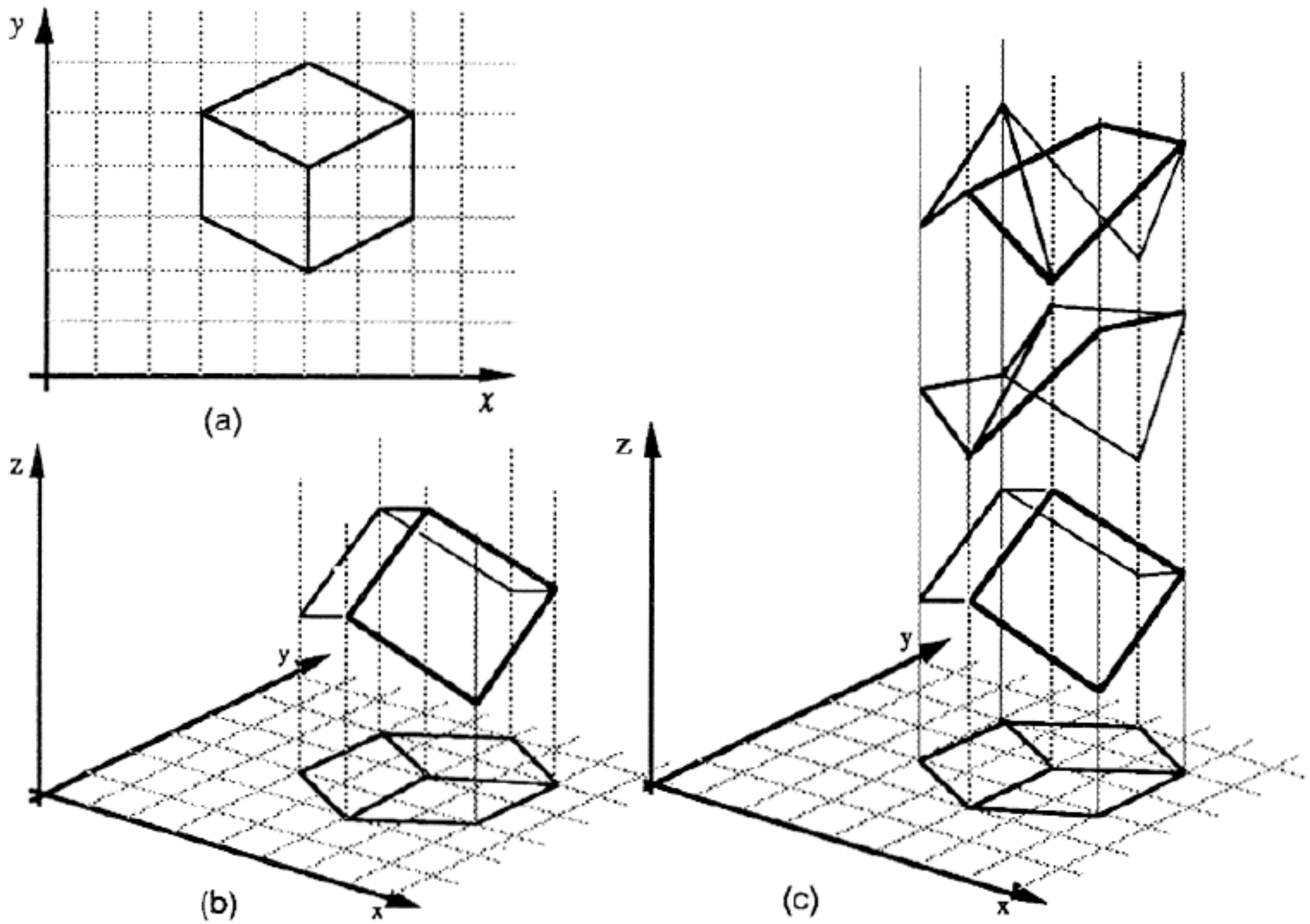


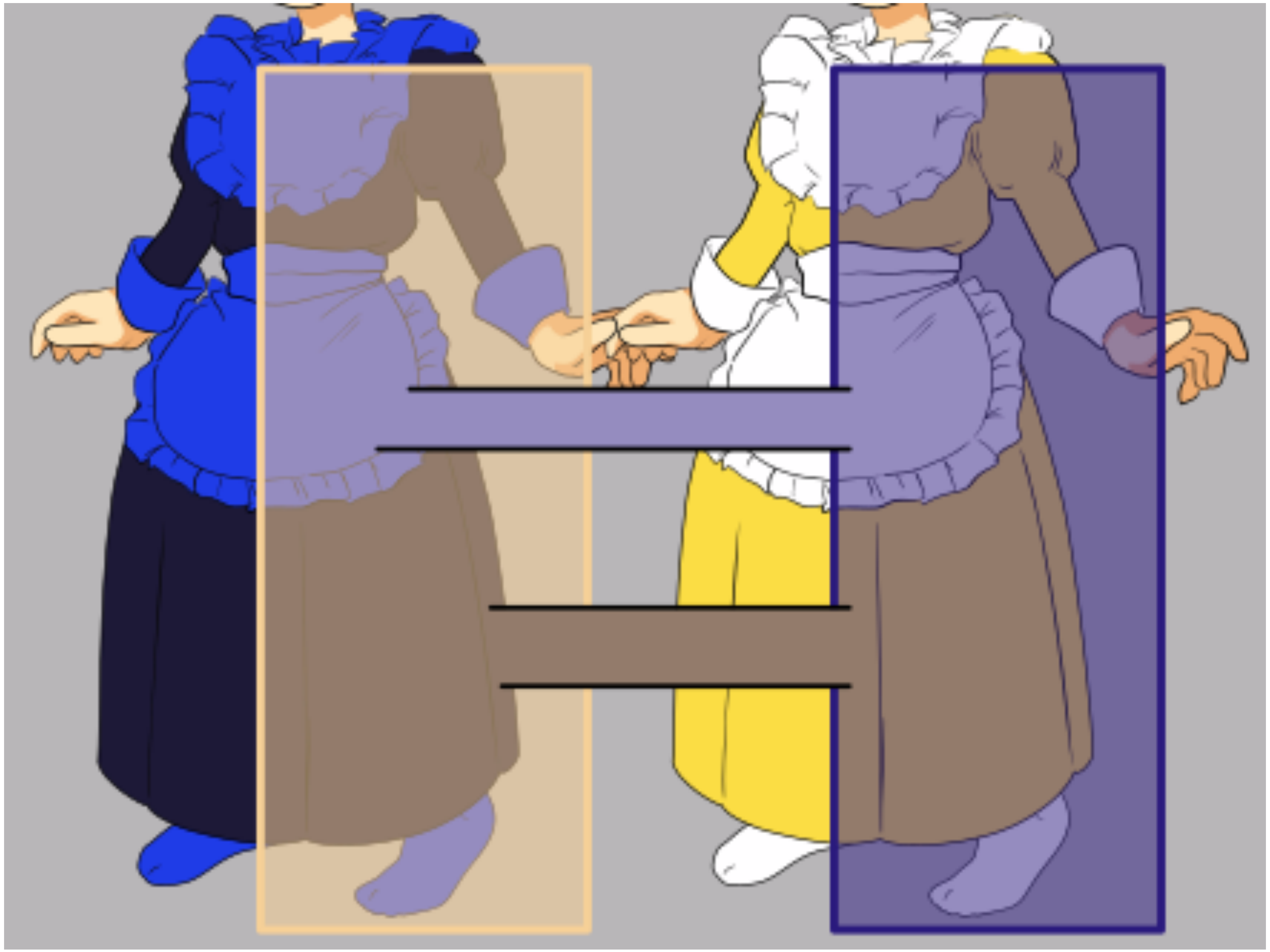
Figure 1. (a) A line drawing provides information only about the x, y coordinates of points lying along the object contours. (b) The human visual system is usually able to reconstruct an object in three dimensions given only a single 2D projection (c) Any planar line-drawing is geometrically consistent with infinitely many 3D structures.

[Sinha & Adelson, 1993]

Source: A. Torralba

Fundamental ambiguities

“The dress”

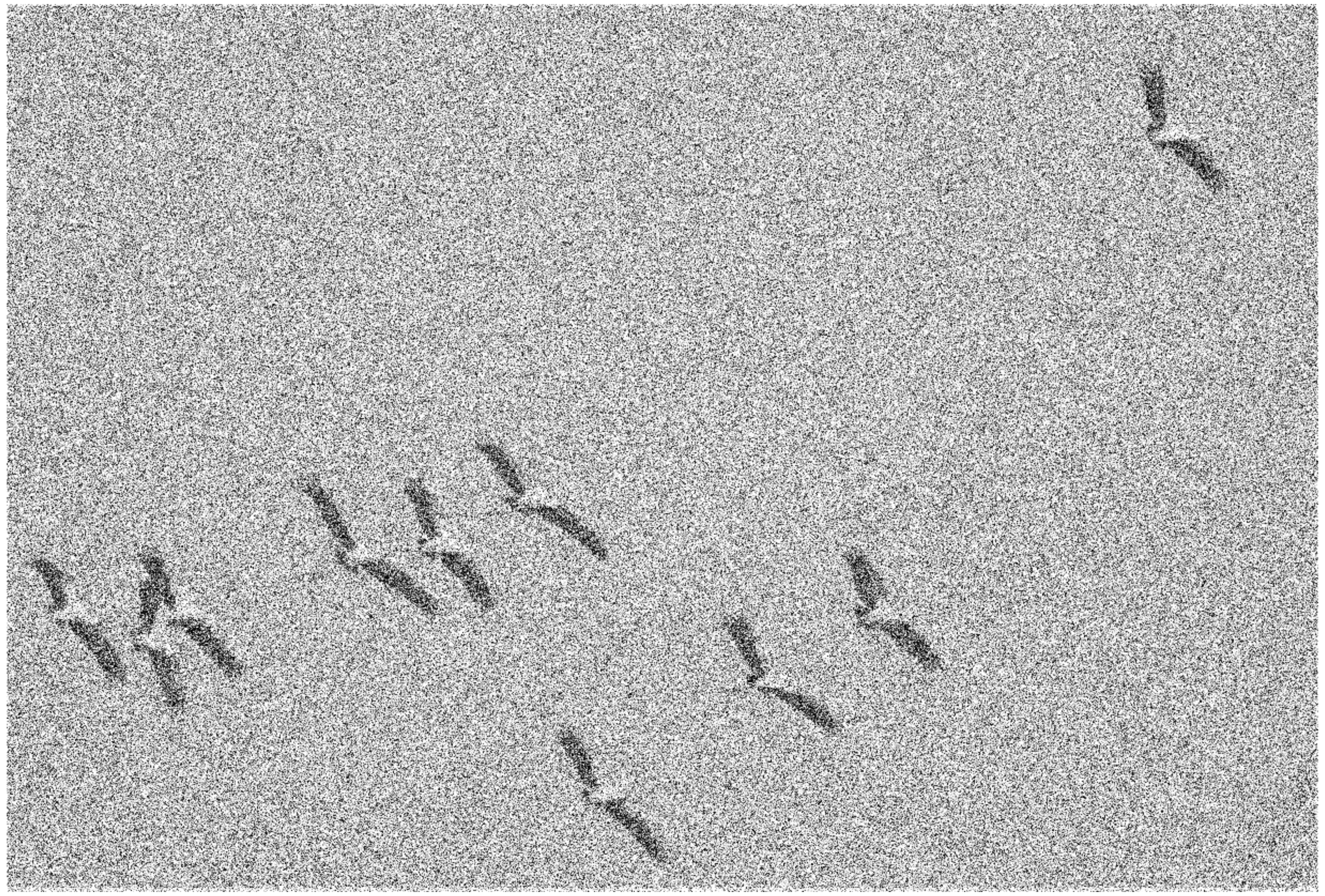


Source: Wikipedia

Today

1. Class logistics
2. Computer vision today
3. What makes vision hard?
4. **Intro to image filtering**

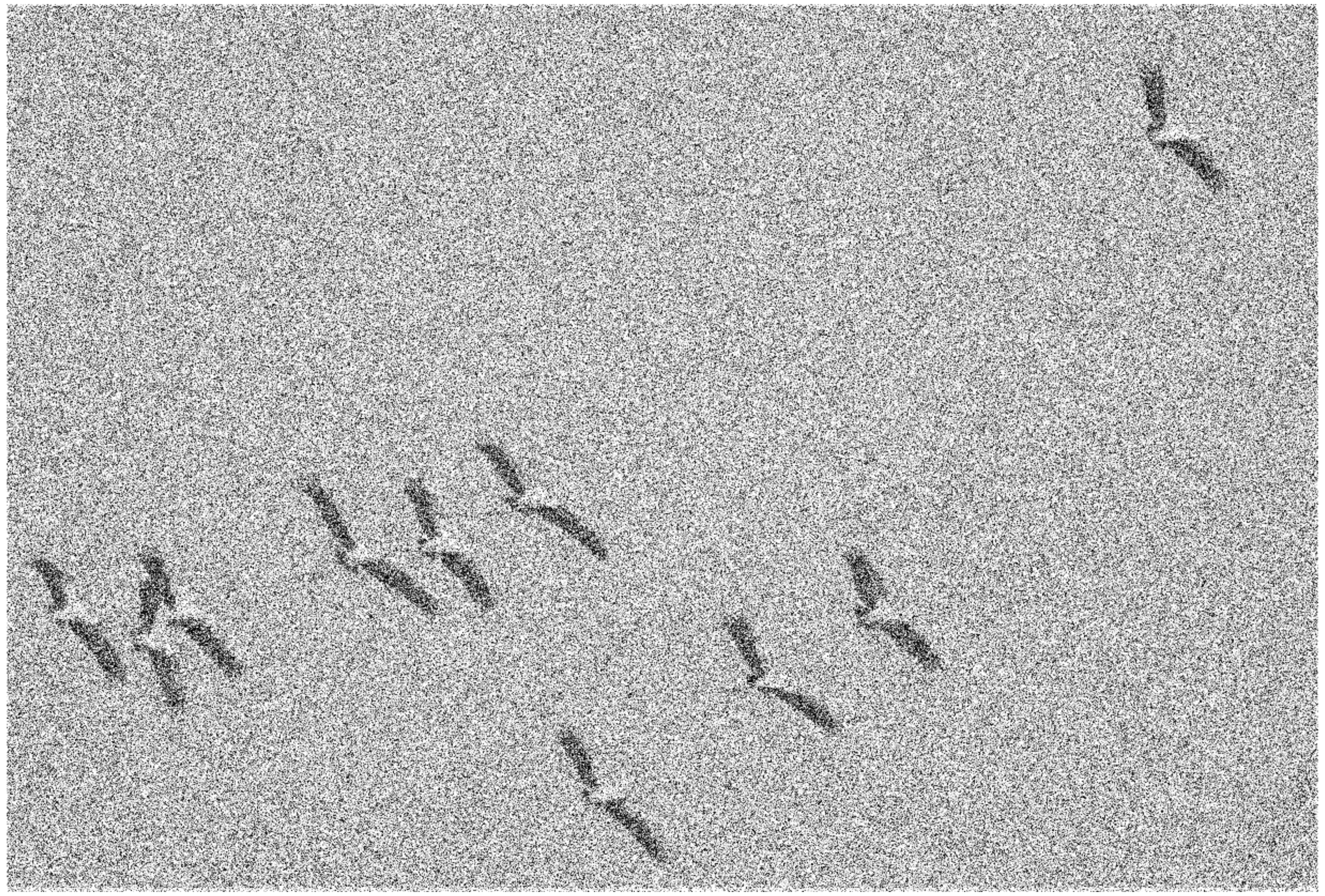
Two computer vision problems



Denoising

Edge detection

Two computer vision problems



Denoising

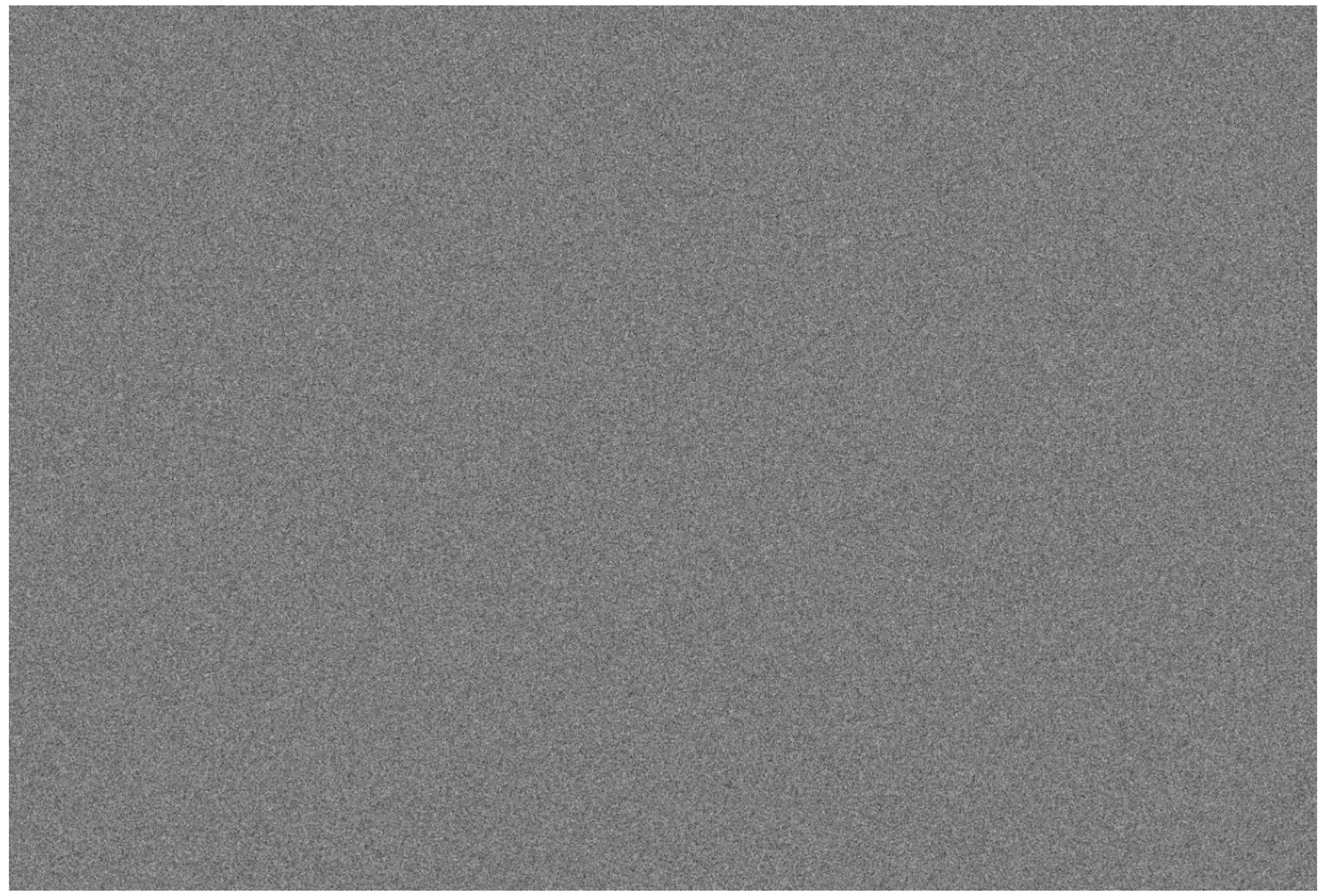
Edge detection

Case study: image denoising

Image

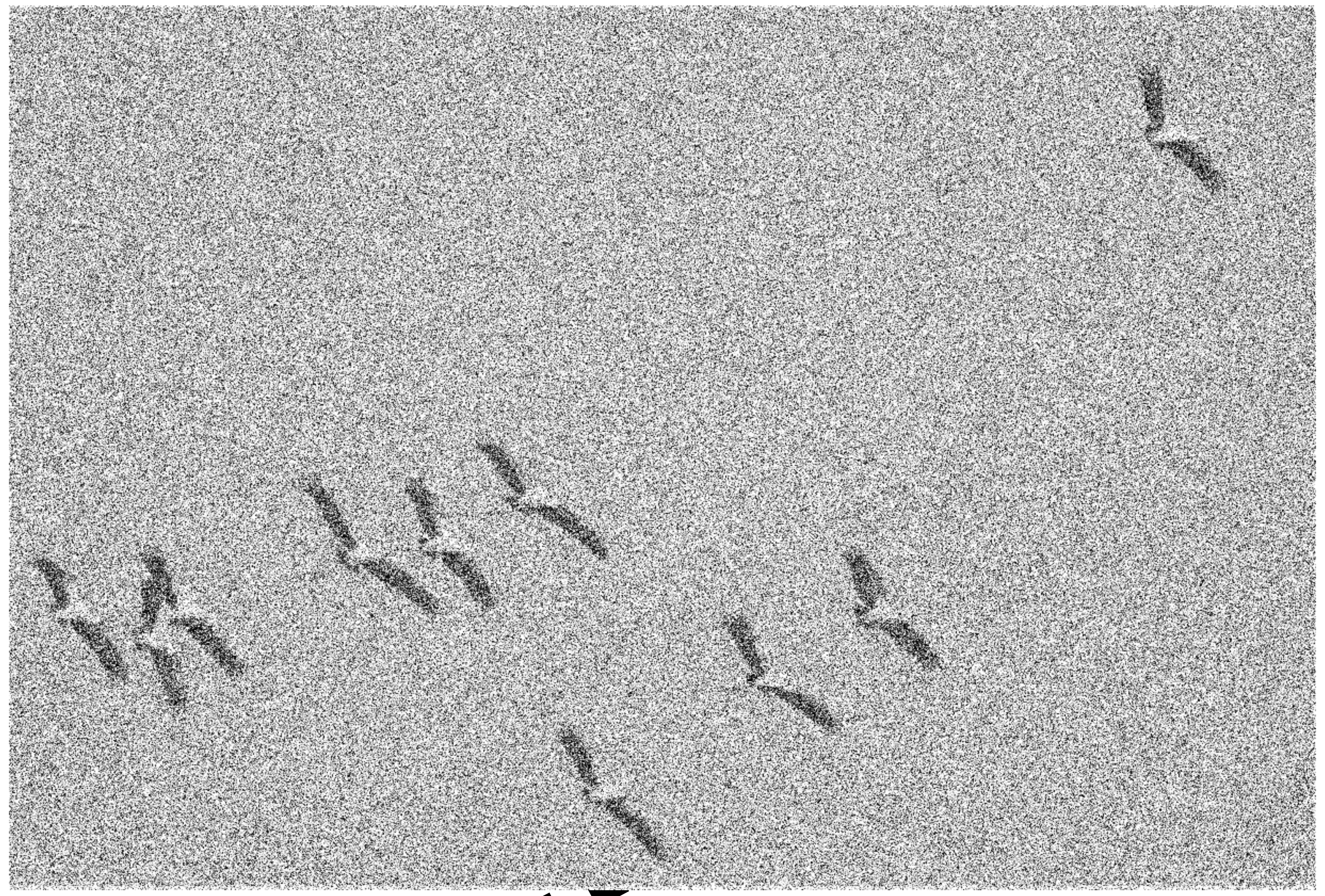
+

Noise



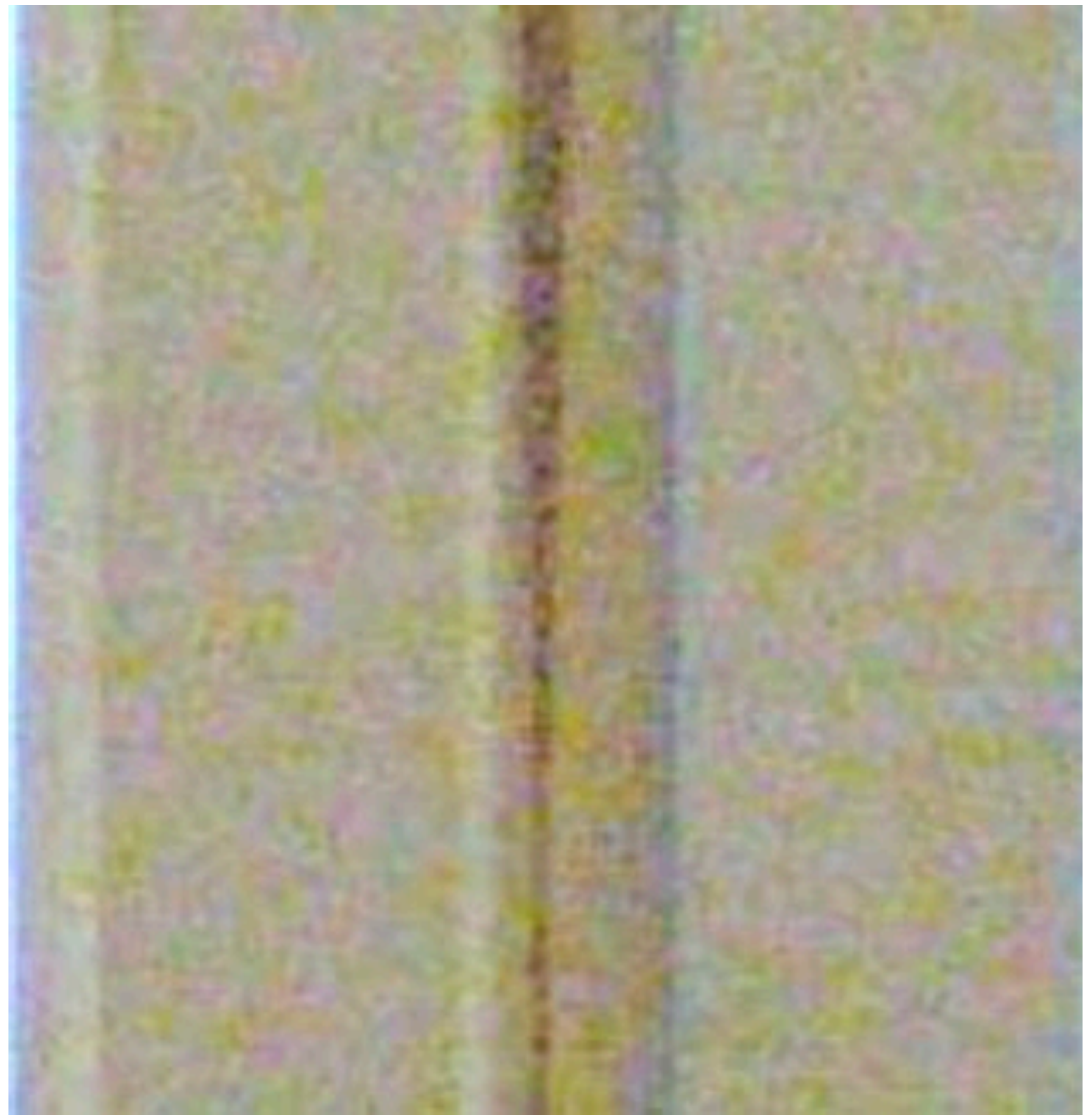
=

Noisy image



Goal: recover the original image

Image denoising problem



In practice: low light photography, “dead” pixels, interference, etc.

We’ll see a lot more later…

Image source: wikipedia

Image denoising

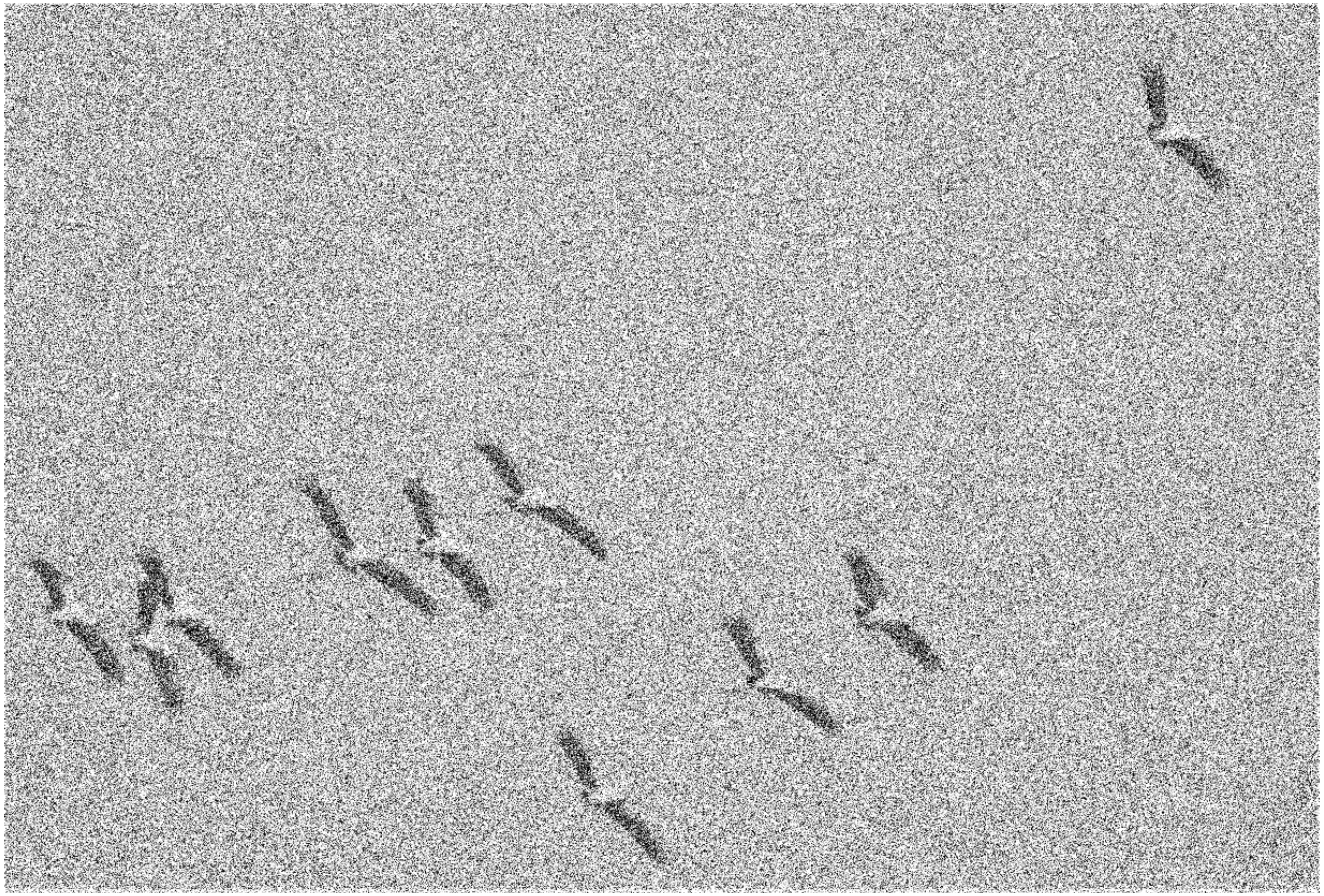


Photo by Fredo Durand

Image denoising

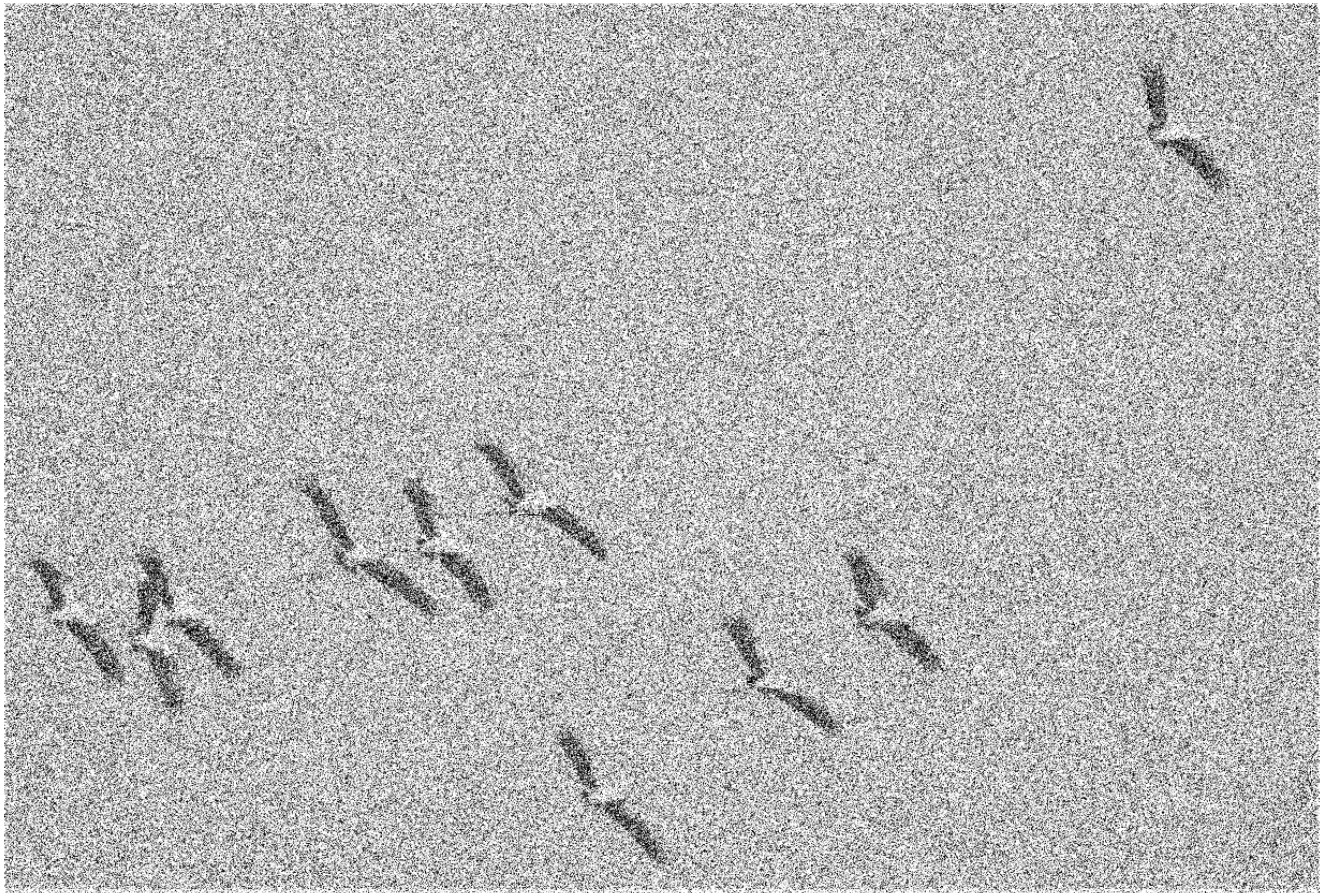
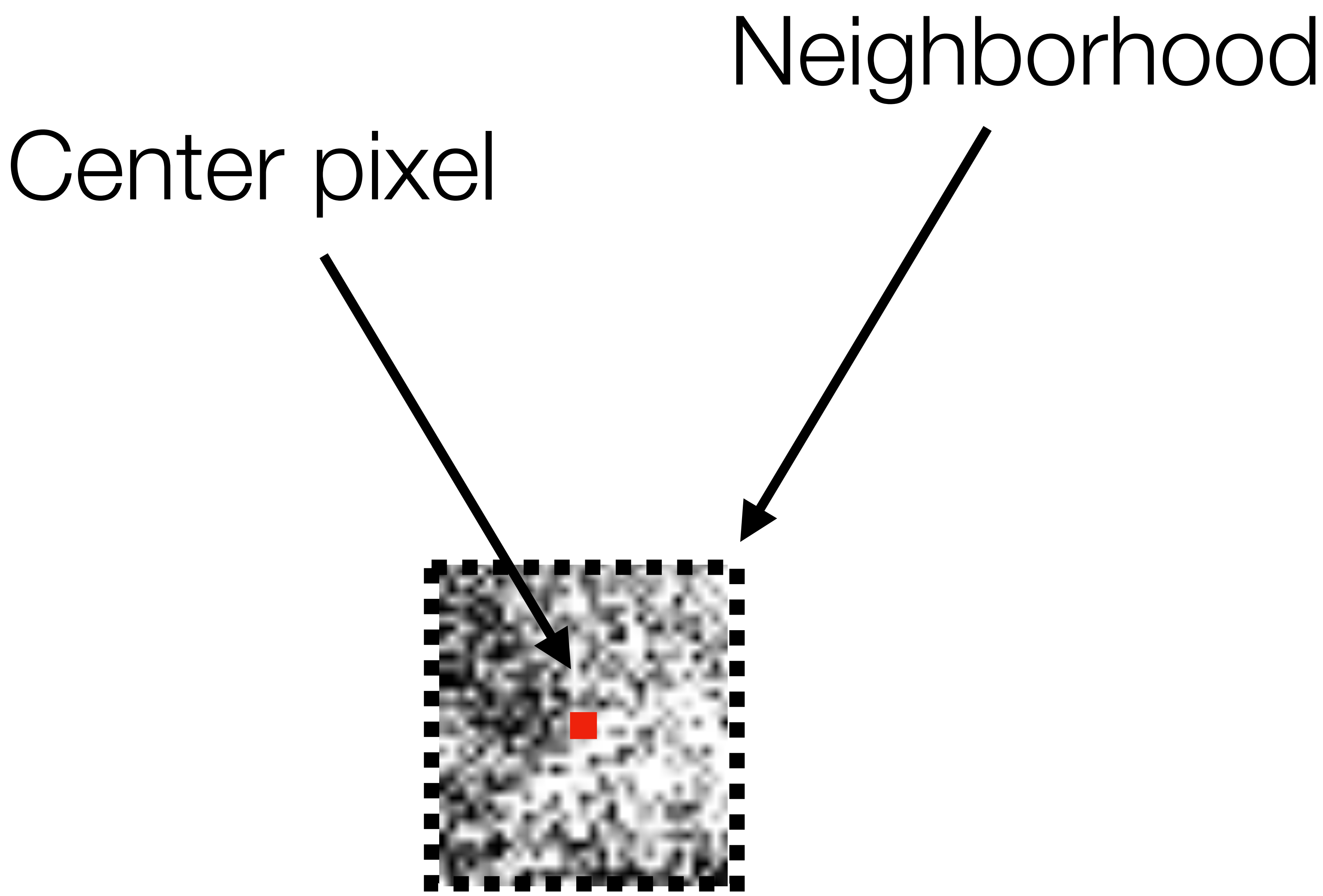


Photo by Fredo Durand

Image denoising

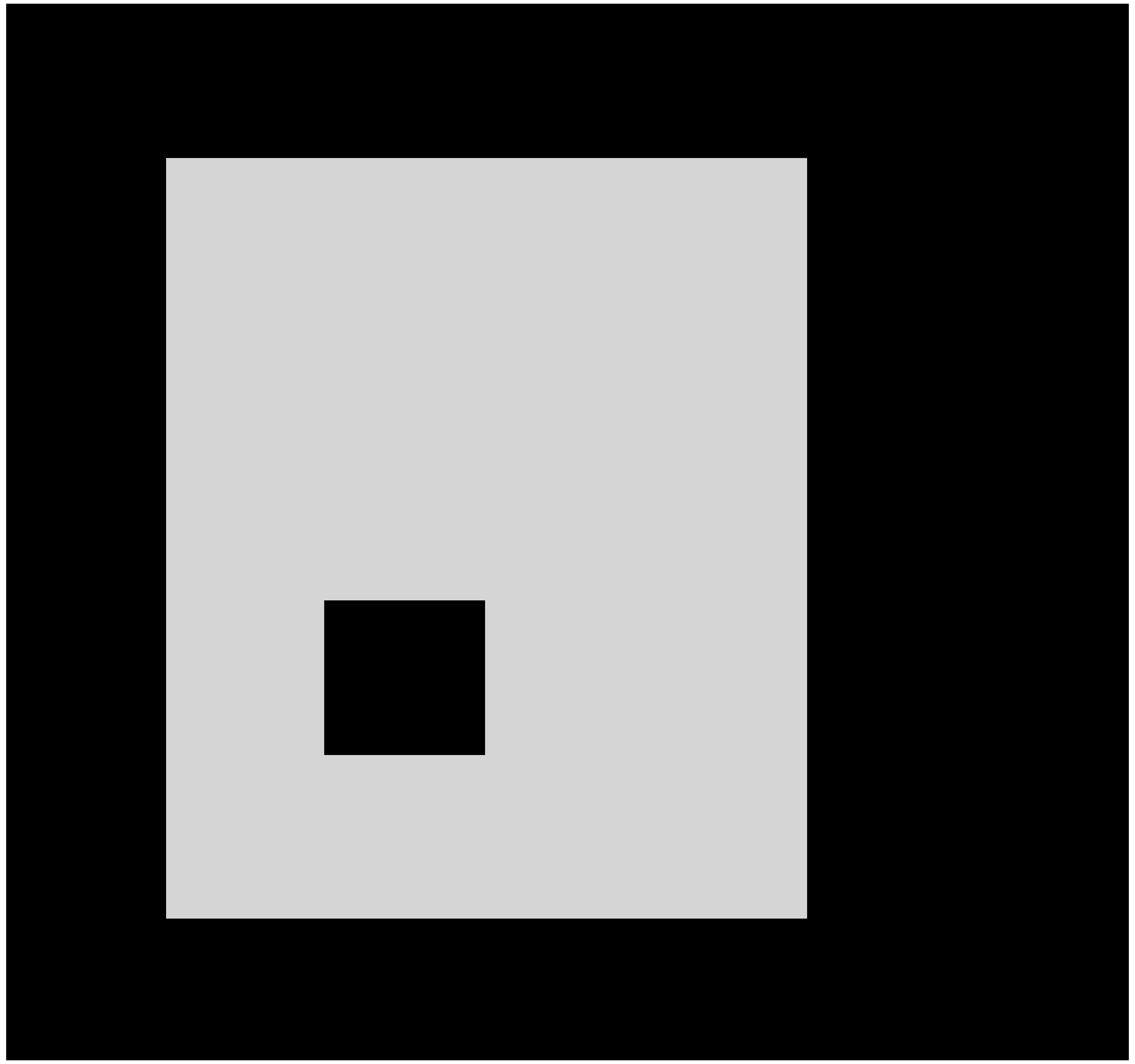


Replace each pixel with a weighted average of its neighborhood

Photo by Fredo Durand

Images as arrays

An image



Images as arrays

How it's represented in the computer

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Images as arrays

How it's represented in the computer

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

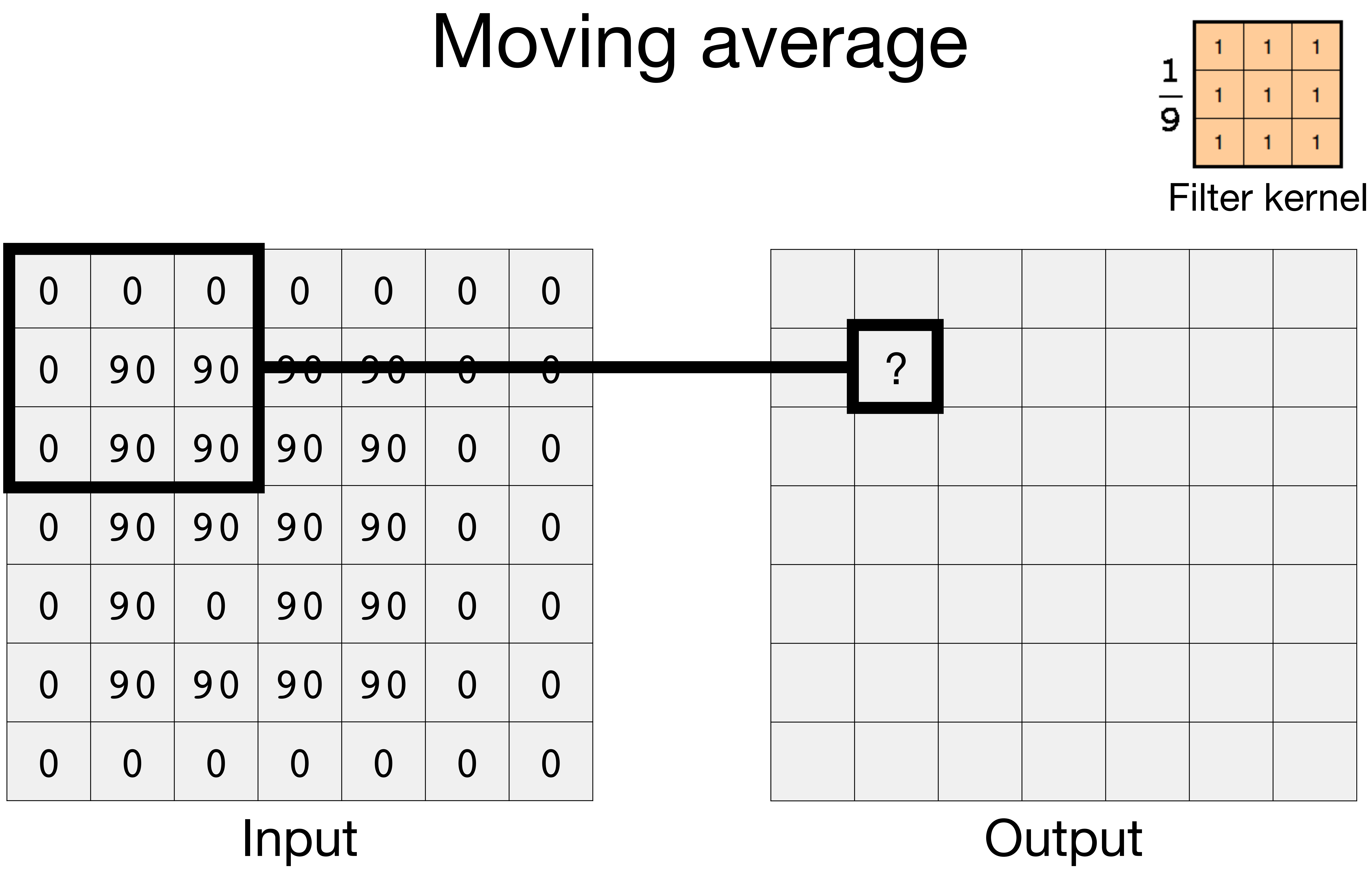
One solution: weighted sum

- Replace each pixel with a weighted average of the pixels around it.
- The weights are called the **filter kernel**.
- The weights for averaging the pixels in a 3x3 pixel neighborhood:

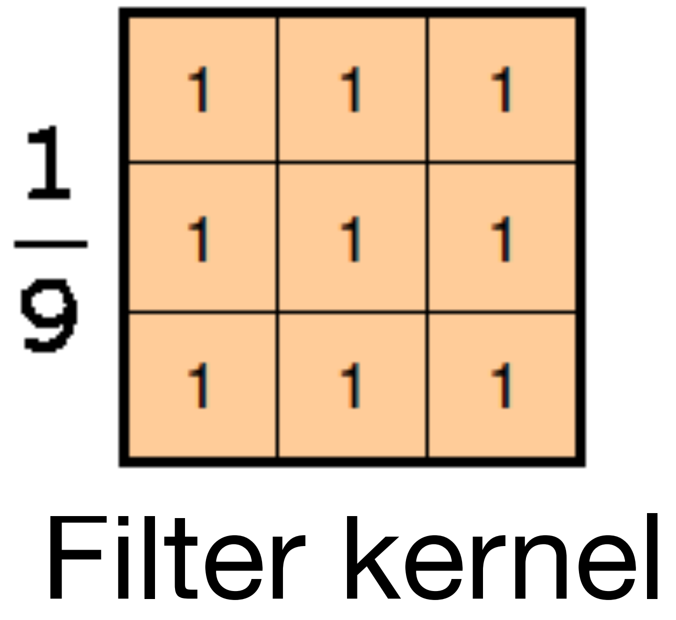
$$\begin{array}{|c|c|c|}\hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \end{array} = \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

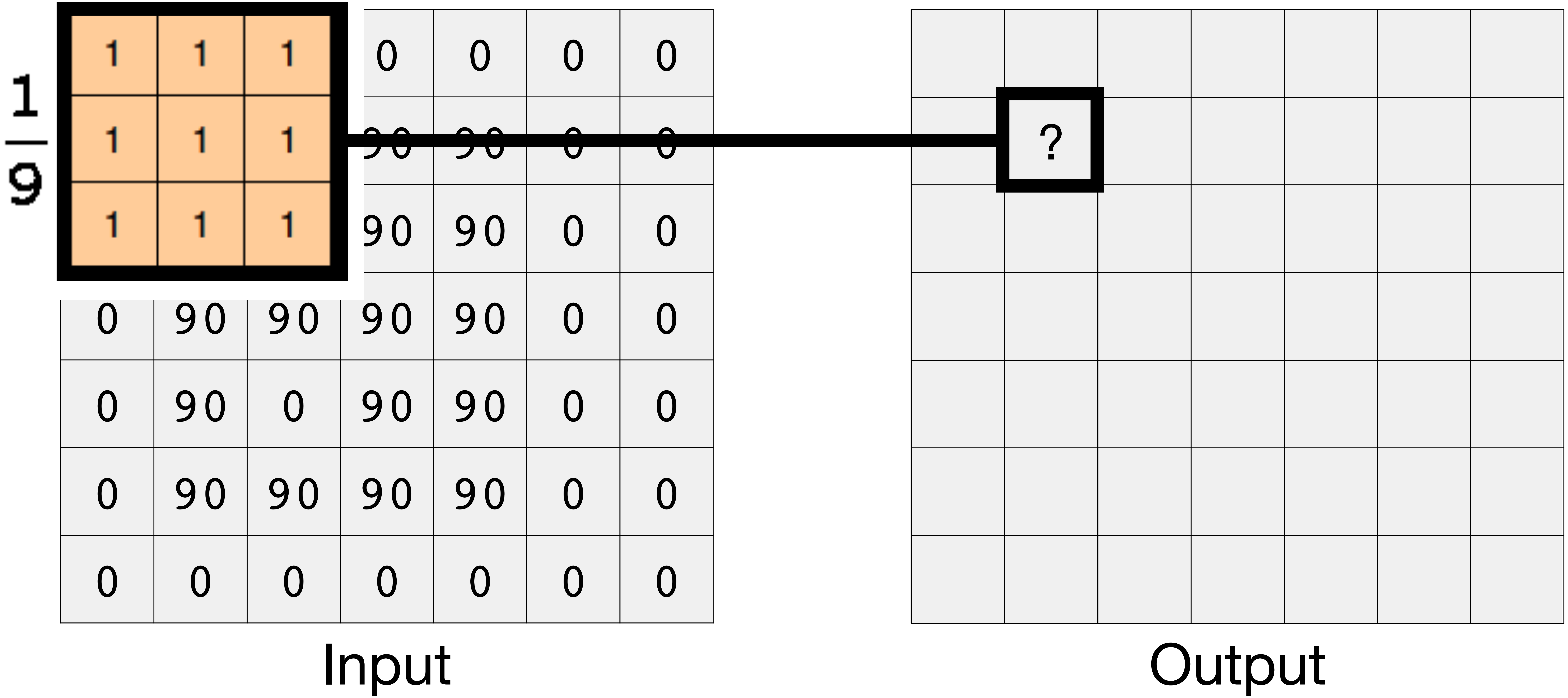
"box filter"

Source: D. Lowe

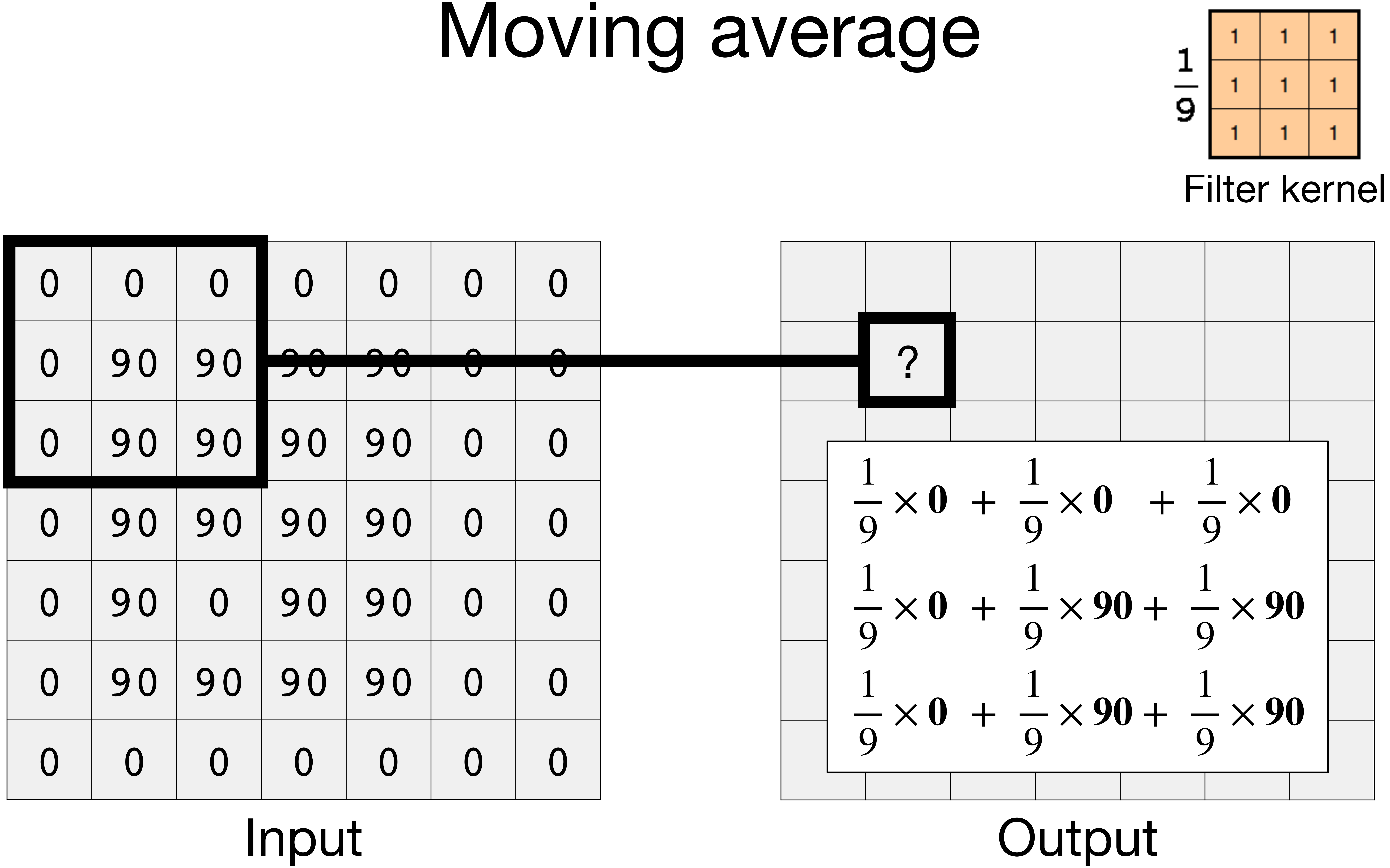

Moving average
$\frac{1}{9}$
1 1 1
1 1 1
1 1 1
Filter kernel
0 0 0 0 0 0 0
0 90 90 90 90 0 0
0 90 90 90 90 0 0
0 90 90 90 90 0 0
0 90 0 90 90 0 0
0 90 90 90 90 0 0
0 0 0 0 0 0 0
?
Input
Output

Moving average

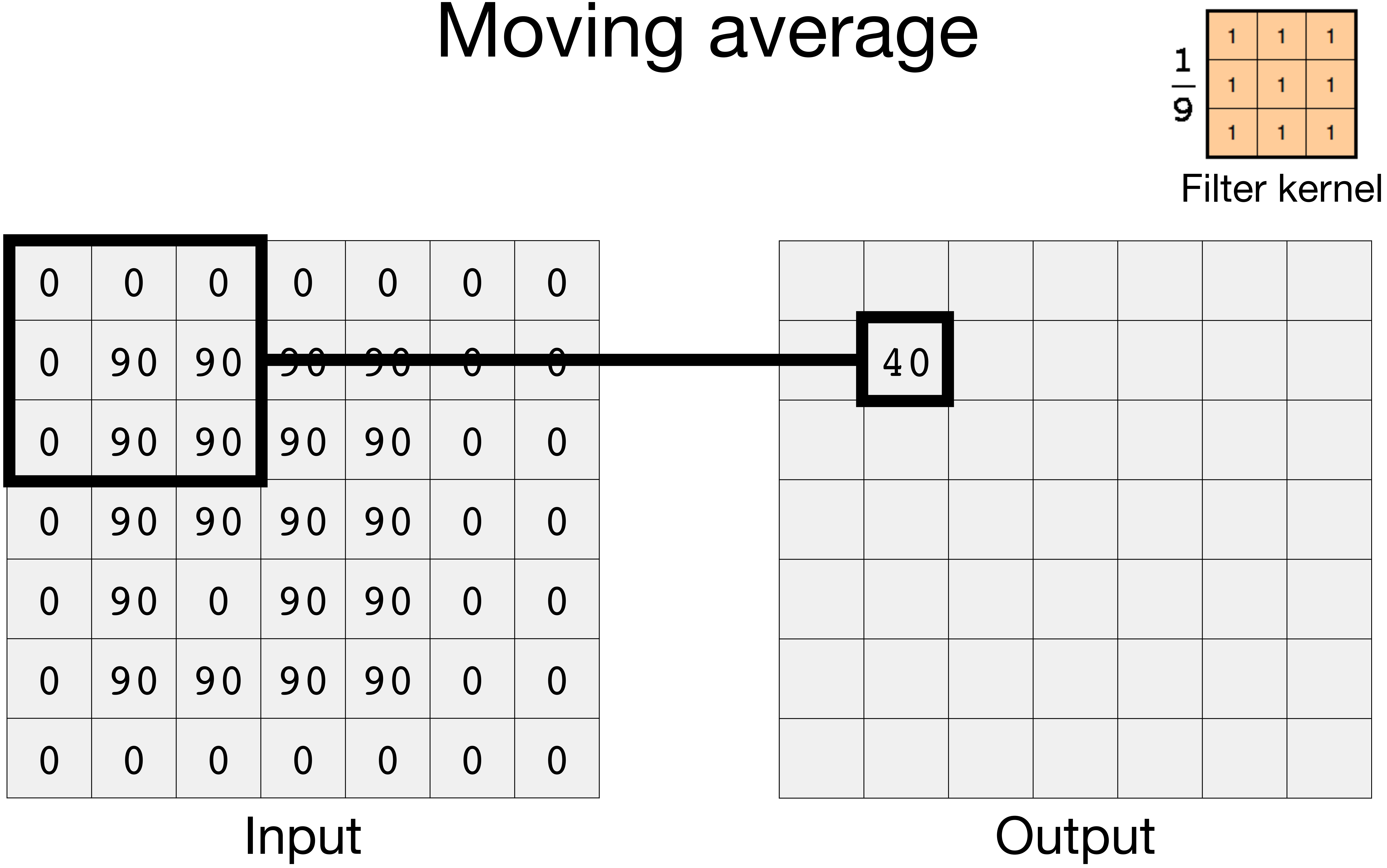




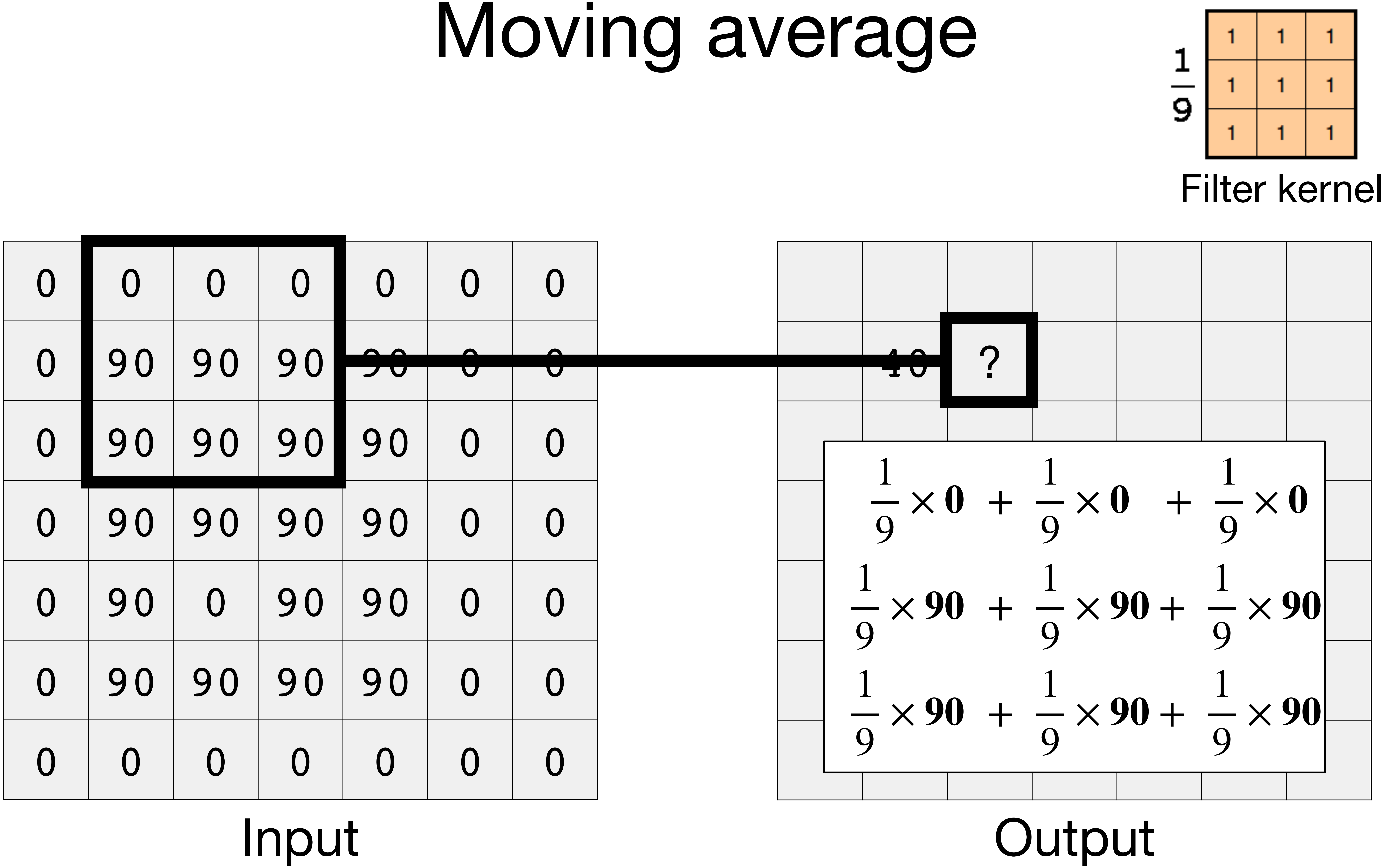
Moving average



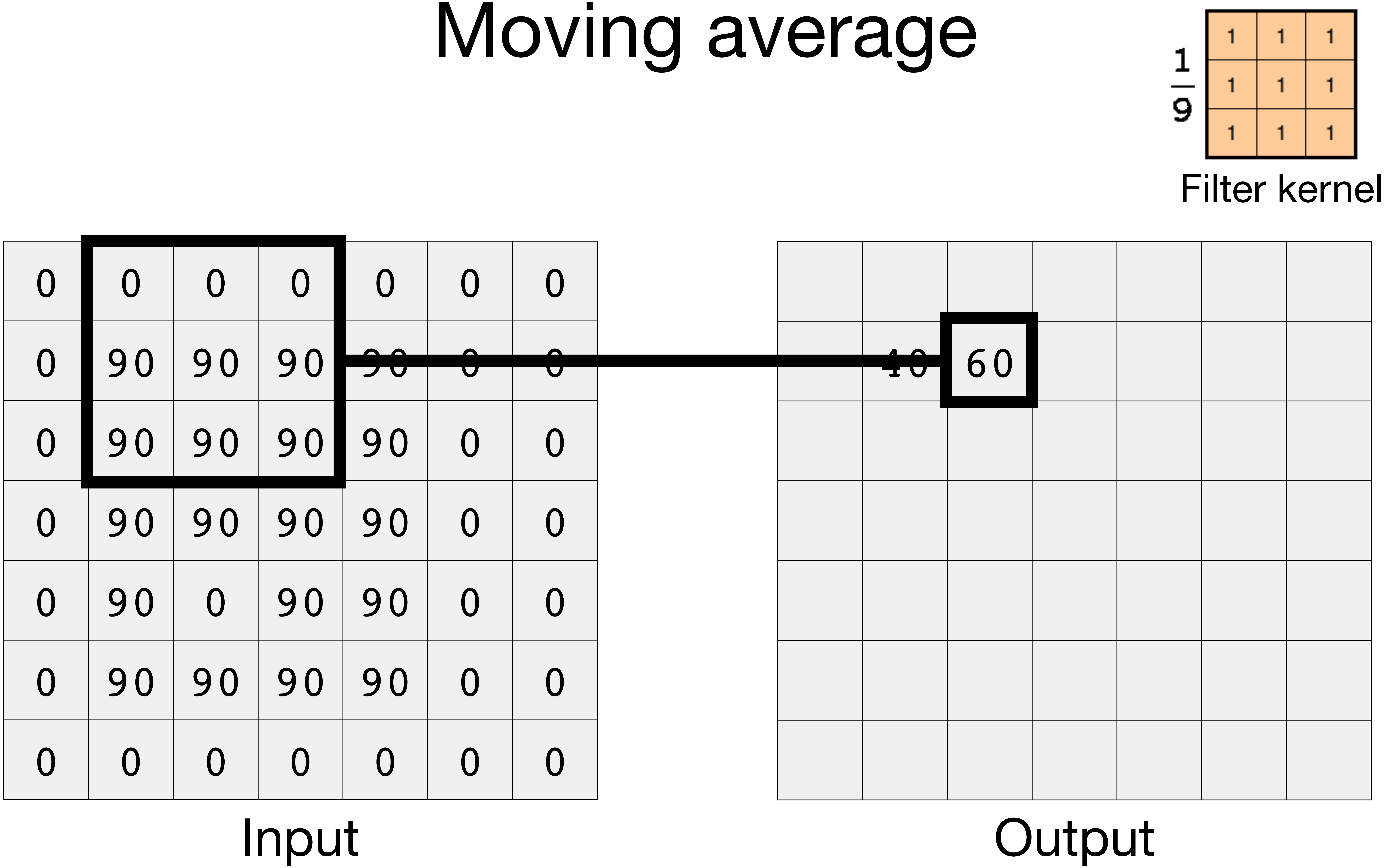
Moving average



Moving average



Moving average



Moving average

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Filter kernel

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60				
		?				

Output

Moving average

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Filter kernel

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60				
		80				

Output

Moving average

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Filter kernel

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60	60	40	20	
	60	90	60	40	20	
	50	80	80	60	30	
	50	80	80	60	30	
	30	50	50	40	20	

Output

Moving average

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Filter kernel

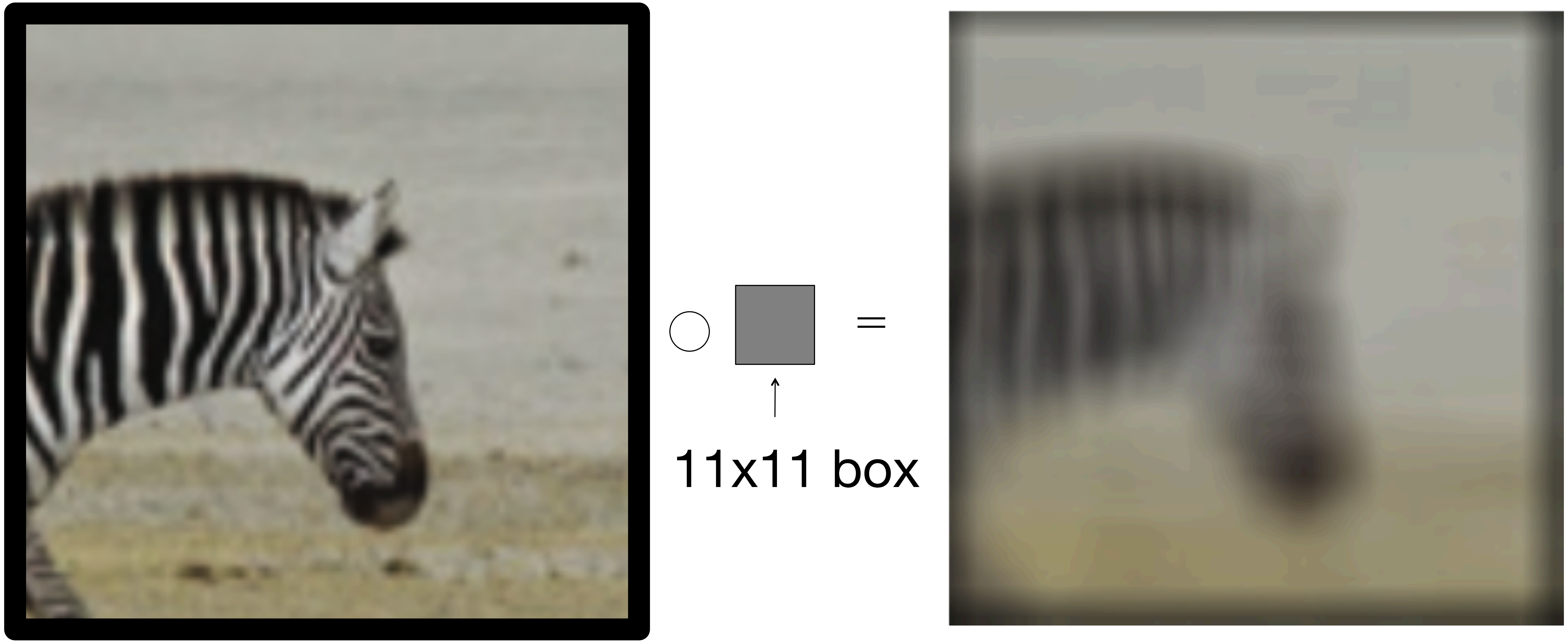
0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60	60	40	20	
	60	90	60	40	20	
	50	80	80	60	30	
?	50	80	80	60	30	
	30	50	50	40	20	

Output

Handling boundaries



Source: Torralba, Freeman, Isola

Moving average

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Filter kernel

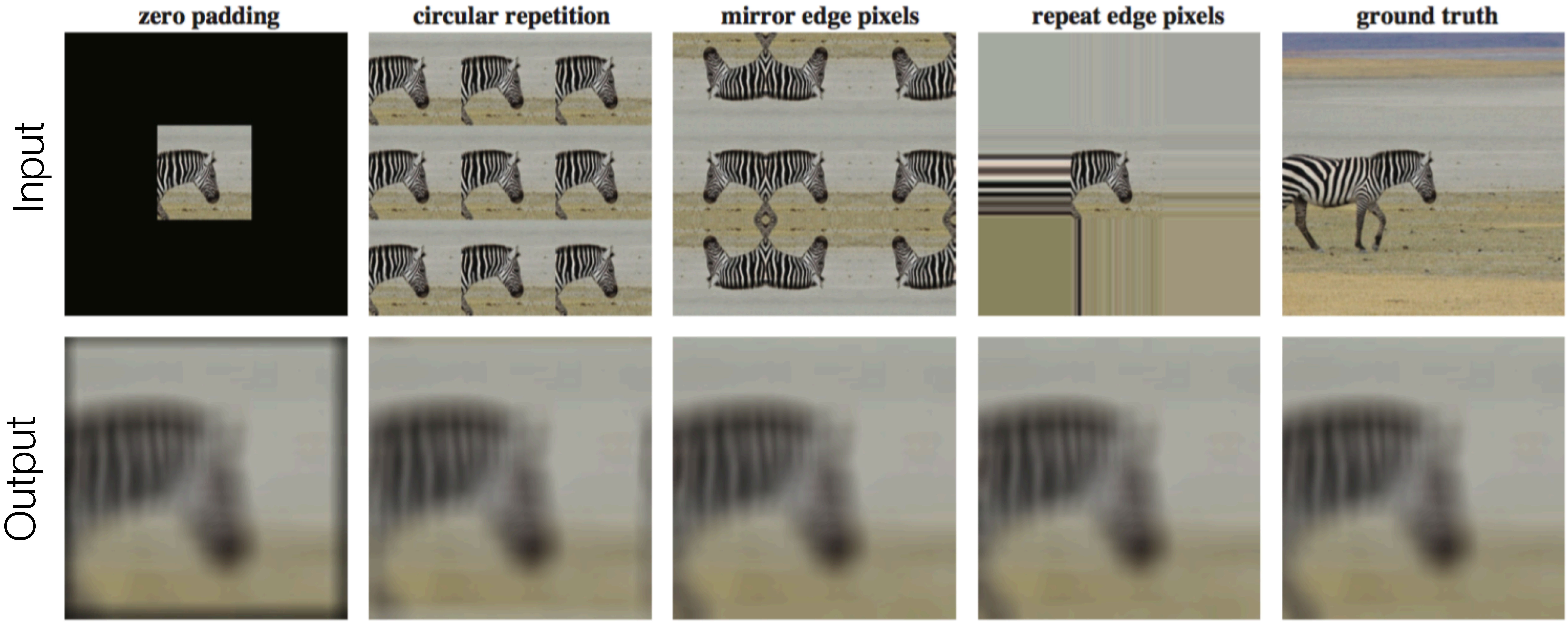
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	90	90	90	90	0	0	0
0	0	90	90	90	90	0	0	0
0	0	90	90	90	90	0	0	0
0	0	90	0	90	90	0	0	0
0	0	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Input

	40	60	60	40	20	
	60	90	60	40	20	
	50	80	80	60	30	
30	50	80	80	60	30	
	30	50	50	40	20	

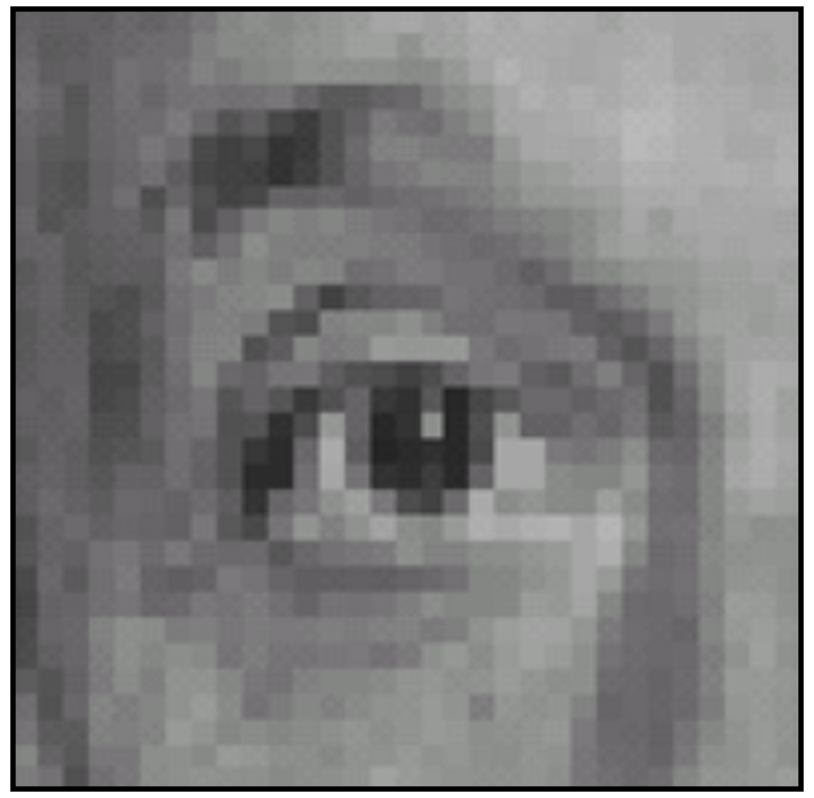
Output

Handling boundaries



Source: Torralba, Freeman, Isola

Practice with neighborhood filtering



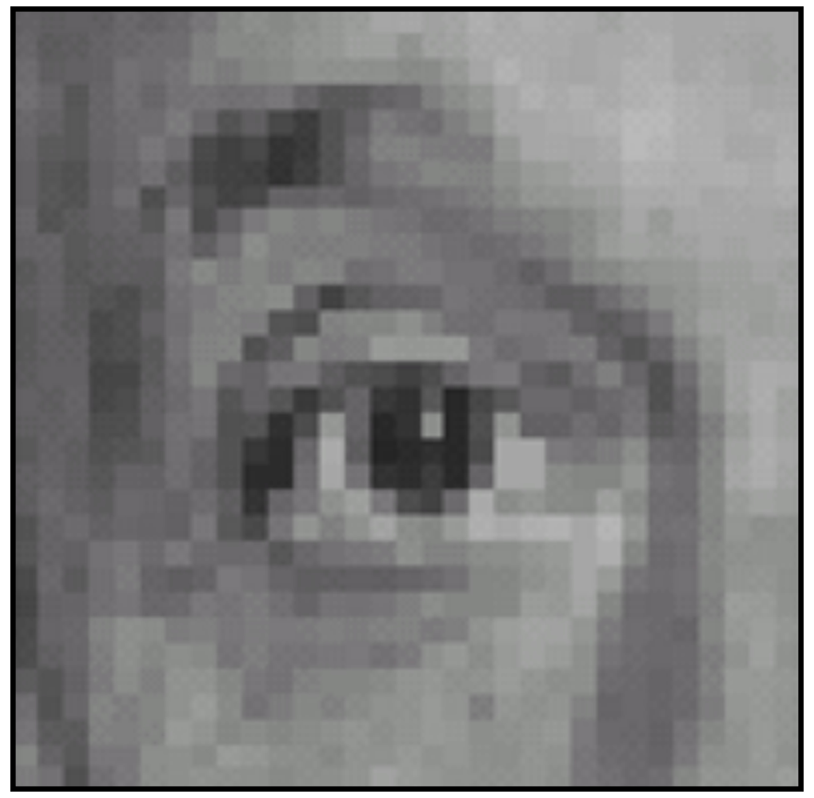
Original

$*$

0	0	0
0	1	0
0	0	0

?

Practice with neighborhood filtering

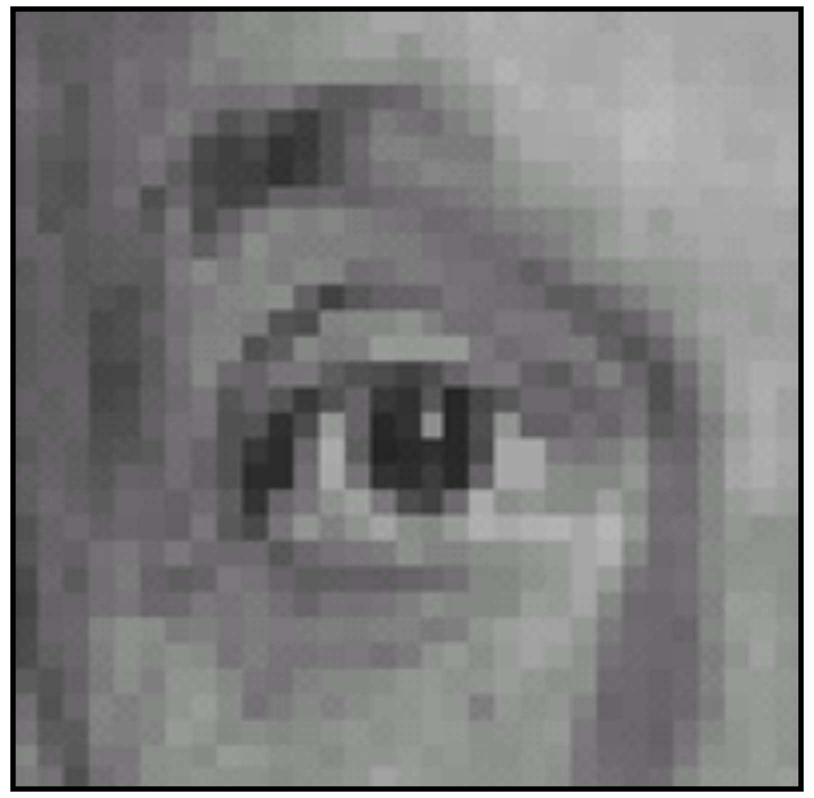


Original

*

0	0	0
0	1	0
0	0	0

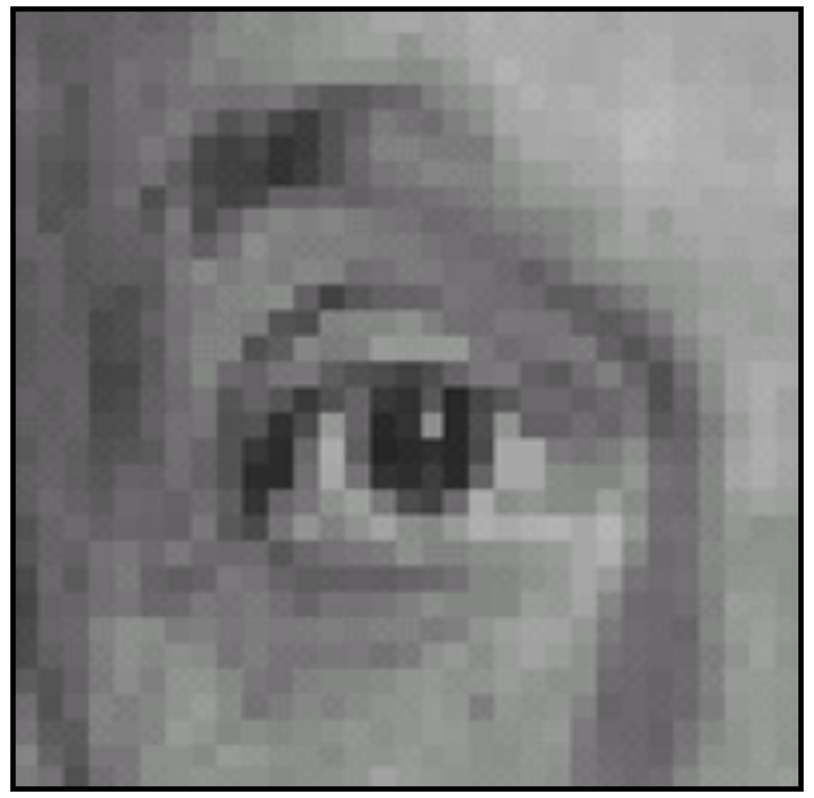
“Impulse”



Filtered
(no change)

Source: D. Lowe

Practice with neighborhood filtering



Original

*

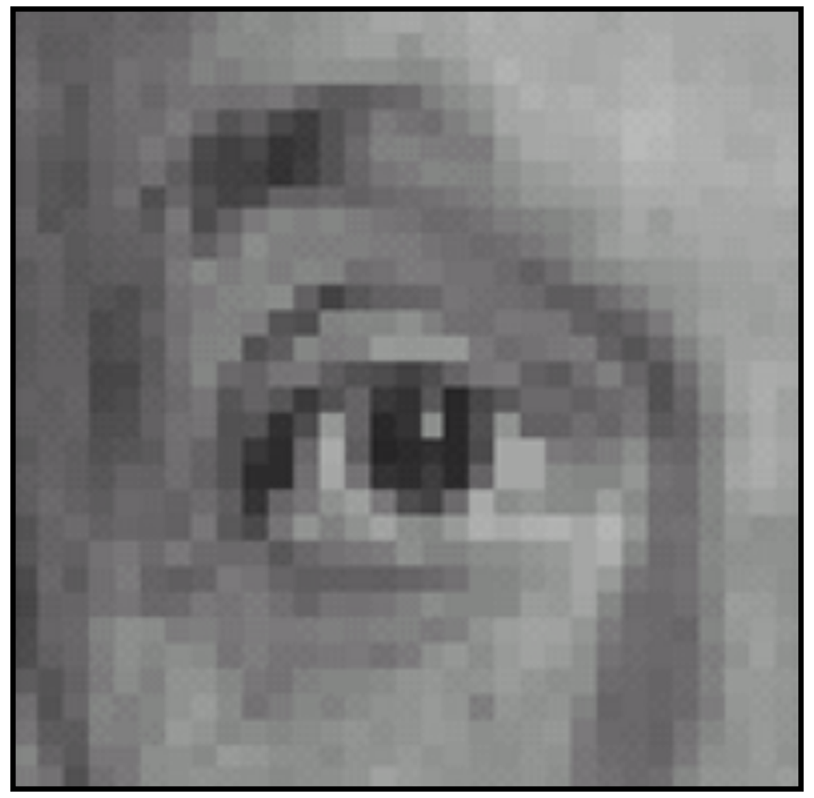
0	0	0
0	0	1
0	0	0

“Translated Impulse”

?

Source: D. Lowe

Practice with neighborhood filtering

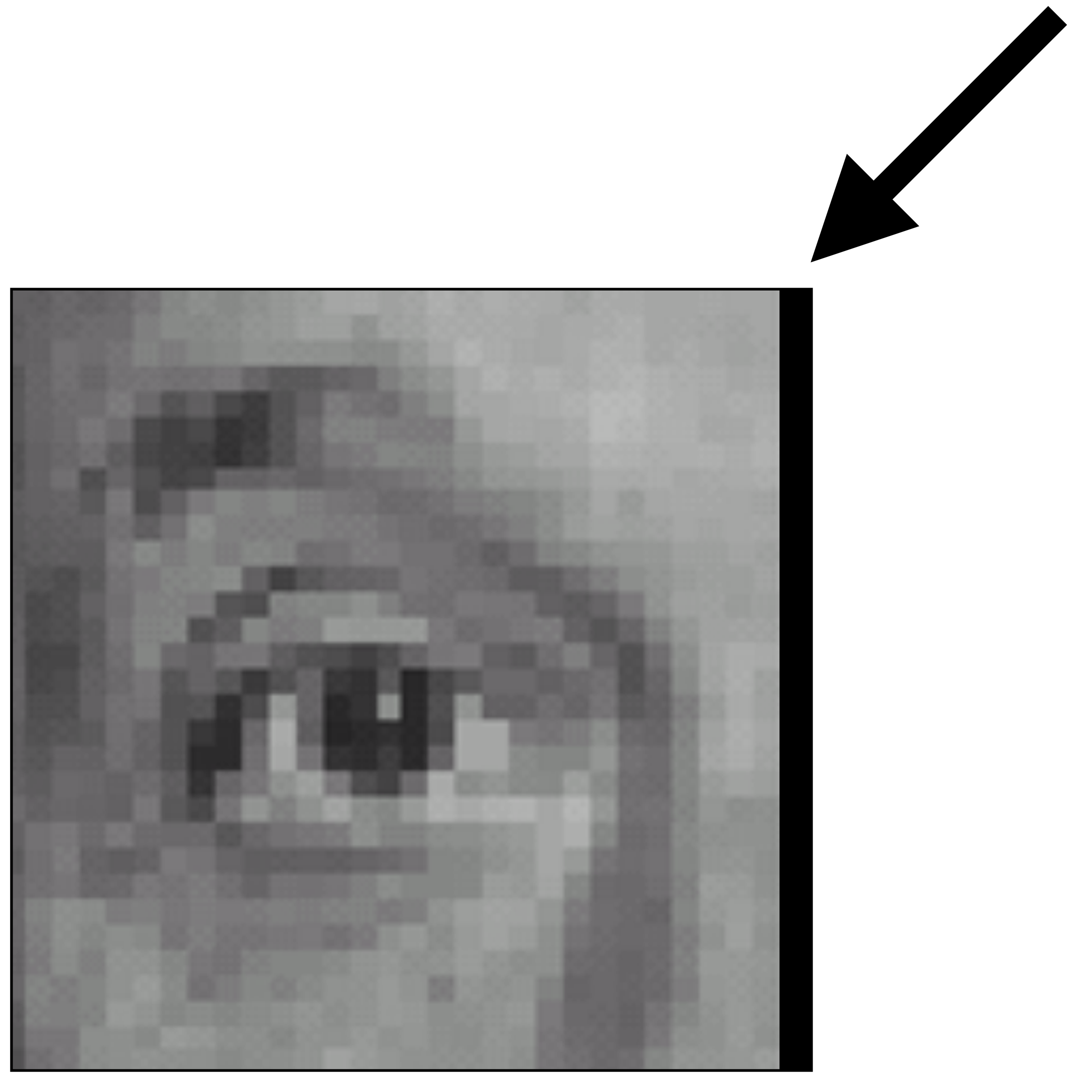


Original

*

0	0	0
0	0	1
0	0	0

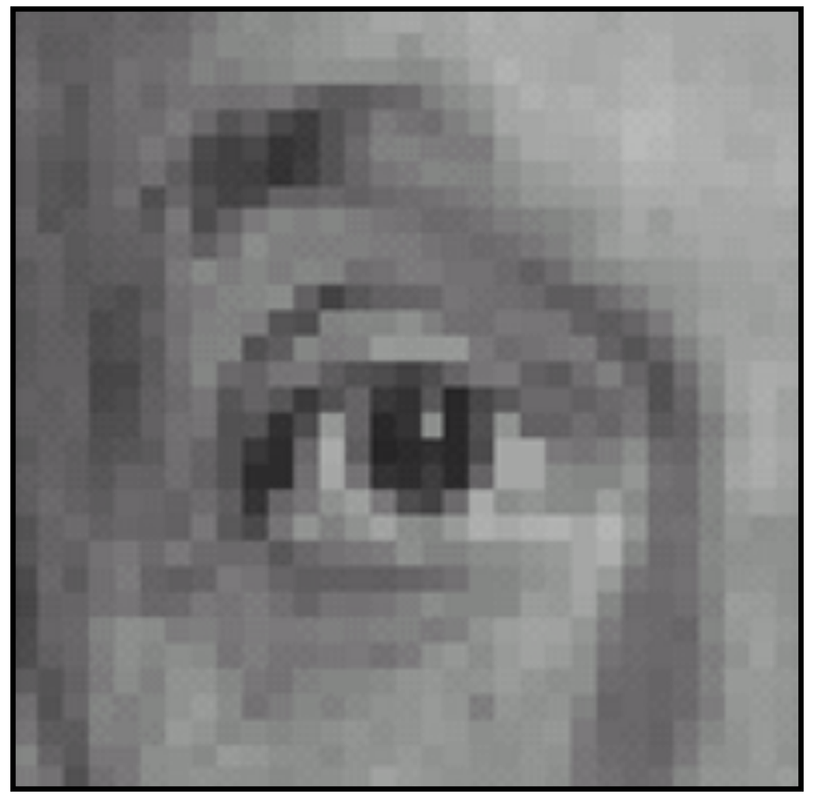
Zero padding



Shifted left
By 1 pixel

Source: D. Lowe

Practice with neighborhood filtering



Original

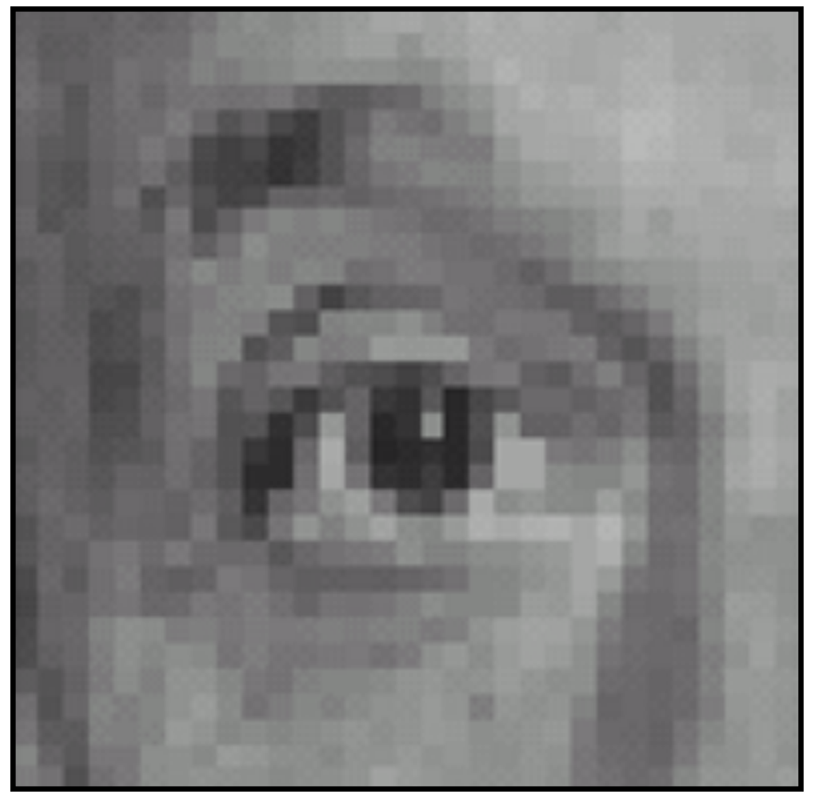
$$* \ \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

"Box filter"

?

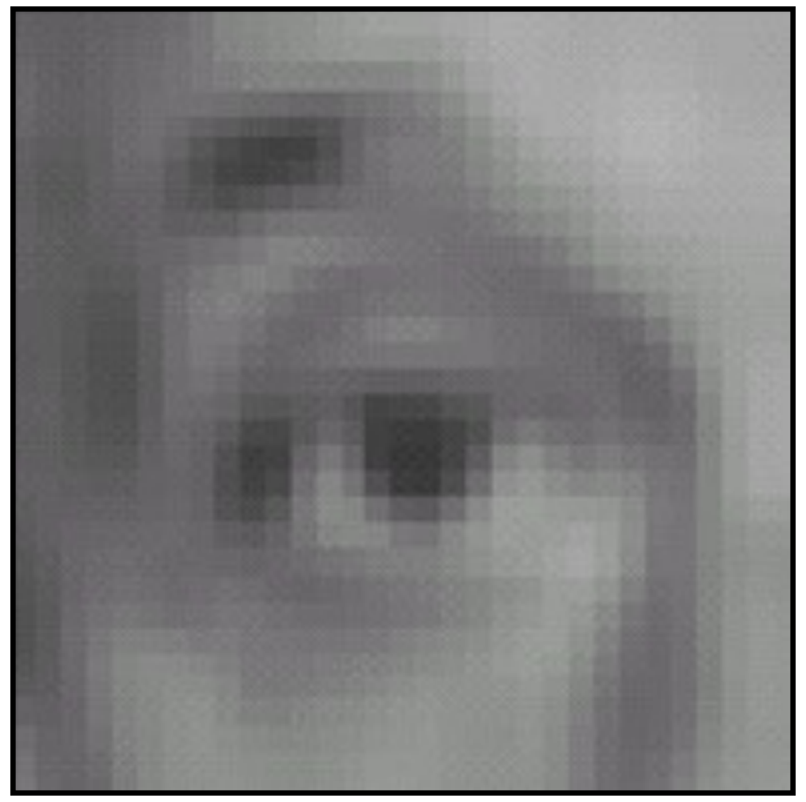
Source: D. Lowe

Practice with neighborhood filtering



Original

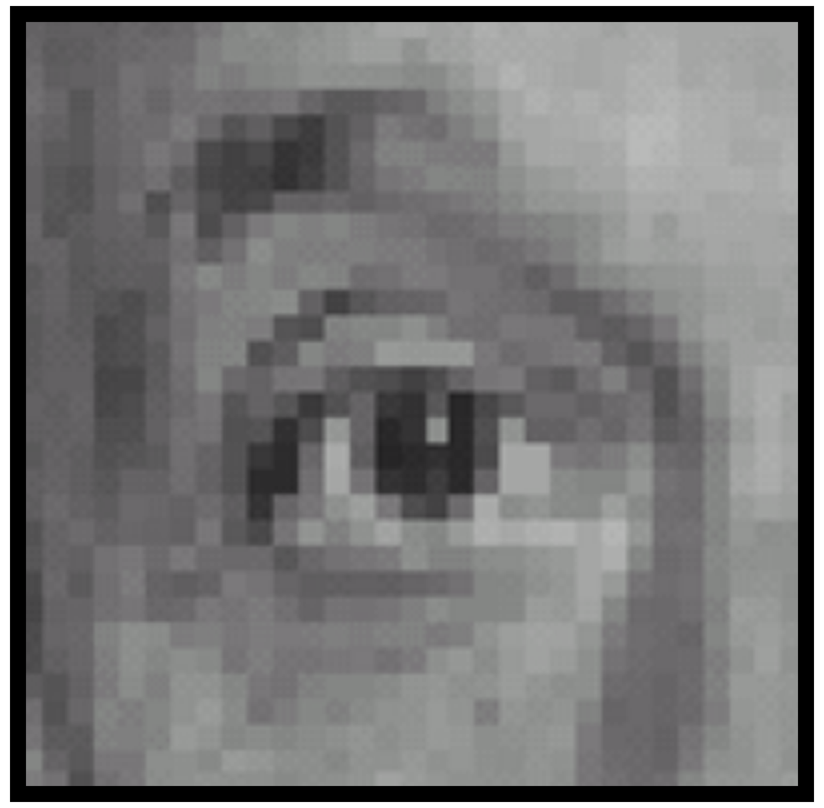
$$* \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$



Blur

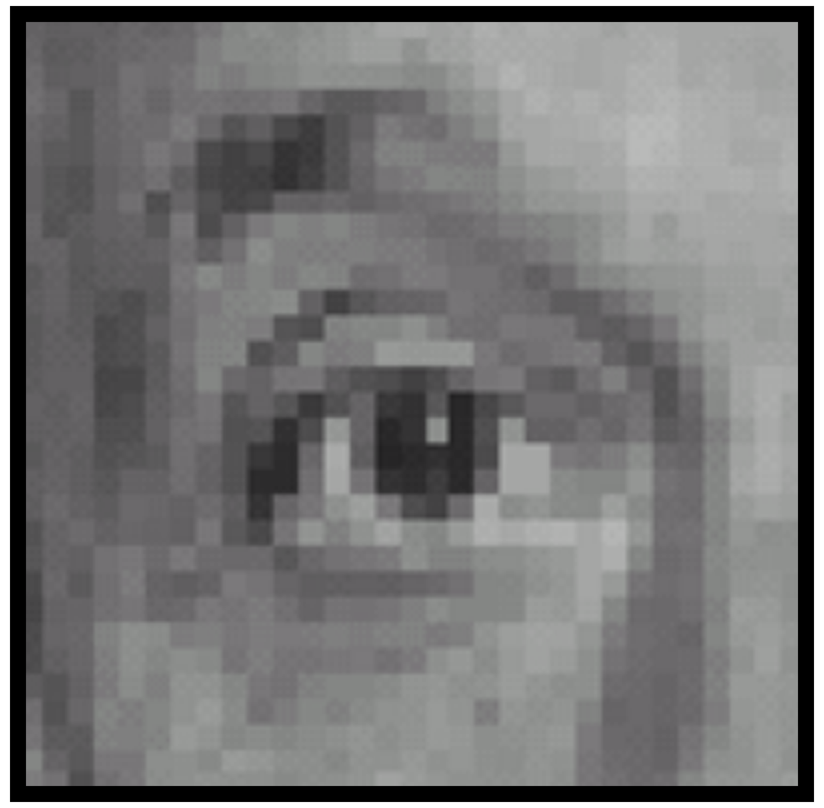
Source: D. Lowe

Practice with neighborhood filtering


Original

$$
* \quad
\begin{bmatrix}
-\frac{1}{9} & -\frac{1}{9} & -\frac{1}{9} \\
-\frac{1}{9} & \frac{17}{9} & -\frac{1}{9} \\
-\frac{1}{9} & -\frac{1}{9} & -\frac{1}{9}
\end{bmatrix}
\quad ?
$$

Practice with neighborhood filtering



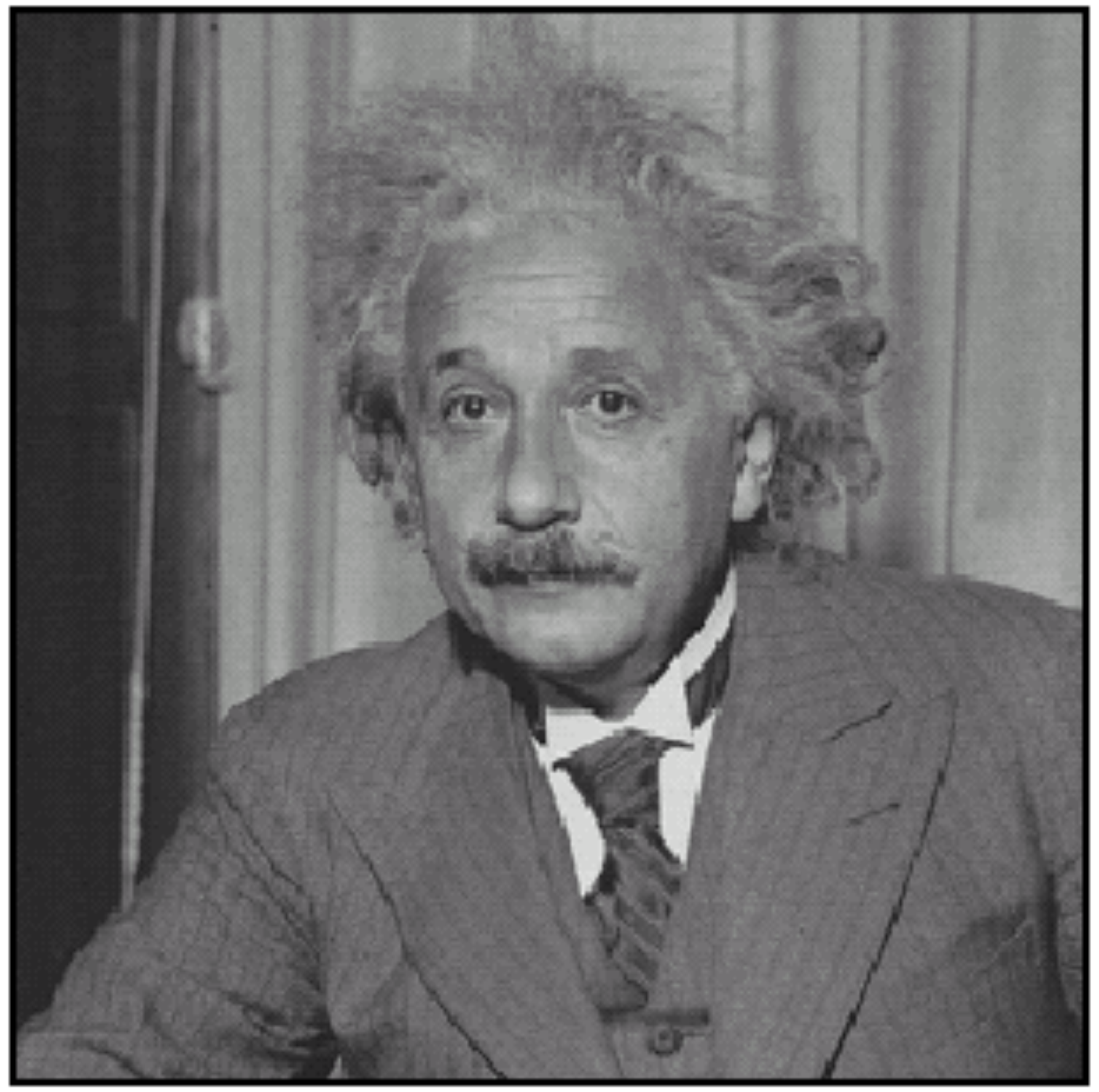
Original

$$* \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right)$$

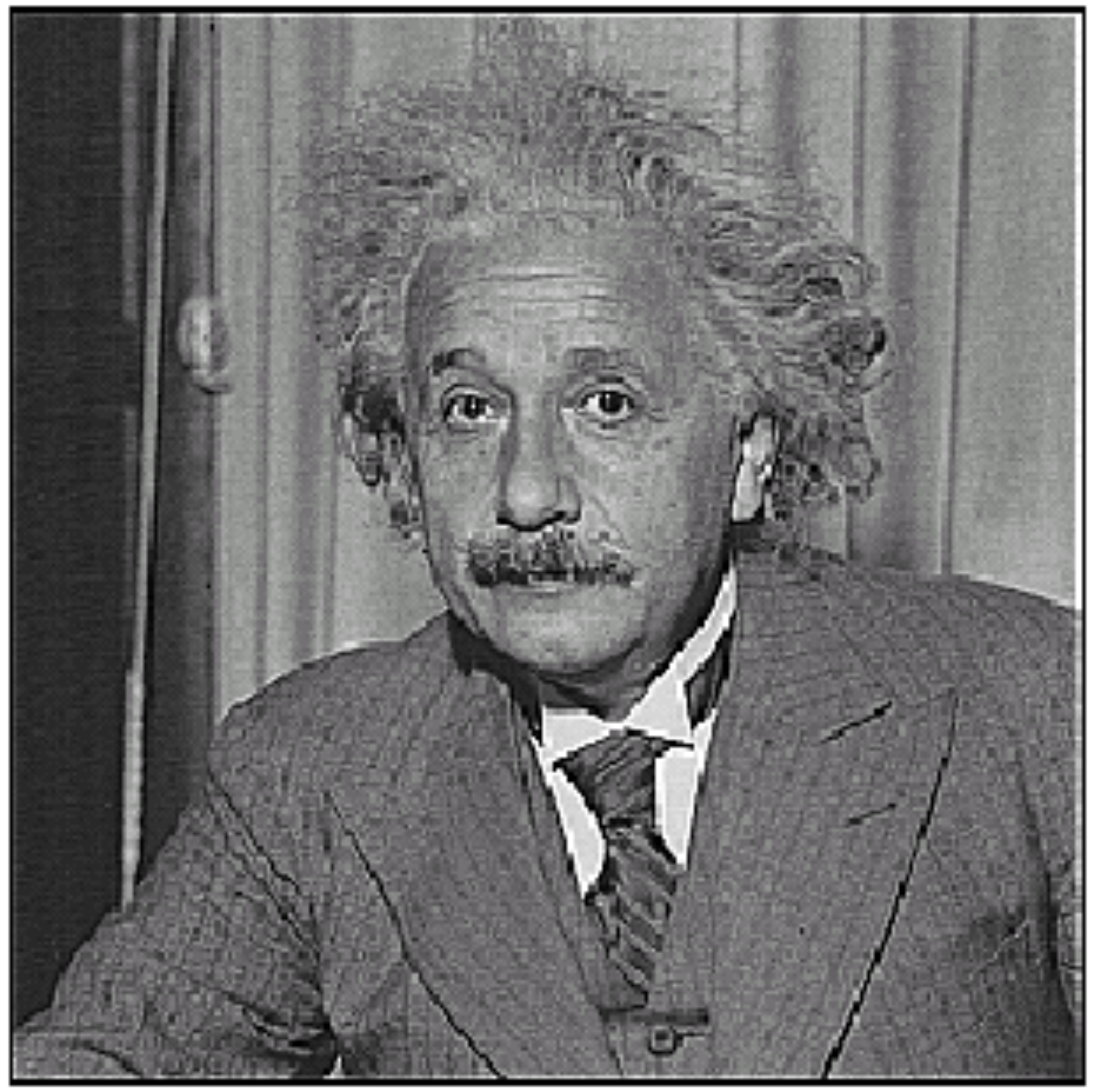
Sharpening filter: Accentuates differences with local average

Source: D. Lowe

Sharpening



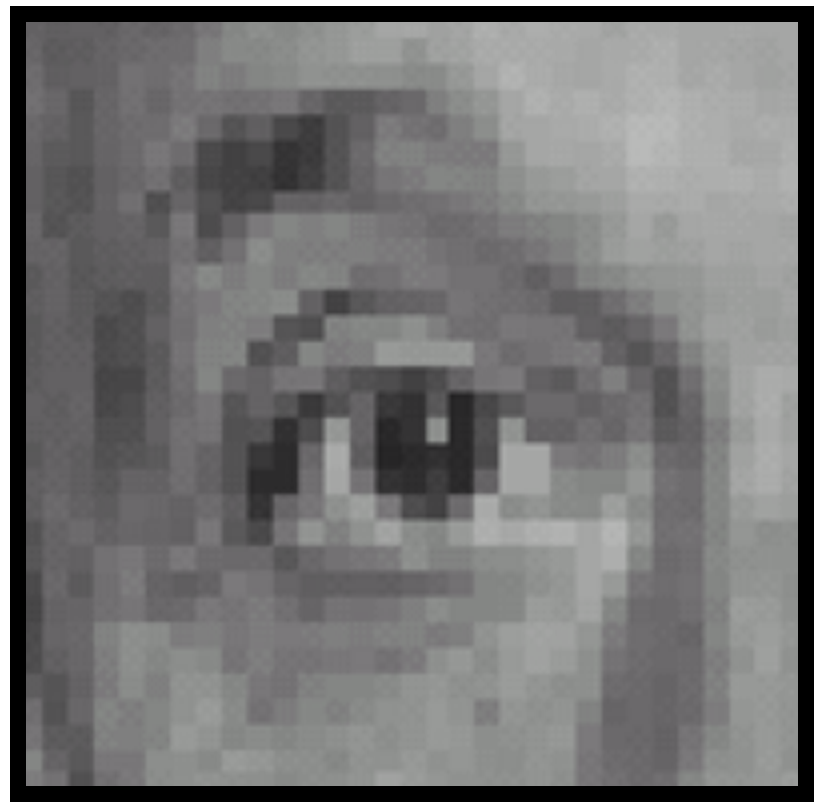
Before



After

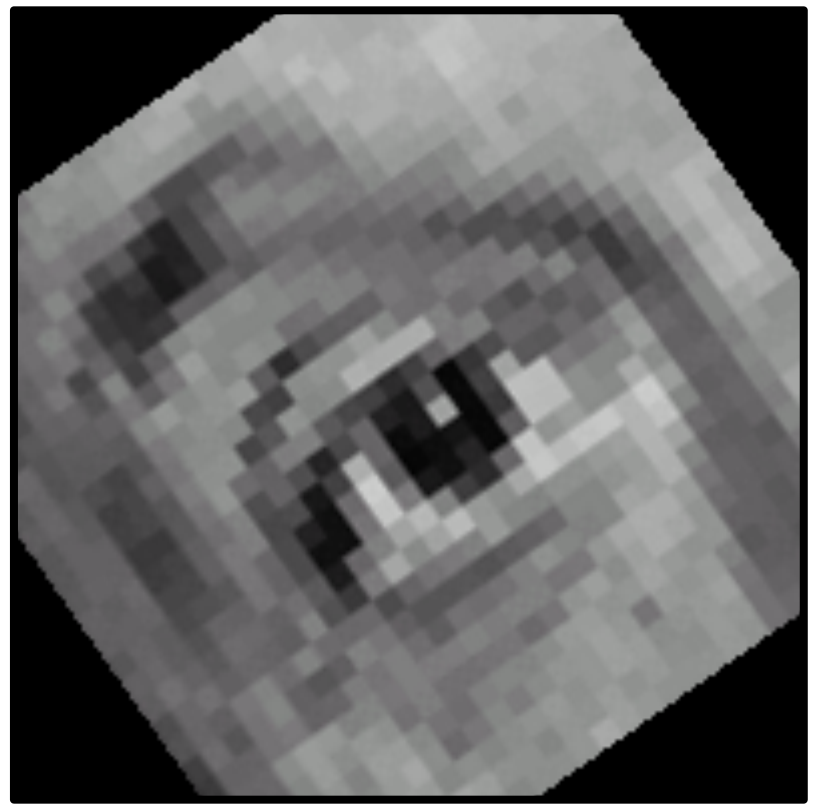
Source: D. Lowe

Practice with neighborhood filtering



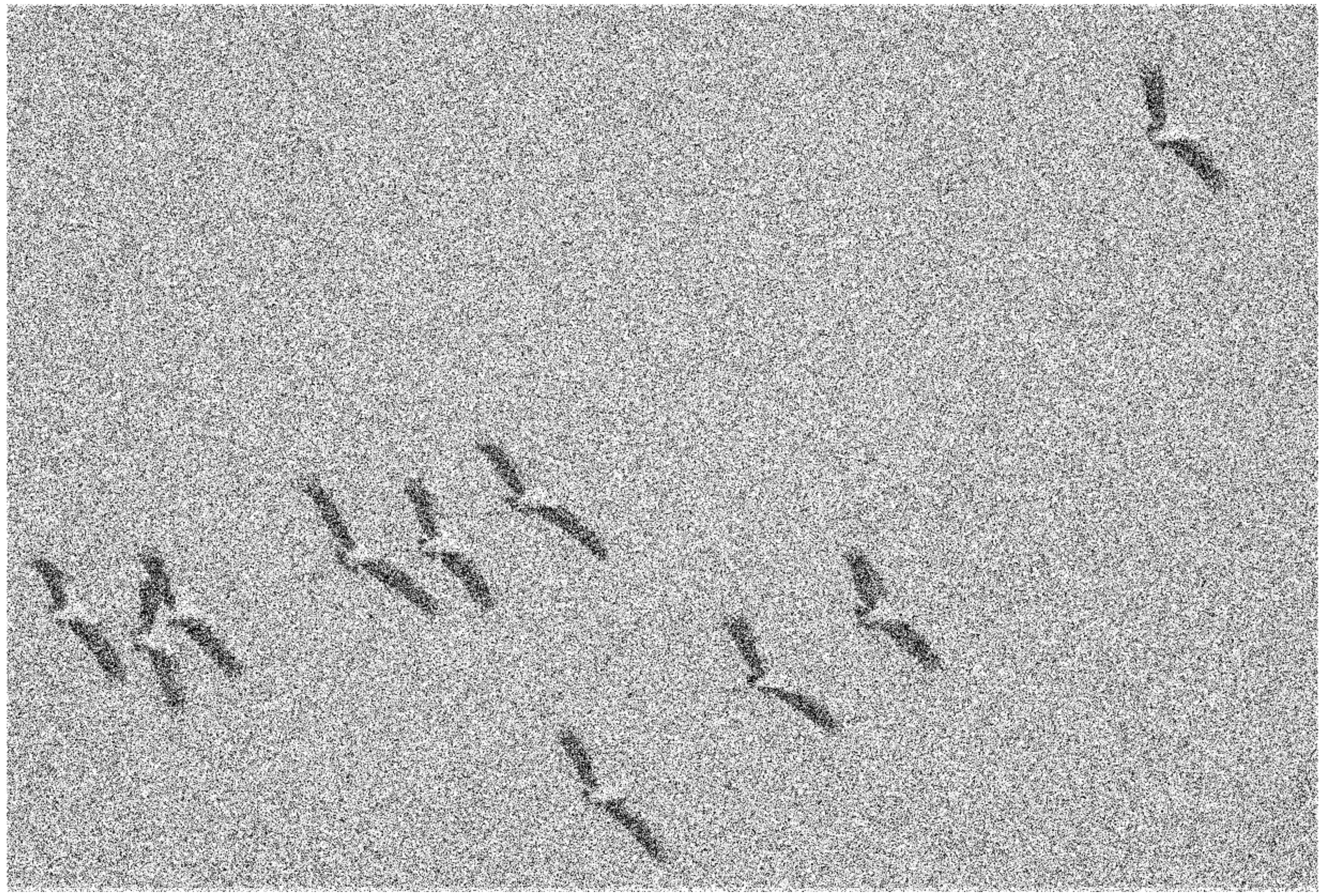
Original

* ?



Is there a neighborhood filter that does this?

Two computer vision problems



Denoising

Edge detection

Edge detection

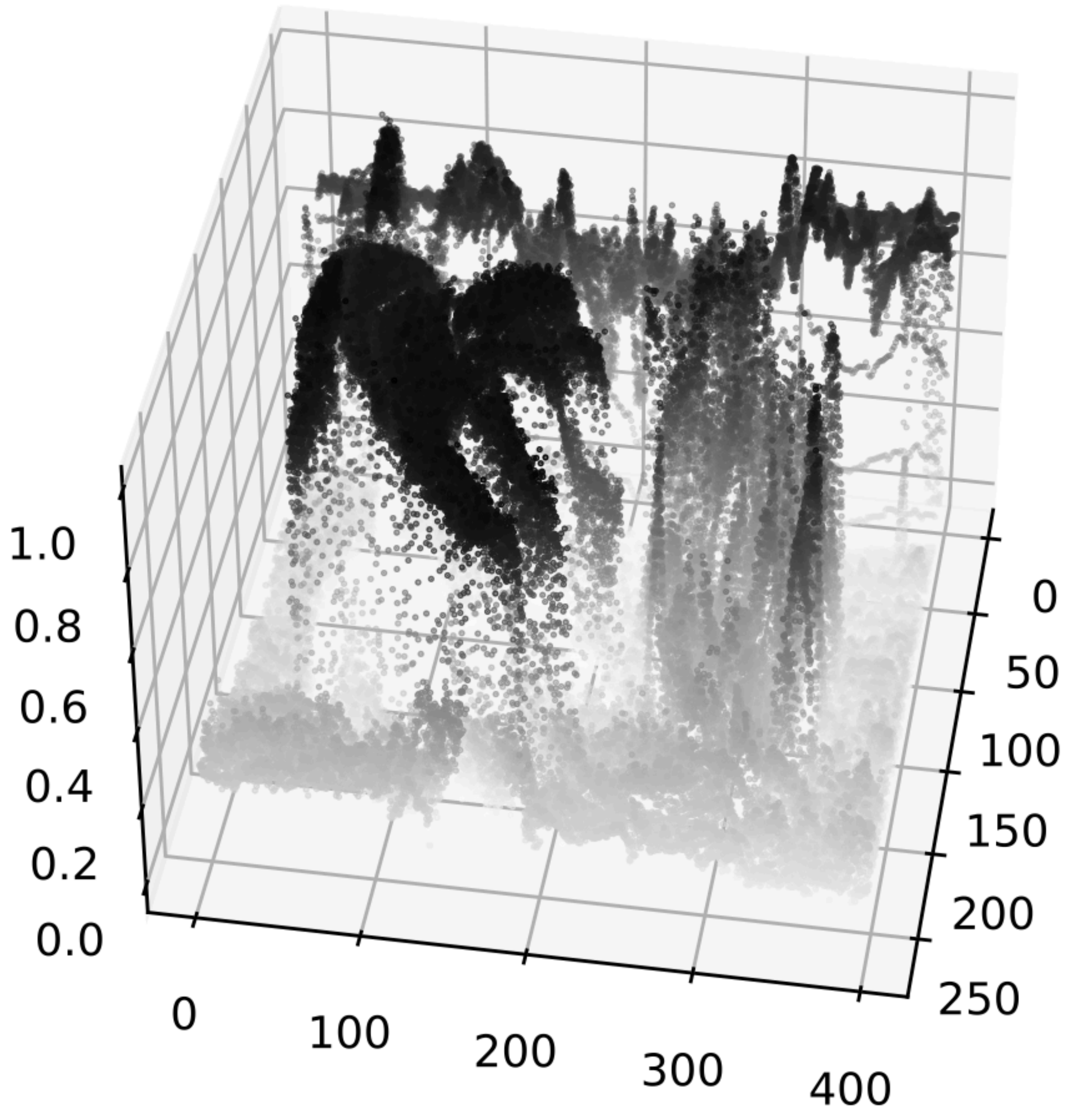
Edge detection

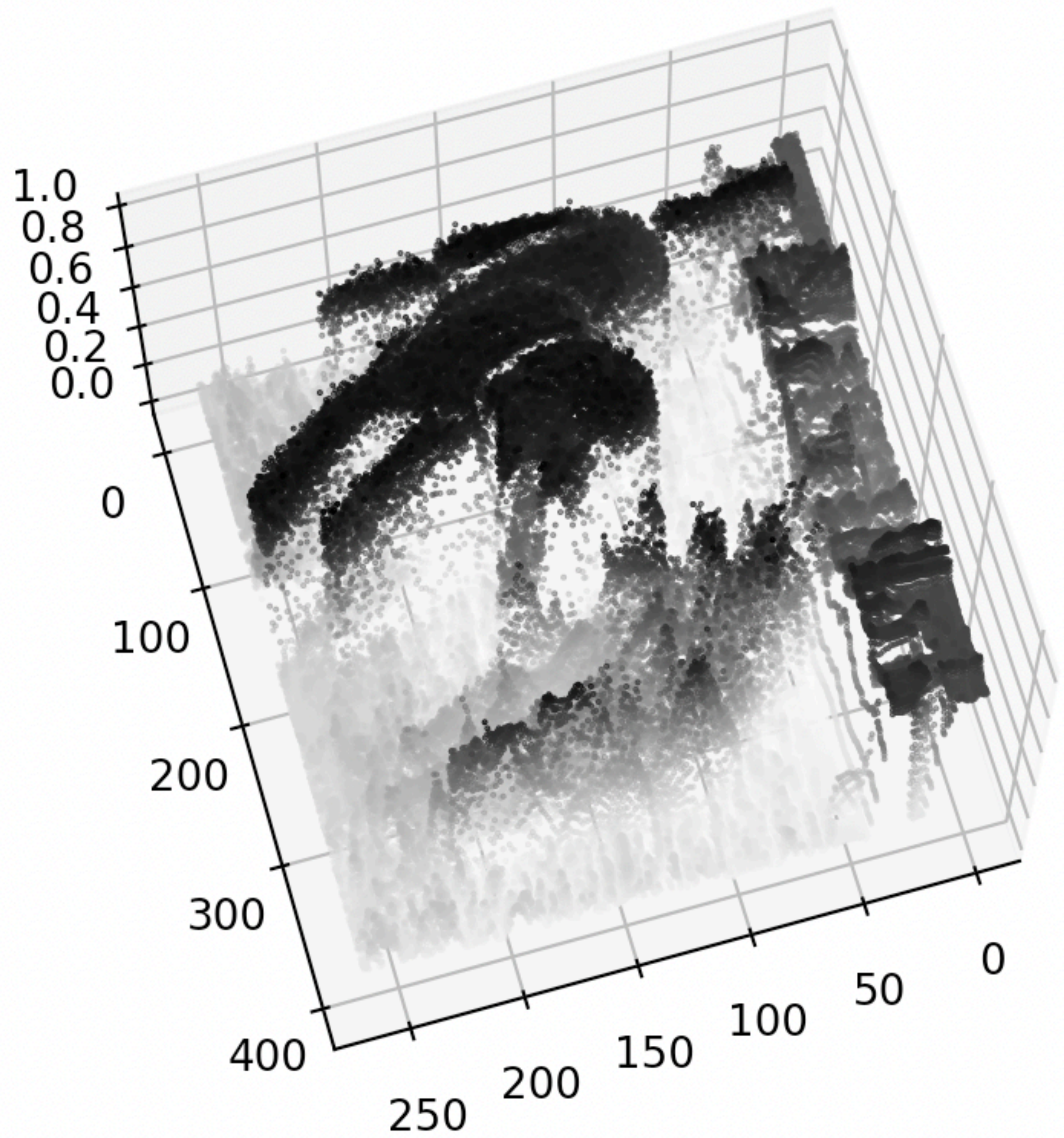
Edge detection on grayscale images

First, convert to grayscale:

$$I_{gray}[y,x] = \frac{1}{3}\left(I_{rgb}[y,x,0] + I_{rgb}[y,x,1] + I_{rgb}[y,x,2]\right)$$

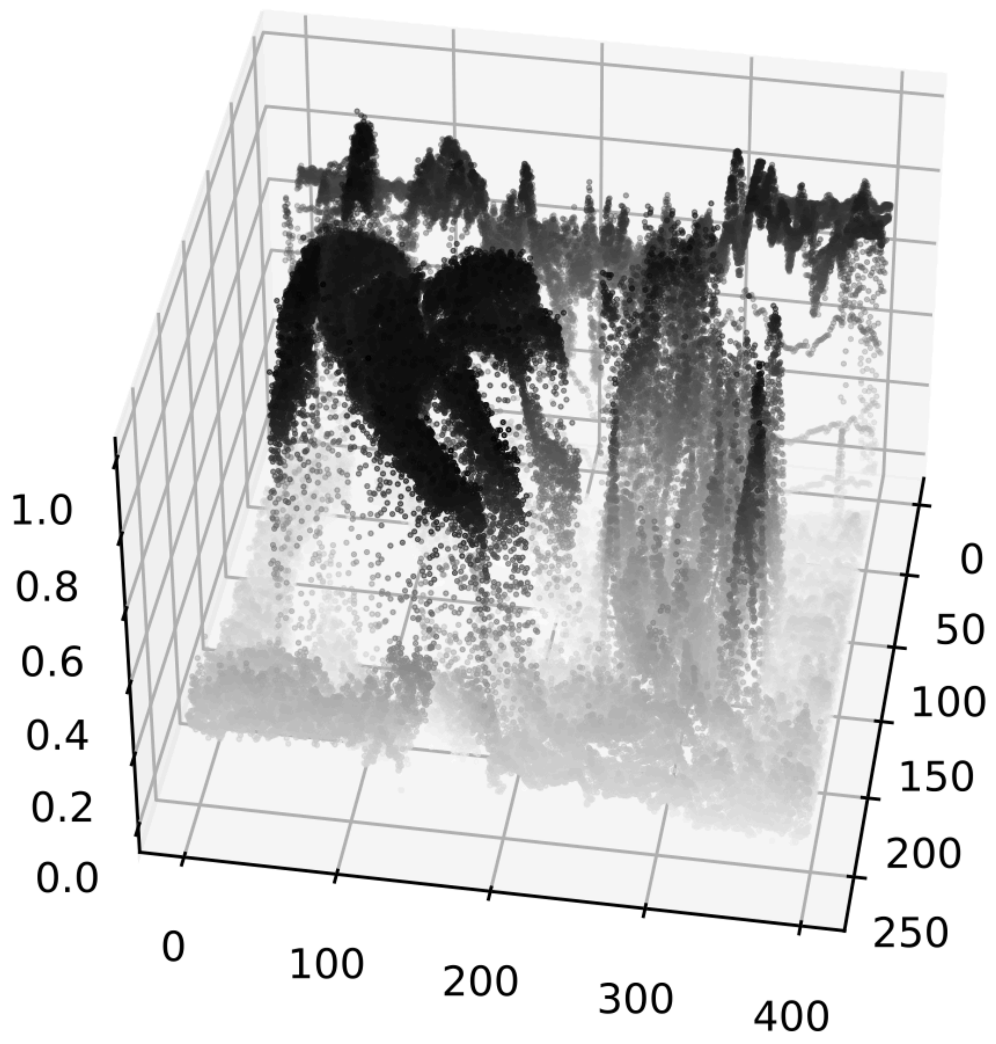
Images as functions





$\mathbf{I}(x, y) =$ intensity at pixel (x, y)

Finding edges in the image



$\mathbf{I}(x, y)$

Image gradient:

$$\nabla \mathbf{I} = \left(\frac{\partial \mathbf{I}}{\partial x}, \frac{\partial \mathbf{I}}{\partial y} \right)$$

Approximation of image derivative:

$$\frac{\partial \mathbf{I}}{\partial x} \simeq \mathbf{I}(x, y) - \mathbf{I}(x - 1, y)$$

Edge strength: $E(x, y) = \|\nabla \mathbf{I}(x, y)\|^2$

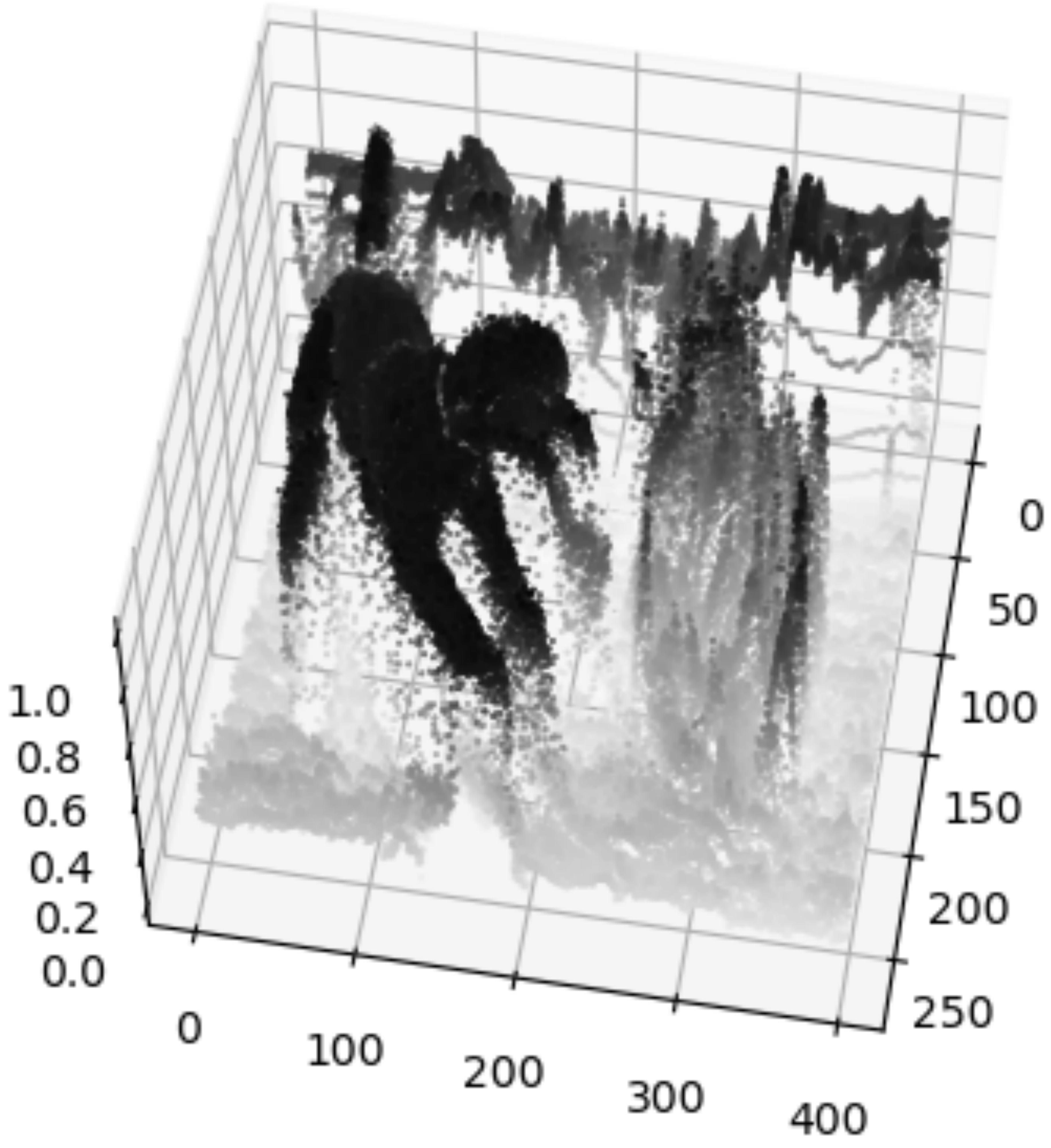
Edge orientation: $\theta(x, y) = \angle \nabla \mathbf{I} = \arctan \frac{\partial \mathbf{I} / \partial y}{\partial \mathbf{I} / \partial x}$

Adapted from A. Torralba

Derivatives

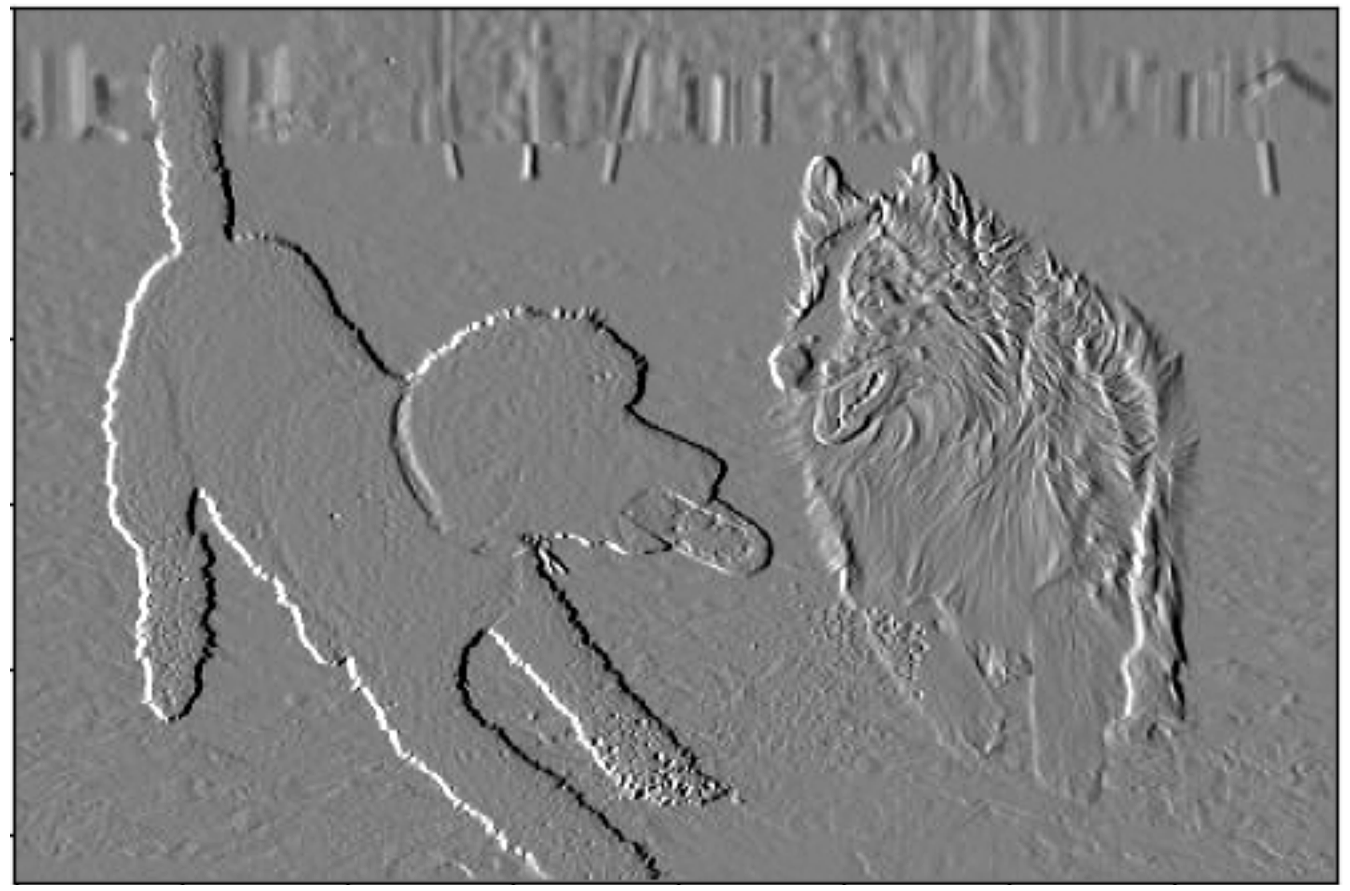
Pseudocode for x derivative:

```
Ix = np.zeros(I.shape)
# I is an H x W image
for y in range(I.shape[0]):
  for x in range(I.shape[1]):
    if x-1 < 0:
      Ix[y, x] = 0
    else:
      Ix[y, x] = I[y, x] - I[y, x-1]
```

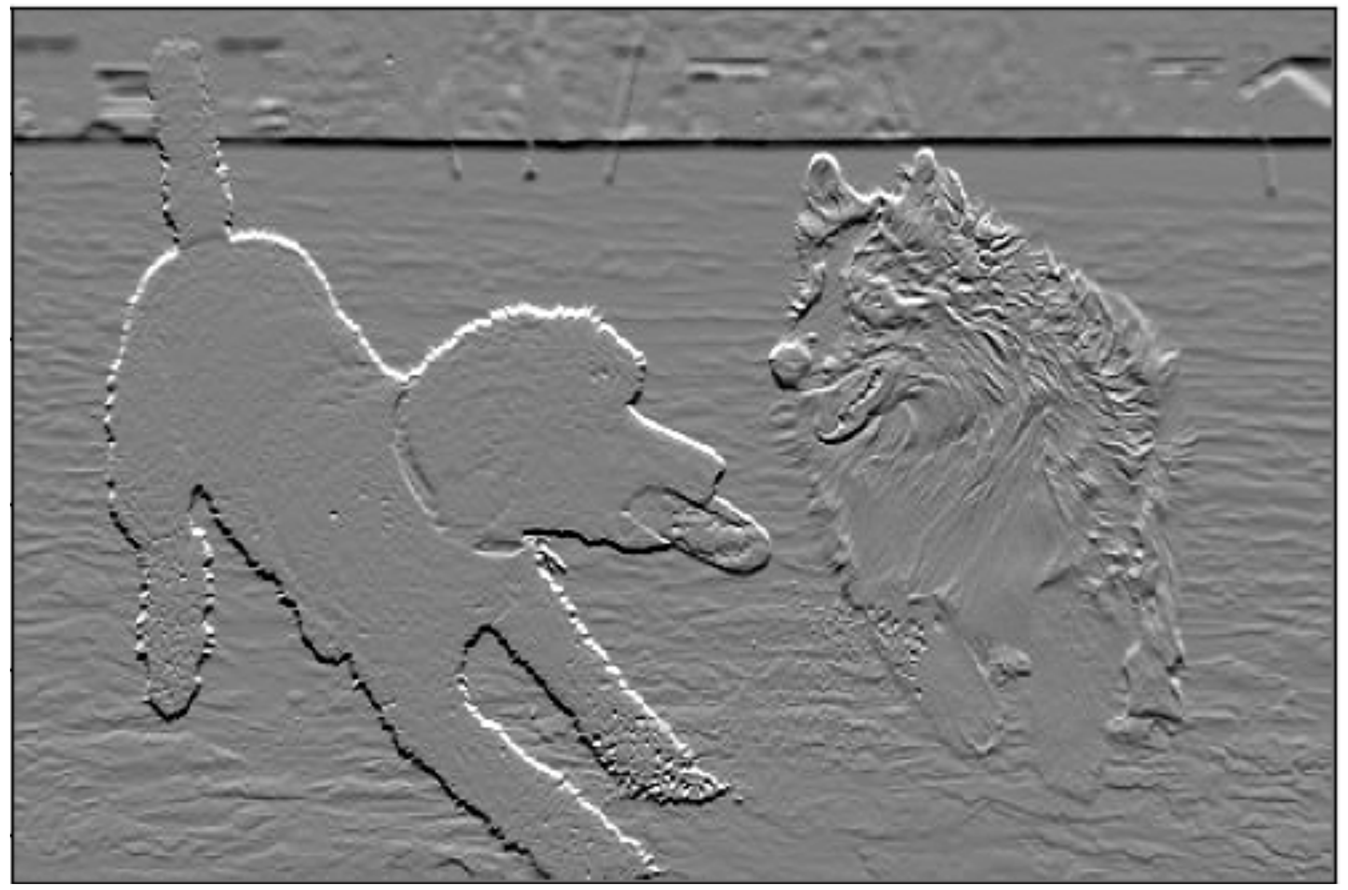


Edge detection

Image



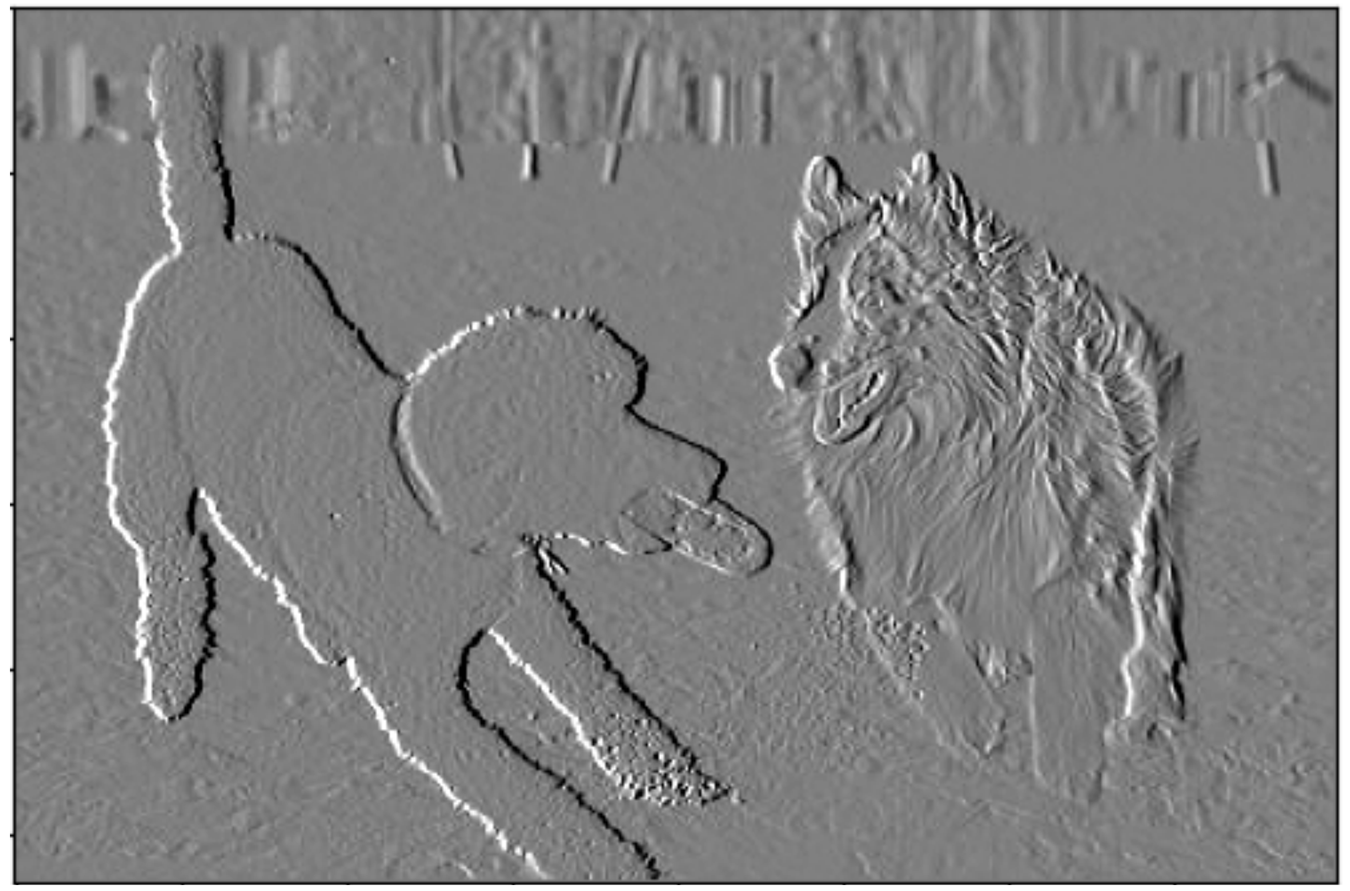
"Change in x"



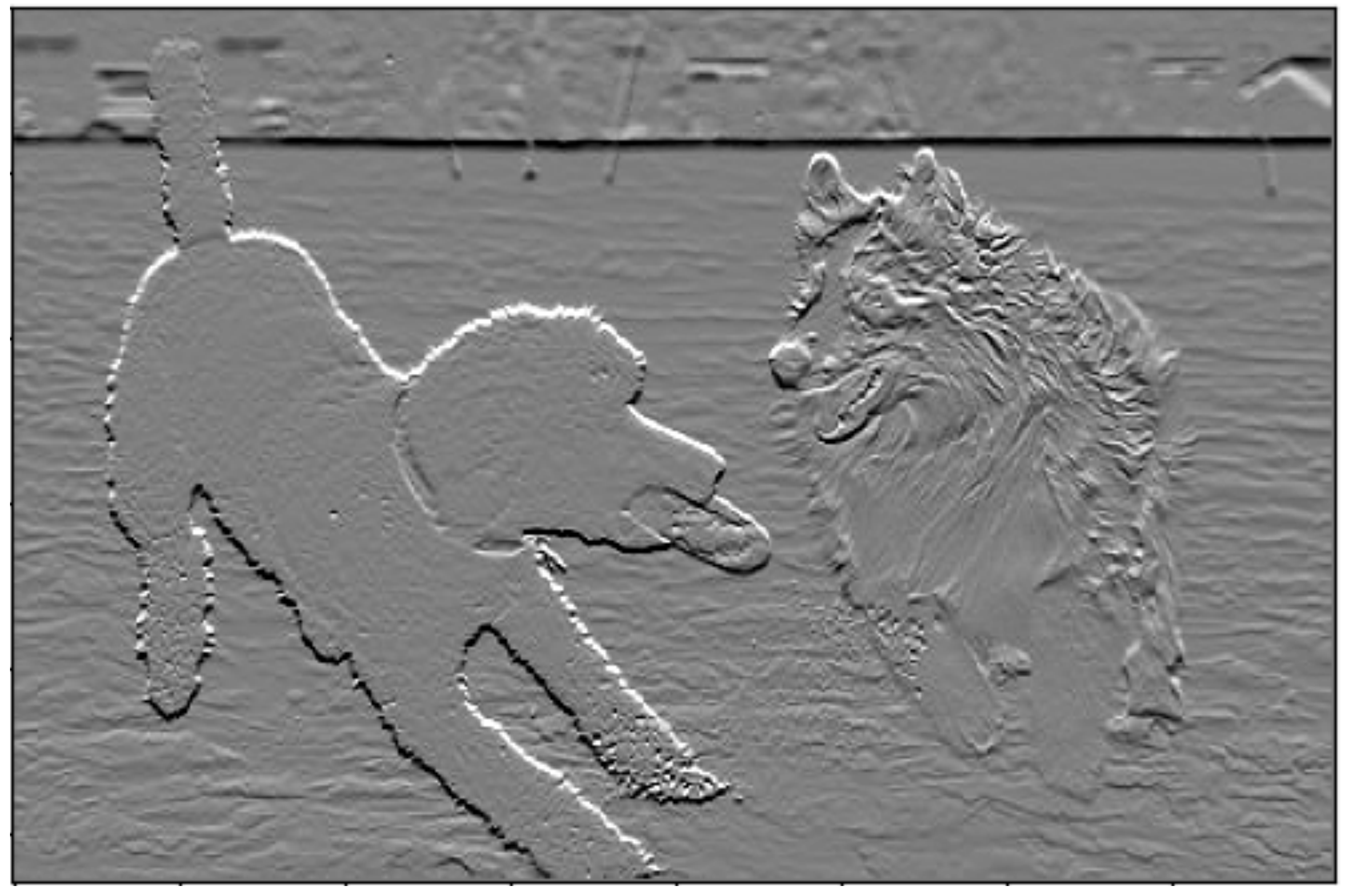
"Change in y"

Edge detection

Image



I_X



I_Y

Total edge strength: $S = I_X^2 + I_Y^2$

Edge detection

Image

Total edge strength

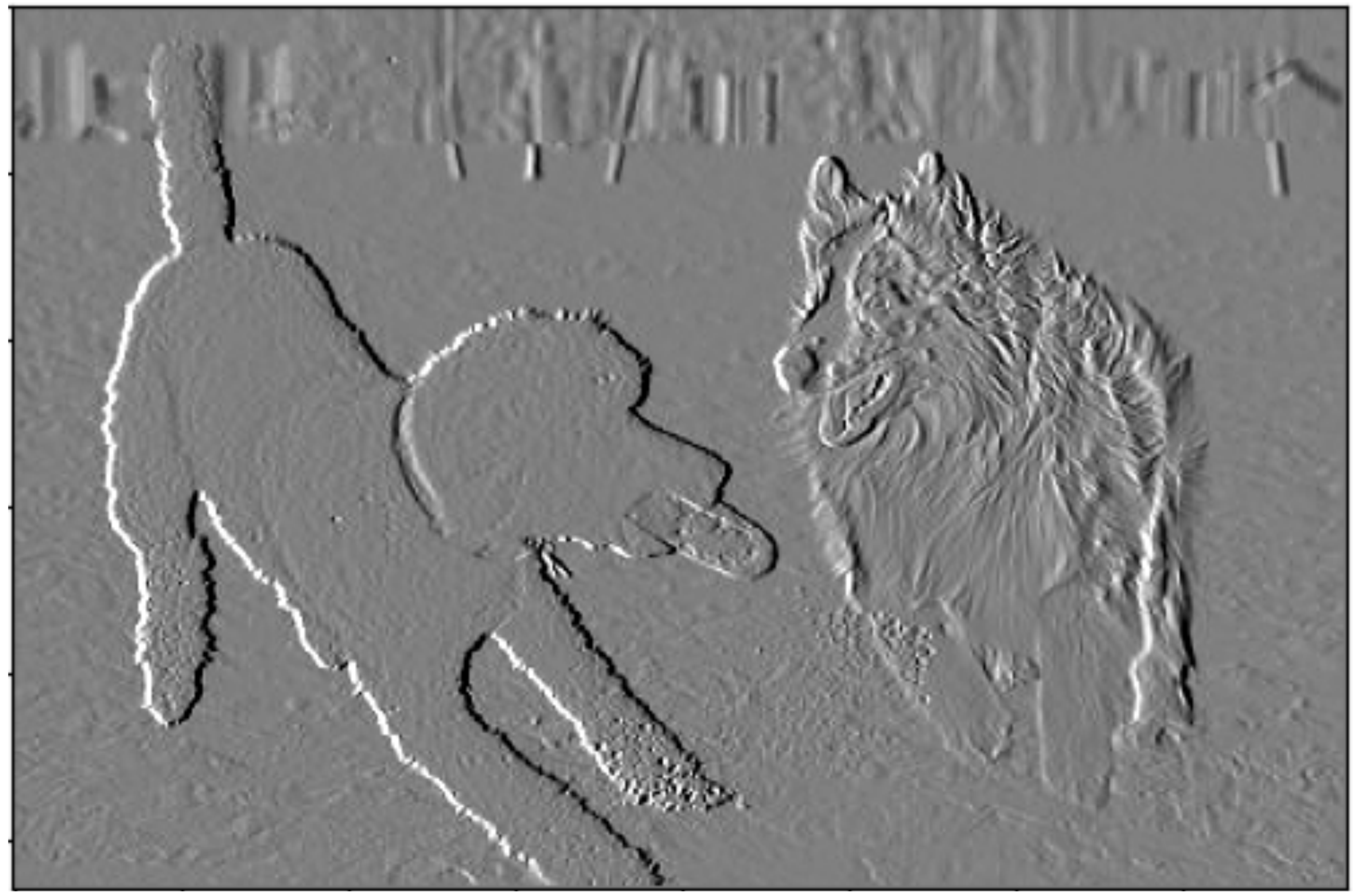
As neighborhood filtering

$$d_x = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad d_y = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

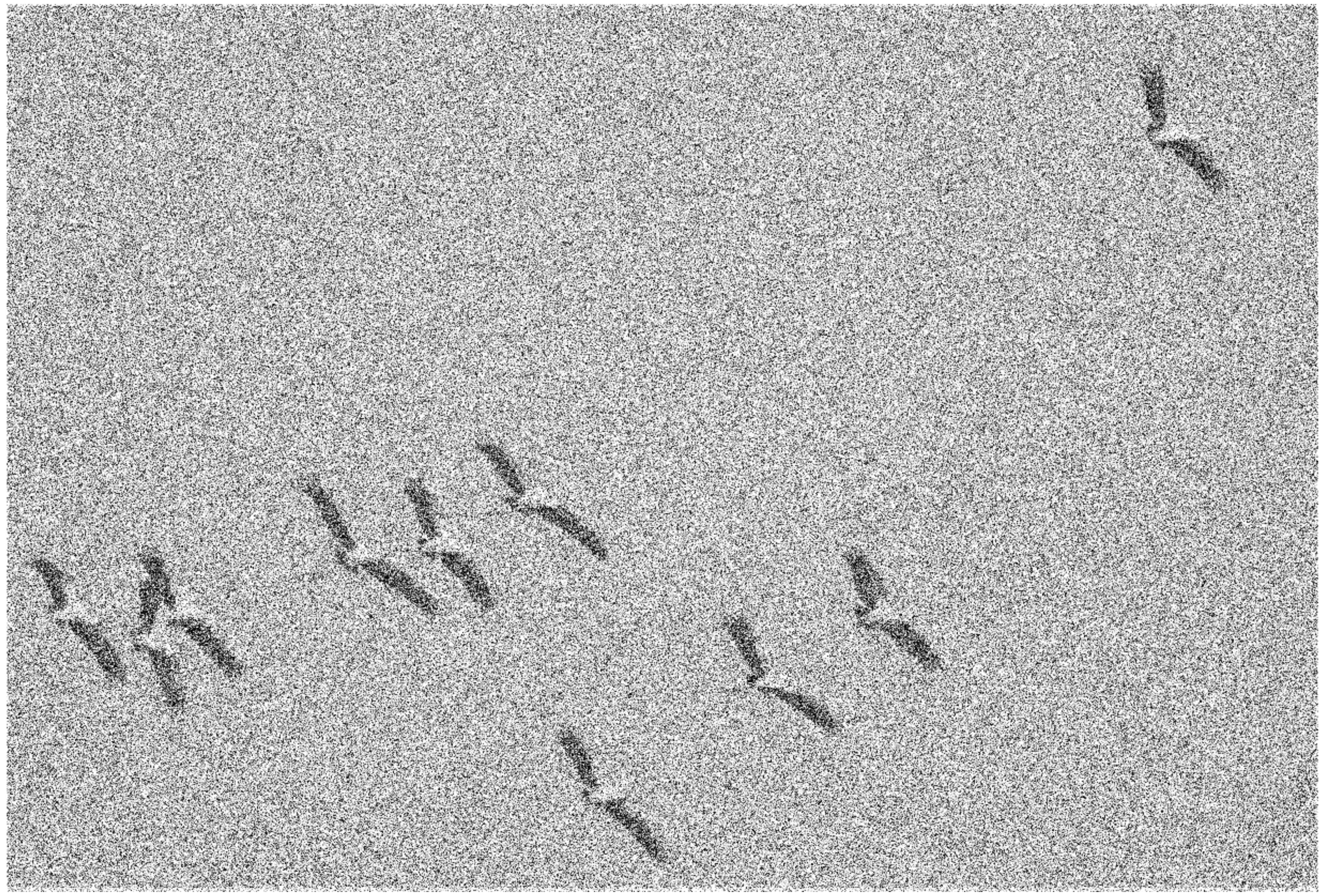
$$\frac{\partial \mathbf{I}}{\partial x} \simeq \mathbf{I}(x, y) - \mathbf{I}(x-1, y)$$

As neighborhood filtering

$$* \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$



Two computer vision problems

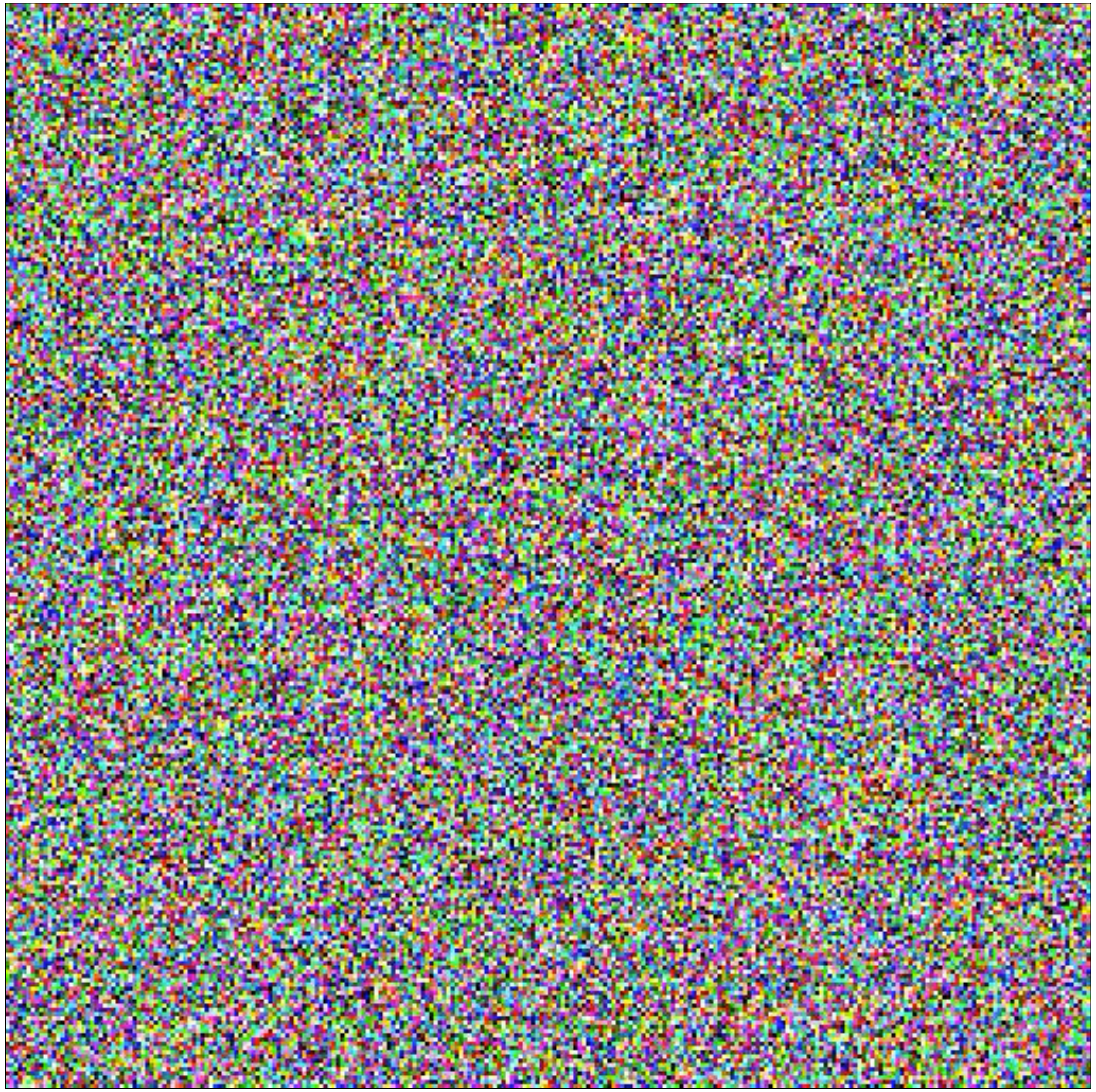


Denoising

Edge detection

Why are we studying these problems, again?

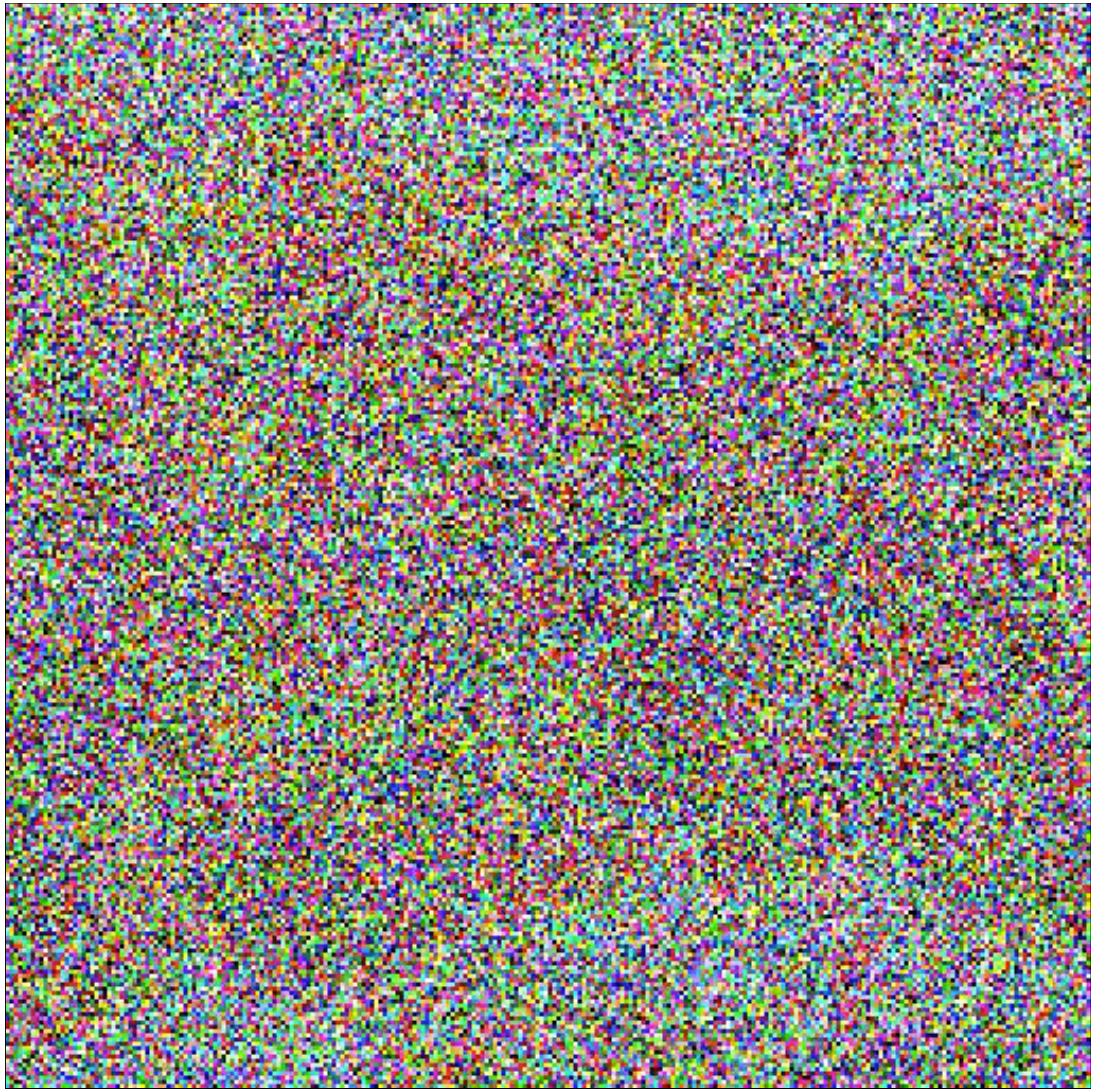
Later in the class: what if you had a *really* good denoiser?



Start with pure random noise, and then denoise!

Source: A. Ramesh

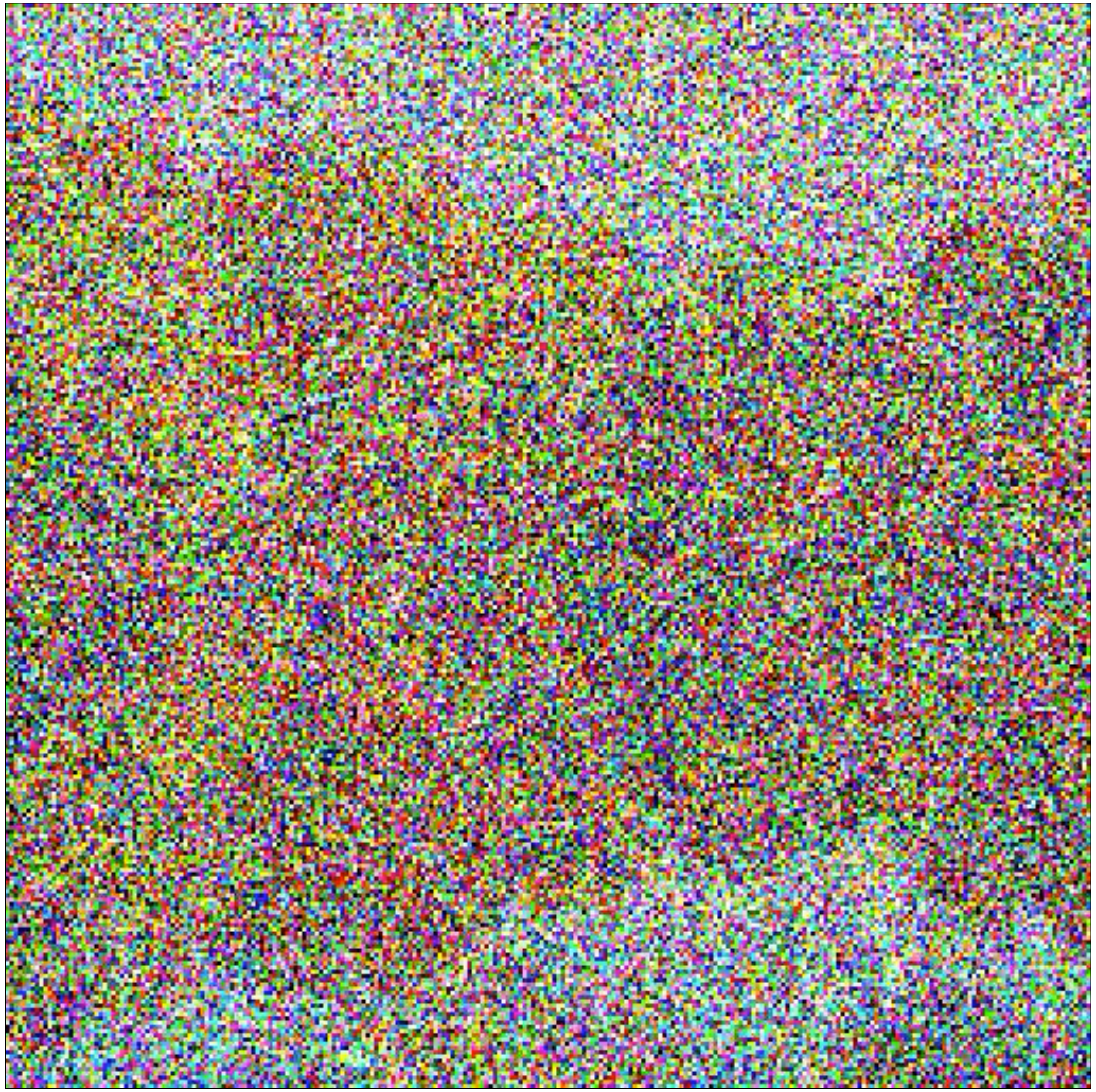
Later in the class: what if you had a *really* good denoiser?



Denoise a bit more…

Source: A. Ramesh

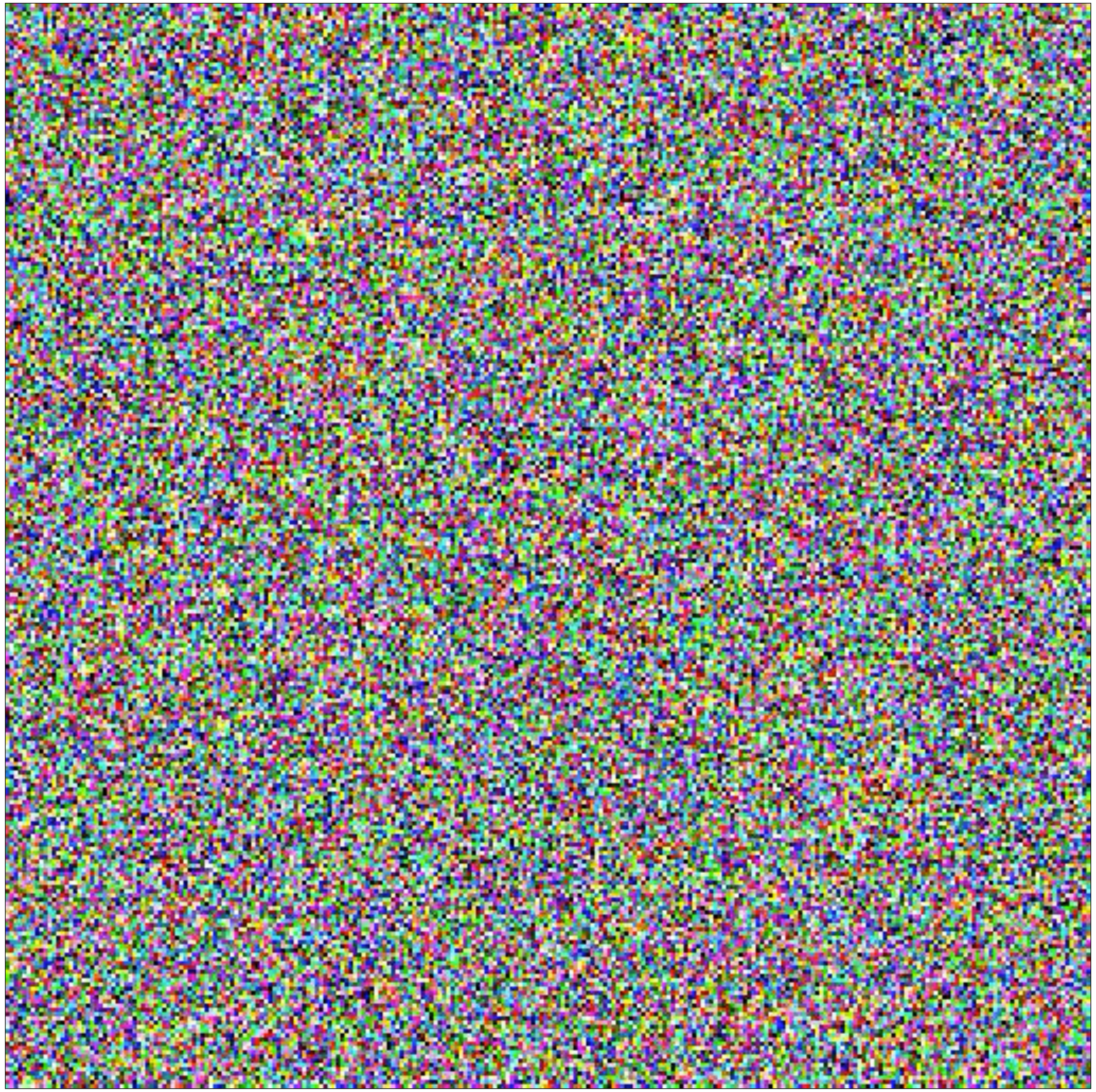
Later in the class: what if you had a *really* good denoiser?



Finally…

Source: A. Ramesh

Later in the class: what if you had a *really* good denoiser?



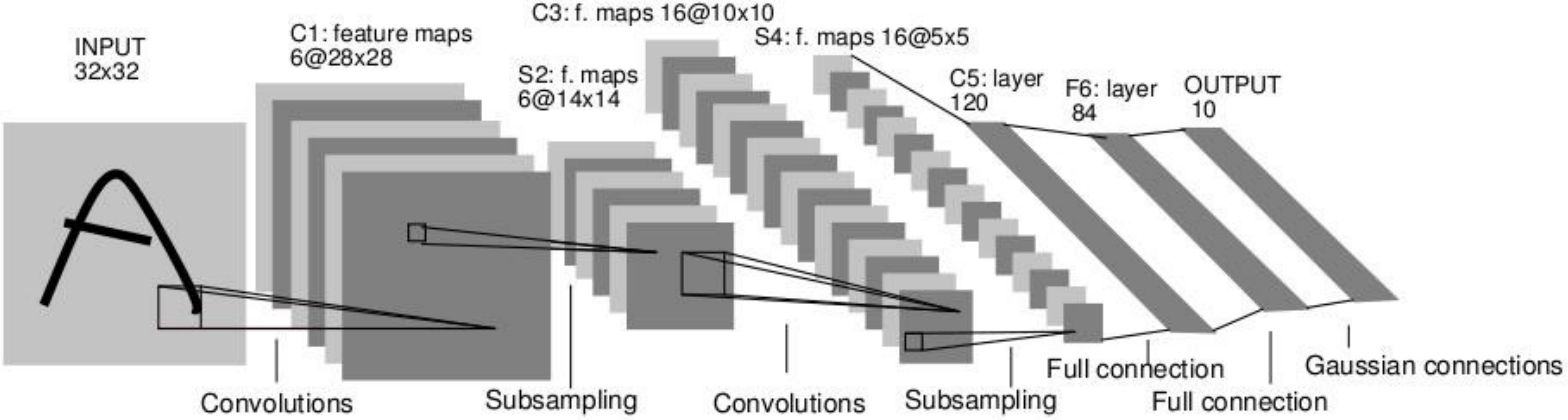
Diffusion: generate images by denoising

(advanced application covered at end of class)

Source: A. Ramesh

Neighborhood filtering: a powerful idea

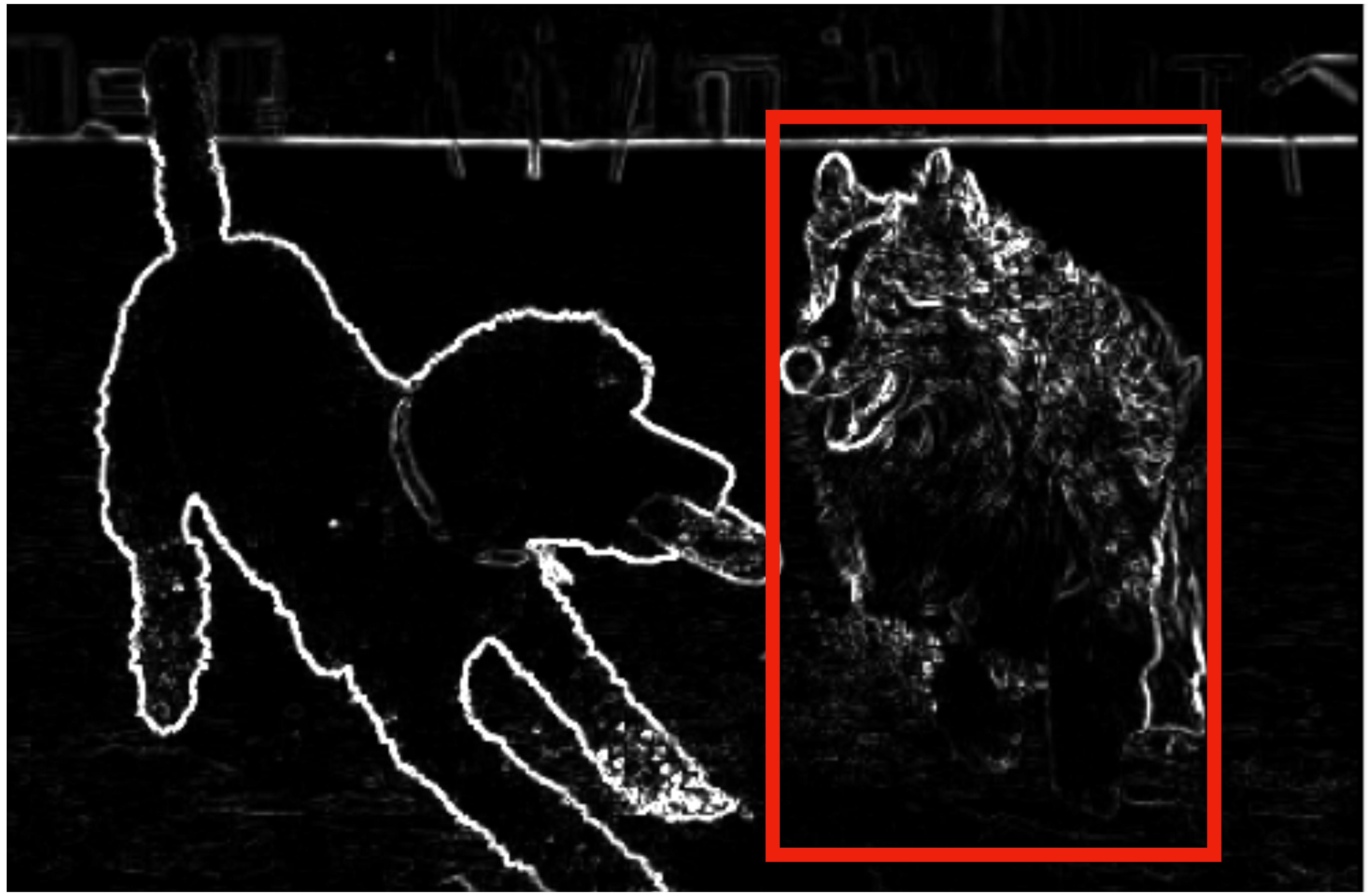
Convolutional neural networks: machine learning systems built on neighborhood filtering.



[LeCun et al. 1989]

Source: Torralba, Freeman, Isola

Is this the best we can do?



Next class: more image filtering