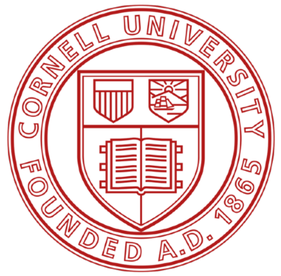
Lecture 29: Course recap and final exam review

CS 5670: Introduction to Computer Vision



Announcements

- All grades except PS6 should be out by tonight
- Would really appreciate it if you did the course evaluation!

Final exam

- You'll have 3 hours
- Compared to midterm, more multiple choice, less (pseudo-)coding
- Compared to the homework: more theoretical questions
- Cumulative. Not included: embodied vision lecture.
- 1-page front and back hand-written single-sided "cheat sheet"
- No calculators (nor are they necessary)

Today

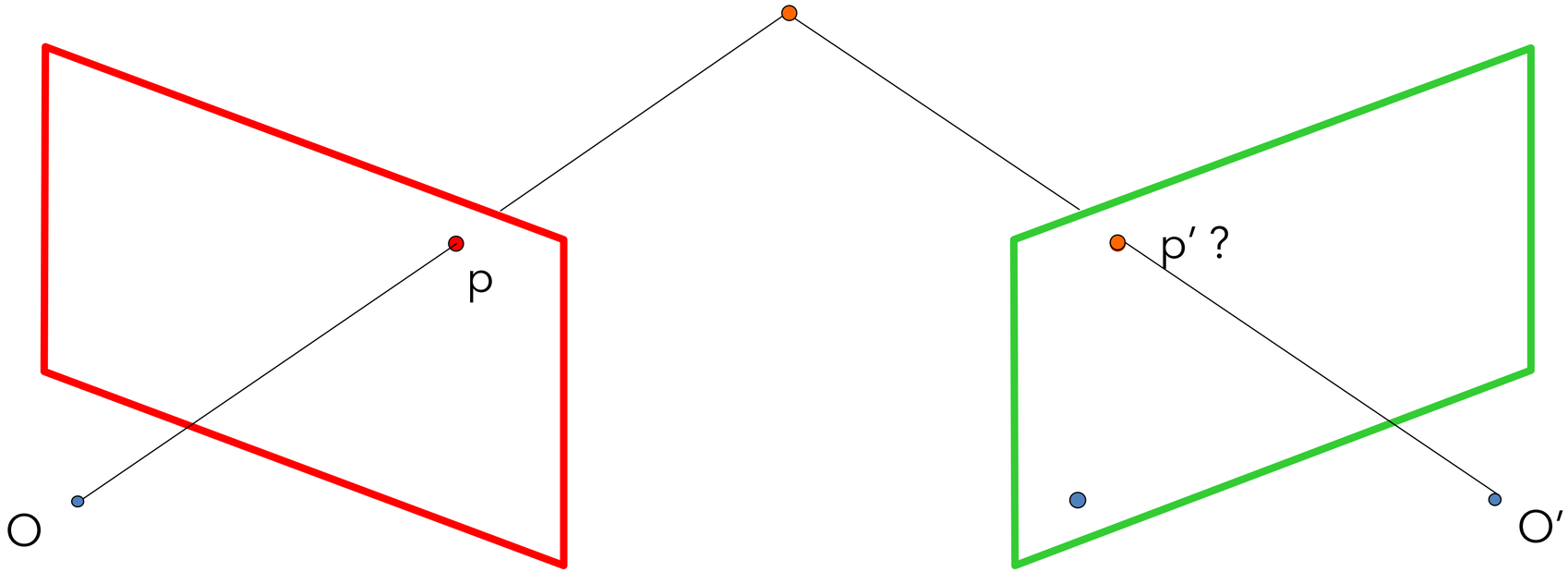
- Practice questions
- Mostly focusing on new material, since you already had practice questions for the midterm
- Plus quick reviews of the course material behind each one
- Not exhaustive!

1. 3D and physically-based vision

3D and physically-based vision

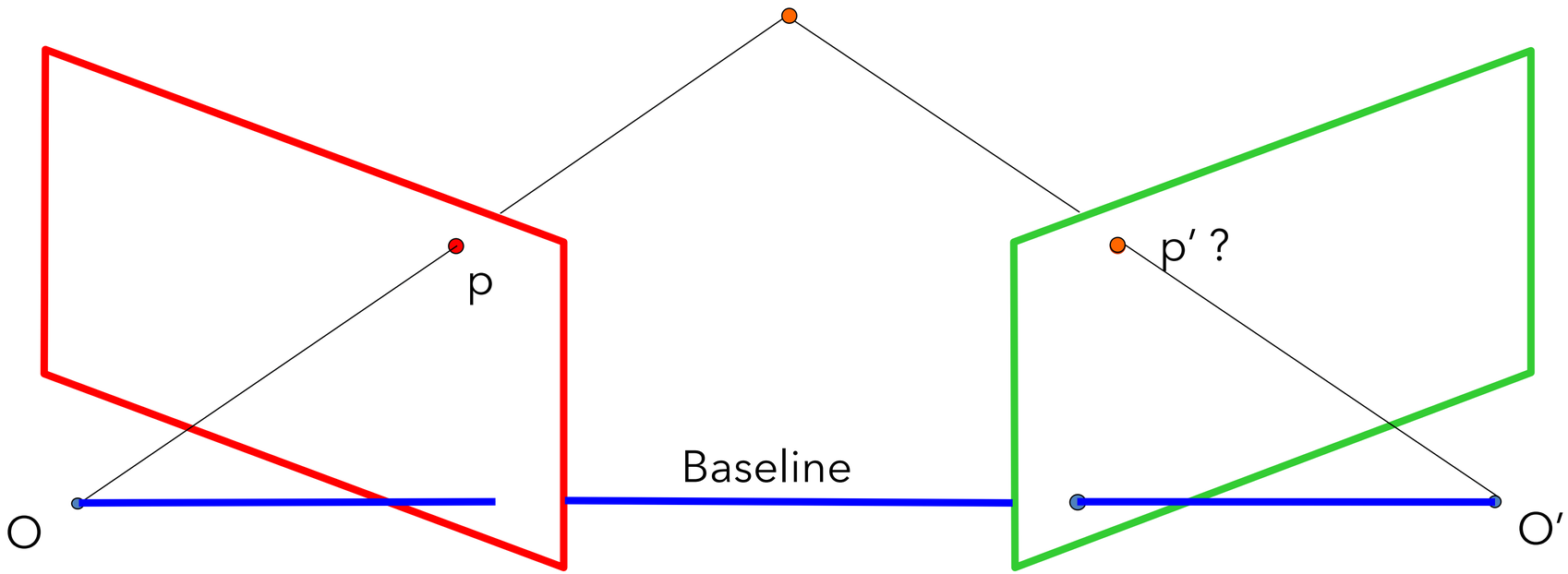
Question: Suppose that we walk along a ray at fixed intervals and project each 3D point into another view. What can you say about how these projections are spatially distributed?

Two-view geometry



Source: Torralba, Isola, Freeman

Two-view geometry

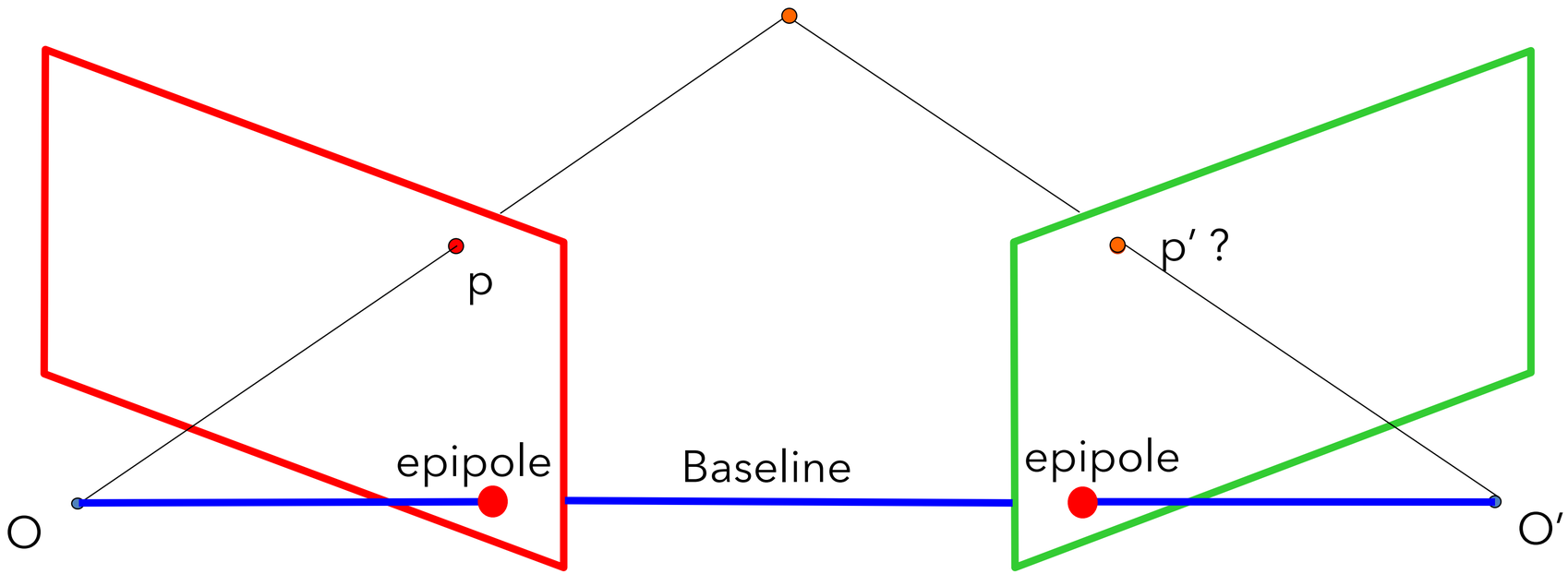


Baseline: the line connecting the two camera centers

Epipole: point of intersection of *baseline* with the image plane

Source: Torralba, Isola, Freeman

Two-view geometry

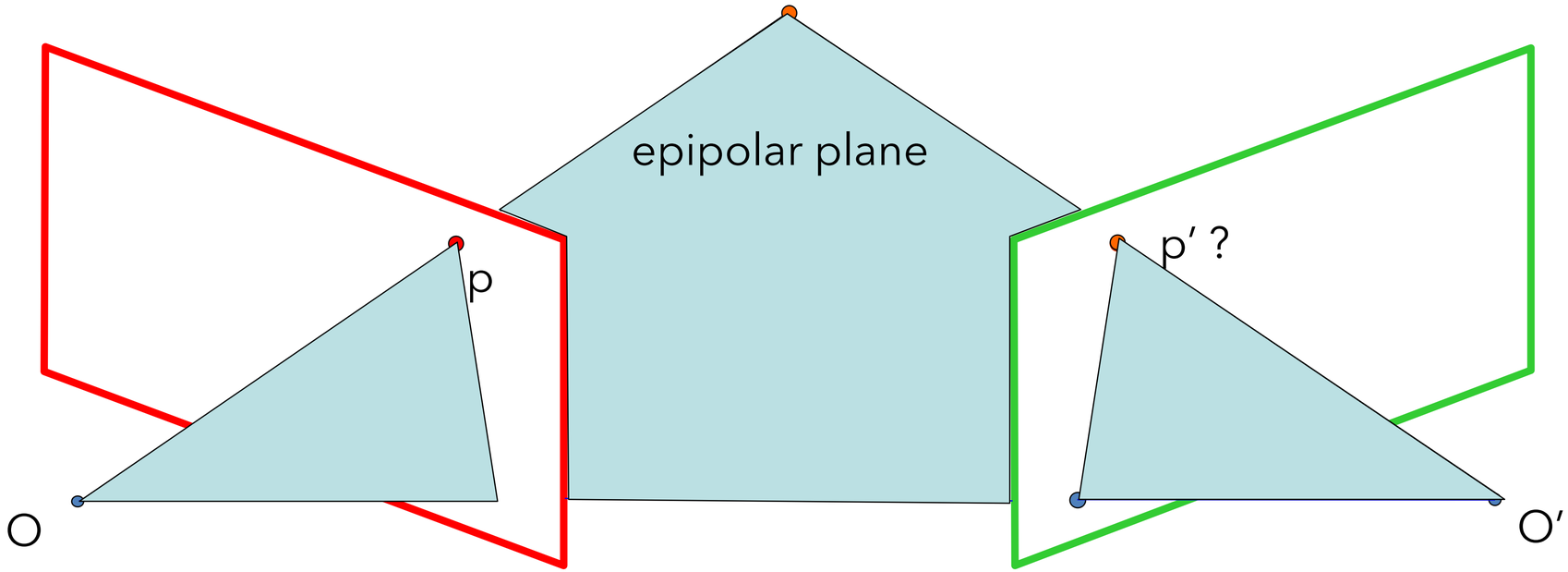


Baseline: the line connecting the two camera centers

Epipole: point of intersection of *baseline* with the image plane

Source: Torralba, Isola, Freeman

Two-view geometry



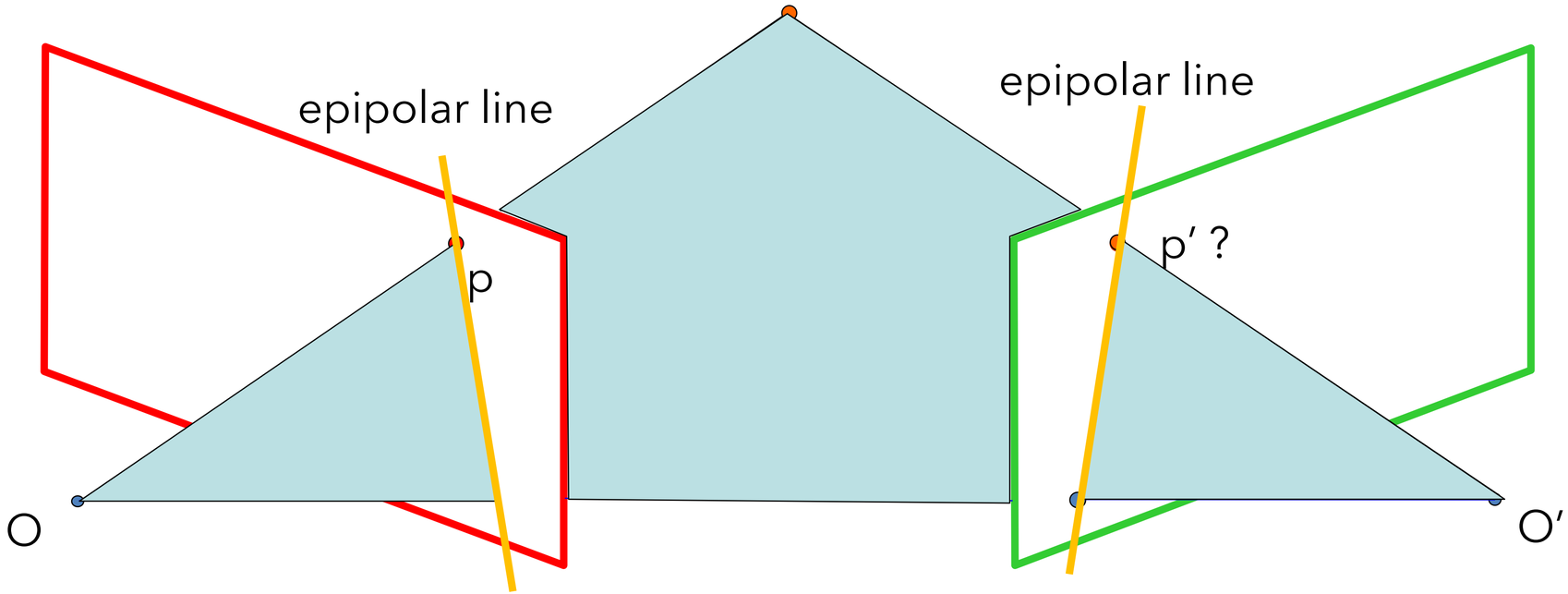
Baseline: the line connecting the two camera centers

Epipole: point of intersection of *baseline* with the image plane

Epipolar plane: the plane that contains the two camera centers and a 3D point in the world

Source: Torralba, Isola, Freeman

Two-view geometry



Baseline: the line connecting the two camera centers

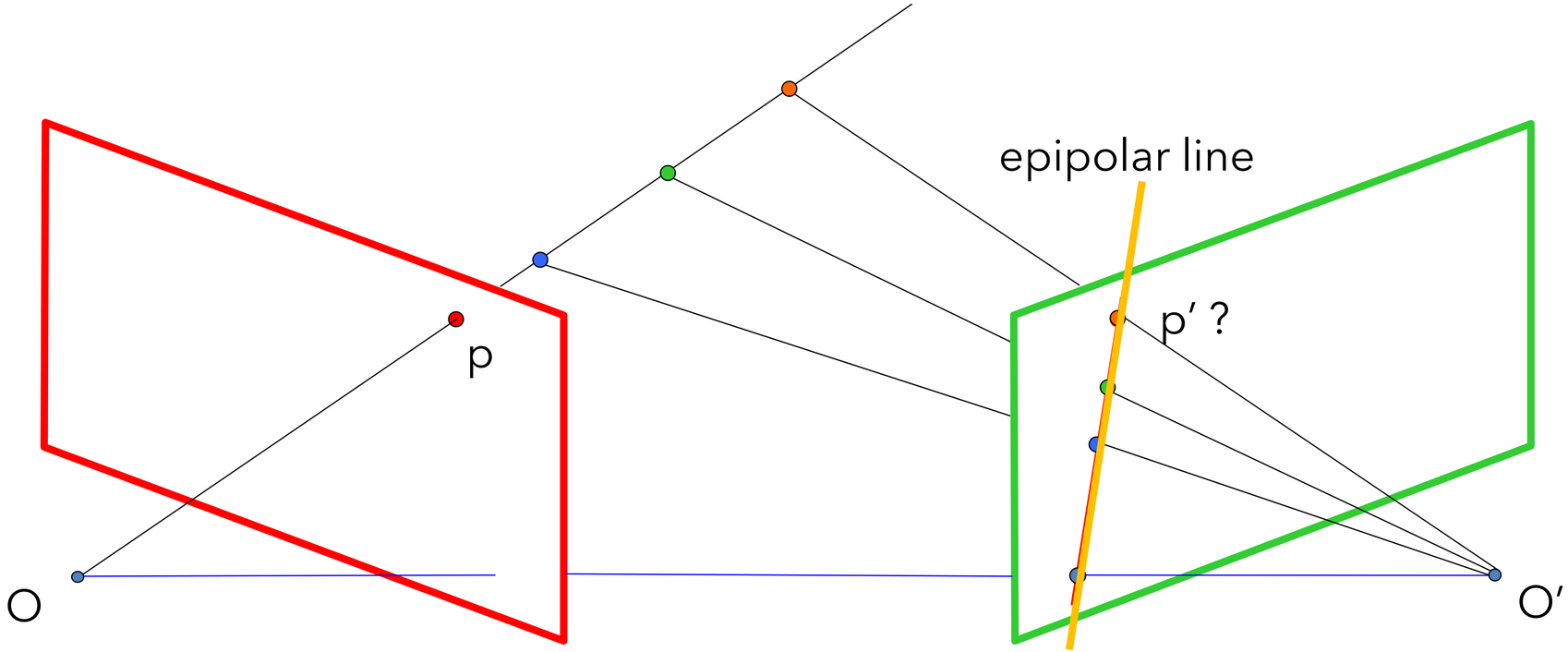
Epipole: point of intersection of *baseline* with the image plane

Epipolar plane: the plane that contains the two camera centers and a 3D point in the world

Epipolar line: intersection of the *epipolar plane* with each image plane

Source: Torralba, Isola, Freeman

Two-view geometry



We can search for matches across epipolar lines

All epipolar lines intersect at the epipoles

Source: Torralba, Isola, Freeman

3D and physically-based vision

Question:

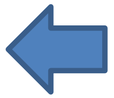
Consider the homography $\mathbf{H} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \\ 4 & 4 & 2 \end{bmatrix}$ Which of the following are equivalent to it?

(a) $\mathbf{H}' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ (b) $\mathbf{H}' = \begin{bmatrix} 2 & 6 & 10 \\ 4 & 8 & 12 \\ 8 & 8 & 4 \end{bmatrix}$ (c) $\mathbf{H} = \begin{bmatrix} -1 & -3 & -5 \\ -2 & -4 & -6 \\ -4 & -4 & -2 \end{bmatrix}$

(d) $\mathbf{H}' = \mathbf{H} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ (e) $\mathbf{H}' = \mathbf{H}\mathbf{H}$

Starting point: affine transformation

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$$



what happens when we change this row?

affine transformation

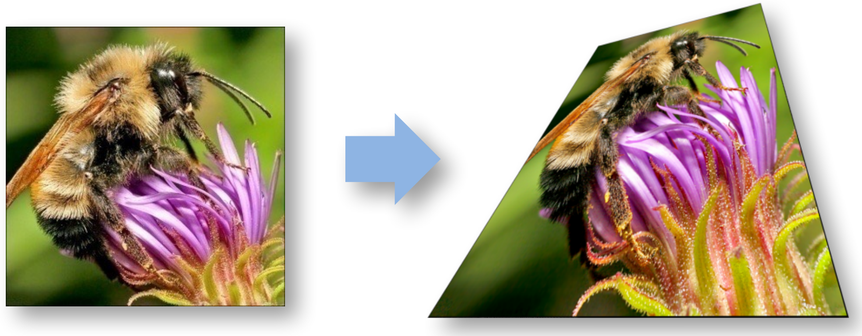
Source: N. Snavely

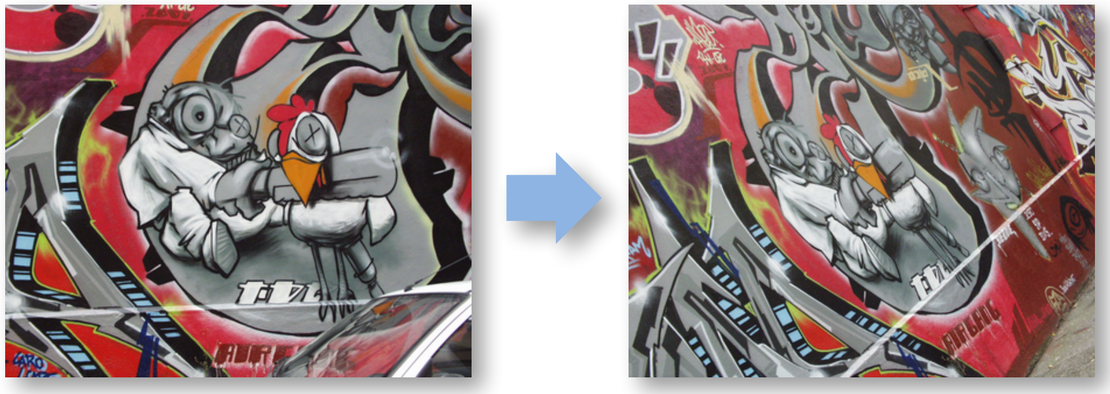
Homographies

$$\mathbf{H} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$$

Called a **homography**

(or planar perspective map)





Source: N. Snavely

Homographies

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\sim \begin{bmatrix} \frac{ax+by+c}{gx+hy+1} \\ \frac{dx+ey+f}{gx+hy+1} \\ 1 \end{bmatrix}$$

Source: N. Snavely

Homography

Example: two pictures taken by rotating the camera:

If we try to build a panorama by overlapping them:

Source: Torralba, Isola, Freeman

Homography

Example: two pictures taken by rotating the camera:

With a homography, you can map both images into a single camera:

Source: Torralba, Isola, Freeman

3D and physically-based vision

Question: Suppose that H is a homography, and X is an eigenvector of H. What result do you get when you transform X using H?

Recall: $\lambda X = HX \rightarrow$ same Cartesian point location!

3D and physically-based vision

Question: Name two ways to prune incorrectly matched descriptors.

Question: Is thresholding feature similarity based on their distance a good way to do this?

Feature matching

Given a feature in I_1, how do we find the best match in I_2?

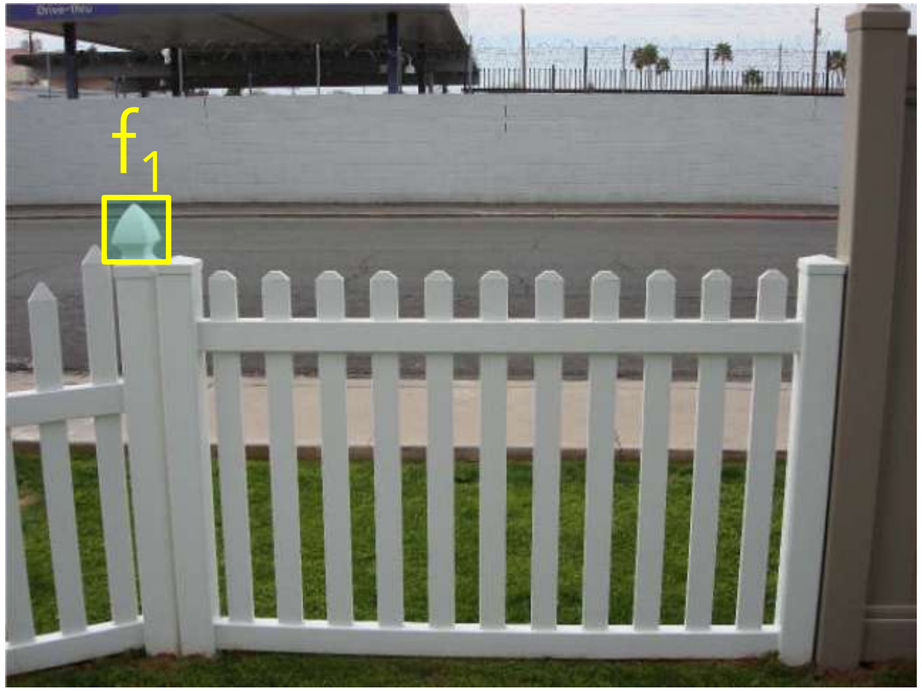
1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the closest one.

Source: N. Snavely

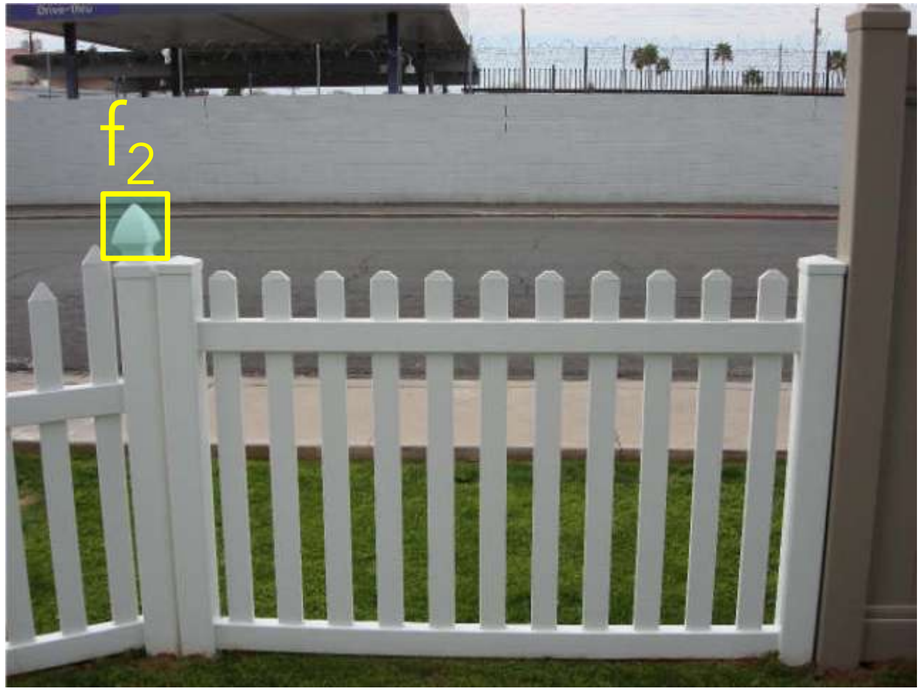
Finding matches

How do we know if two features match?

- Simple approach: are they the nearest neighbor in L_2 distance, $||f_1 - f_2||$?



I_1



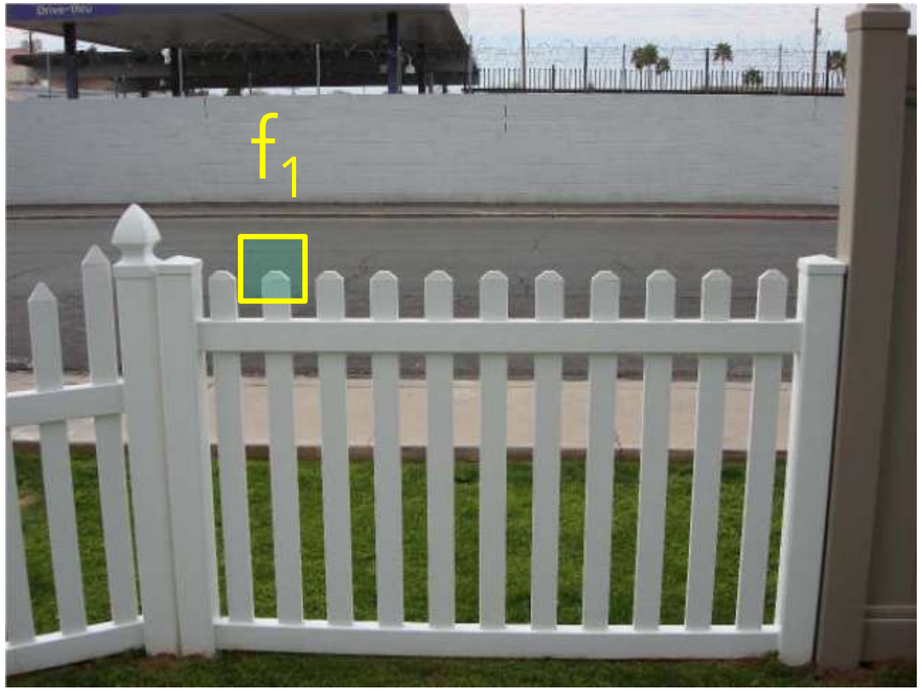
I_2

Source: N. Snavely

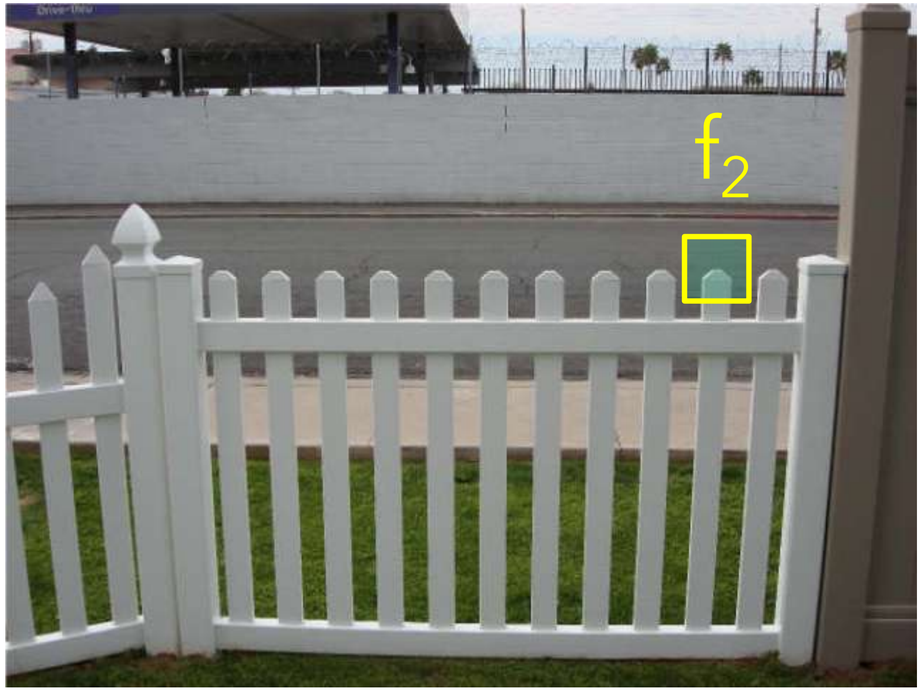
Finding matches

How do we know if two features match?

- Simple approach: are they the nearest neighbor in L_2 distance, $||f_1 - f_2||$?
- Can give good scores to ambiguous (incorrect) matches.



I_1



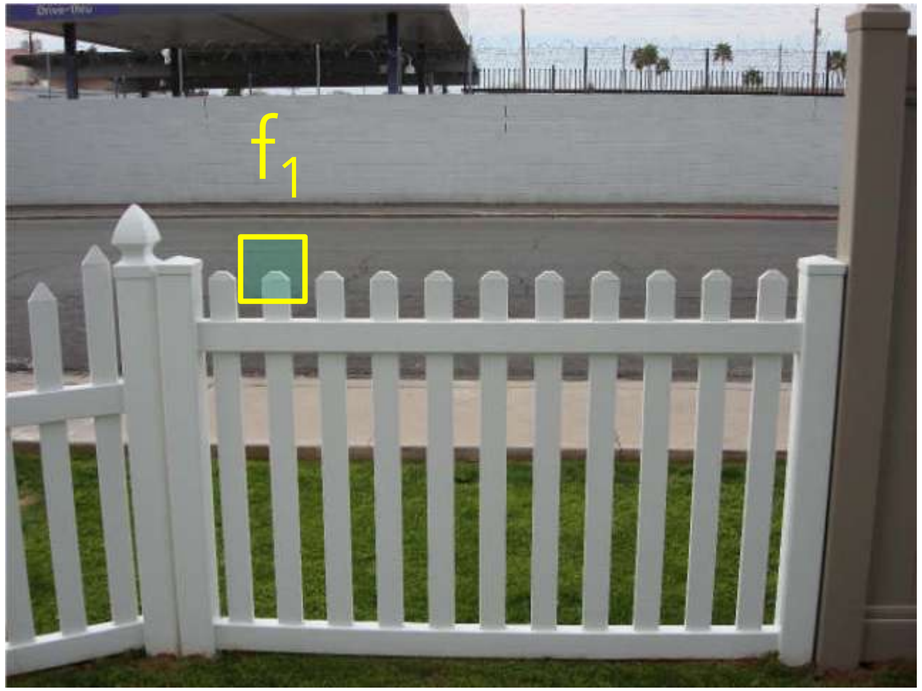
I_2

Source: N. Snavely

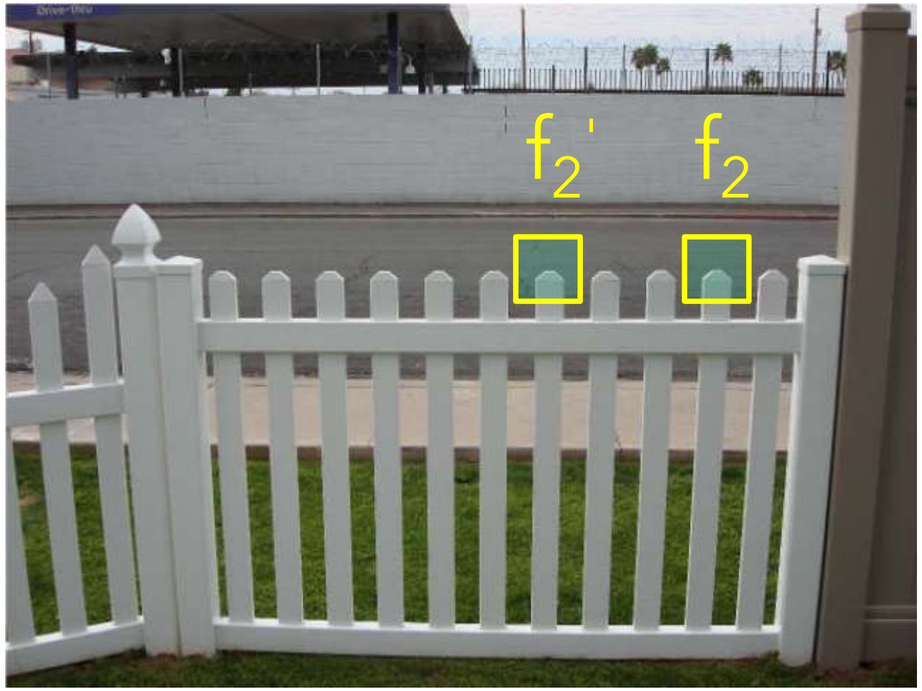
Finding matches

Throw away matches that fail tests:

- **Ratio test**: this *by far* the best match? Compare best and 2nd-best matches.
  - Ratio distance = $\|f_1 - f_2\| / \|f_1 - f_2'\|$
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2nd best SSD match to f_1 in I_2
- **Forward-backward consistency**: f_1 should also be nearest neighbor of f_2



I_1



I_2

Source: N. Snavely

3D and physically-based vision

Question: Which of the following best describes the aperture problem?

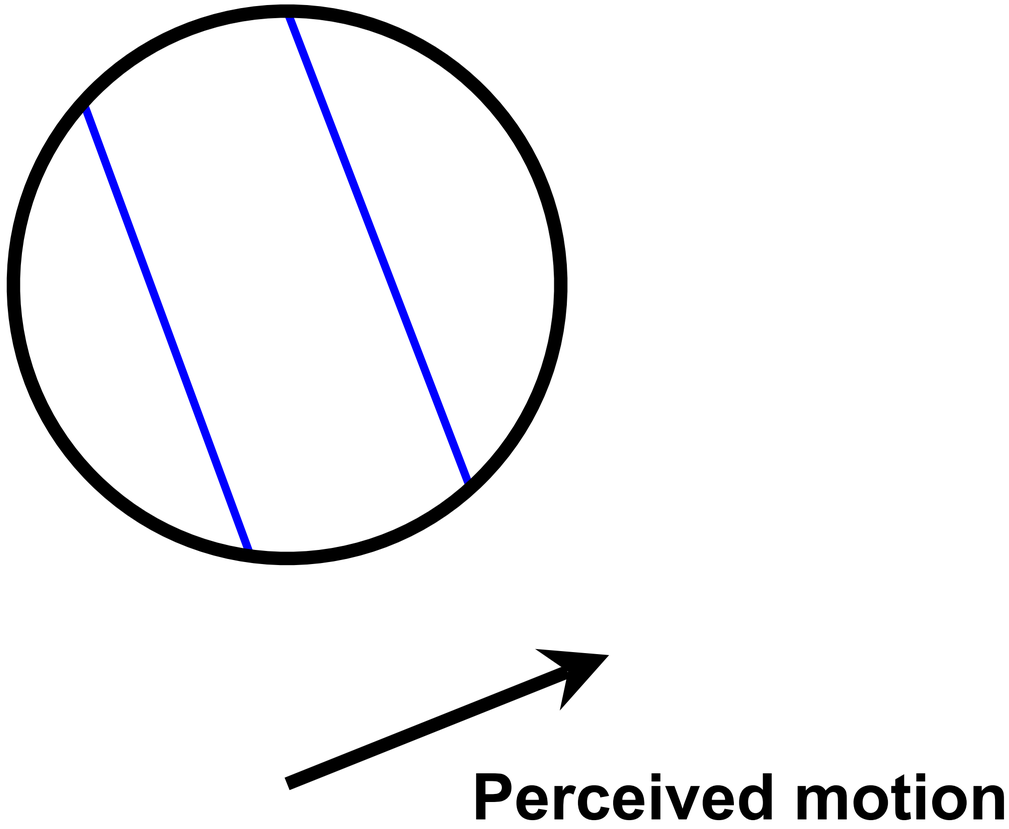
A. The problem of estimating camera aperture from motion.

B. Difficulty in estimating motion when the camera focal length is unknown.

C. Ambiguity in motion direction when viewing a straight edge through a small window.

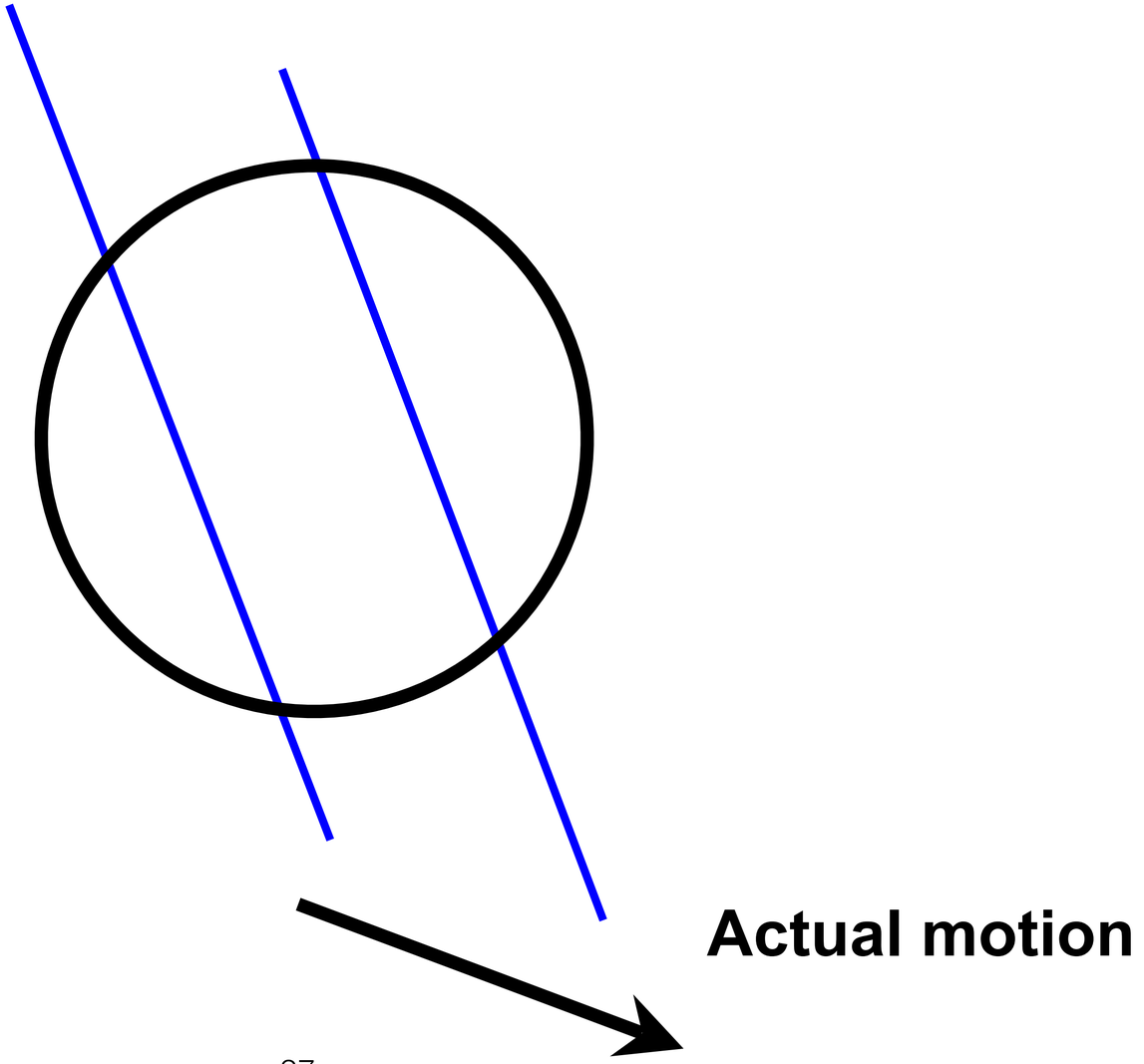
D. Difficulty in computing optical flow due to low resolution or low frame rate.

The aperture problem



Source: S. Lazebnik

The aperture problem



Source: S. Lazebnik

The barber pole illusion

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

Source: S. Lazebnik

The barber pole illusion

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

Source: S. Lazebnik

The brightness constancy constraint

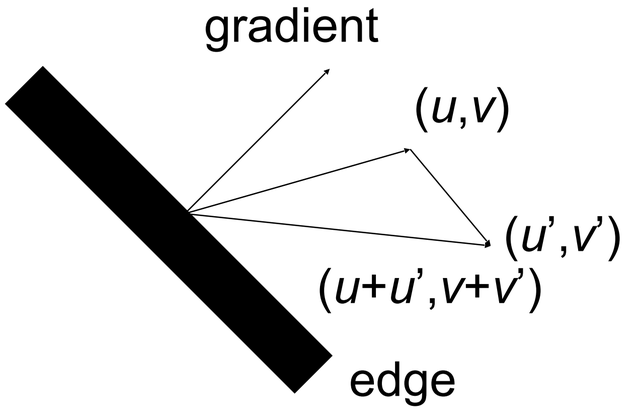
$$I_x u + I_y v + I_t = 0$$

- How many equations and unknowns per pixel?
  - One equation, two unknowns
- Under-constrained. Let's rewrite it:

$$\nabla I \cdot (u, v) + I_t = 0$$

- The component of the flow perpendicular to the image gradient (i.e., parallel to the edge) is unknown!

If (u, v) satisfies the equation,
so does (u+u', v+v') if $\nabla I \cdot (u', v') = 0$



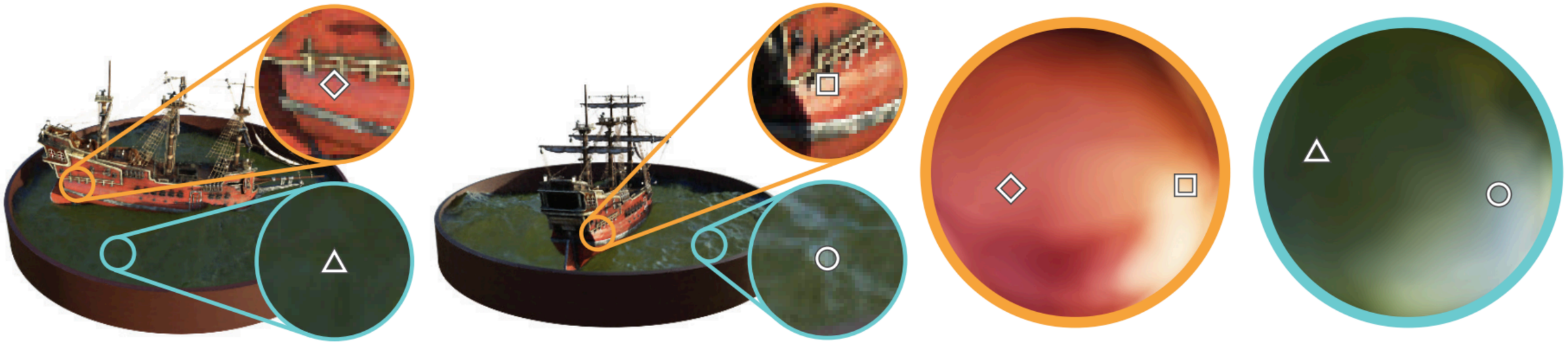
Source: S. Lazebnik

3D and physically-based vision

Question: Which of the following statements correctly describes the view-dependence property in the NeRF formulation?

A. Both volume density (σ) and color (c) are dependent on the viewing direction.
B. Volume density (σ) is dependent on viewing direction, while color (c) is only dependent on spatial location.
C. Volume density (σ) is only dependent on spatial location, while color (c) is dependent on both spatial location and viewing direction.
D. Neither volume density (σ) nor color (c) depends on the viewing direction.

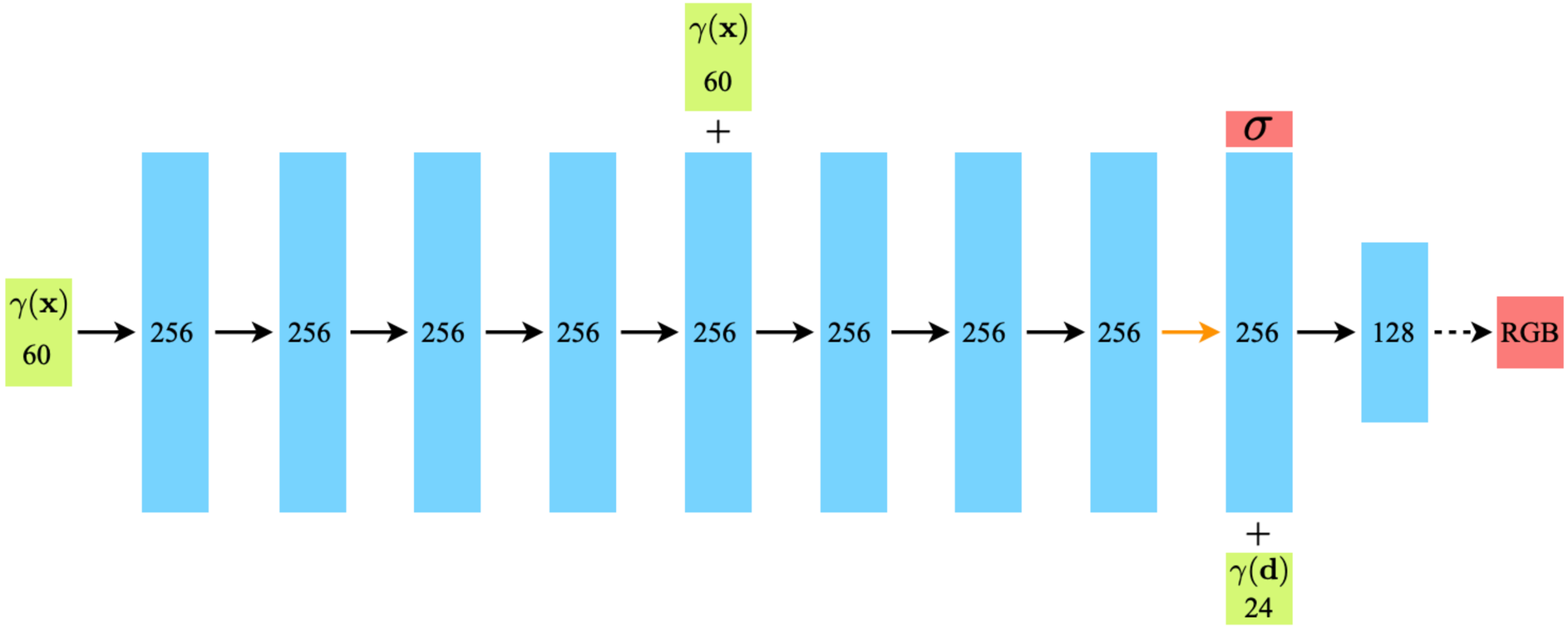
Why is it good to be view-dependent?



(a) View 1 (b) View 2 (c) Radiance Distributions

[Source: Mildenhall et al., “NeRF”, 2020]

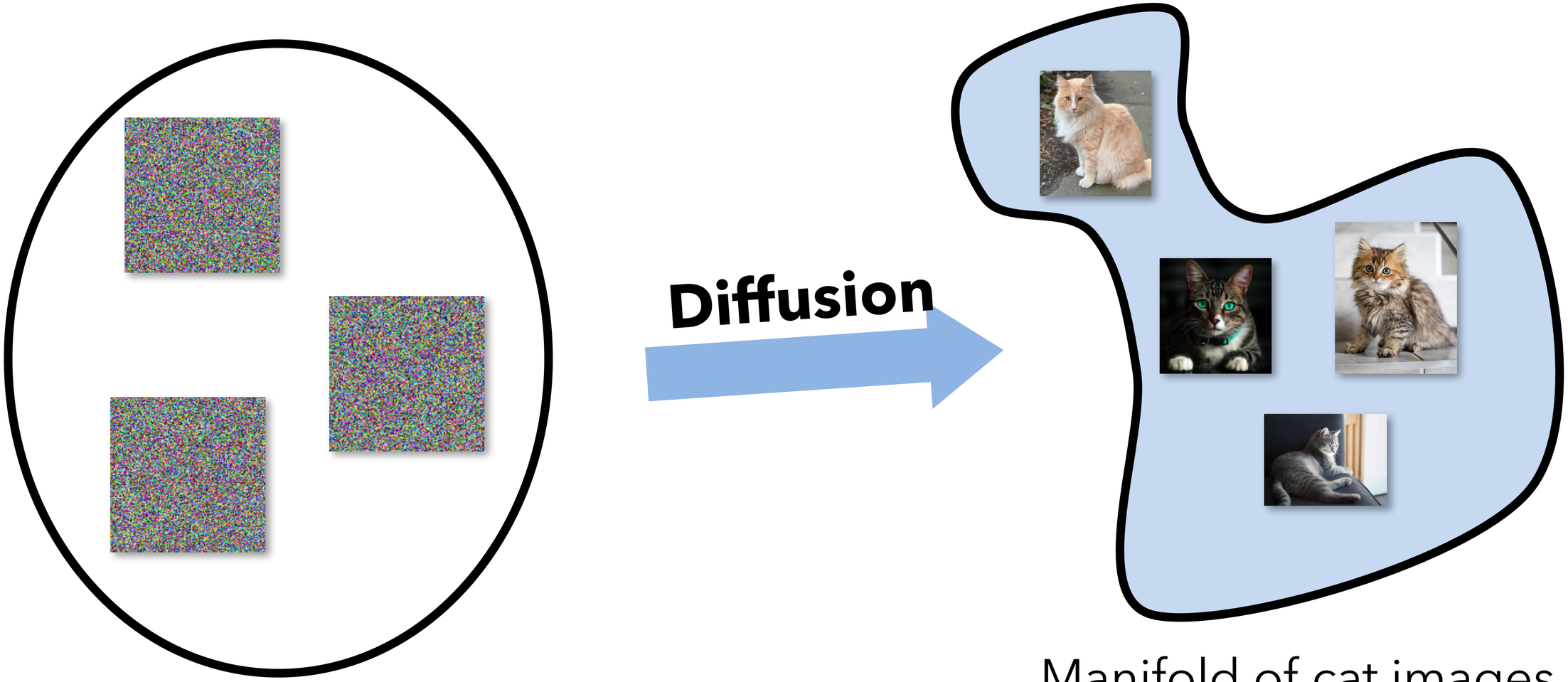
NeRF MLP architecture



$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots, \sin\left(2^{L-1}\pi p\right), \cos\left(2^{L-1}\pi p\right)\right)$$

[Source: Mildenhall et al., "NeRF", 2020]

2. Generative models



Random images

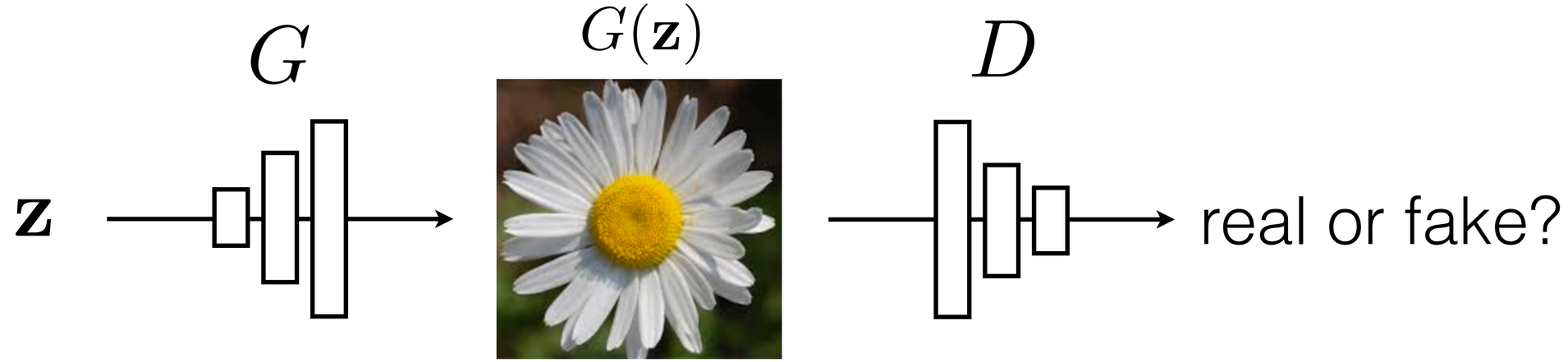
Manifold of cat images

Slide source: N. Snavely & S. Seitz

Generative models

Question: What makes it hard to train GANs?

Training



G tries to synthesize fake images that fool **D**

D tries to identify the fakes

- Training: iterate between training D and G with backprop.
- Global optimum when G reproduces data distribution (see book)
- Note: when training G, we take derivatives through discriminator!

[Goodfellow et al., 2014]

Source: Isola, Freeman, Torralba

Generative models

Question: Match the image to the t value that was used to perform Sketch-to-Image translation using SDEdit for a diffusion model. The input is a sketch, the output is an image.

(i)

(ii)

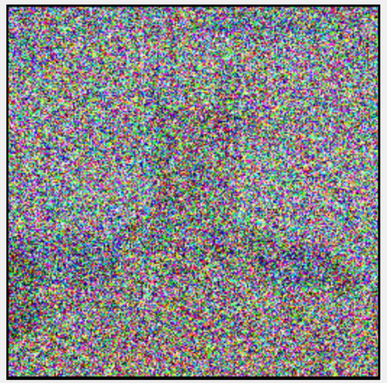
(a) Low t

(b) Large t

Source: [Meng et al., “SDEdit”, 2022]

Sketch to photo

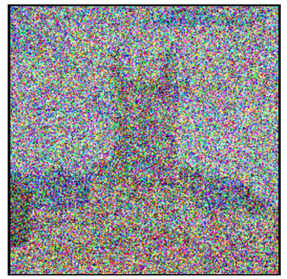
Add noise to a sketch



$\mathbf{x}_t$

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \text{ for } \epsilon \sim N(0,1)$$

Denoise from t to 0

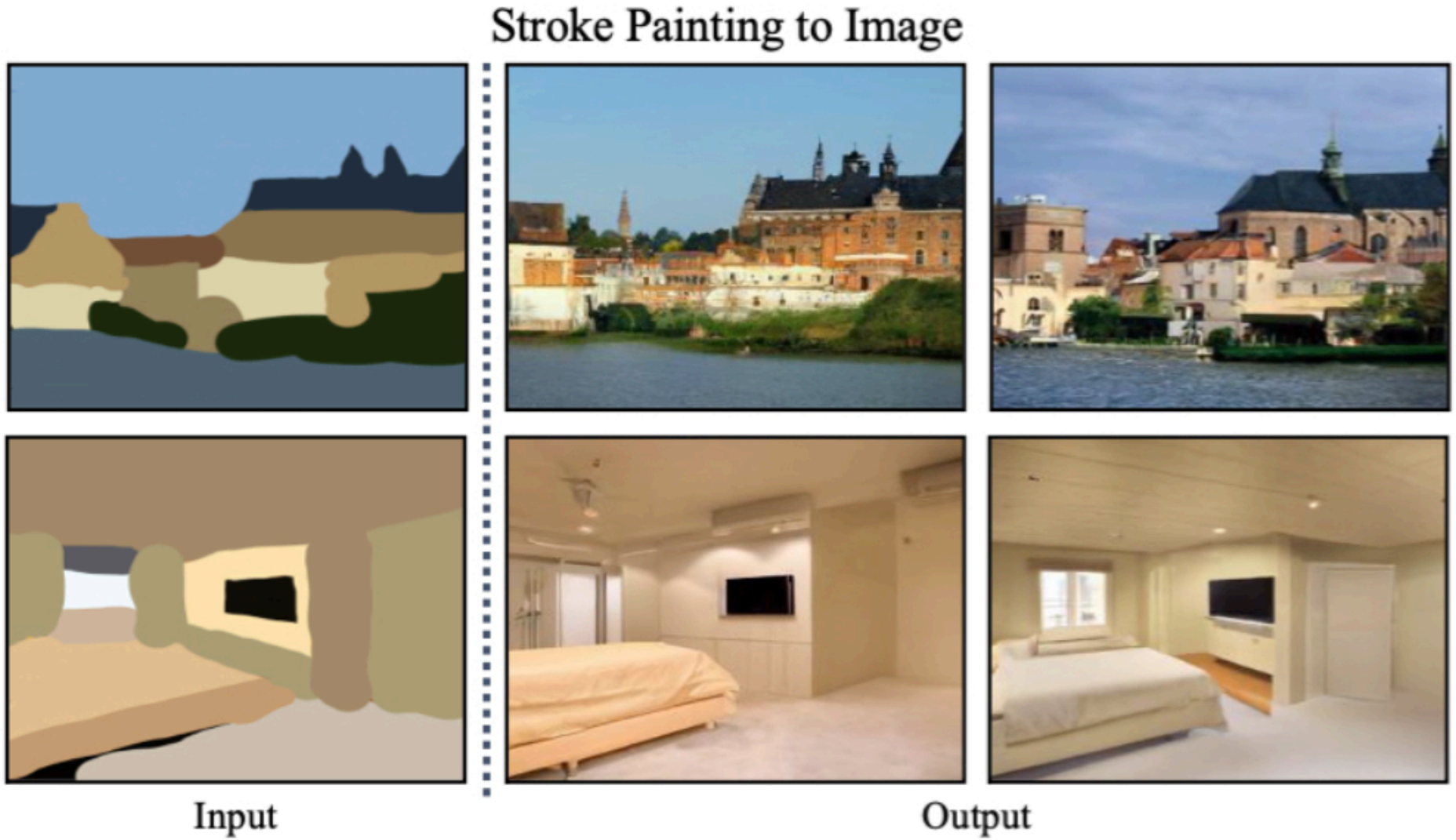


$\mathbf{x}_0$

Denoise using diffusion model trained on *real* images.

[Meng et al. "SDEdit", 2022]

Sketch to photo



Source: [Meng et al. “SDEdit”, 2022]

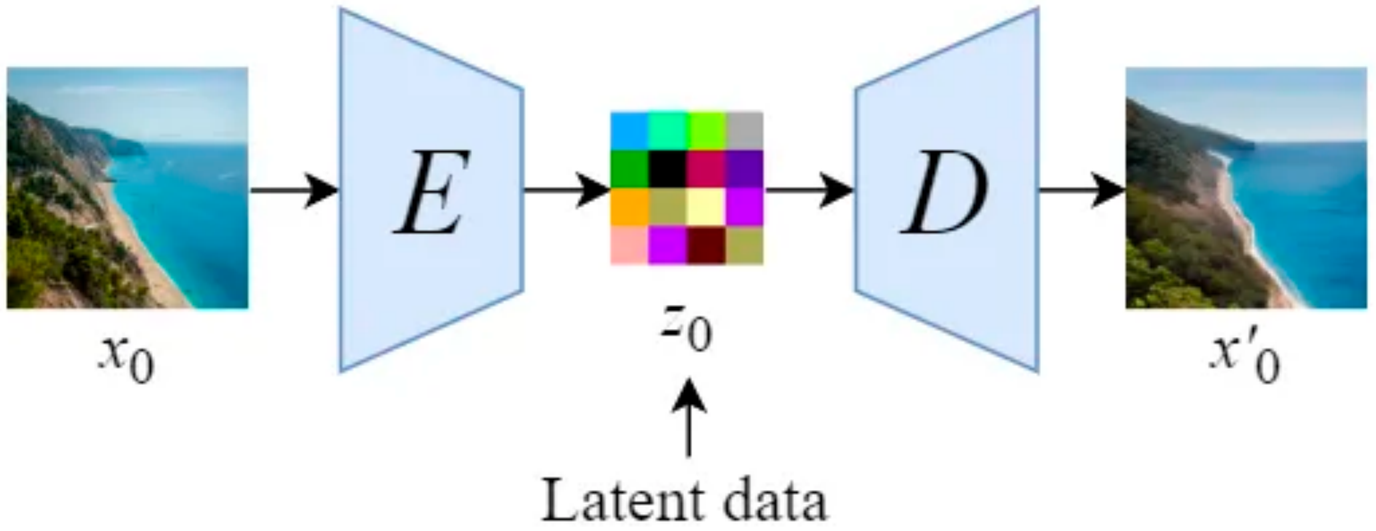
Generative models

Question: Explain how GANs are used in latent diffusion models.

Question: Suppose that you are training an autoregressive image generator. How do you represent the output of the model? I.e., what are the variables that you should sequentially predict?

Latent diffusion models

- Use a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run diffusion in latent space.
- Faster and focuses more on “perceptually important” details.

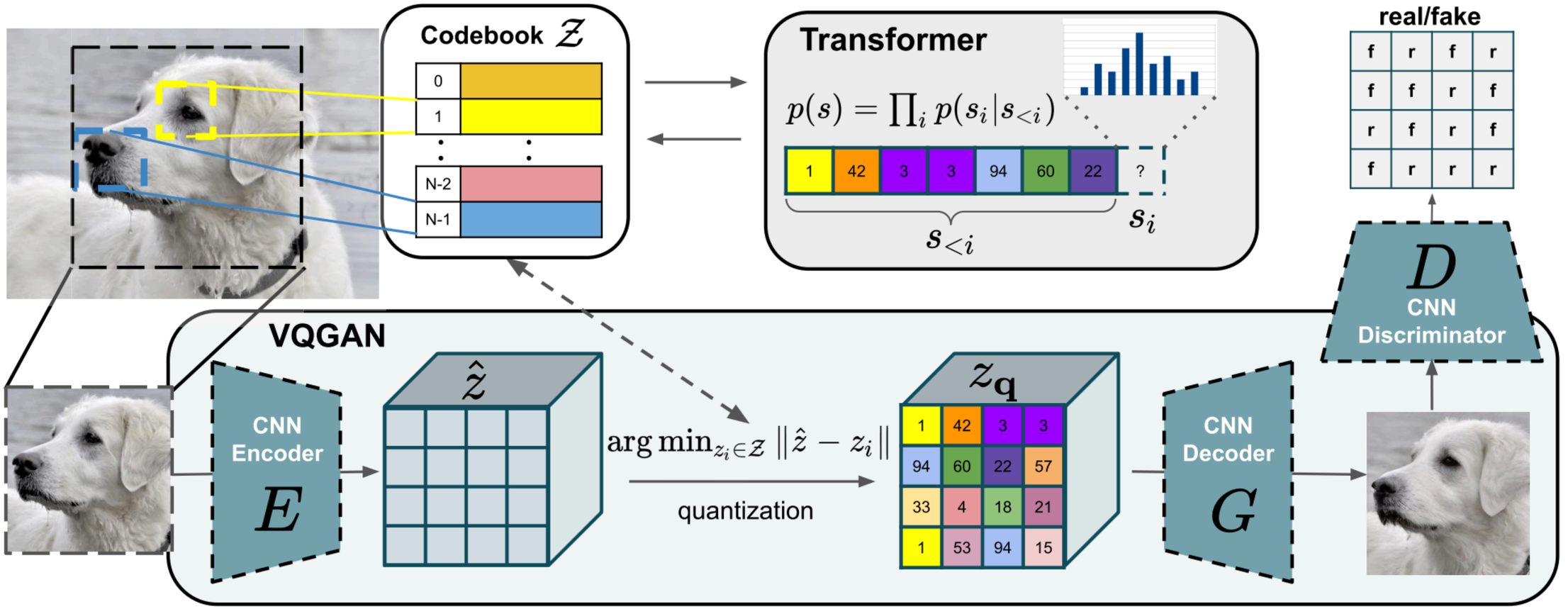


R. Rombach et al. High-Resolution Image Synthesis with Latent Diffusion Models. CVPR 2022

Figure source

Slide source: S. Lazebnik.

Vector quantized GAN (VQ-GAN)



Add a GAN loss to get crisper images

[Esser et al., "Taming Transformers for High-Resolution Image Synthesis", 2021]

3. Learning for vision

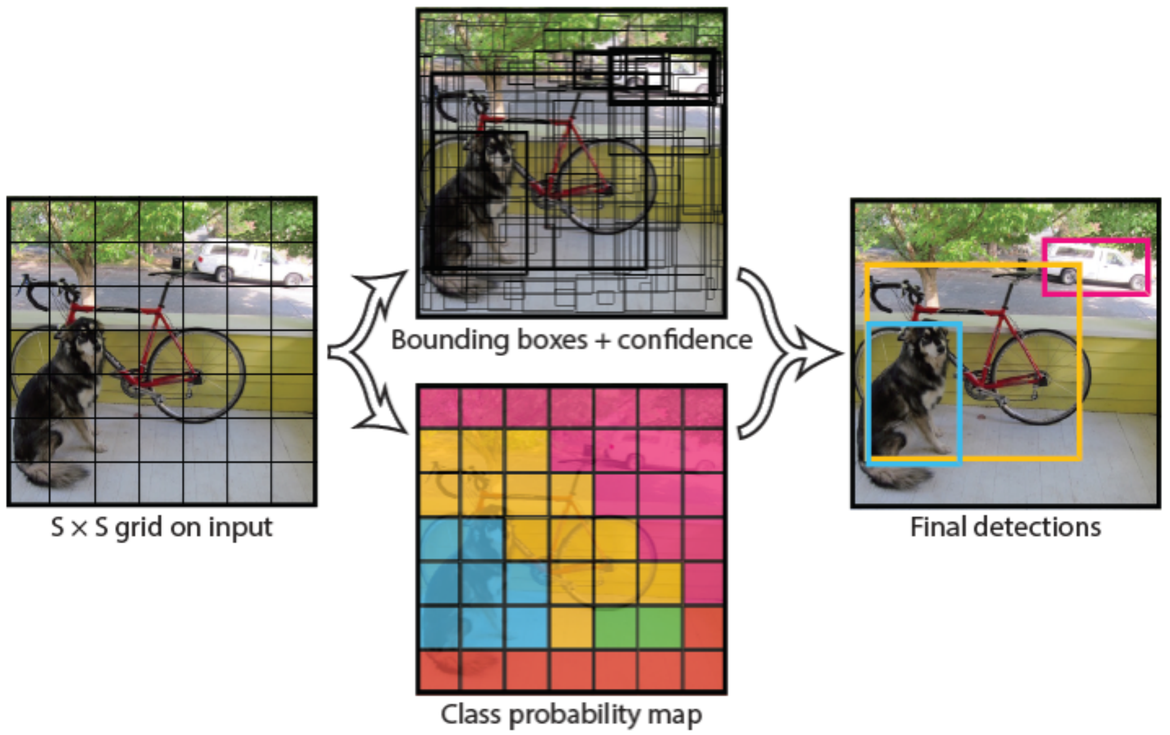
Learning for vision

Question: Explain the difference between one-stage and two-stage object detection and the trade-offs between the two approaches.

Single-stage object detector

- Divide the image into a coarse grid using a fully convolutional net

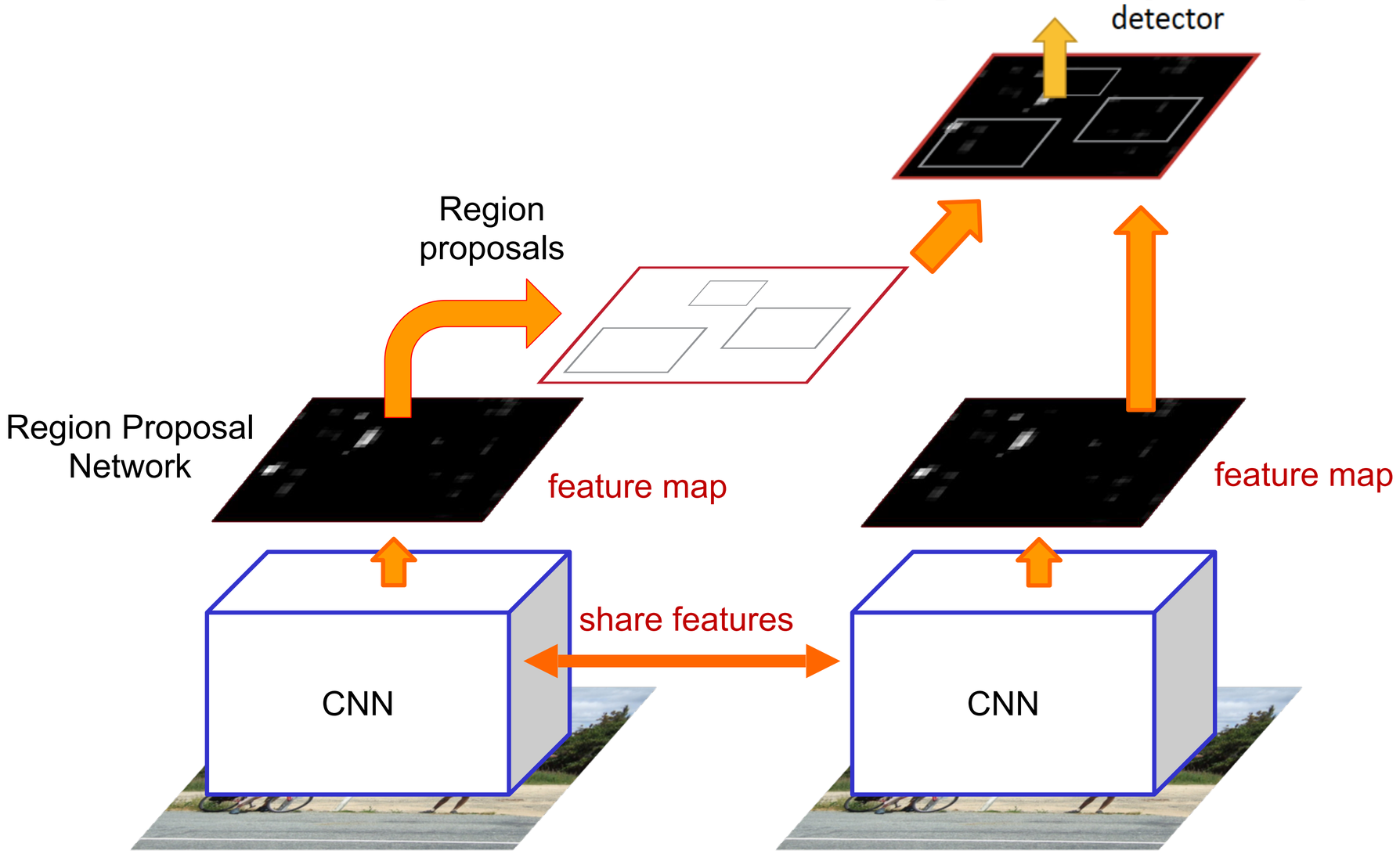
- Directly predict class label, confidence, and a few candidate boxes for each grid cell.



J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, CVPR 2016

Source: S. Lazebnik

“Faster” R-CNN: learn region proposals

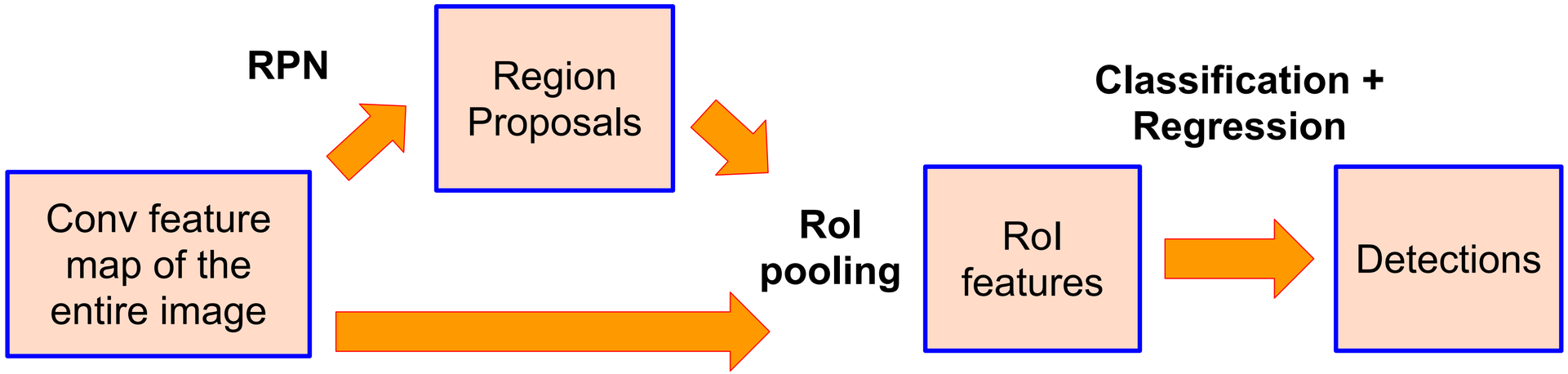


[Ren et al., “Faster R-CNN”, 2015]

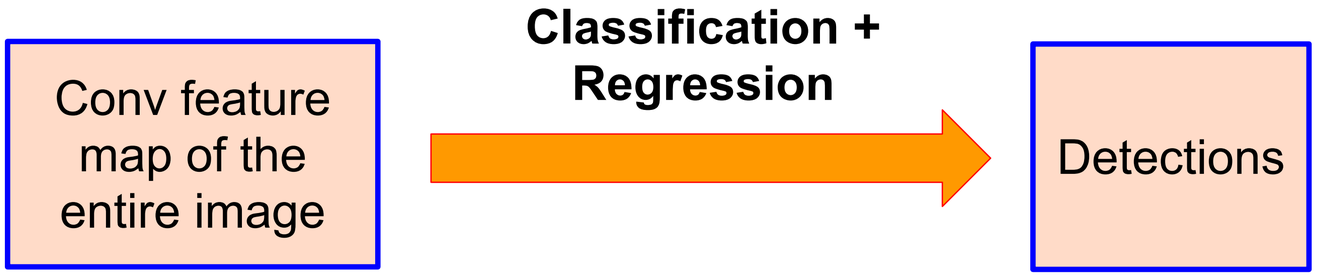
Source: S. Lazebnik

Streamlined detection architectures

- The Faster R-CNN pipeline separates proposal generation and region classification:



- Is it possible do detection in one shot?

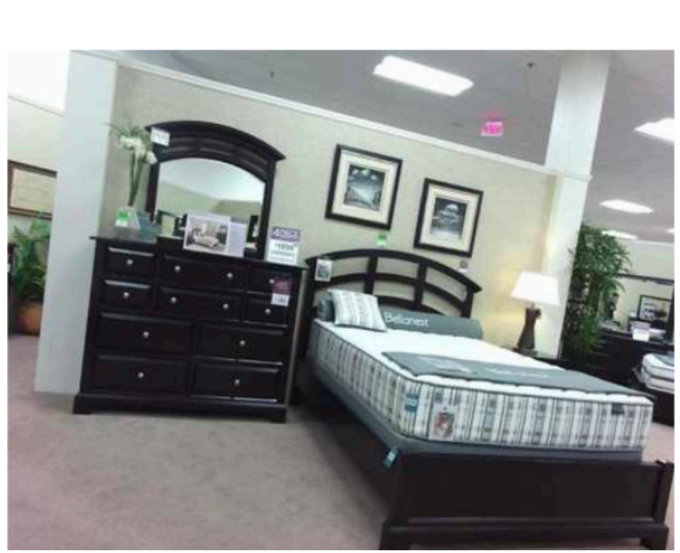


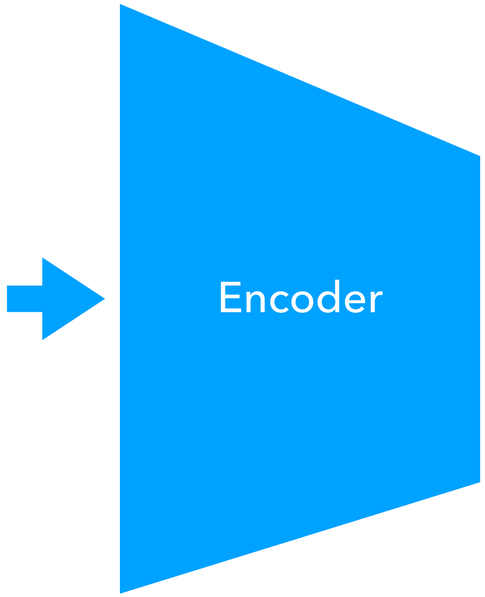
Source: S. Lazebnik

Learning for vision

Question: Suppose that you removed the skip connections in a U-net and retrained the model What would happen to output?

Encoder-decoder architectures

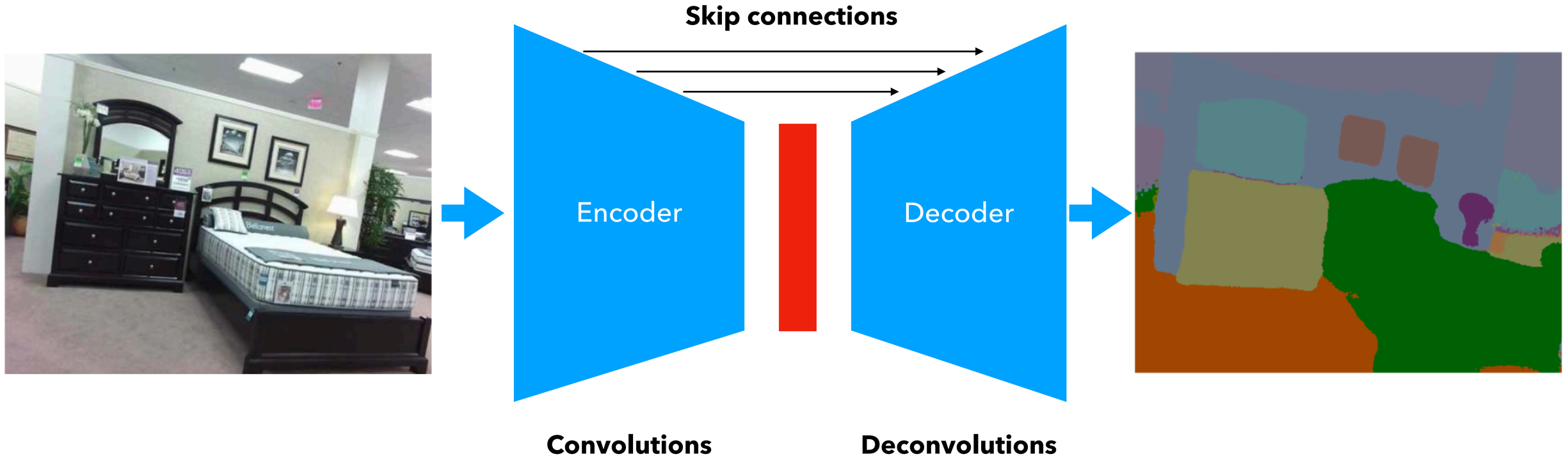




Convolutions **Deconvolutions**

Source: Torralba, Freeman, Isola

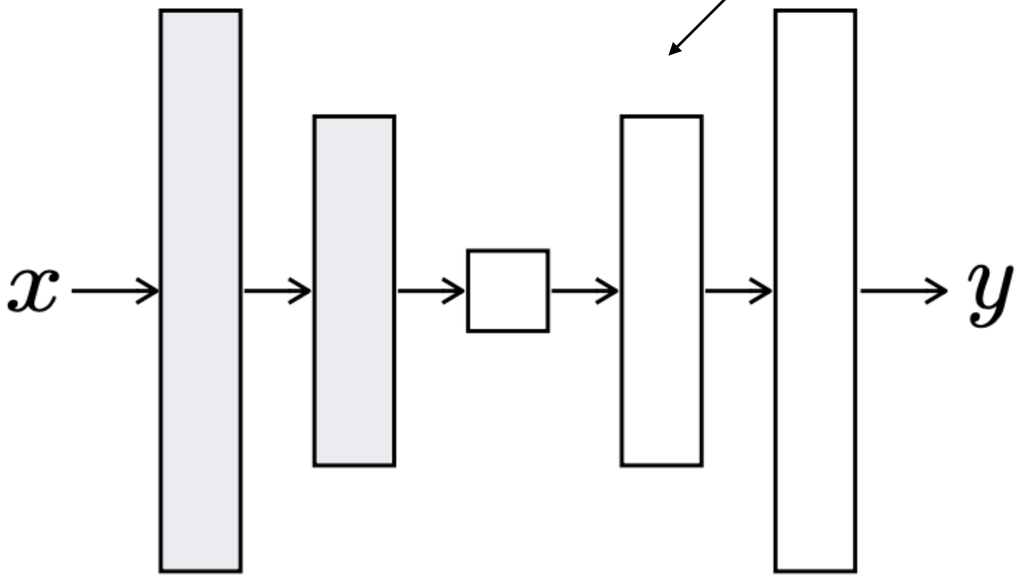
Can we prevent blurry predictions?



Source: Torralba, Freeman, Isola

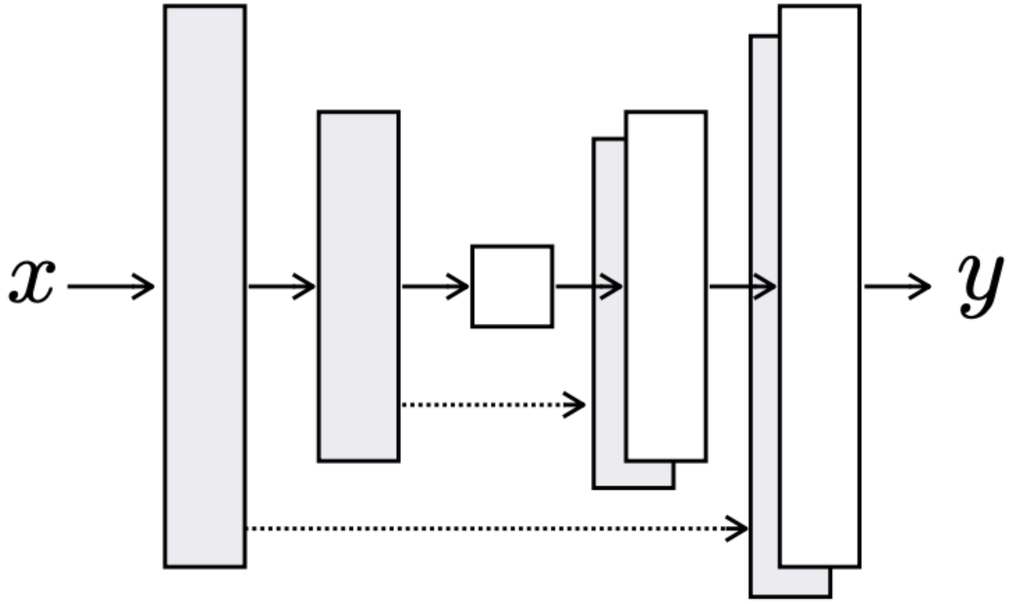
Encoder-decoder architectures

Transposed convolution



"Vanilla" encoder-decoder architecture

Early layers and late layers have same shape. Concatenate channel-wise!

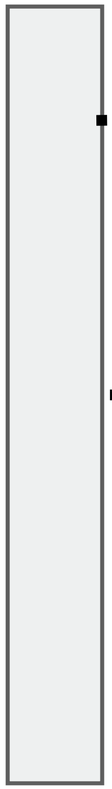


U-Net

Figures from [Isola et al., "Image-to-Image Translation with Conditional Adversarial Networks",

U-net

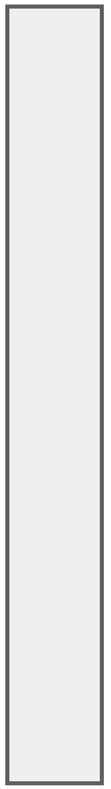
$W \times H$



$\frac{W}{2} \times \frac{H}{2}$

concat

$\cdots$

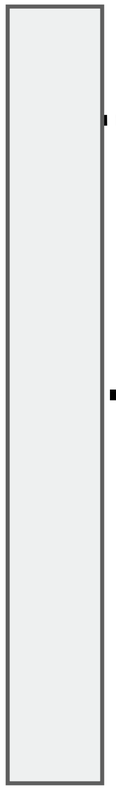


$\frac{W}{2} \times \frac{H}{2}$

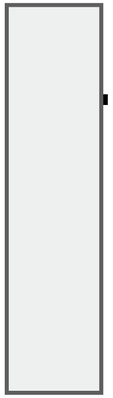
$W \times H$

U-net

$W \times H$



$\frac{W}{2} \times \frac{H}{2}$

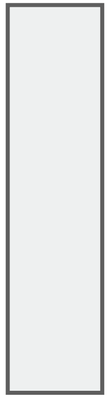


$\frac{W}{4} \times \frac{H}{4}$

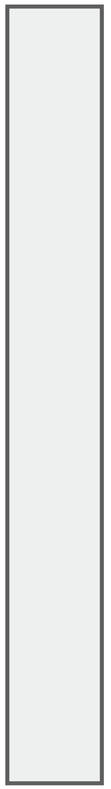
concat

concat

• • •



$\frac{W}{4} \times \frac{H}{4}$



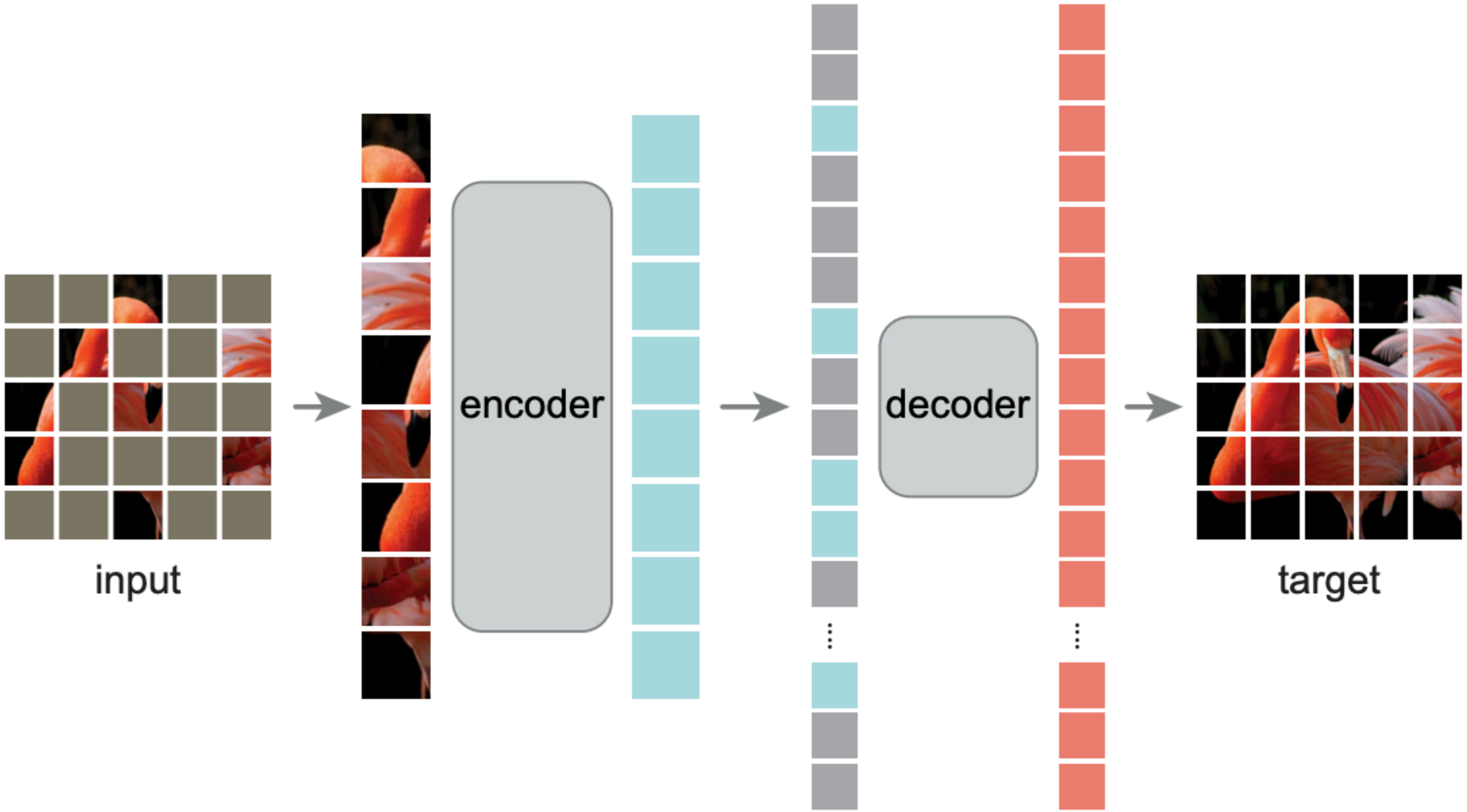
$\frac{W}{2} \times \frac{H}{2}$

$W \times H$

Learning for vision

Question: What are the challenges of implementing a masked autoencoder using a CNN?

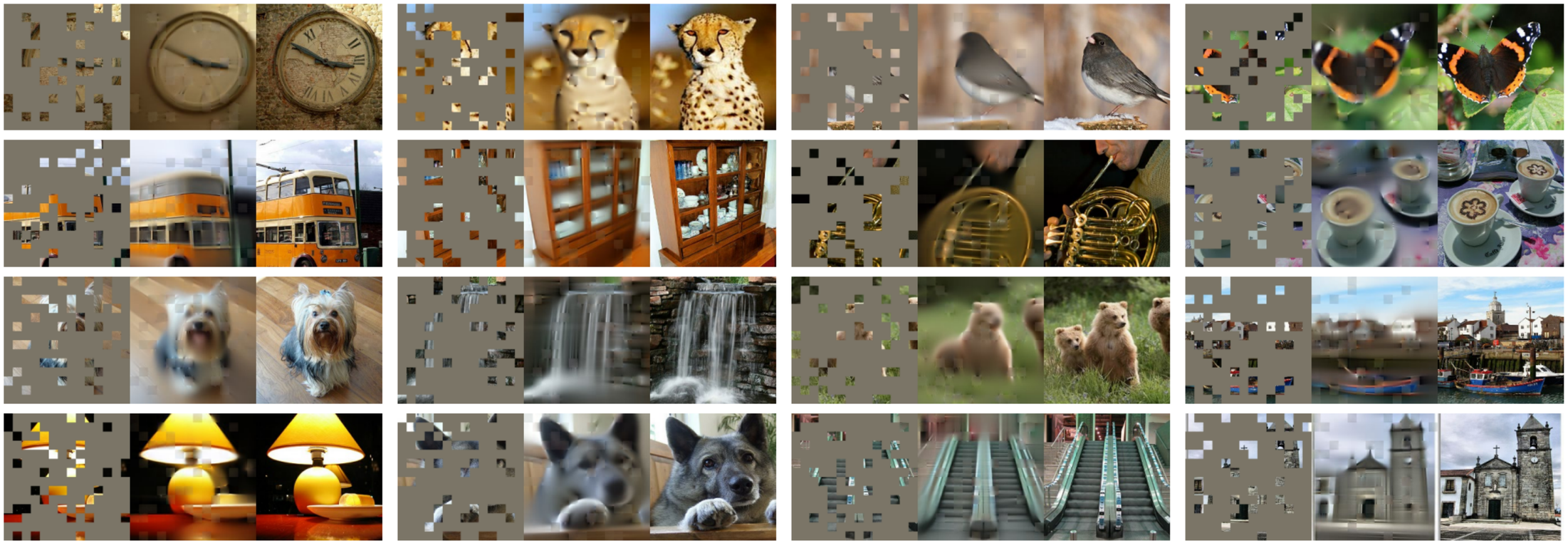
Application: self-supervised learning with masked autoencoders



Source: S. Lazebnik

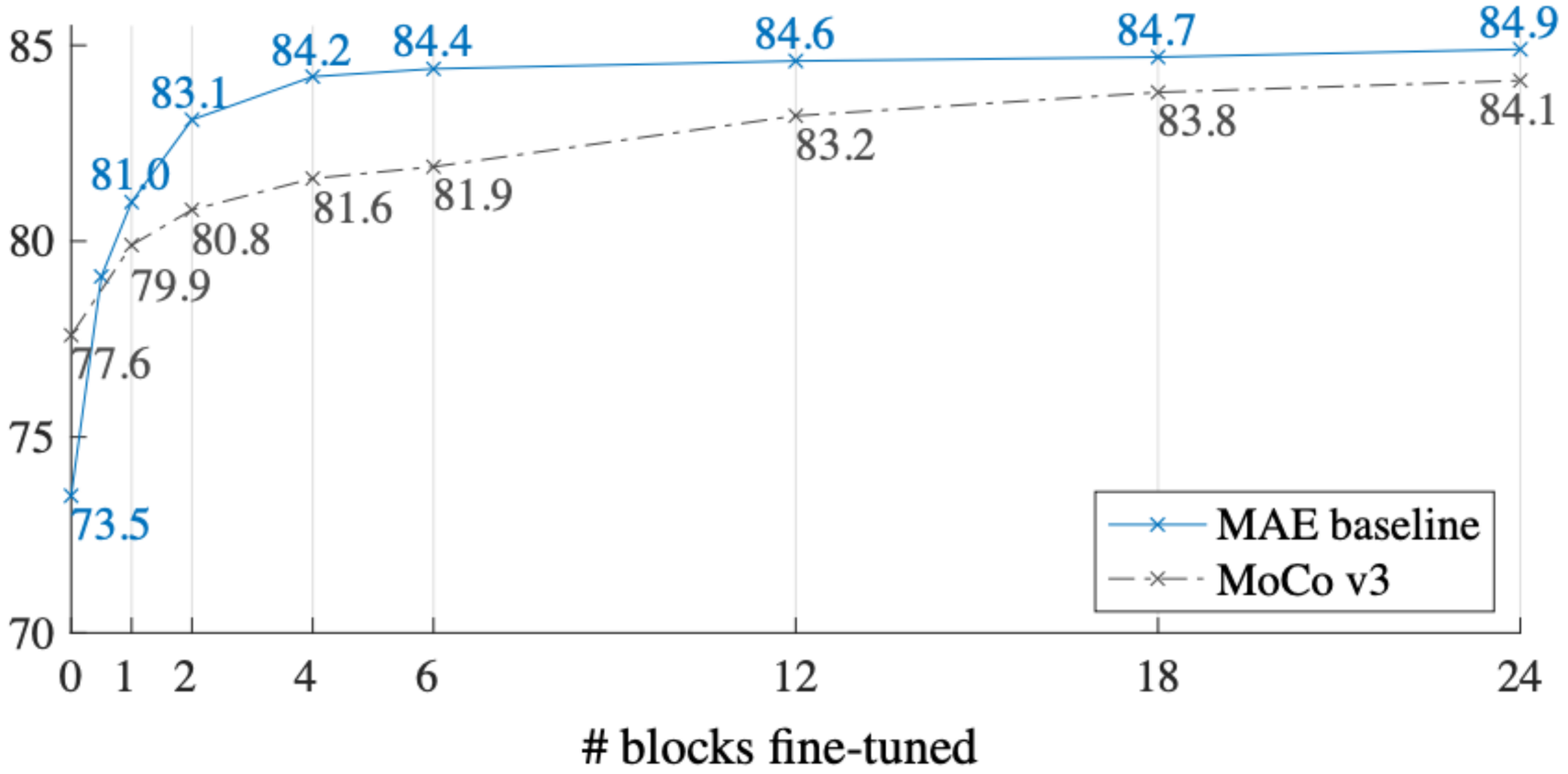
[Kaiming He et al., "Masked Autoencoders Are Scalable Vision Learners", 2021]

Application: self-supervised learning



[Kaiming He et al., “Masked Autoencoders Are Scalable Vision Learners”, 2021]

Application: self-supervised learning with masked autoencoders



[Kaiming He et al., "Masked Autoencoders Are Scalable Vision Learners", 2021]

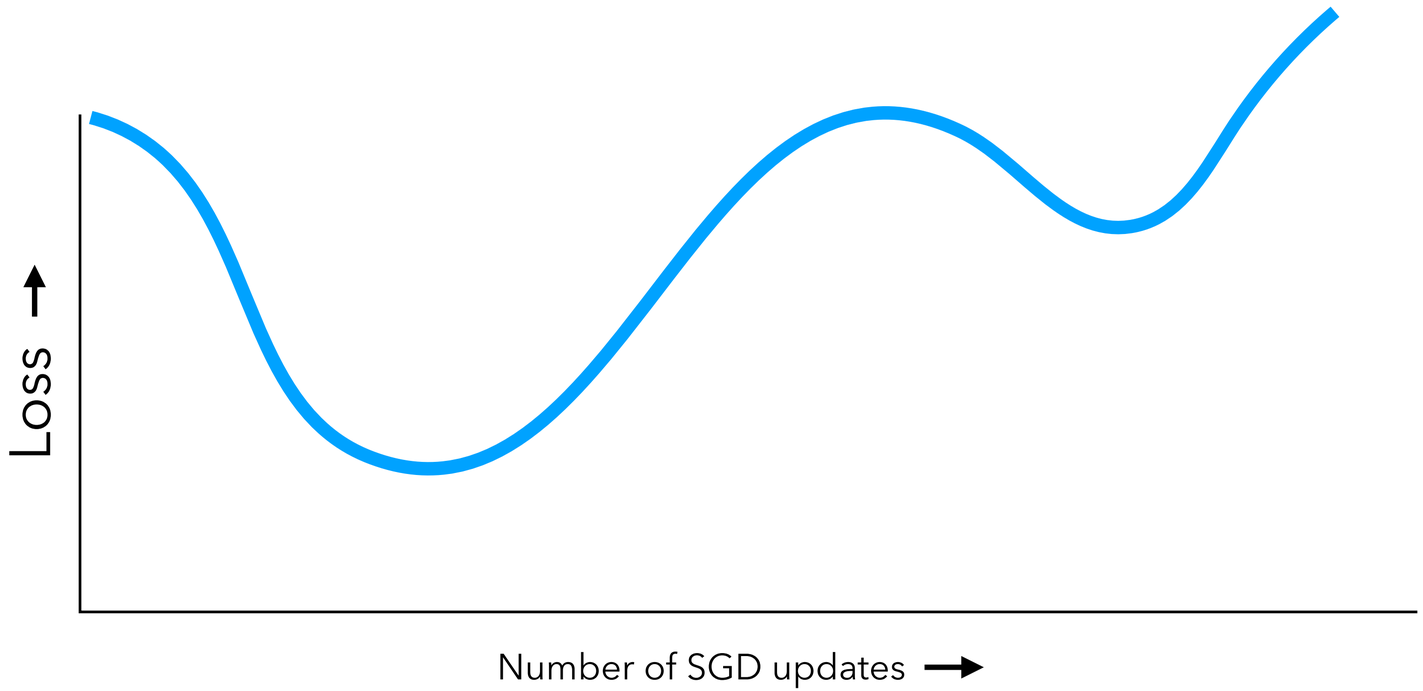
4. Machine learning basics

Learning

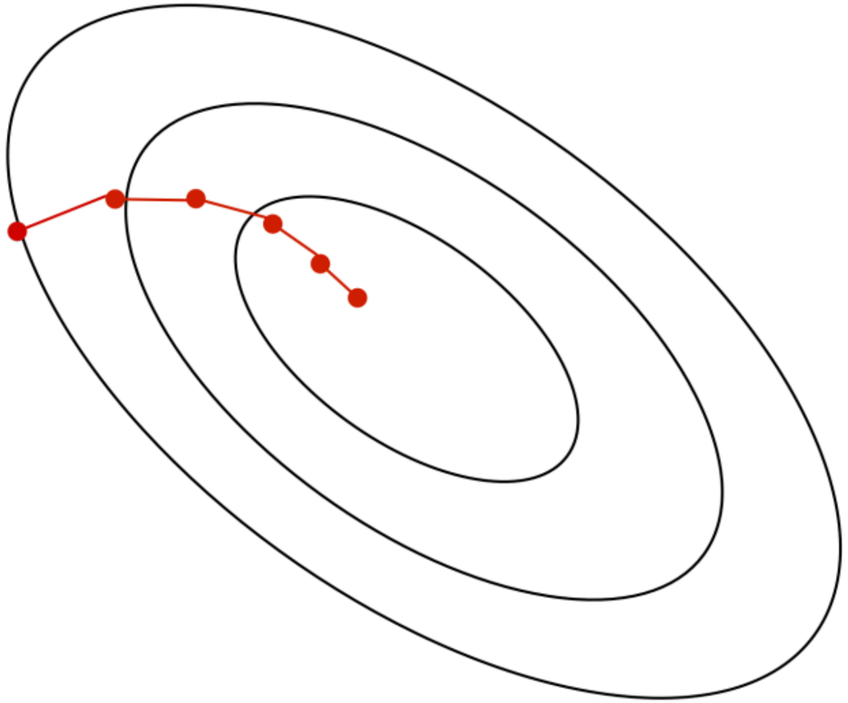
Question: Why might Stochastic Gradient Descent (SGD) be more effective than Batch Gradient Descent (BGD) when training neural networks?

Learning

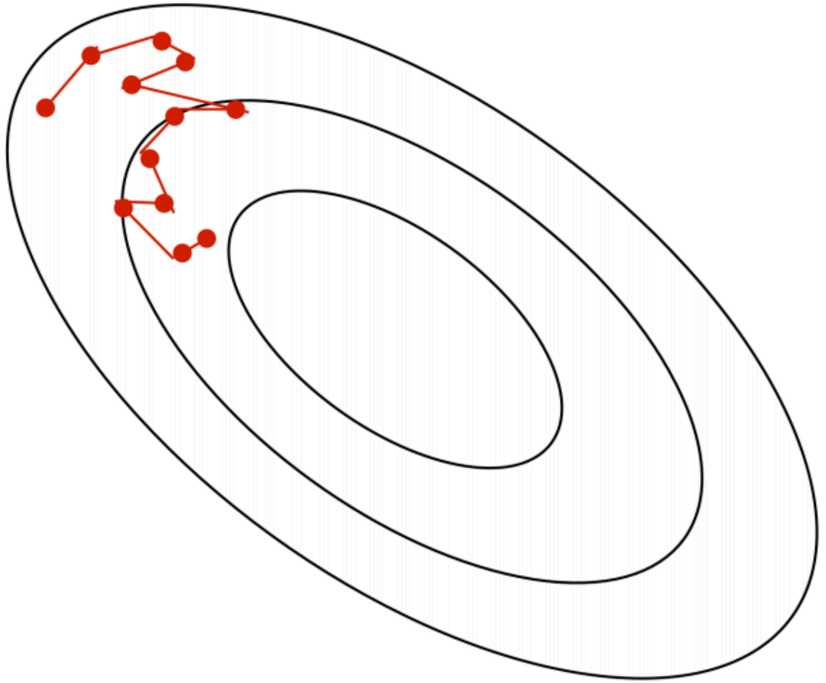
Question: Here is a cartoon learning curve. How might you adjust the model's learning rate to address the problem?



Stochastic gradient descent



Batch gradient descent

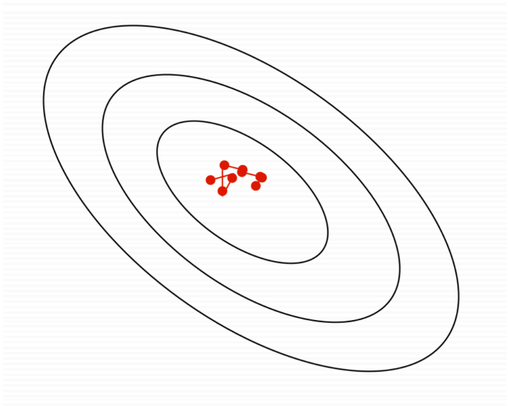


Stochastic gradient descent

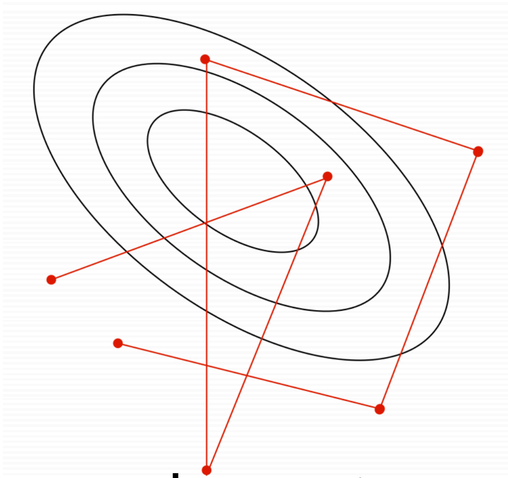
Source: R. Grosse

Learning rate

- Sensitive to the learning rate: $\theta^{t+1} = \theta^t - \eta_t \nabla_\theta J(\theta)\Big|_{\theta=\theta^t}$



Small learning rate

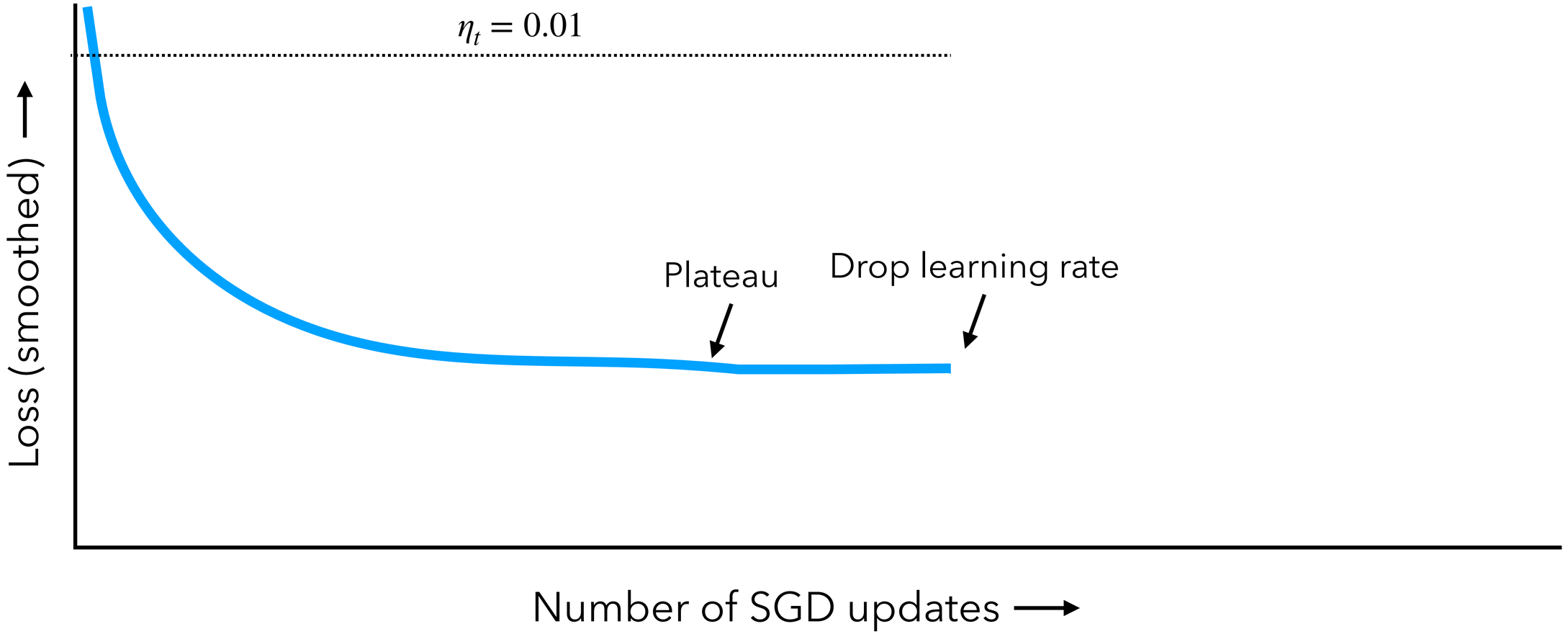


Large learning rate

Figure source: R. Grosse

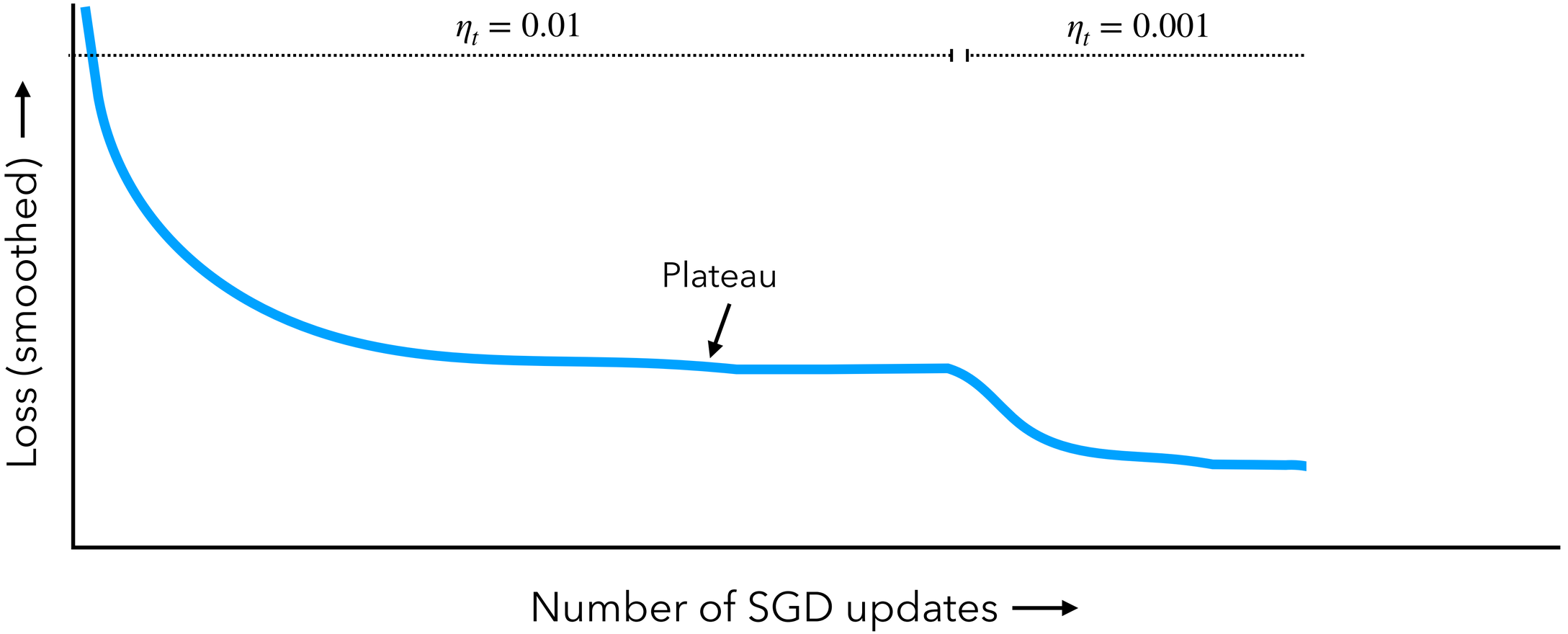
Learning rate schedules

- Start with high learning rate, and decrease over time.



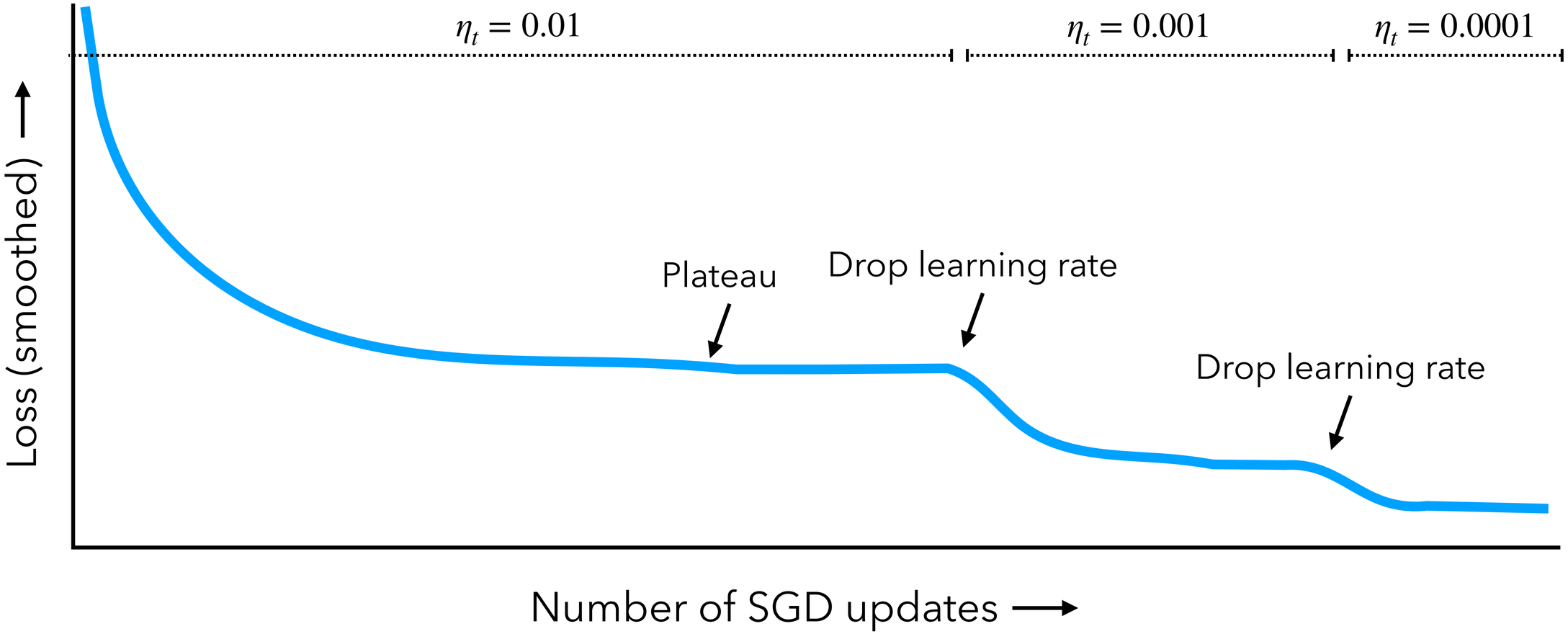
Learning rate schedules

- Start with high learning rate, and decrease over time.



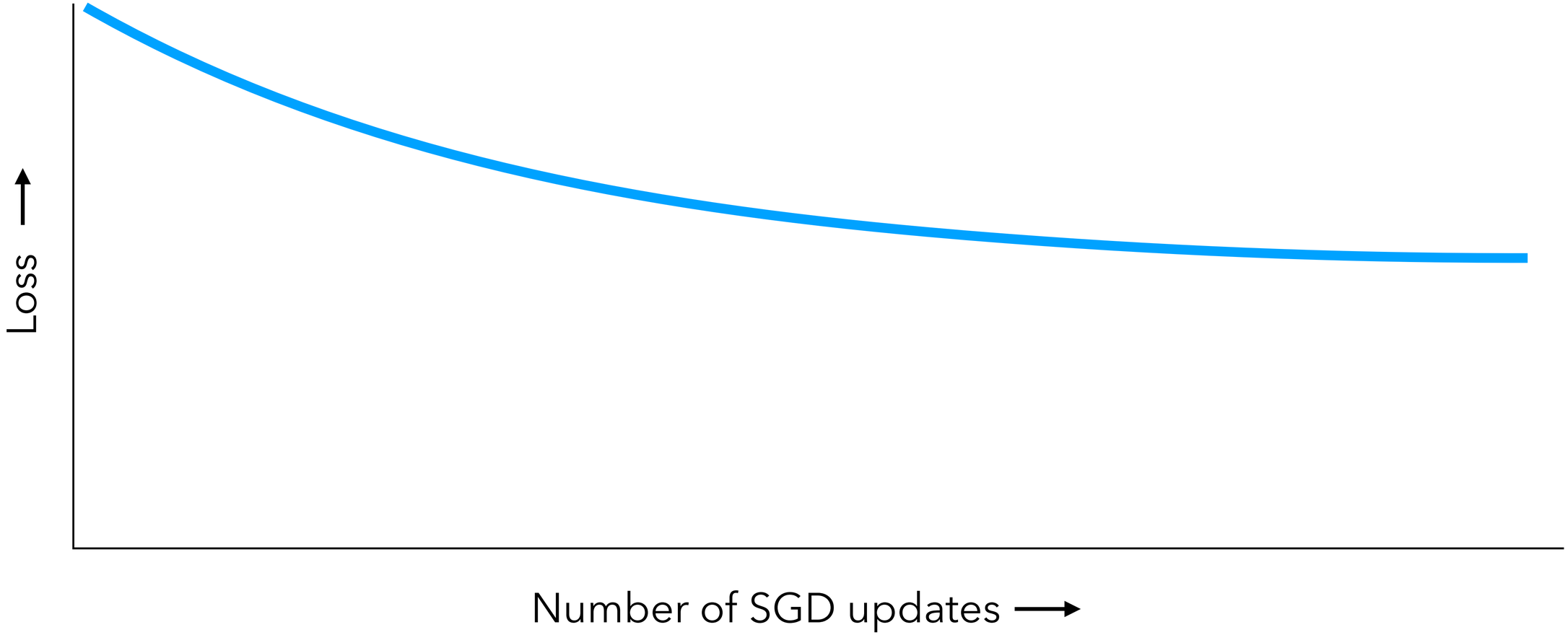
Learning rate schedules

- Start with high learning rate, and decrease over time.



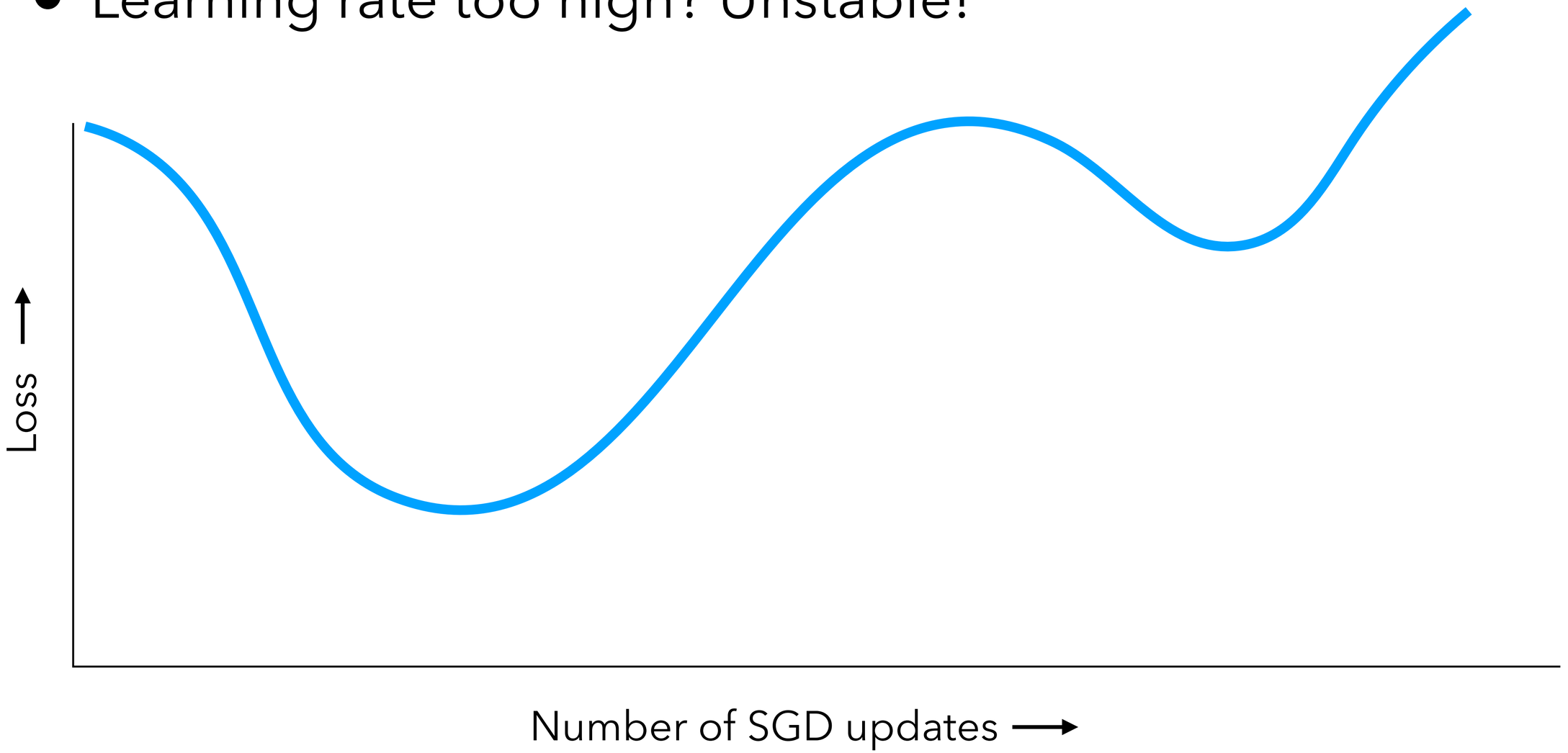
Learning rate schedules

- What if you always use a small learning rate?
- Loss goes down extremely slowly



Learning rate schedules

- Learning rate too high? Unstable!



Learning rate schedules

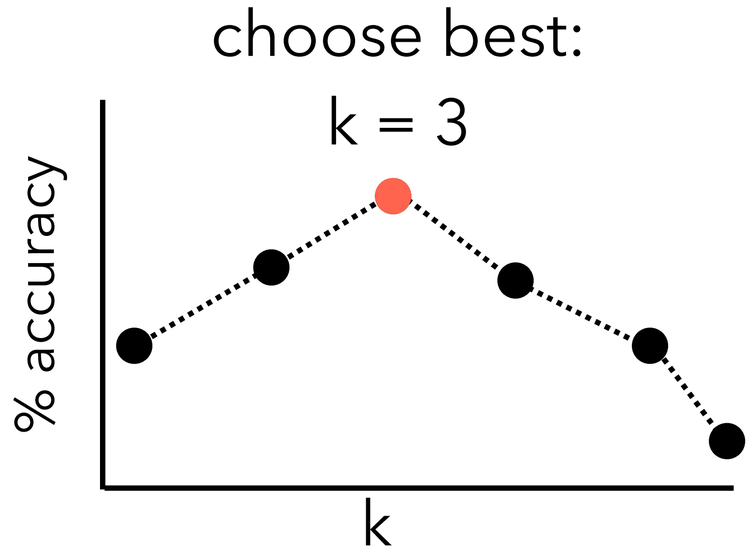
- Choosing the initial learning rate is surprisingly hard. Often requires grid search, i.e., trying many rates and choosing the best.
- When do you drop the learning rate? Some strategies:
  - Wait until validation or training loss plateaus, then drop it (e.g., by a factor of 10).
  - Smoothly drop the learning rate over time. Requires choosing the rate of dropping.
  - Warm up: make the beginning of training easier. Start learning rate at 0 and gradually increase for the first few iterations.
- Another option: decrease *and* increase the learning rate using a periodic function (e.g., cosine) [Loshchilov & Hutter, 2017]

Learning

Question: Explain how to choose *k* in k-nearest neighbor learning.

How do we choose k?

How do we choose k?



Pool of nearest neighbors

Training Set | Validation Set | Test Set

Measures *generalization*

Choose **hyperparameters** like k, feature space, similarity function, etc.

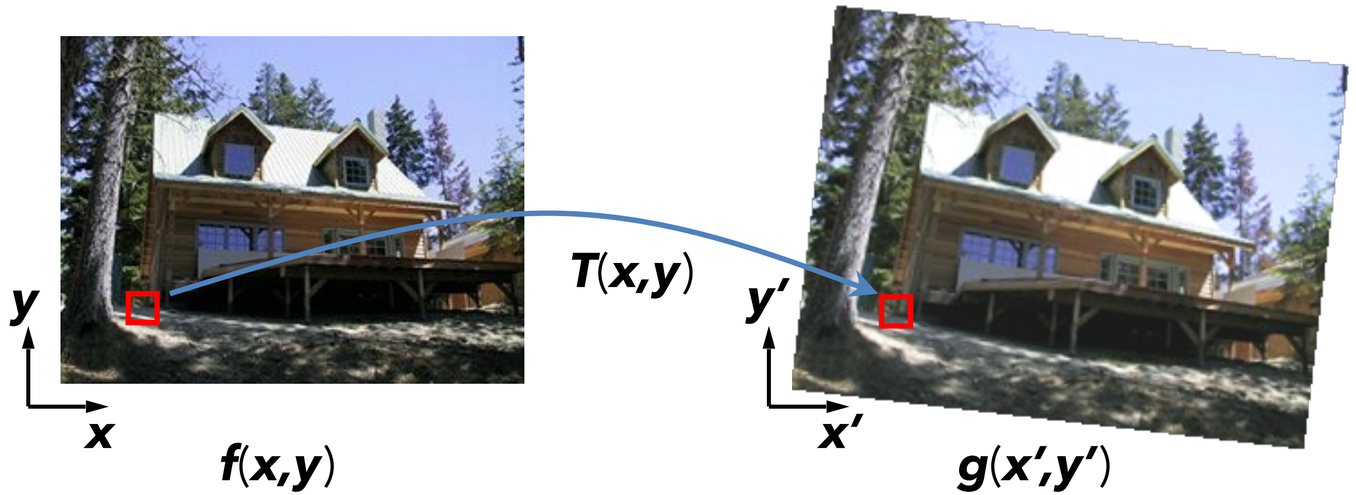
5. Image processing

Image processing

Question: What are the advantages of backward warping over forward warping?

Image warping

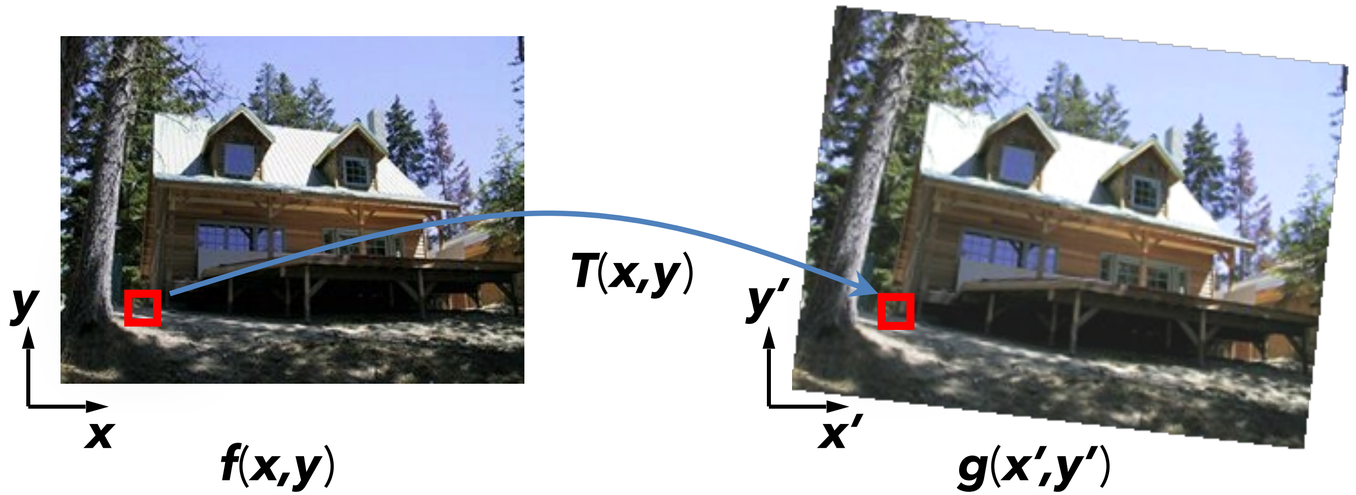
Given a coordinate transformation $(x', y') = T(x, y)$ and a source image $f(x, y)$, how do we compute a transformed image $g(x', y') = f(T(x, y))$?



Source: N. Snavely

Forward warping

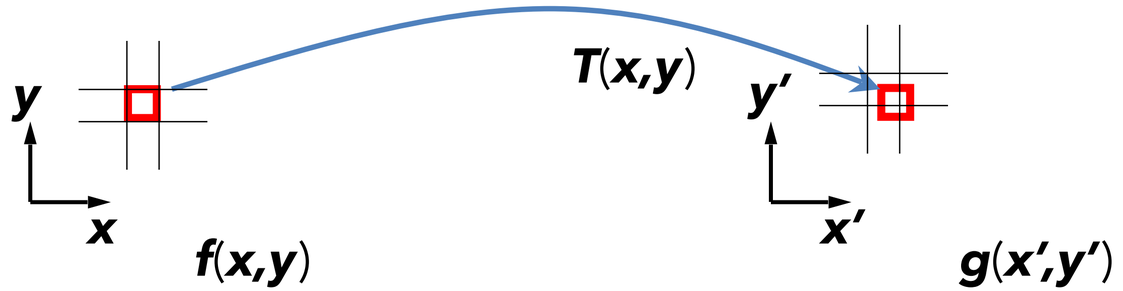
- Send each pixel $\boldsymbol{f(x)}$ to its corresponding location $\boldsymbol{(x',y')} = \boldsymbol{T(x,y)}$ in $\boldsymbol{g(x',y')}$
- What if a pixel lands "between" two pixels?



Source: N. Snavely

Forward warping

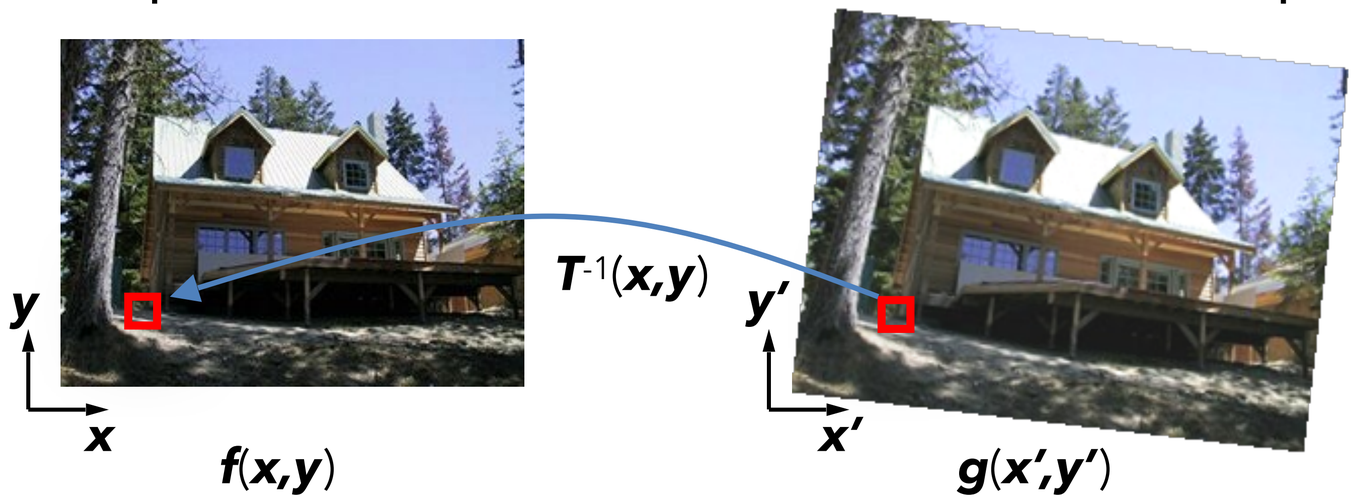
- Send each pixel $\boldsymbol{f(x)}$ to its corresponding location $\boldsymbol{(x',y')} = \boldsymbol{T(x,y)}$ in $\boldsymbol{g(x',y')}$
- What if a pixel lands "between" two pixels?
  - Answer: add "contribution" to several pixels, normalize later (*splatting*)
  - Can still result in holes



Source: N. Snavely

Backward (inverse) warping

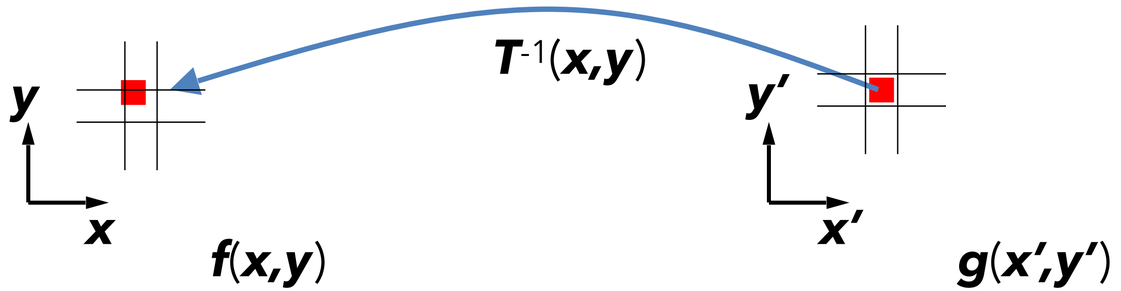
- Get each pixel $g(x',y')$ from its corresponding location $(x,y) = T^{-1}(x,y)$ in $f(x,y)$
  - Requires taking the inverse of the transform
  - What if pixel comes from "between" two pixels?



Source: N. Snavely

Backward (inverse) warping

- Get each pixel $g(x')$ from its corresponding location $x' = h(x)$ in $f(x)$
  - What if pixel comes from "between" two pixels?
  - Answer: *resample* color value from *interpolated* source image



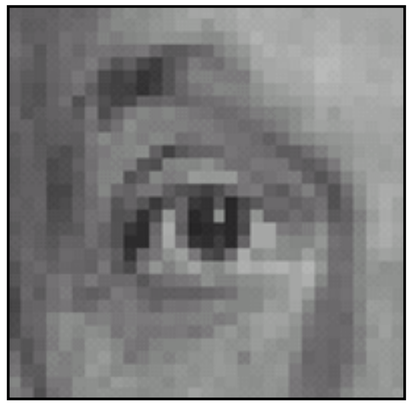
Source: N. Snavely

Image processing

Question: Can you do the following using a **convolution**? If so, give a filter that performs it.

(a) Increase the brightness of an image by a factor of 2.
(b) Zoom in the image by a factor of 2.
(c) Shift the image to the left by 2 pixels.

Practice with neighborhood filtering



Original

$*$

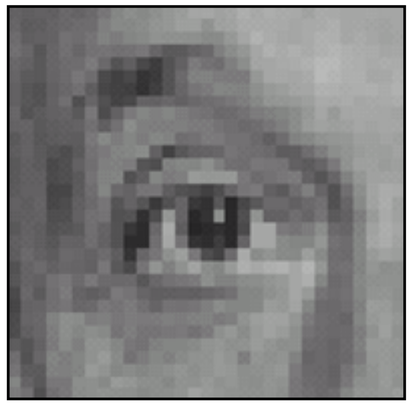
0	0	0
0	1	0
0	0	0

?

The operation we've seen is known as **cross correlation**.

Source: D. Lowe

Practice with neighborhood filtering

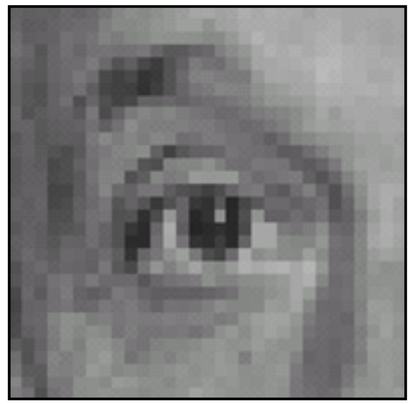


Original

$*$

0	0	0
0	1	0
0	0	0

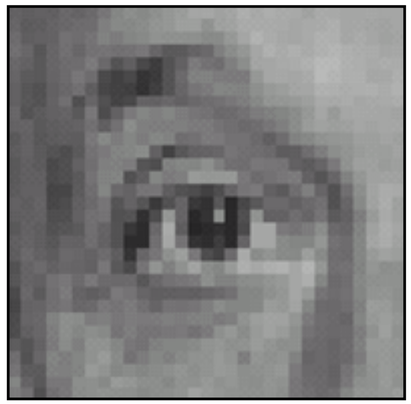
"Impulse"



Filtered
(no change)

Source: D. Lowe

Practice with neighborhood filtering



Original

$*$

0	0	0
0	0	1
0	0	0

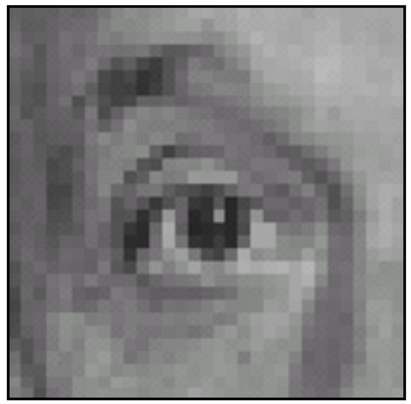
"Translated Impulse"

?

Source: D. Lowe

Practice with neighborhood filtering

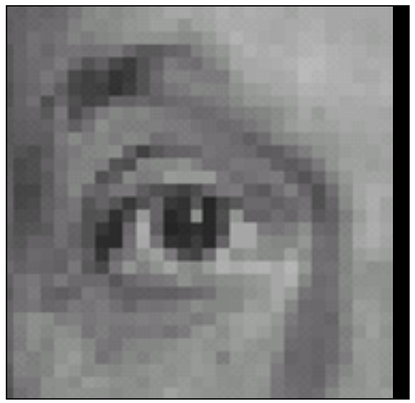
Zero padding



Original

$*$

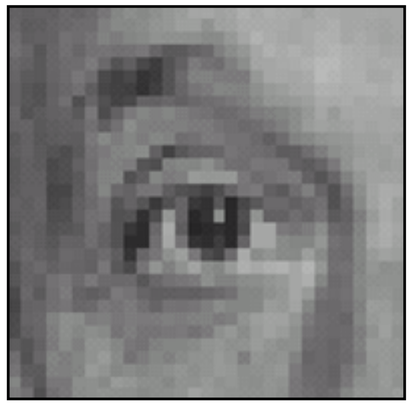
0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with neighborhood filtering



Original

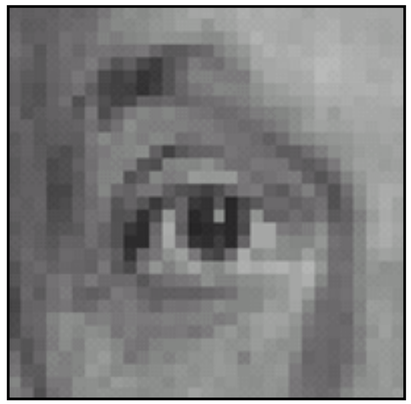
$$* \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

"Box filter"

?

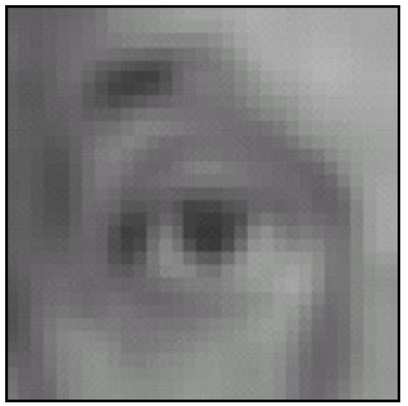
Source: D. Lowe

Practice with neighborhood filtering



Original

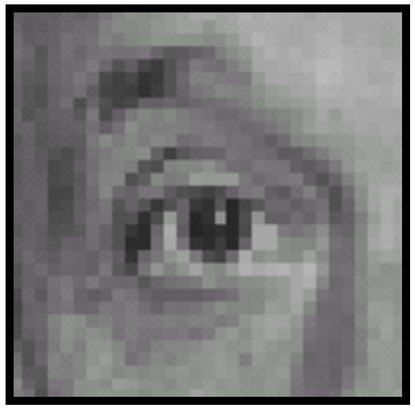
$$* \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



Blur

Source: D. Lowe

Practice with neighborhood filtering



Original

$*$

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

?

Thank you!