Detour: Time complexity of convolution

- Image is w x h
- Filter is k x k
- Every entry takes $O(k^2)$ operations
- Number of output entries:
  - (w+k-1)(h+k-1) for full
  - wh for same
- Total time complexity:
  - $O(whk^2)$

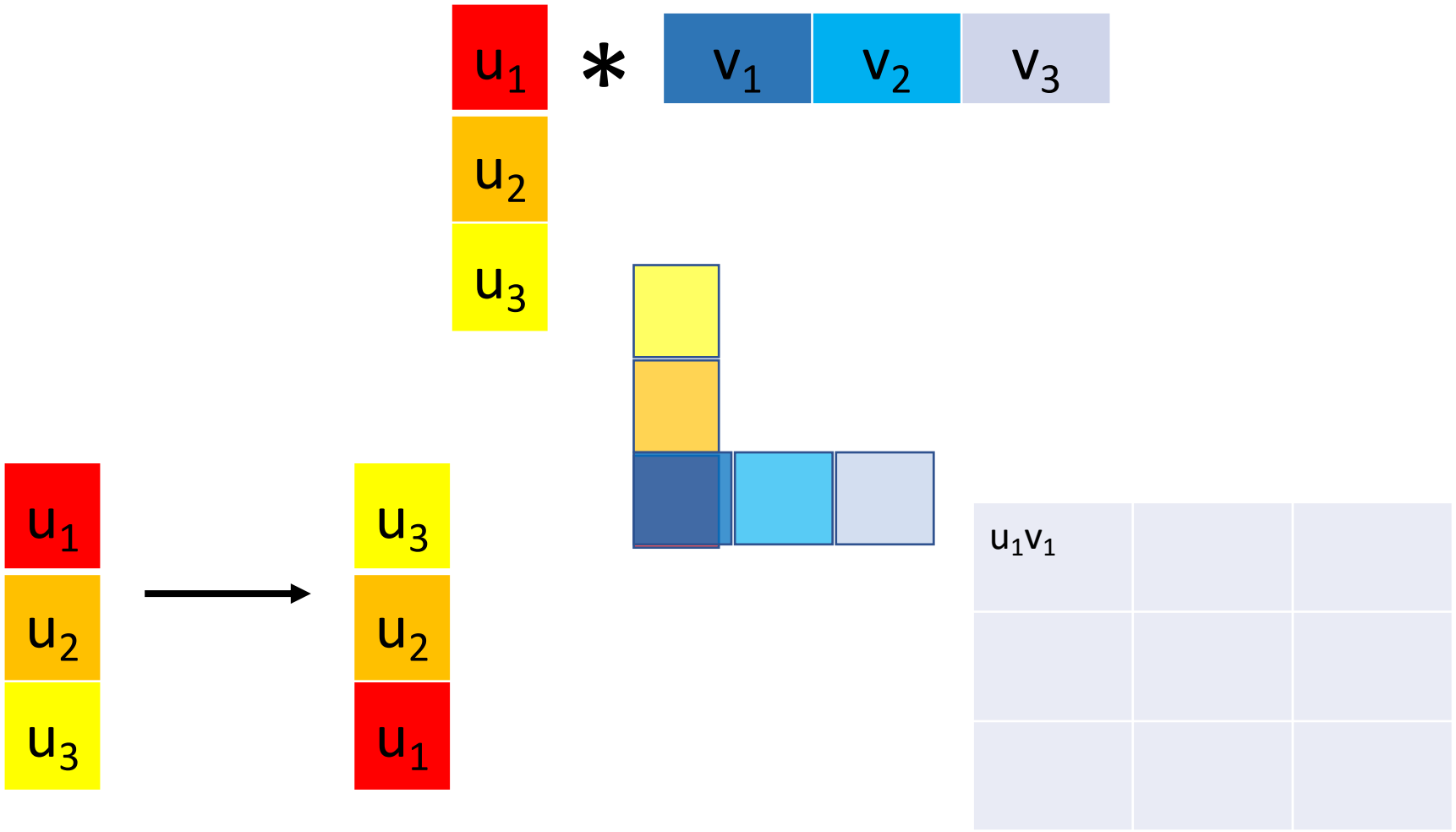
Optimization: separable filters

- basic alg. is $O(r^2)$: large filters get expensive fast!
- definition: $w(x,y)$ is *separable* if it can be written as:

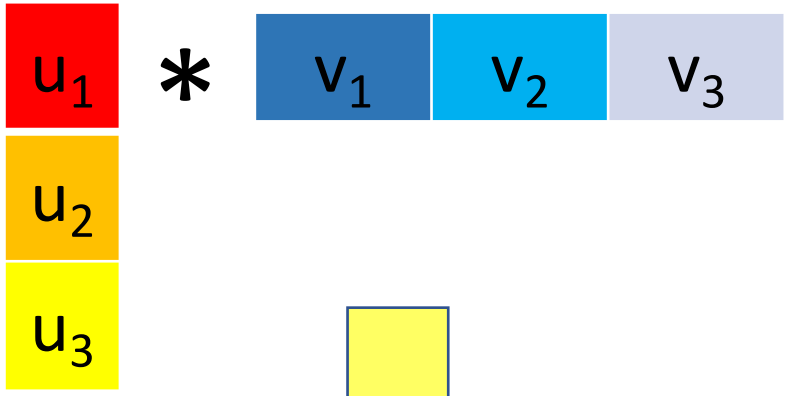
$$w(i,j) = u(i)v(j)$$

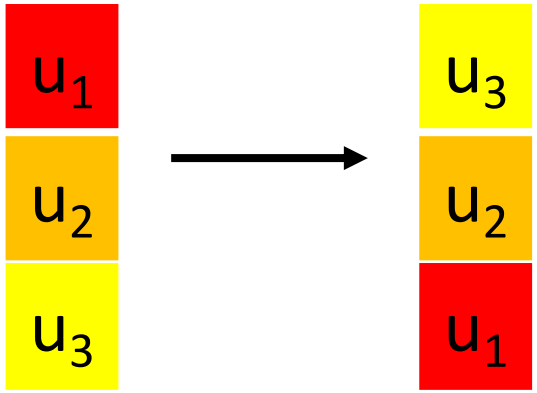
- Write u as a k x 1 filter, and v as a 1 x k filter
- Claim: $w = u * v$

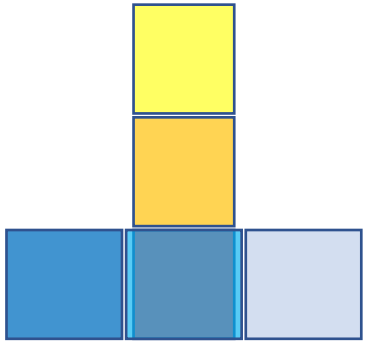
Separable filters



Separable filters

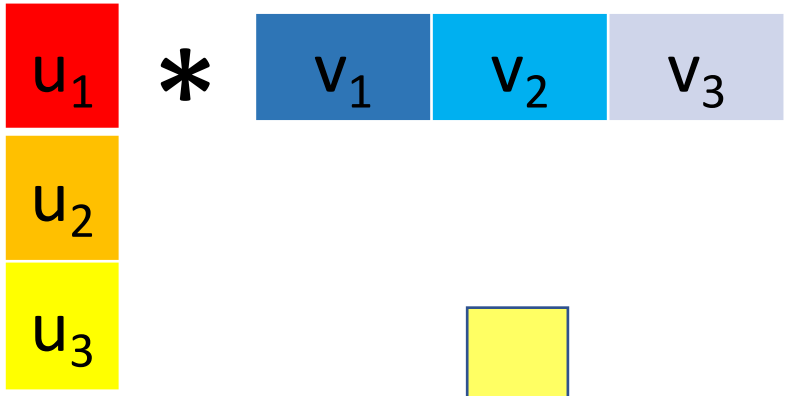


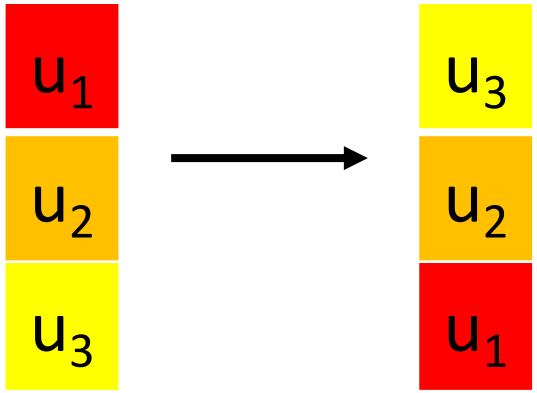


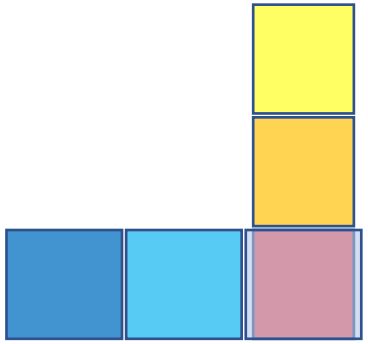


u_1v_1	u_1v_2	

Separable filters

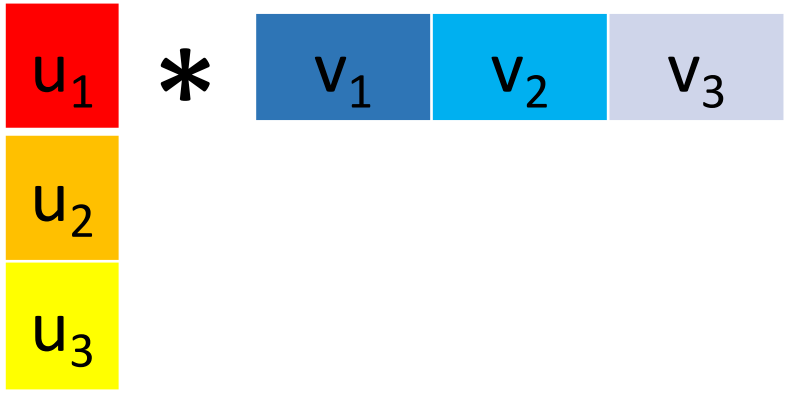


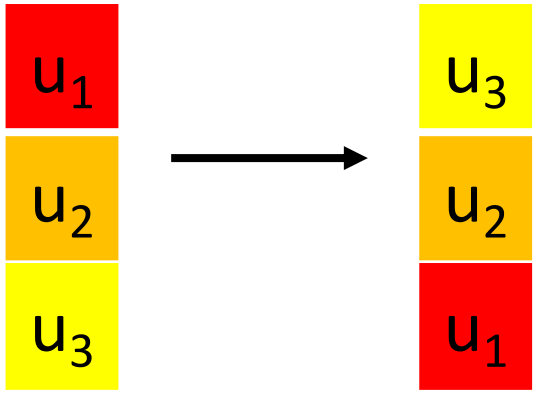


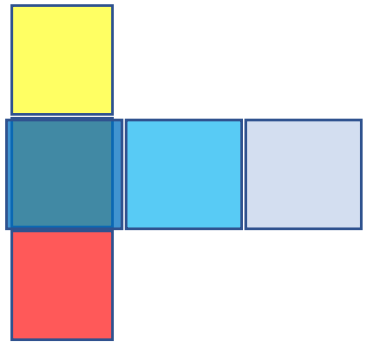


u_1v_1	u_1v_2	u_1v_3

Separable filters

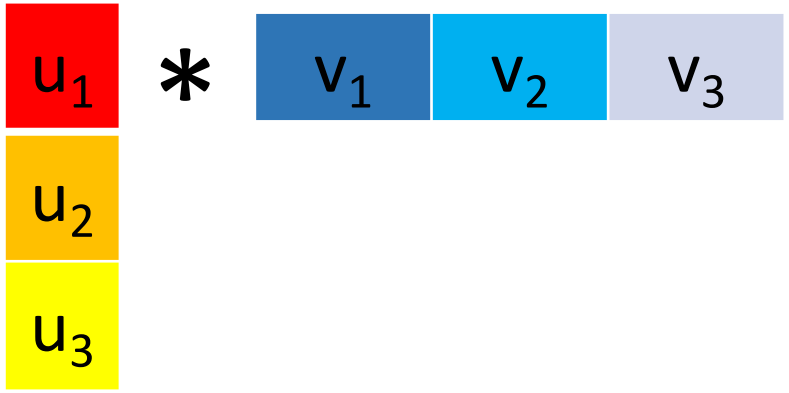


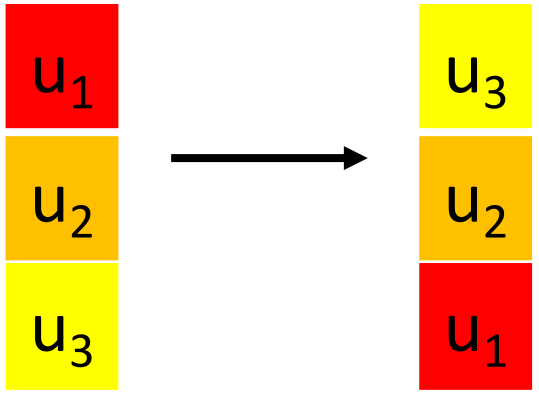


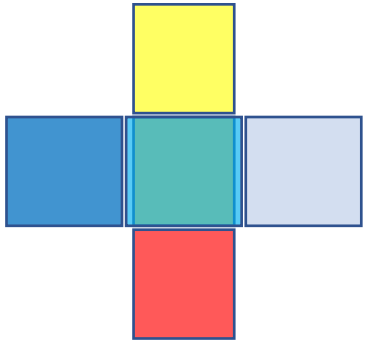


u_1v_1	u_1v_2	u_1v_3
u_2v_1		

Separable filters

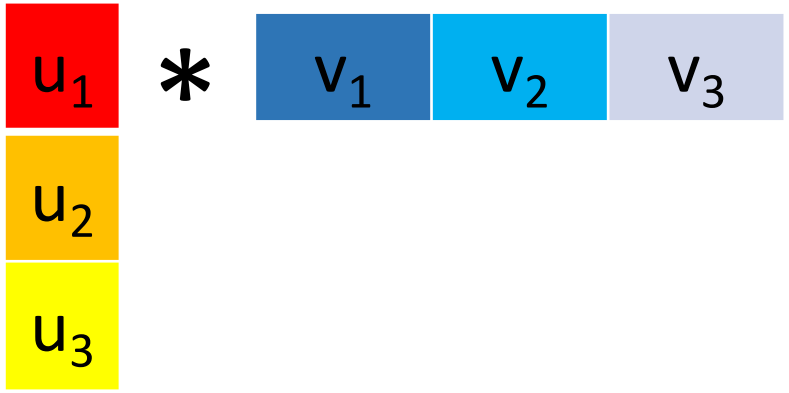


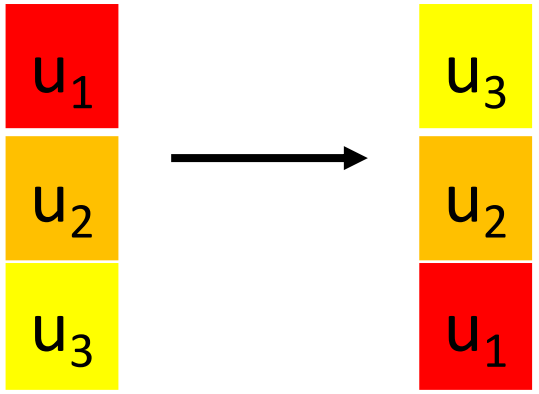


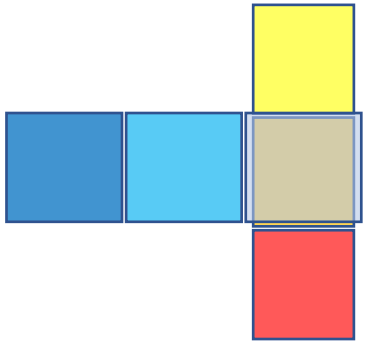


u_1v_1	u_1v_2	u_1v_3
u_2v_1	u_2v_2	

Separable filters

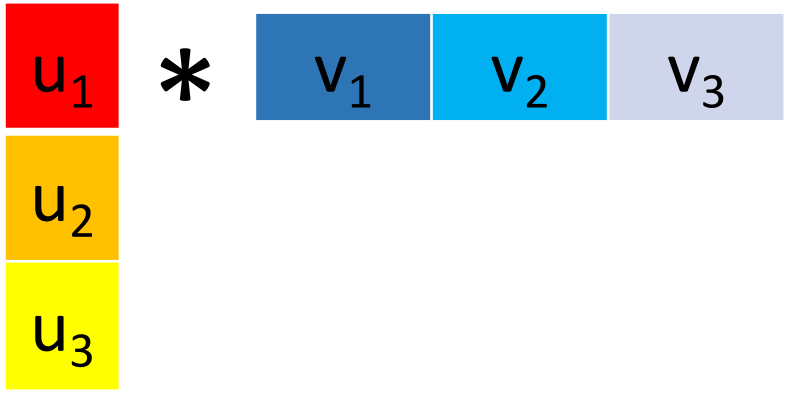


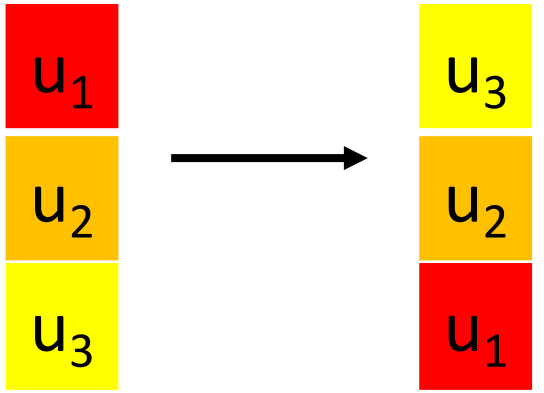


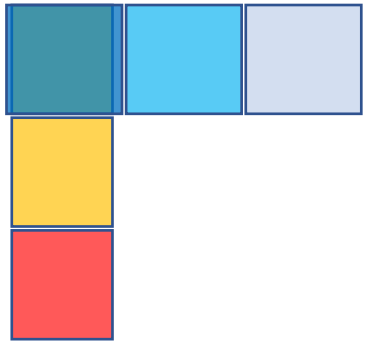


u_1v_1	u_1v_2	u_1v_3
u_2v_1	u_2v_2	u_2v_3

Separable filters

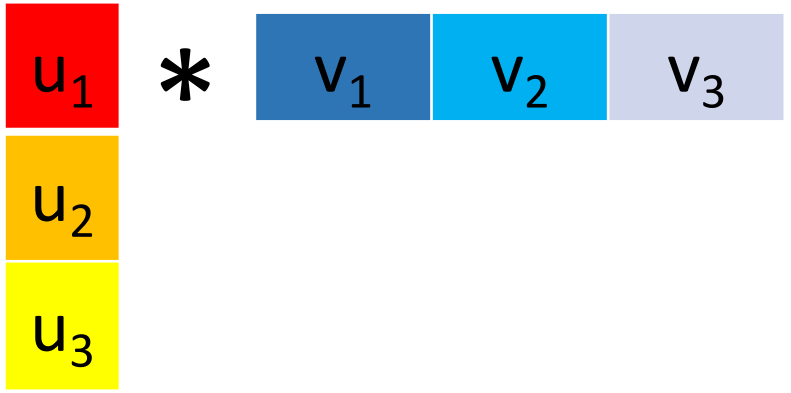


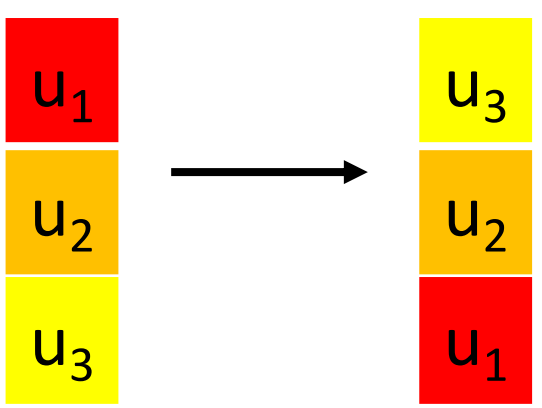


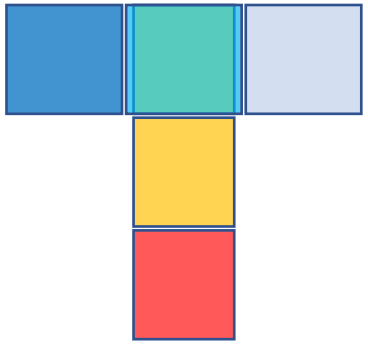


u_1v_1	u_1v_2	u_1v_3
u_2v_1	u_2v_2	u_2v_3
u_3v_1		

Separable filters

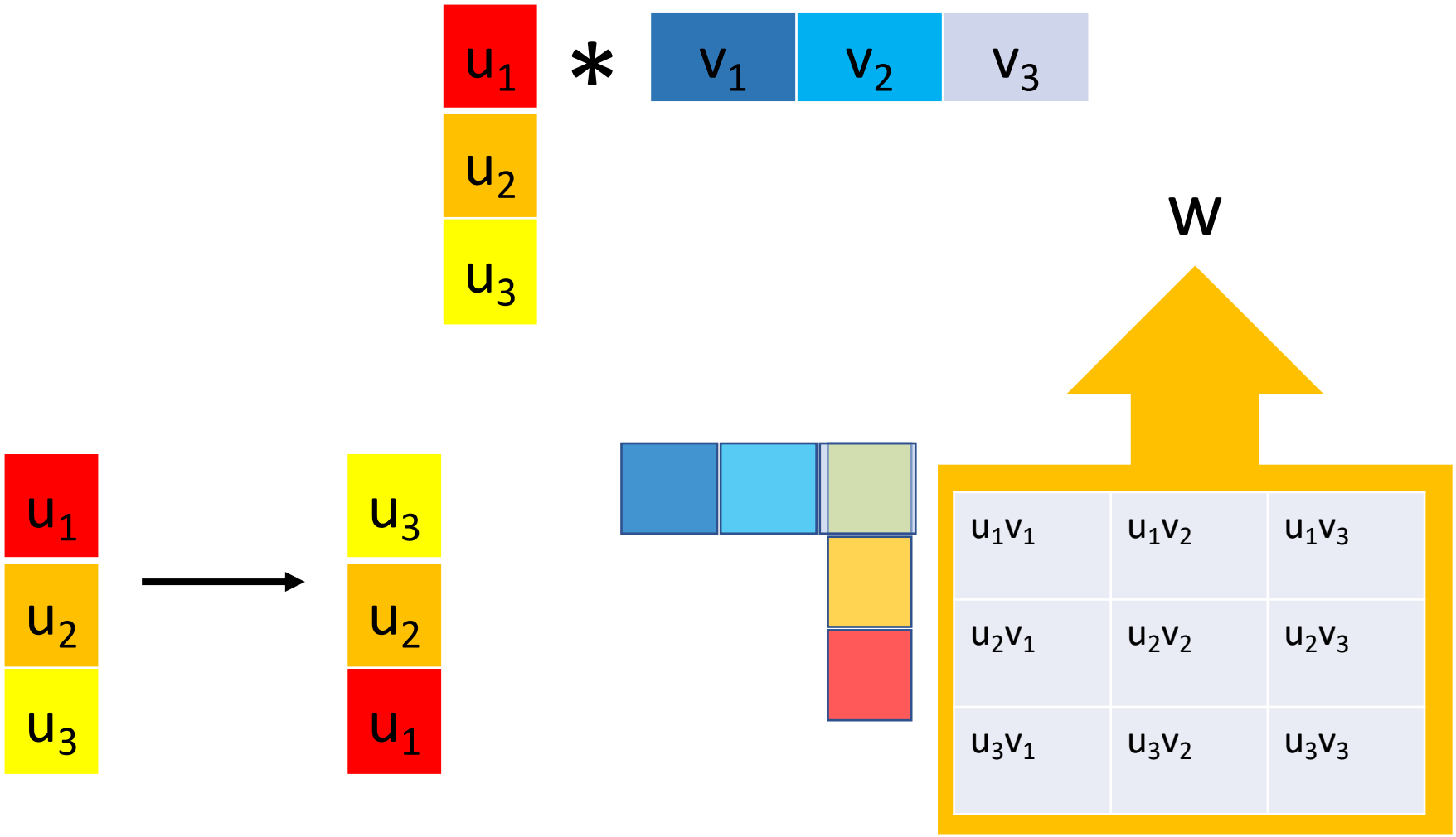






u_1v_1	u_1v_2	u_1v_3
u_2v_1	u_2v_2	u_2v_3
u_3v_1	u_3v_2	

Separable filters



Separable filters

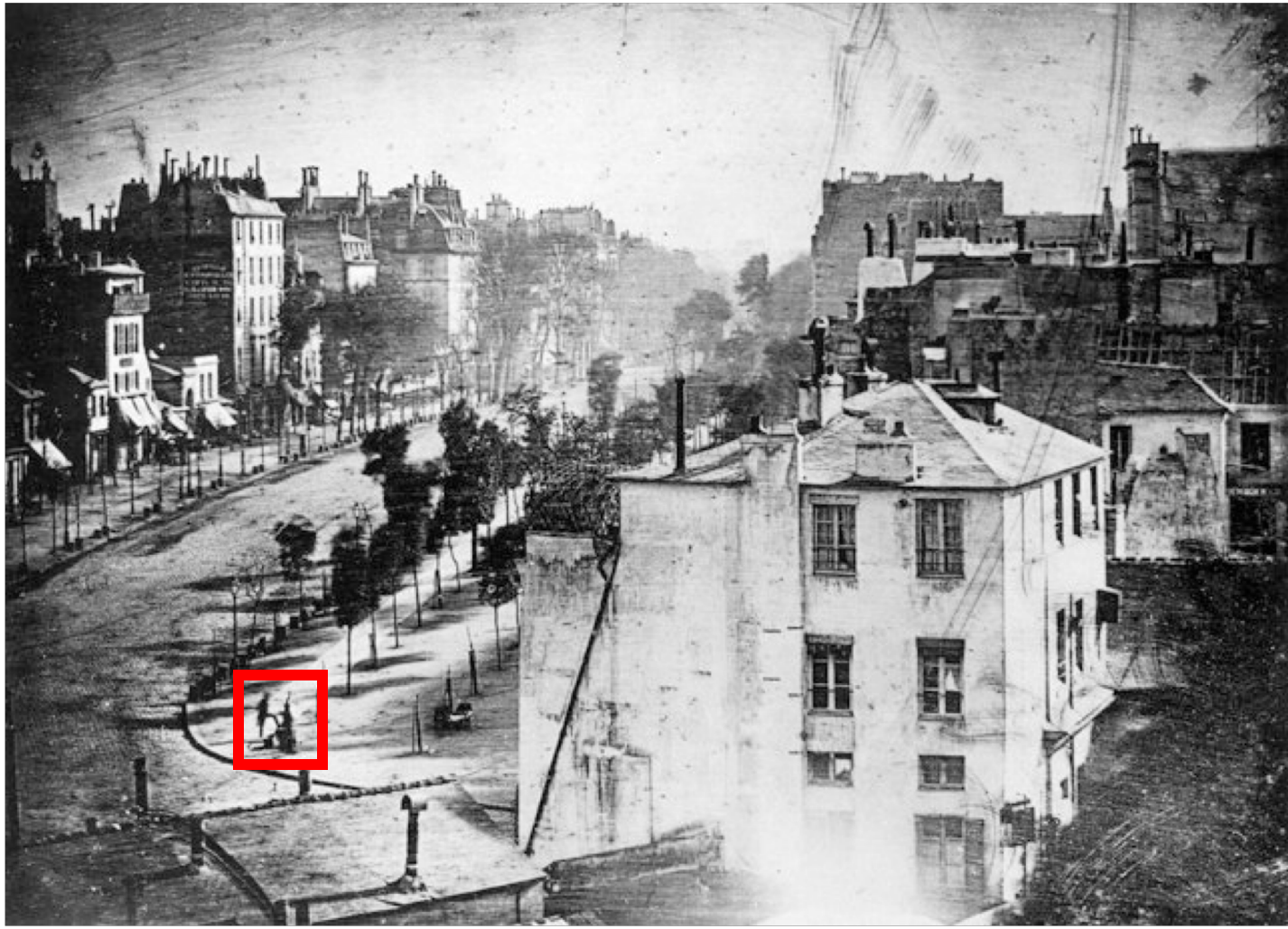
$$w * f = (u * v) * f$$
$$= u * (v * f)$$

- Time complexity of original : O(whk^2)
- Time complexity of separable version : O(whk)

Image resizing

Why do we need to talk about resizing?

- Need to zoom in to a region to get more details
- Can we get more details?



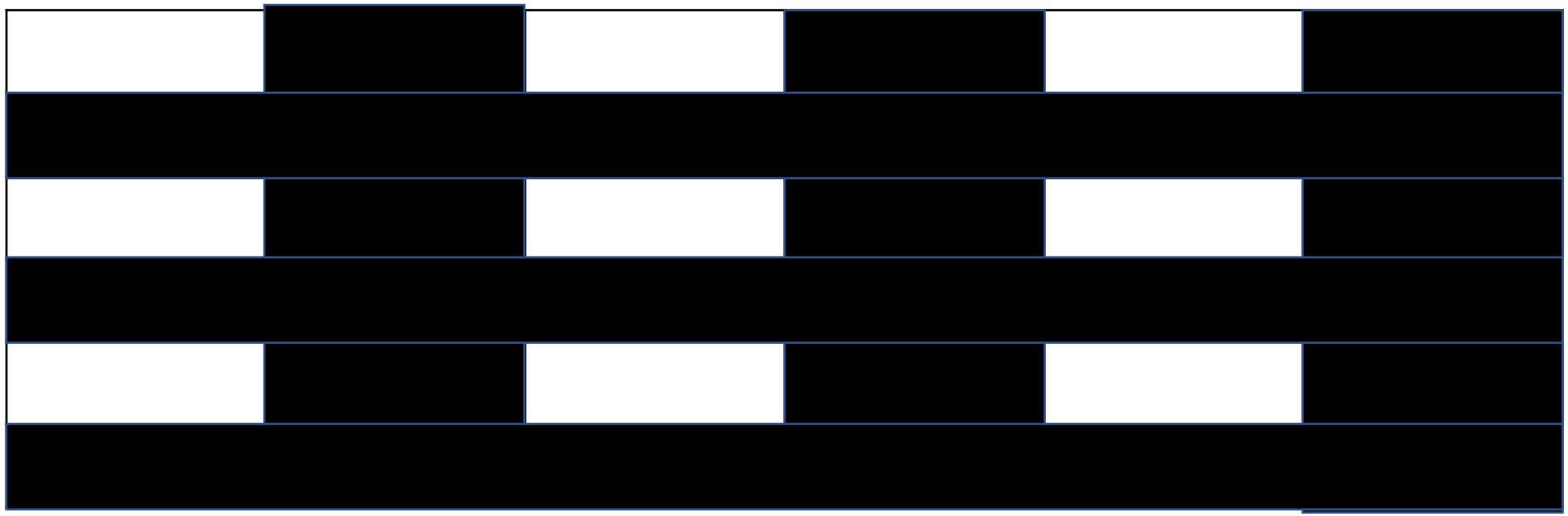
Louis Daguerre, 1838

Why do we need to talk about resizing?

- Far away objects appear small, nearby objects appear larger
- Need to recognize objects at multiple scales
- Resizing images to same size helps recognition

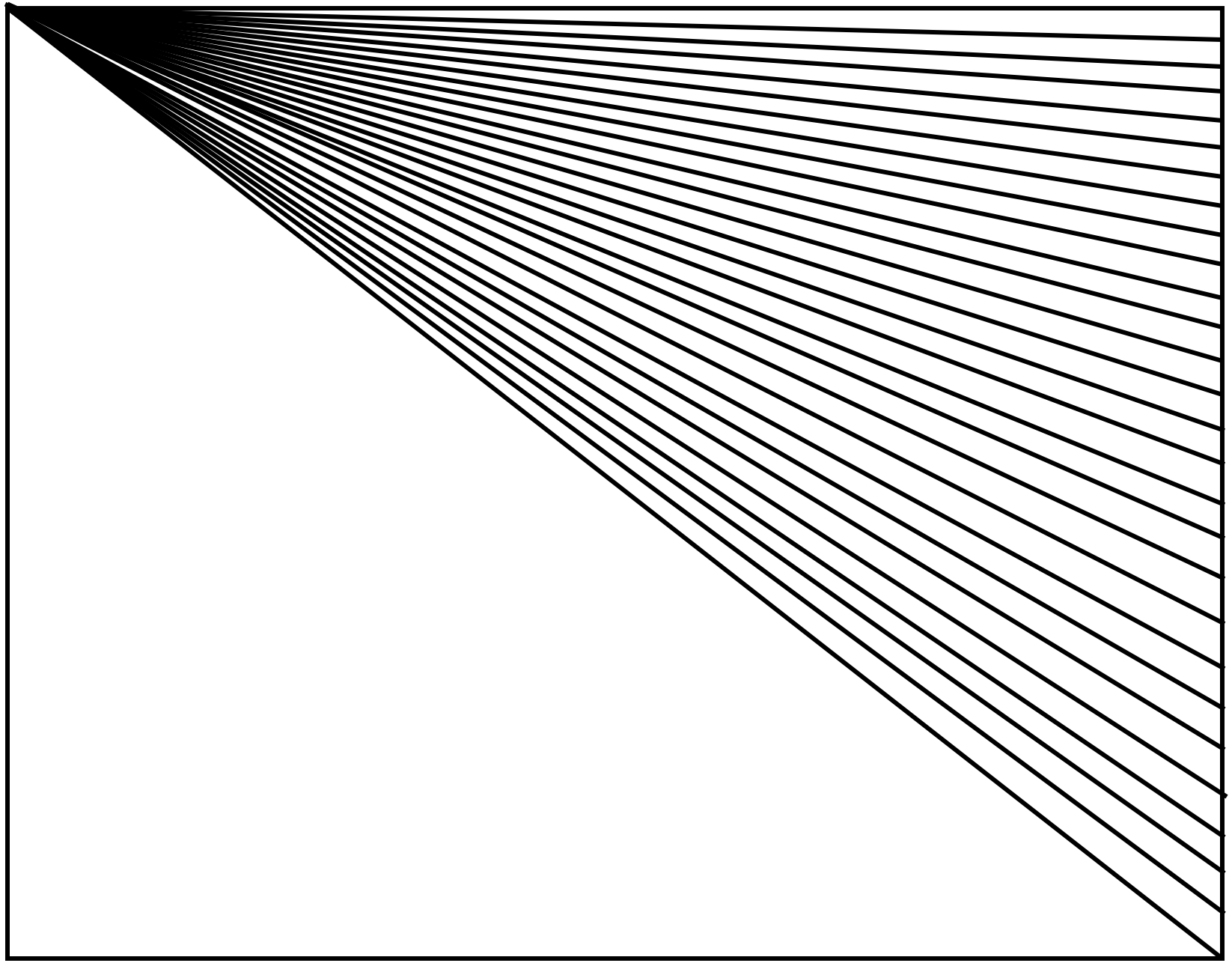
Why is resizing hard?

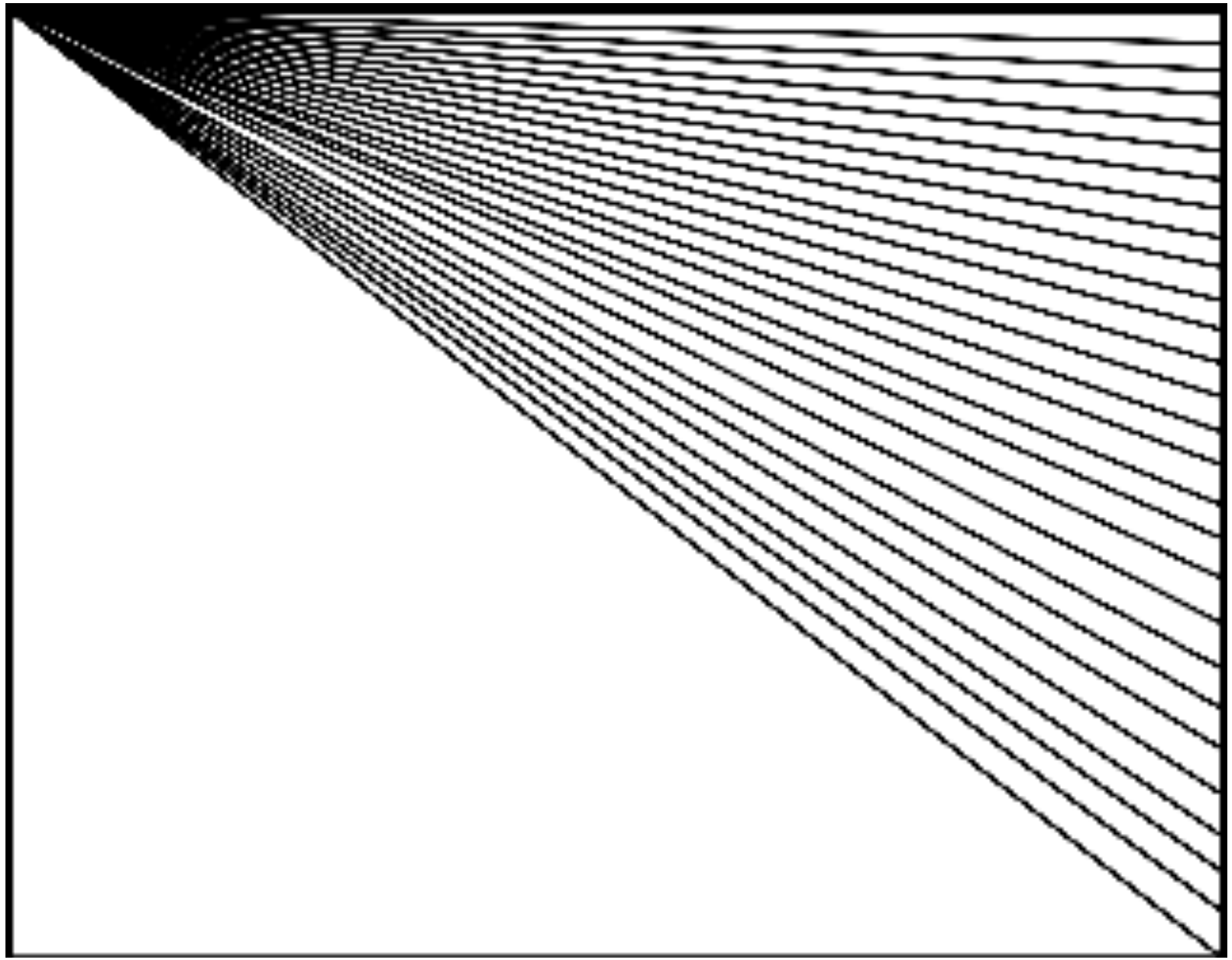
- E.g, consider reducing size by a factor of 2
- Simple solution: subsampling
- Example: subsampling by a factor of 2



Why is resizing hard?

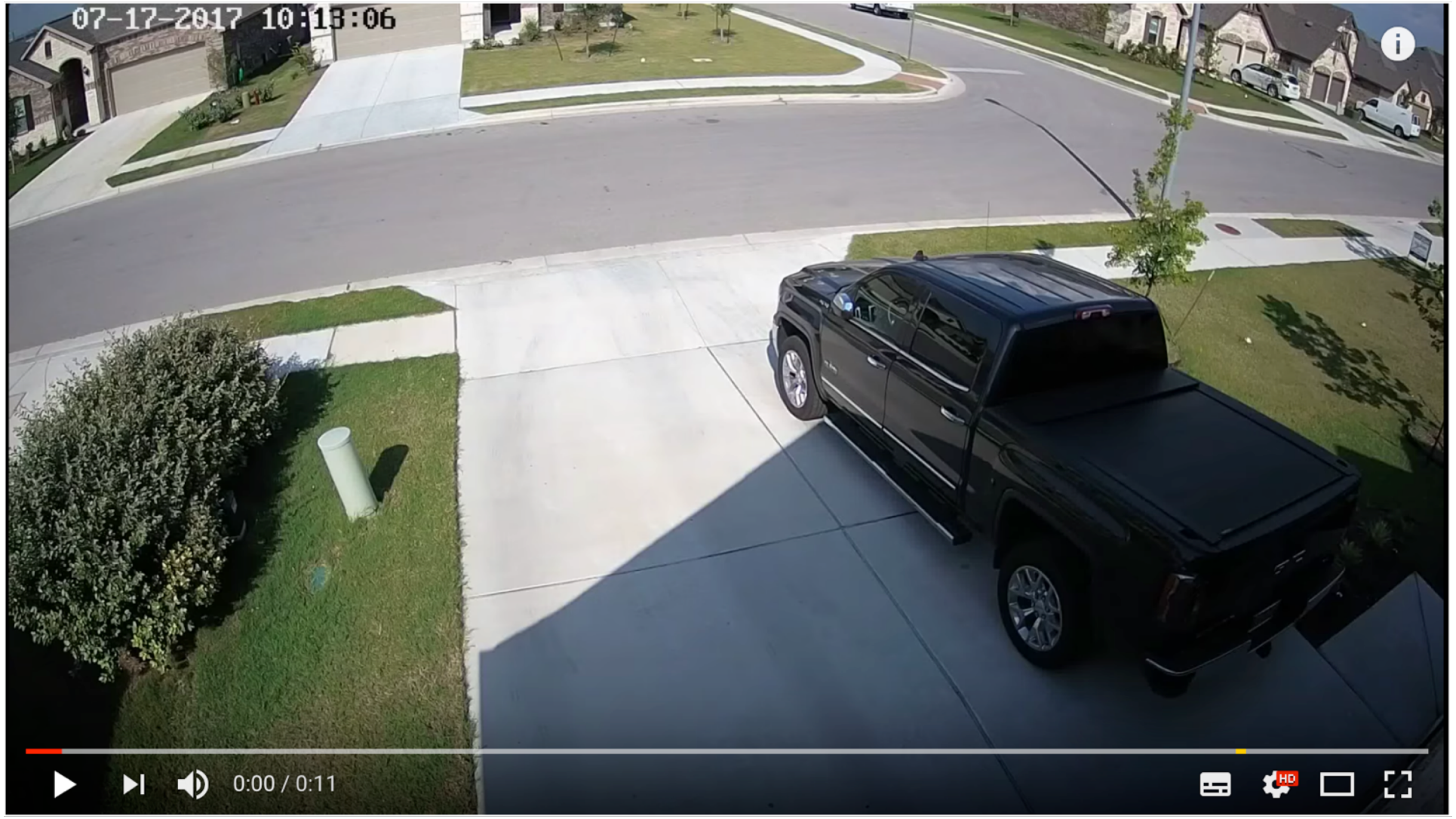
- Dropping pixels causes problems





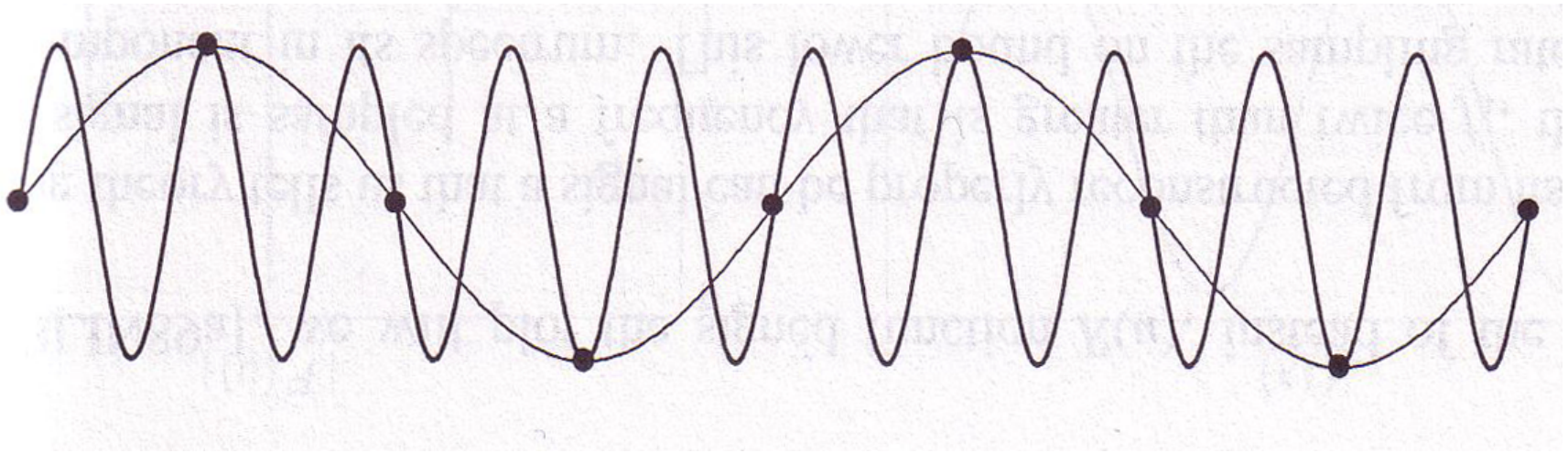
Aliasing in time

Aliasing in time



Why does aliasing happen?

- We ”miss” things between samples
- High frequency signals might appear as low frequency signals
- Called “aliasing”



Let's look at it mathematically

- Fourier basis signals
  - $B_k(n) = e^{\frac{i2\pi kn}{N}}$, time period N/k, frequency k/N
- Suppose we sample every P pixels. Then we are only looking at the values
  - $[B_k(P), B_k(2P), \dots, B_k(nP), \dots]$
- Consider two basis vectors B_k and $B_{k+N/P}$
- $B_{k+N/P}\,(nP) = e^{\frac{i2\pi\left(k+\frac{N}{P}\right)Pn}{N}} = e^{\frac{i2\pi kPn}{T}+i2\pi n(\frac{N}{P})(\frac{P}{N})} = e^{\frac{i2\pi kPn}{T}+i2\pi n}$

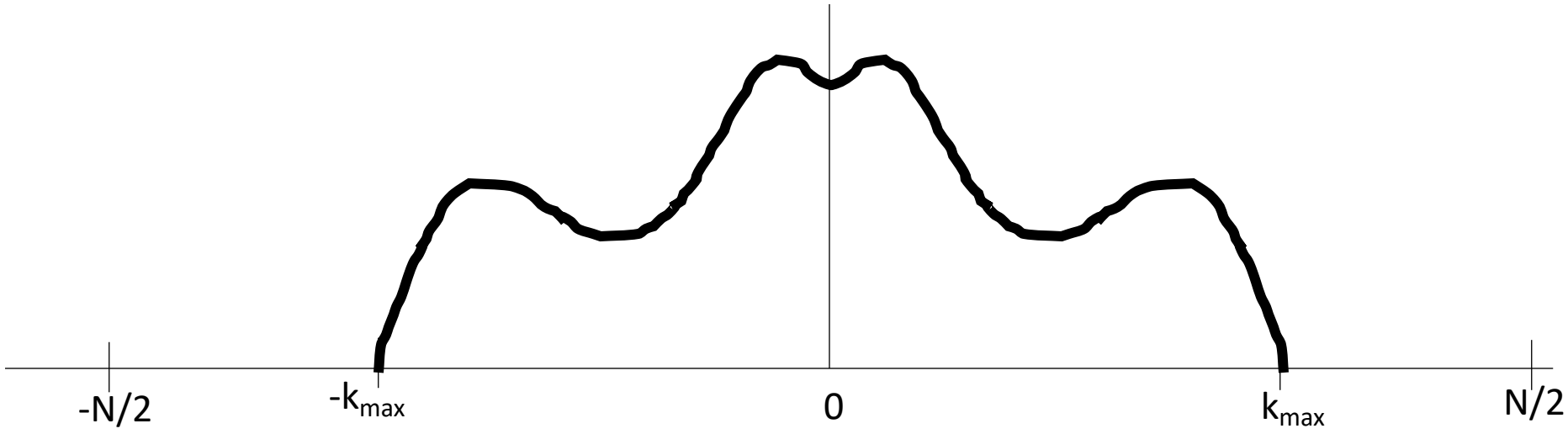
$$= e^{\frac{i2\pi kPn}{N}} = B_k(nP)$$

Let’s look at it mathematically

- $B_k(nP) = B_{k+\frac{N}{P}}(nP)$
- The two basis vectors look identical if we sample every P pixels
- Our eyes see the lower frequency: aliasing
- How do we avoid aliasing?
  - *Remove high frequencies that may be aliased*
  - *Sample at a high enough rate (lower P)*

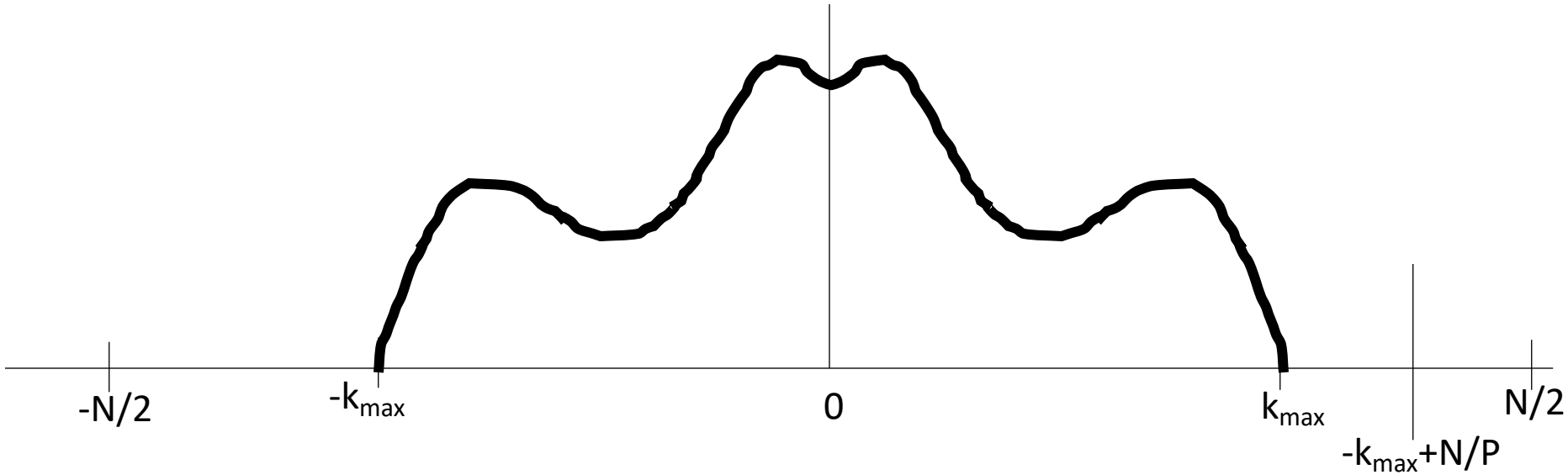
Avoiding aliasing

- Suppose image has limited high frequency components
- Fourier transform of x is X
- Suppose X is non-zero only for $-k_{max}$ to k_{max}



Avoiding aliasing

- When sampling once every P pixels, $-k_{max}$ is indistinguishable from $-k_{max} + N/P$
- If image has both coefficients, they will get mixed
- So we want that $k_{max} < -k_{max} + \frac{N}{P} \Rightarrow\ 2k_{max} < \frac{N}{P}$



Nyquist Sampling Theorem

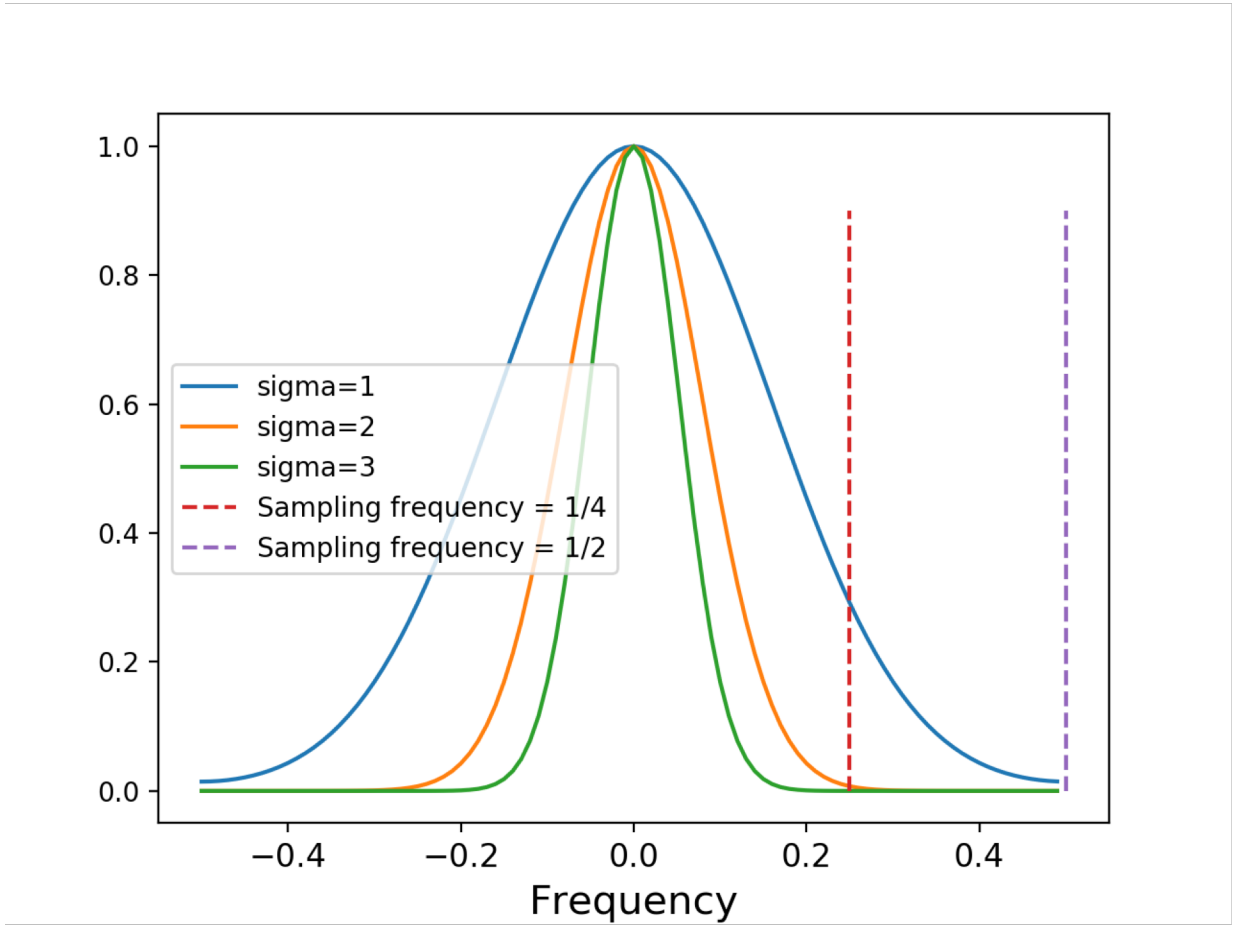
- $2k_{max} < \frac{N}{P}$
- Basis B_k has a time period of $\frac{N}{k}$ or frequency of $\frac{k}{N}$
-

$$2k_{max} < \frac{N}{P}$$
$$\Rightarrow \frac{2k_{max}}{N} < \frac{1}{\mathrm{P}}$$
$$\Rightarrow 2\nu_{max} < \nu_{sample}$$

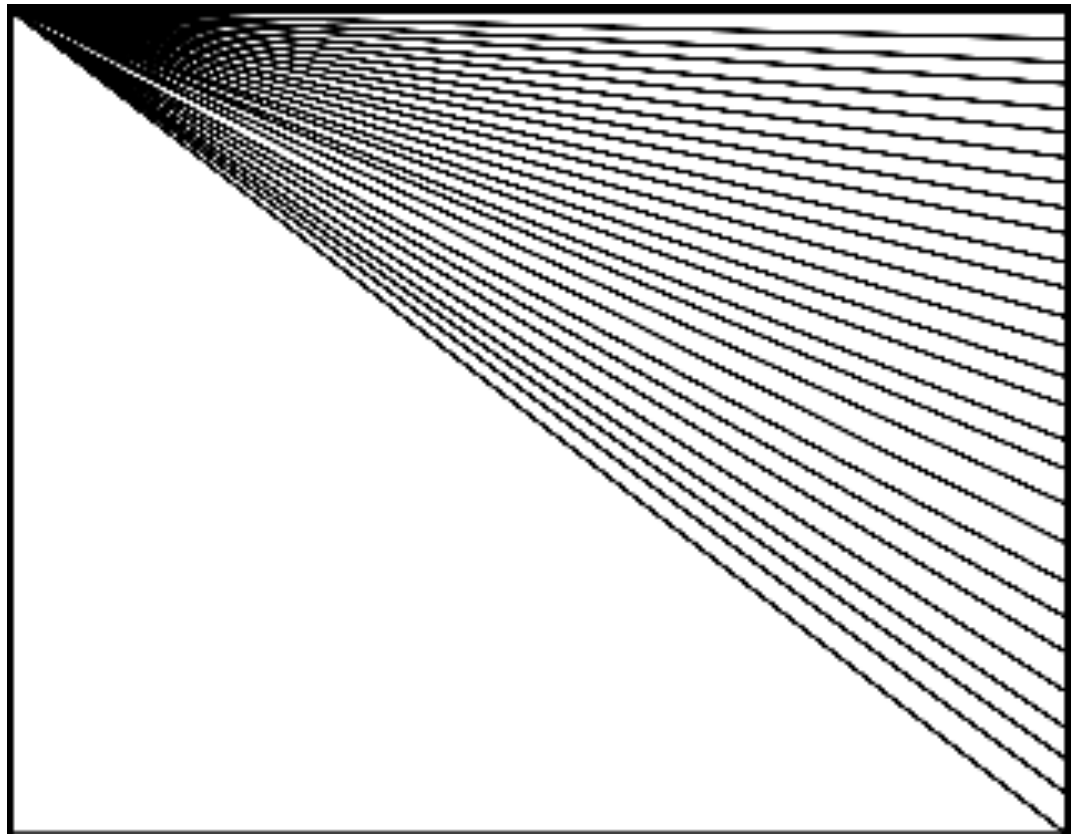
- **Need to sample at at least twice the rate of the highest frequency component**

Subsampling the image

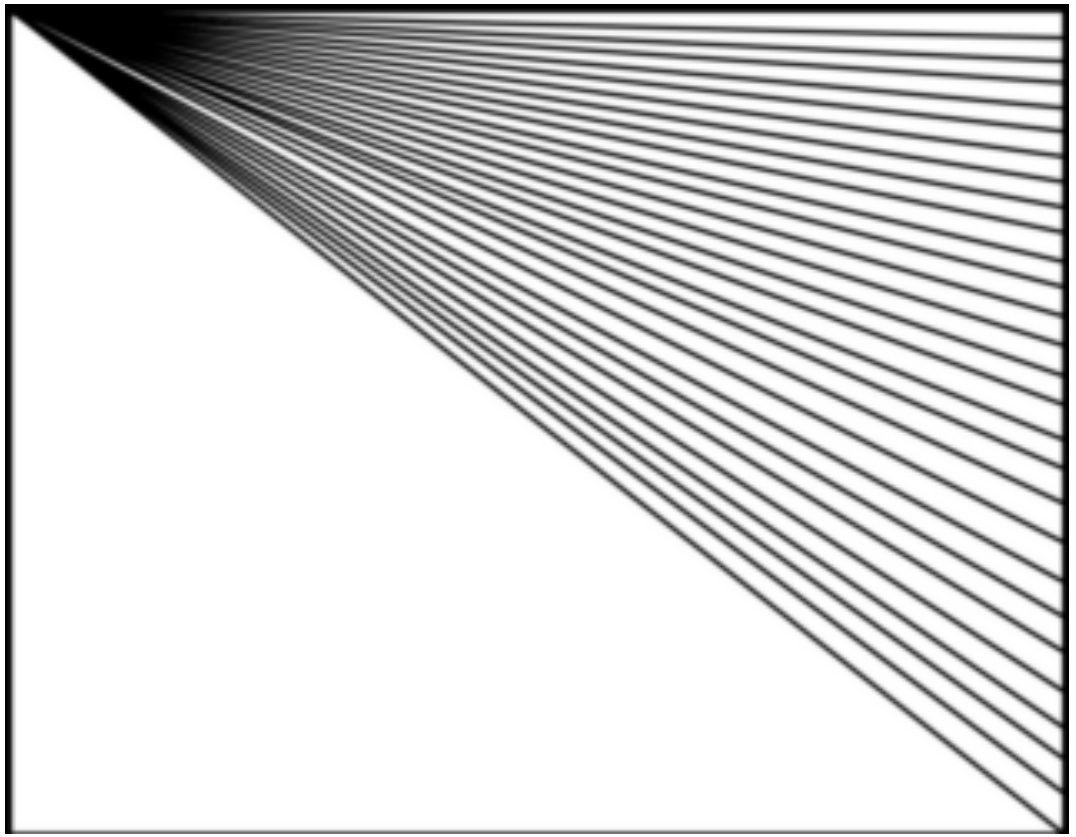
- First smooth the image to remove high frequency components
- How should we smooth?
- One simple answer: blur with Gaussian
  - Recall that Fourier transform of Gaussian is Gaussian
  - Choose sigma so that frequencies above $\frac{\nu_{sample}}{2}$ get killed



Subsampling before and after smoothing

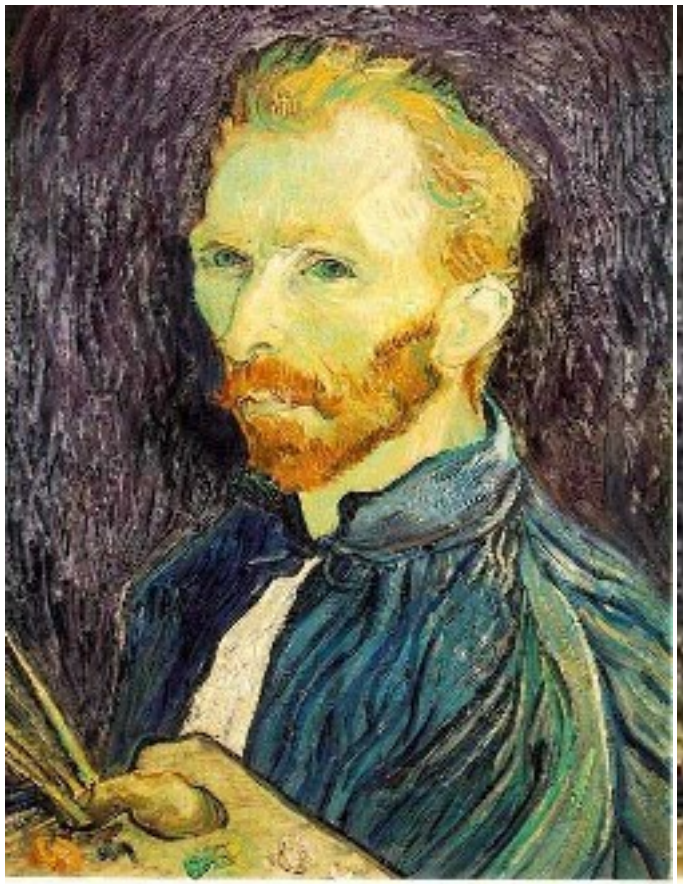


Before



After

Image sub-sampling



1/2

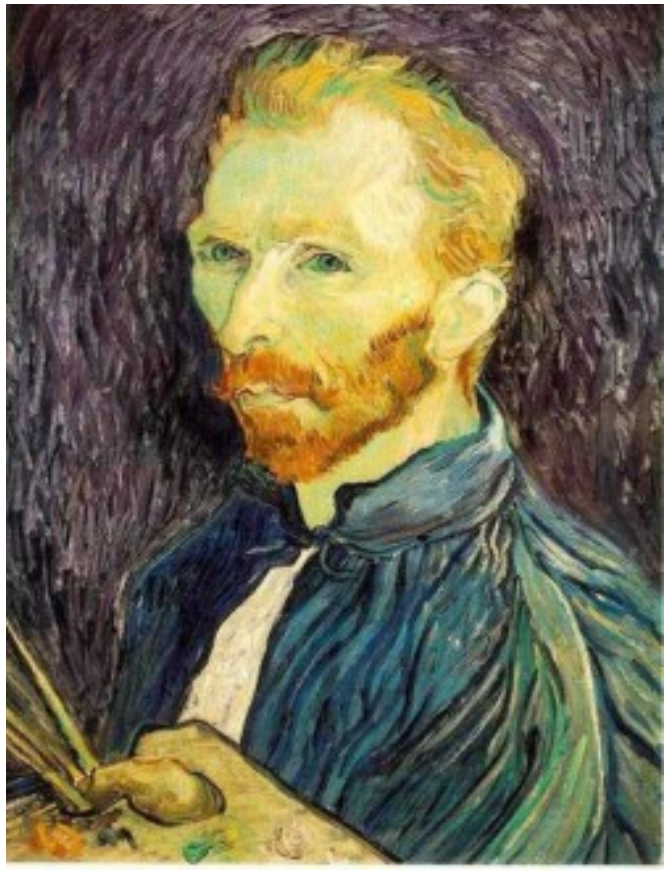
1/4 (2x zoom)

1/16 (4x zoom)

Why does this look so crufty? Aliasing!

Source: S. Seitz

Subsampling images correctly



Gaussian 1/2

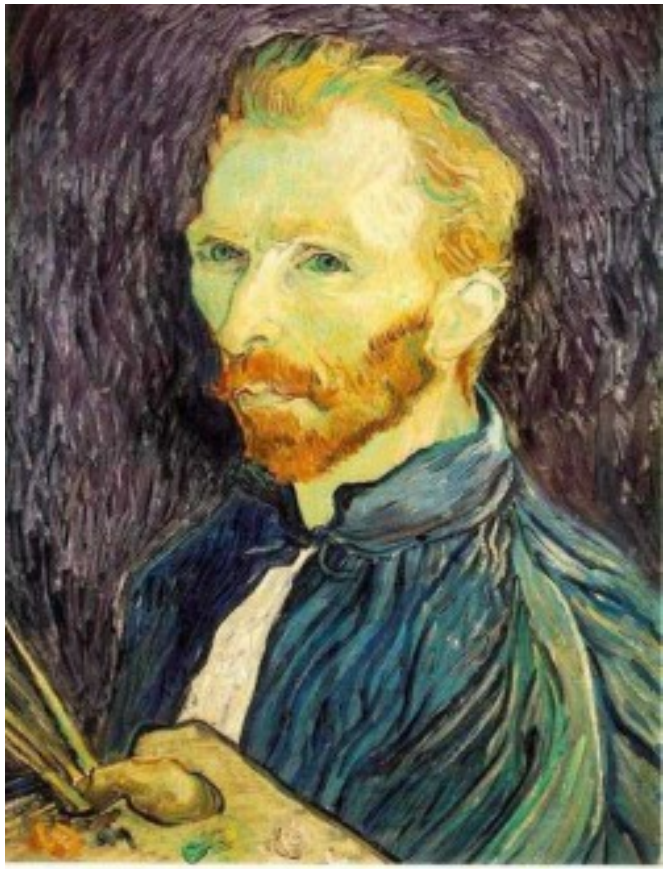
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Subsampling with Gaussian pre-filtering



Gaussian 1/2

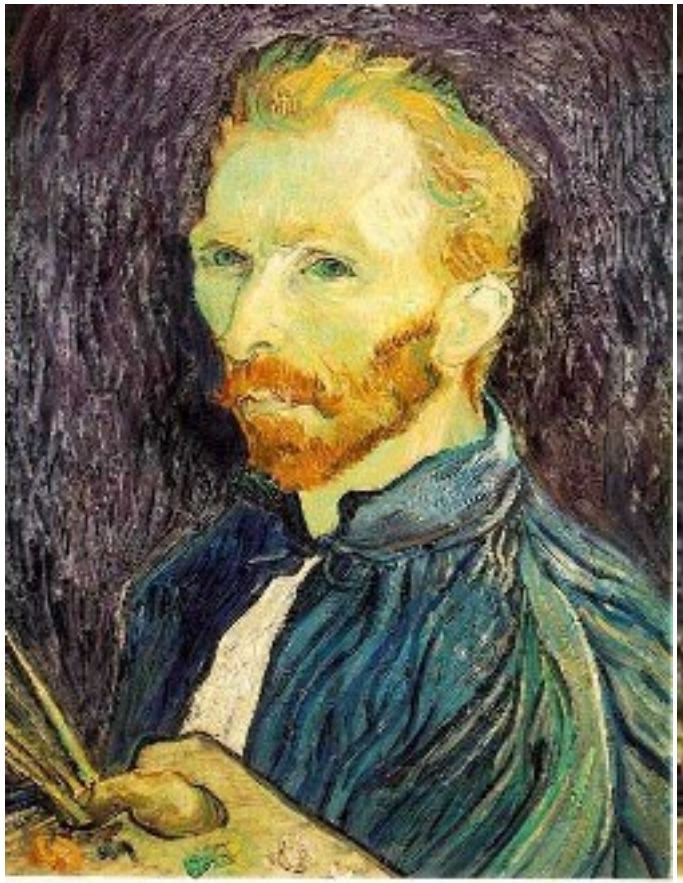
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Compare with...



1/2

1/4 (2x zoom)

1/8 (4x zoom)

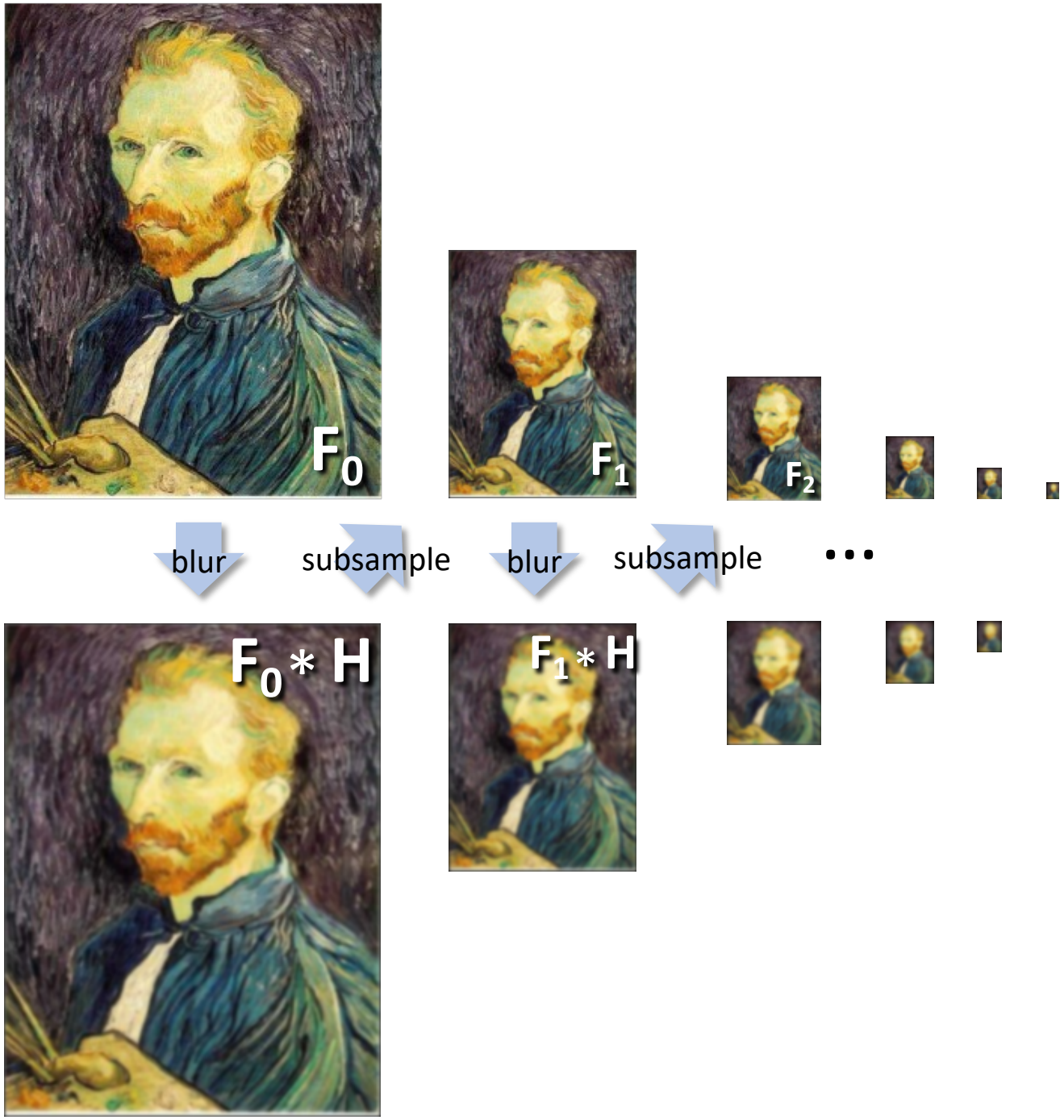
Source: S. Seitz

Low-pass filtering

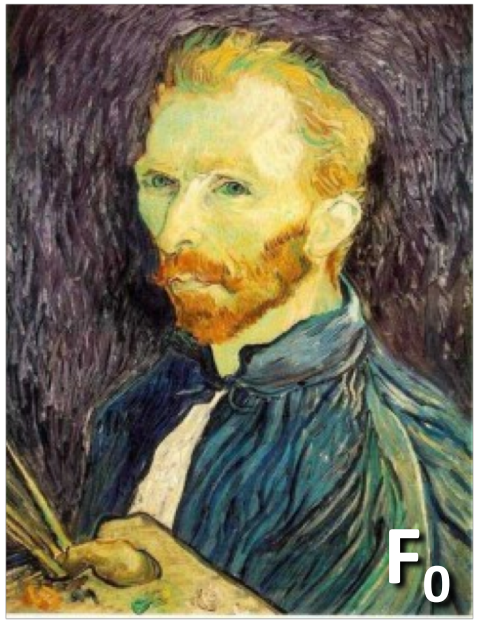
- Convolving with a Gaussian = remove high frequencies
- “Low-pass” filtering: low frequencies “pass” through filter, high frequencies don’t
- Identity – Low-pass filtered image = “High-pass filtering”

Gaussian pre-filtering

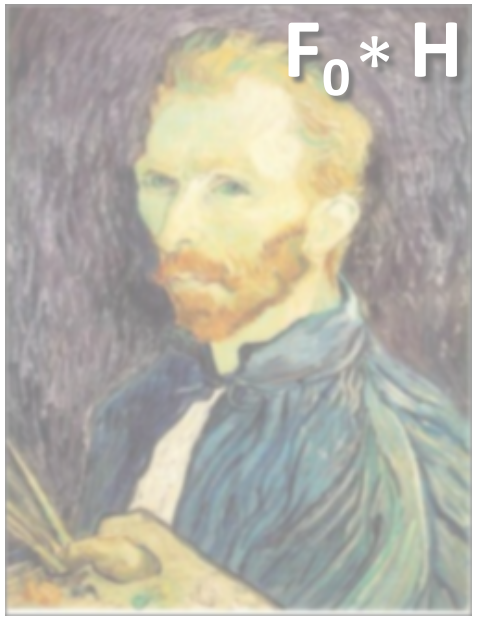
- Solution: filter the image, *then* subsample

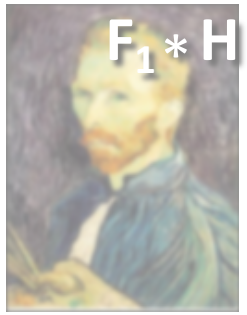


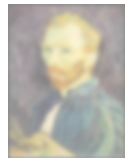
Gaussian pyramid



blur subsample blur subsample • • •

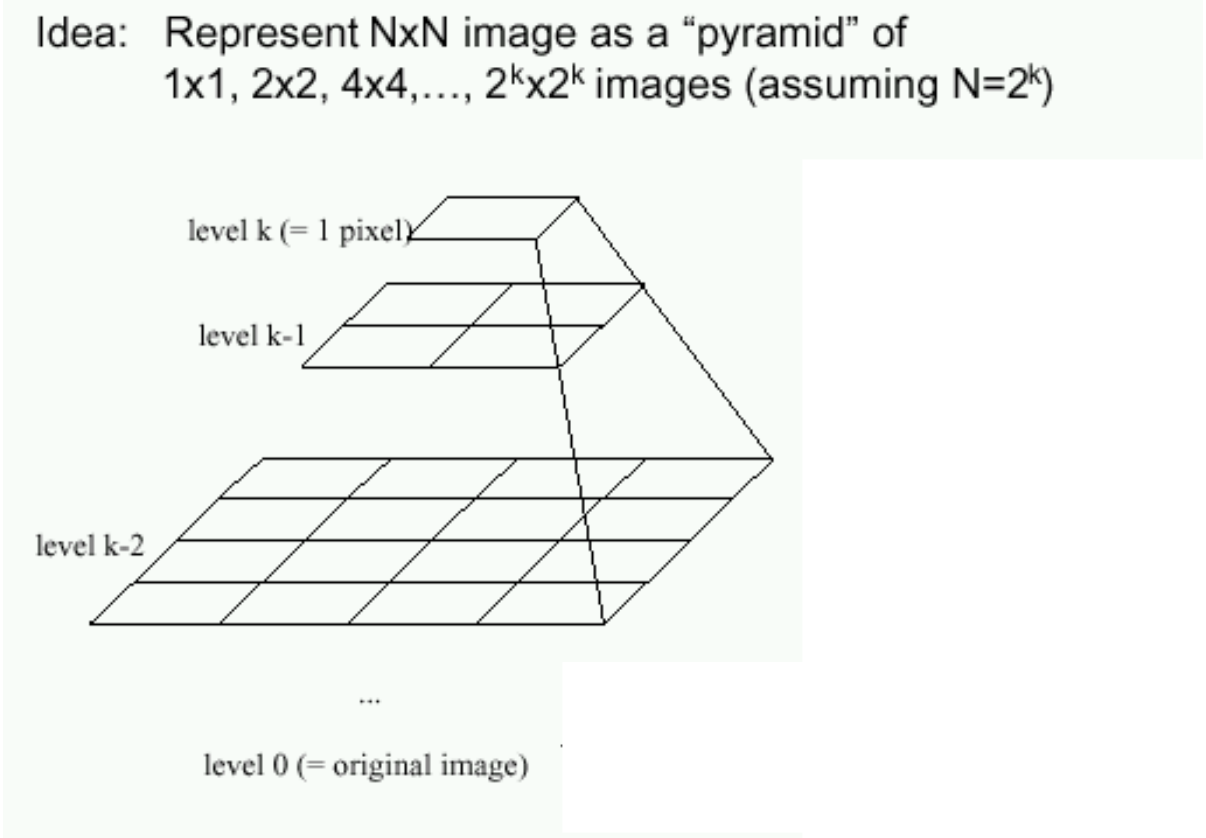






Gaussian pyramids [Burt and Adelson, 1983]

Idea: Represent NxN image as a "pyramid" of 1x1, 2x2, 4x4,…, 2^kx2^k images (assuming N=2^k)

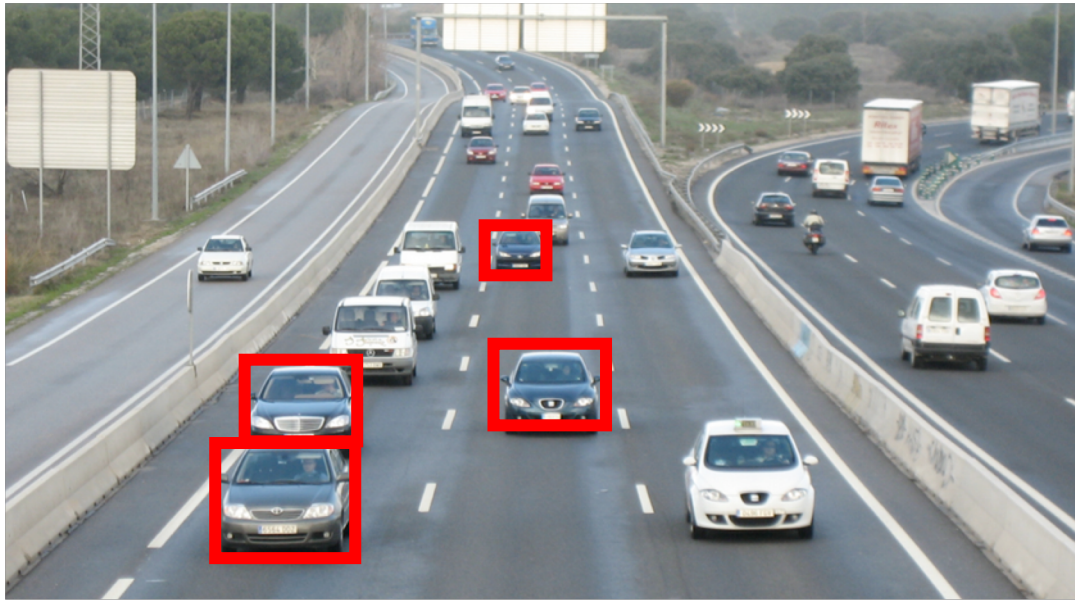


- In computer graphics, a *mip map* [Williams, 1983]

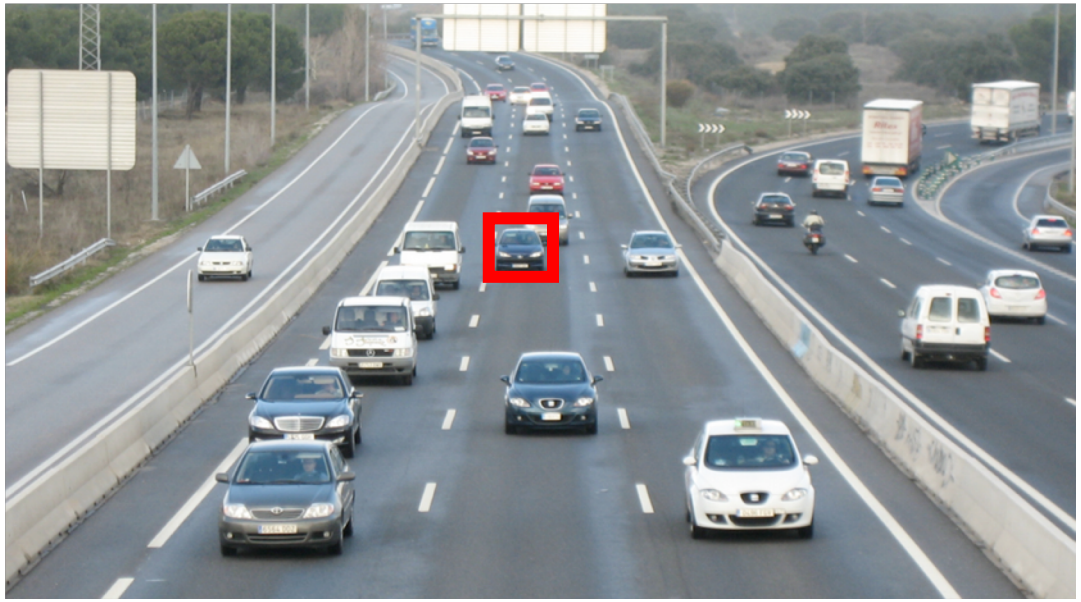
Gaussian Pyramids have all sorts of applications in computer vision

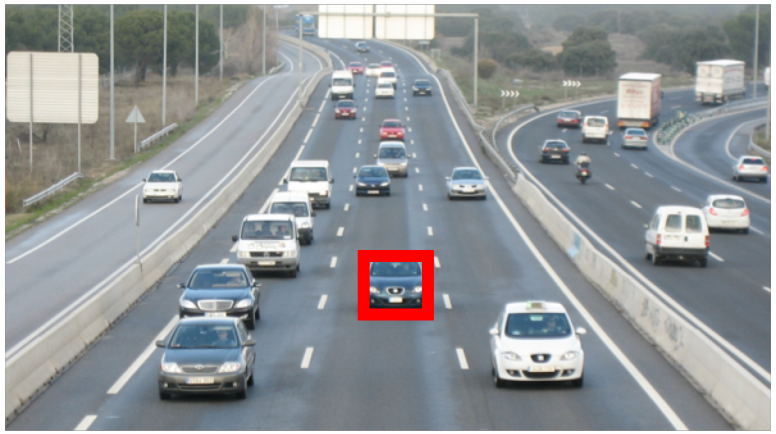
Source: S. Seitz

Gaussian pyramids - Searching over scales

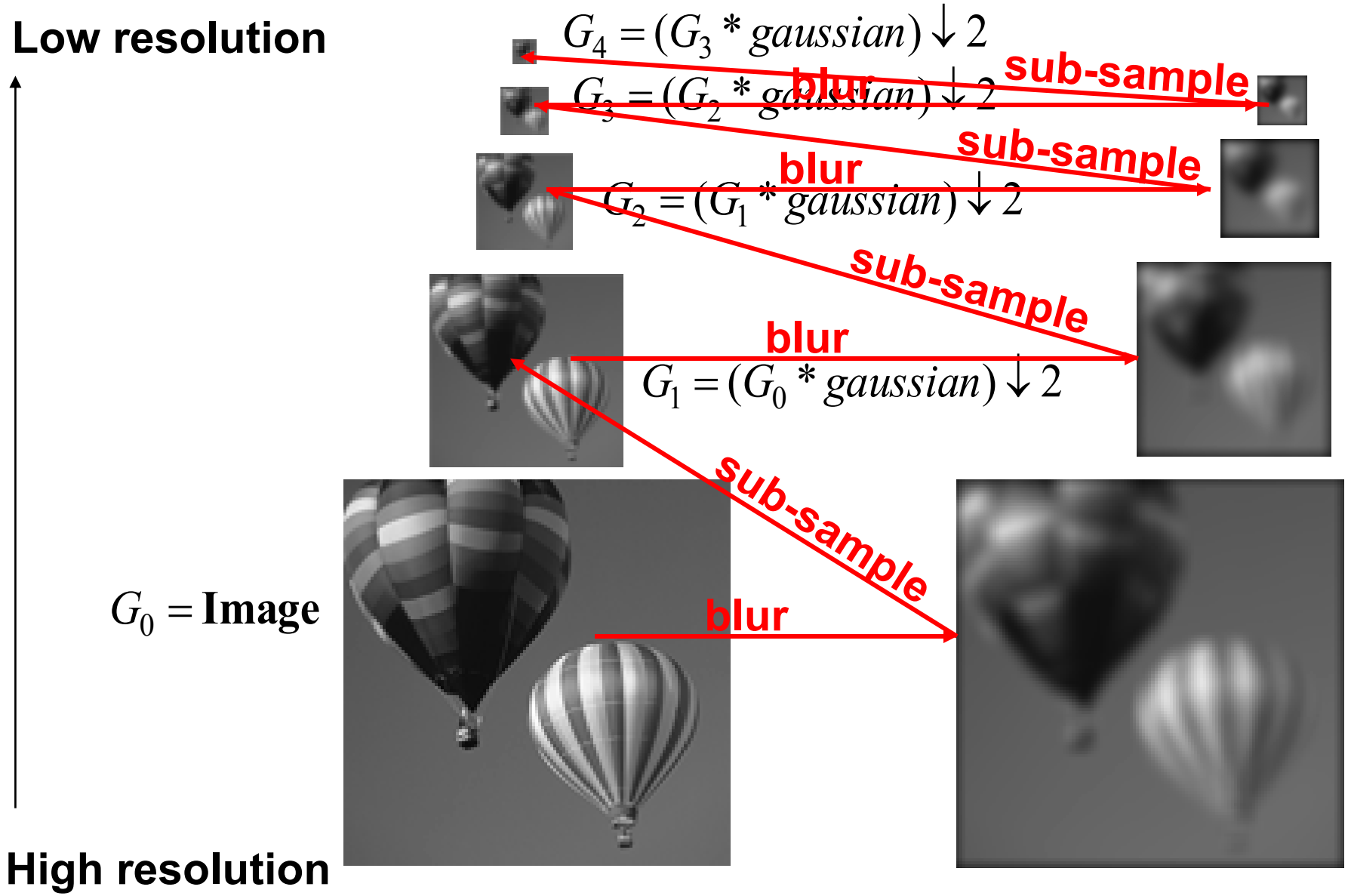


Gaussian pyramids - Searching over scales

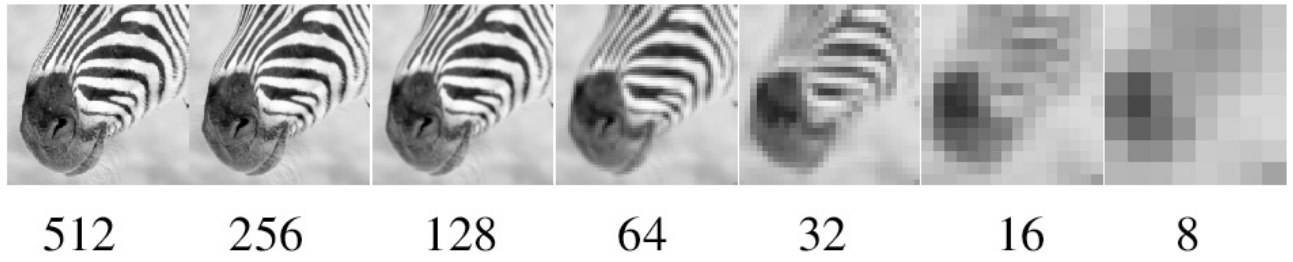




The Gaussian Pyramid



Gaussian pyramid and stack



Source: Forsyth

Memory Usage

- What is the size of the pyramid?