

Aliasing and Scale-invariance



Antonio Carta

2026-09-24

University of Pisa

antonio.carta@unipi.it

We saw how many filters depend on a scale parameter σ .

- **Scale-space**: a principled, continuous family of representations indexed by scale
- **Image pyramids**: discrete, subsampled multiscale representations (efficient)
- **Applications**: coarse-to-fine search, template matching, image blending

multiscale representations are everywhere in vision, from SIFT to DNNs

Aliasing

- images are signals in the continuous domain, but we work with discrete measurements
- discrete measurements result in a limited **frequency resolution**
- we need to be aware of what information we lost
- in most CV pipelines we continuously subsample images, so we need to understand sampling limitations

Questions:

- given a sampling frequency (i.e. how many measurements per second or meter), what is the maximum signal frequency (i.e. the Fourier decomposition) that we can represent without errors?
- what happens when we upsample or downsample an image?

Desiderata:

- **maximize information**, which requires as many samples as possible
- **minimize compute and memory cost**, which are often at least linear in image size

The sampling frequency controls this tradeoff.

- we can also downsample to reduce computational cost
- we cannot upsample to recover lost information
- low resolution doesn't only lose information, it can also add artifacts

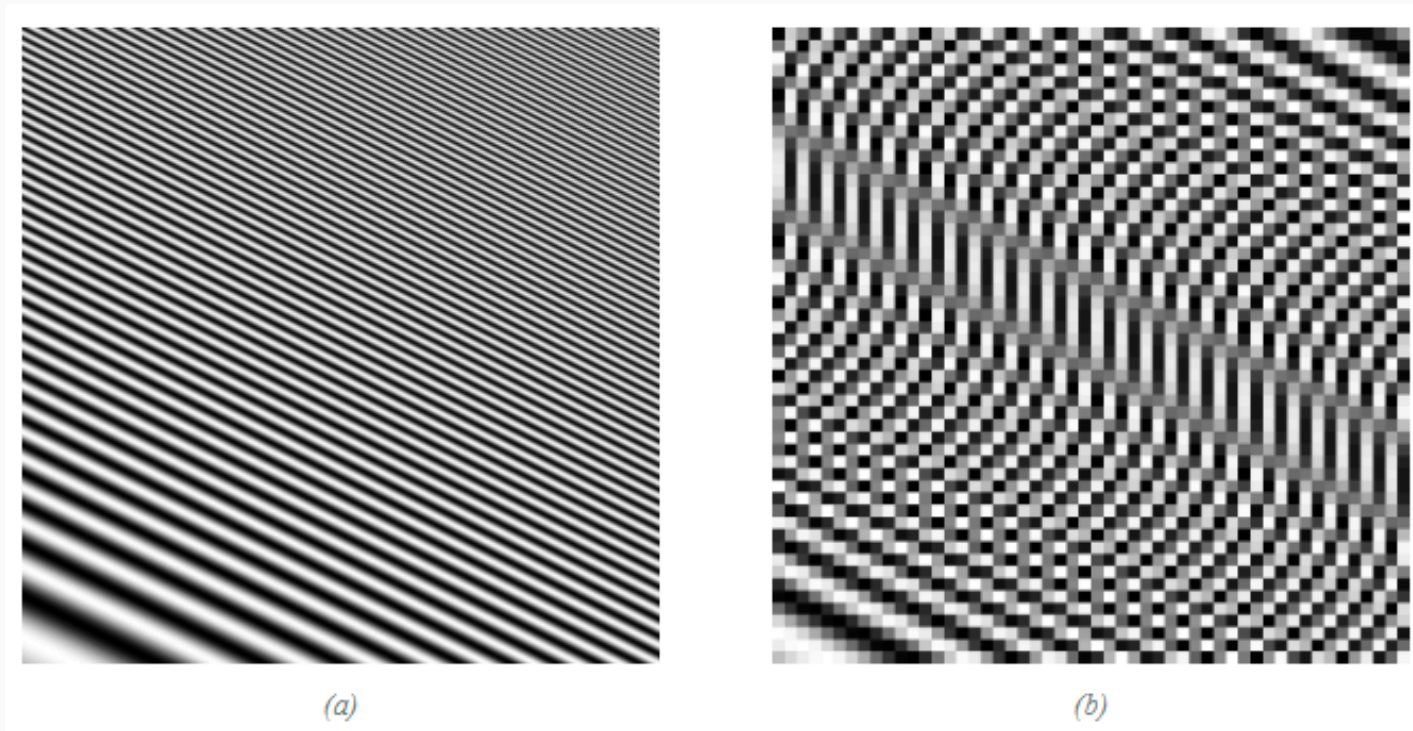


Figure 1: (left) high-resolution image. (right) its sub-sampled versions. Source: visionbook

Aliasing Example

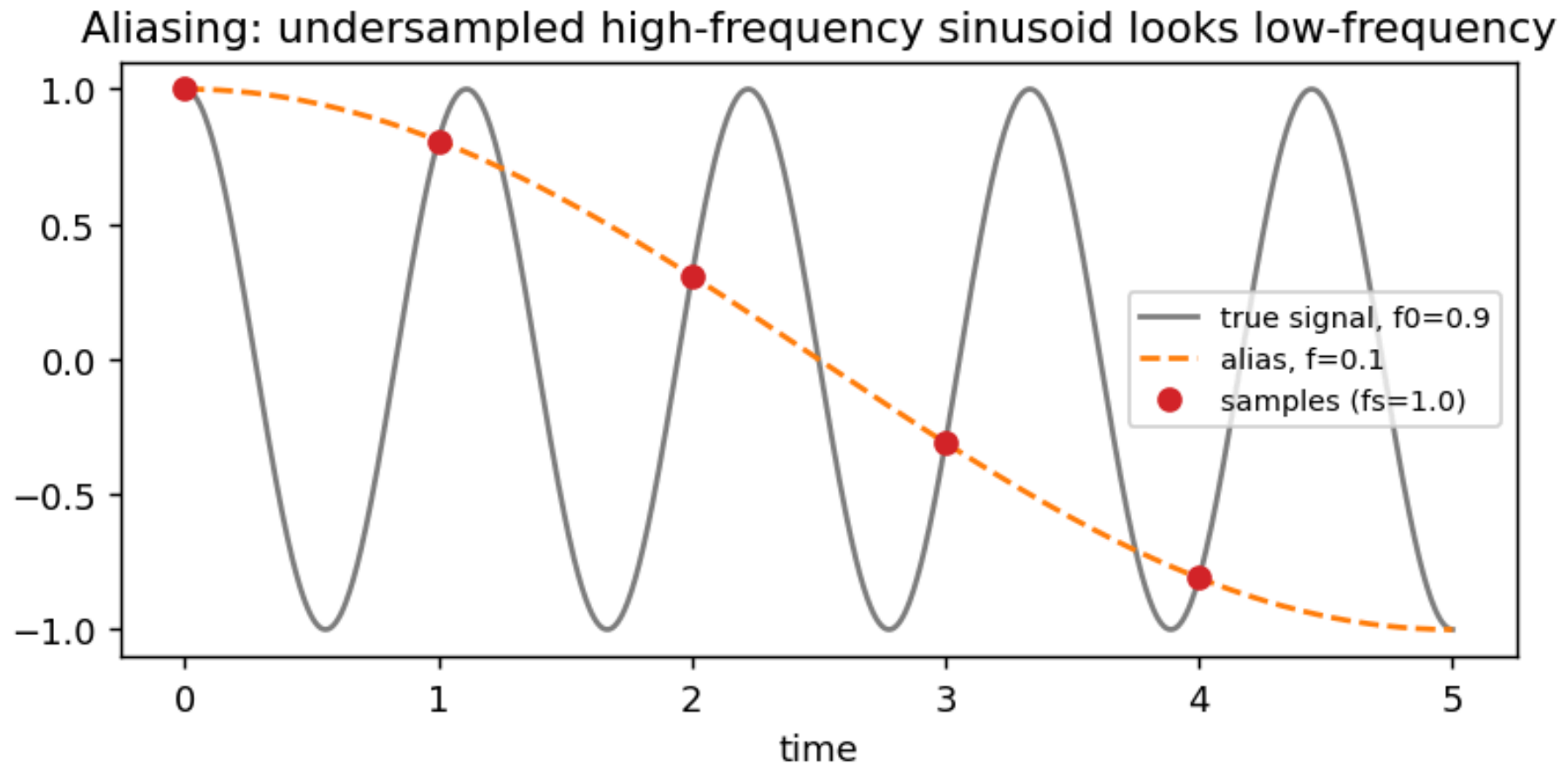


Figure 2: A high-frequency sinusoid ($f_0 = 0.9 f_s$) sampled at rate f_s looks identical to a low-frequency sinusoid ($f = 0.1 f_s$)

Consider $f(x) = \cos(2\pi f_0 x)$ sampled at rate f_s :

- If $f_s > 2f_0$: the samples uniquely identify the sinusoid
- If $f_s < 2f_0$: the samples are **indistinguishable** from a lower-frequency sinusoid at $|f_0 - kf_s|$ for some integer k
 - **aliasing** is the problem of higher frequency content being incorrectly measured as a lower frequency due to a sampling frequency that is too small

Slow-and-smooth prior: notice that we implicitly assume that the slowest frequency is the correct one.

- it's how the mathematical tools will work
- it's consistent with our perceptual bias

- **Aliasing** = high-frequency content masquerading as low-frequency content after sampling, due to spectral replica overlap
- Occurs whenever sampling rate $< 2 \times$ (highest frequency present in the signal)
- Once aliased, the original high-frequency information is **unrecoverable**, it's mixed into the low frequencies, not just lost

Sampling Theorem

The Sampling Theorem (Nyquist–Shannon)

Sampling Theorem:

- let $l(t)$ be a band-limited signal with maximum frequency ω_{max}
- given a sampling frequency $\omega_s = 2\frac{\pi}{T_s}$, where T_s is the sampling period

Then, $l(t)$ can be perfectly reconstructed if $\omega_s > 2\omega_{max}$ (the **Nyquist condition**)

In other words: **the sampling frequency needs to be more than twice the maximum frequency of the signal**

- **Nyquist frequency:** $\omega_f = \frac{f_s}{2}$, that is the maximum frequency correctly reconstructed given a sampling frequency f_s

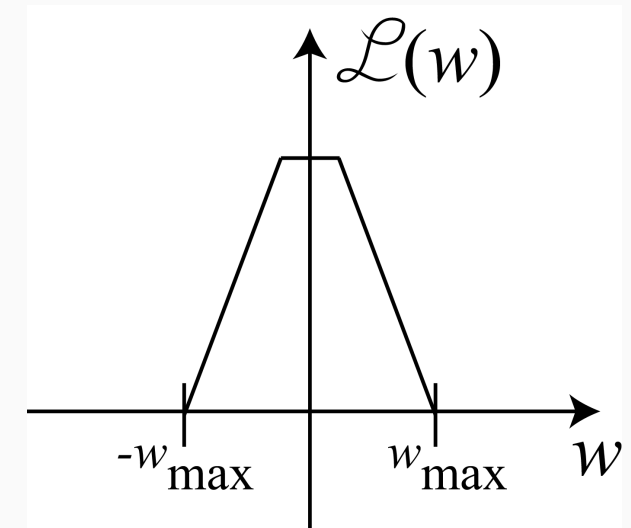


Figure 3: source: visionbook

Aliasing and sampling theorem

- Aliasing is a consequence of the sampling theorem
- notice that, when the Nyquist condition is not satisfied, we do not just lose information. Instead, we get aliasing

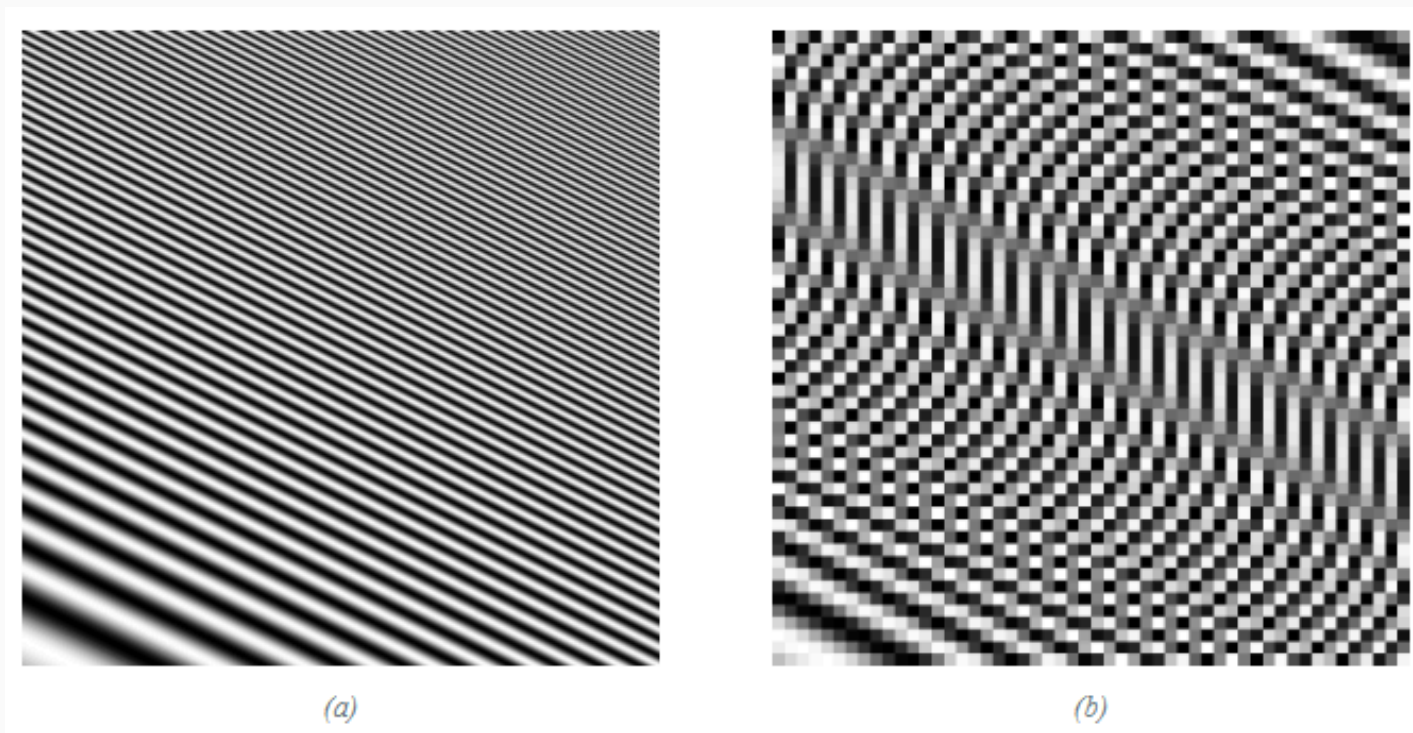


Figure 4: (left) high-resolution image. (right) its sub-sampled versions. Source: visionbook

- **Spatial - Moiré patterns**: overlapping fine grids/textures (e.g., striped clothing on camera, screen photographed by another screen)
- **Temporal - Wagon-wheel effect**: wheel/fan appears to rotate at a different (lower) frequency than its real one due to temporal sampling (frame rate) vs. rotation frequency
 - examples: https://en.wikipedia.org/wiki/Wagon-wheel_effect

- **Jagged edges (jaggies)**: high-frequency content of sharp diagonal edges aliases into spurious low-frequency stair-step patterns
- **Texture aliasing**: fine repeating textures (bricks, fabric) can produce false low-frequency patterns when downsampled without filtering
- Naive image resizing results in aliasing

- **Sampling**: discretization in space/time (e.g., 10 measurements per second, where we measure)
- **Quantization**: discretization in amplitude (e.g., 8-bit pixel values, how precisely we represent each measurement)
- Aliasing arises from sampling, not quantization

- Strided convolutions and max/average pooling are **downsampling operations**
- Downsampling can introduce artifacts in the latent representations
- **Consequence:** small input shifts can cause large changes in feature maps (loss of shift-invariance/equivariance)

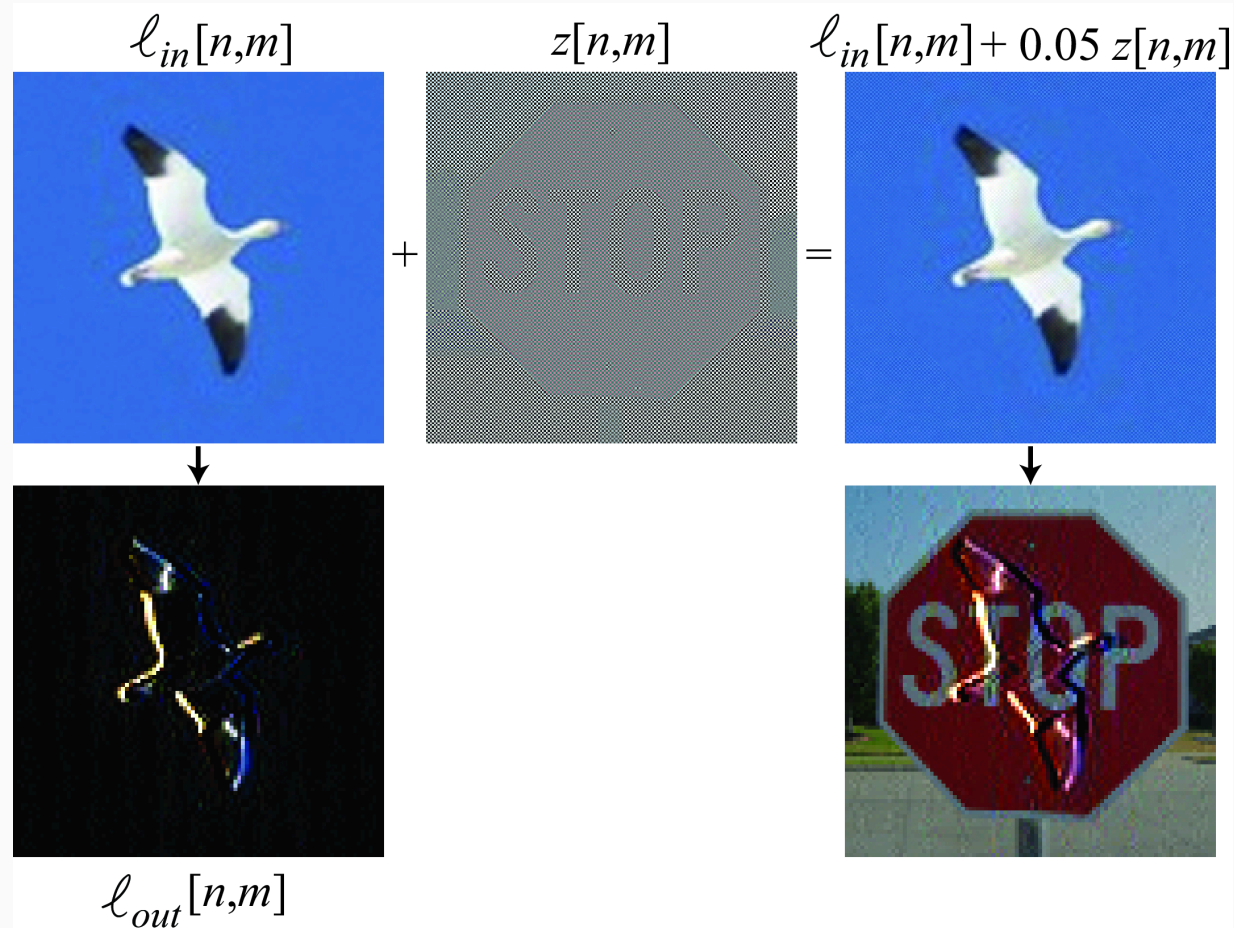


Figure 5: Bottom row: output of image derivative -> downsampling -> ReLU. Source: visionbook

- The sampling frequency limits the frequency that we can represent
- If the signal has higher frequency content, we get aliasing
- Aliasing affects most computer vision pipelines because resizing images is a common operation

Downsampling

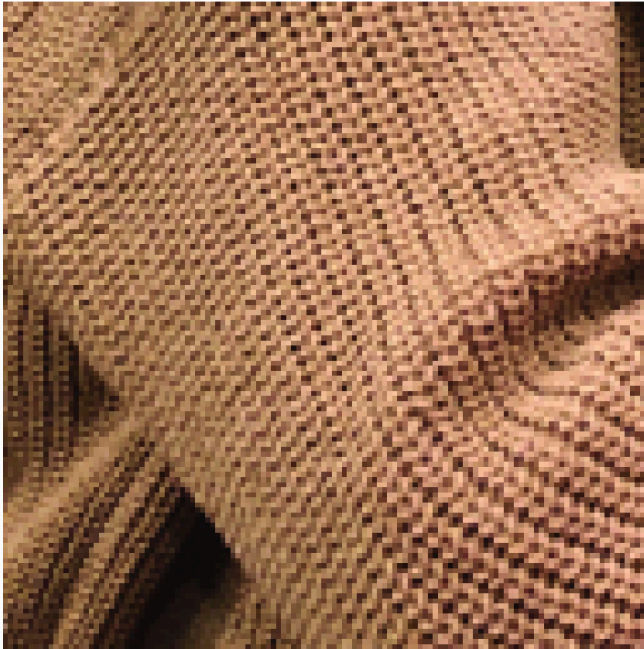
- we need to study how to do downsampling to minimize the amount of lost information

Decimation: downsample an image by dropping pixels

$$l_{\downarrow k}[n, m] = l[kn, km]$$

- this naive approach introduces aliasing

128x128



64x64



32x32



Figure 6: Source: visionbook

2D Delta Train

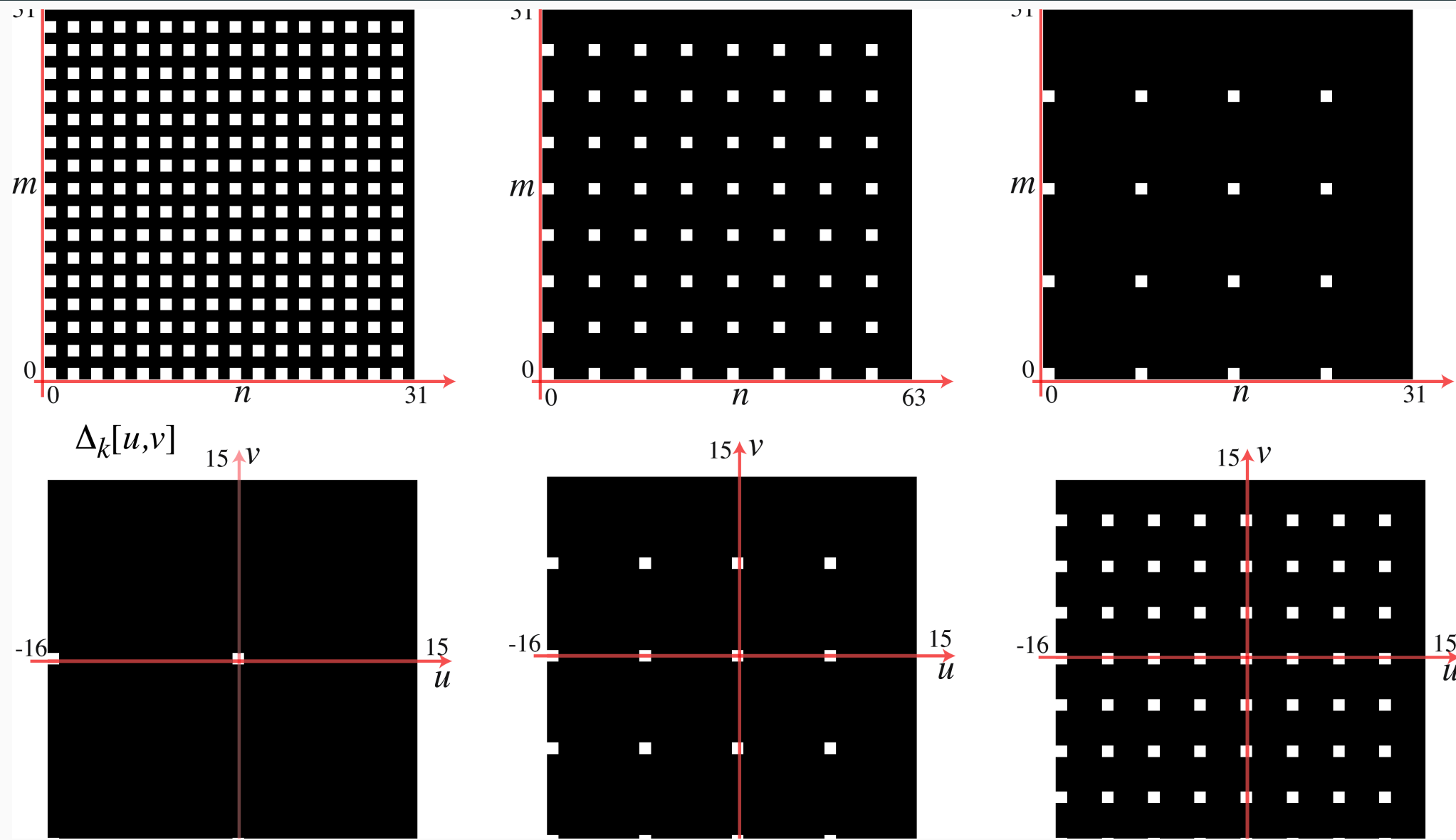


Figure 7: Source: visionbook

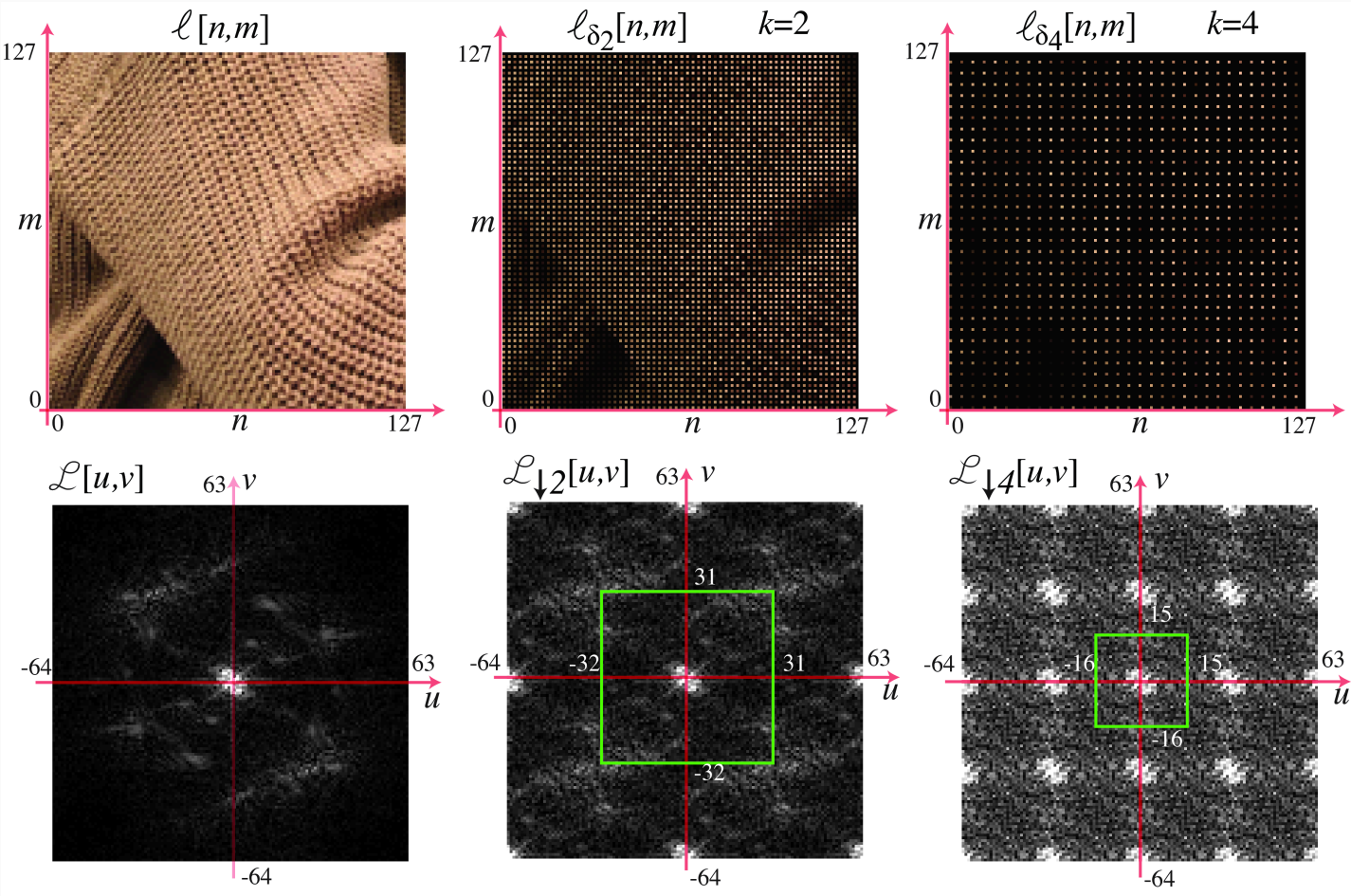


Figure 8: Source: visionbook

- To satisfy the sampling theorem when the input is *not* band-limited: **low-pass filter before sampling**
- This removes frequencies above the new Nyquist rate, preventing them from folding back (aliasing)
- Camera sensors include an **optical low-pass (anti-aliasing) filter** in front of the sensor for exactly this reason



Figure 9: note the ringing artifacts with the ideal filter. Source: visionbook

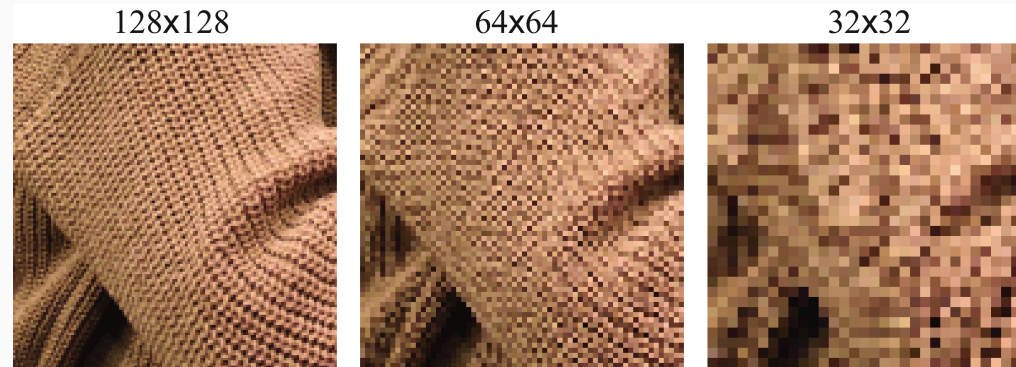


Figure 10: Source: visionbook

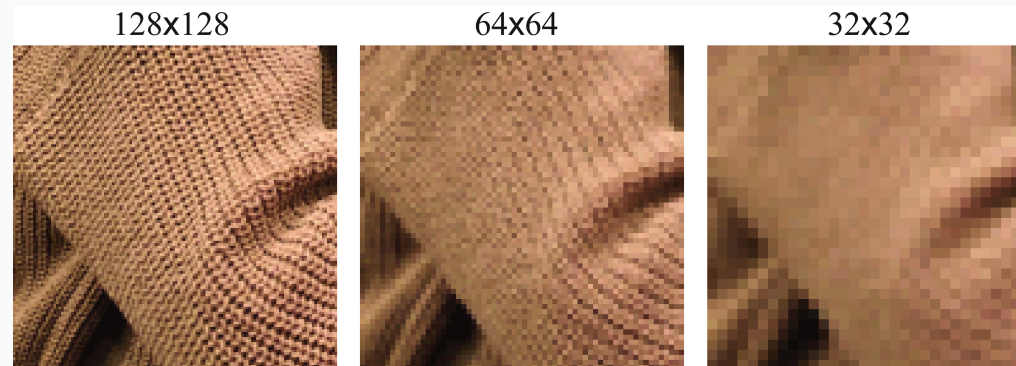


Figure 11: Source: visionbook

- A **zone plate** (concentric rings with increasing frequency toward the center) is a classic test pattern for visualizing aliasing
- When downsampled without filtering, the high-frequency outer rings produce spurious low-frequency moiré patterns

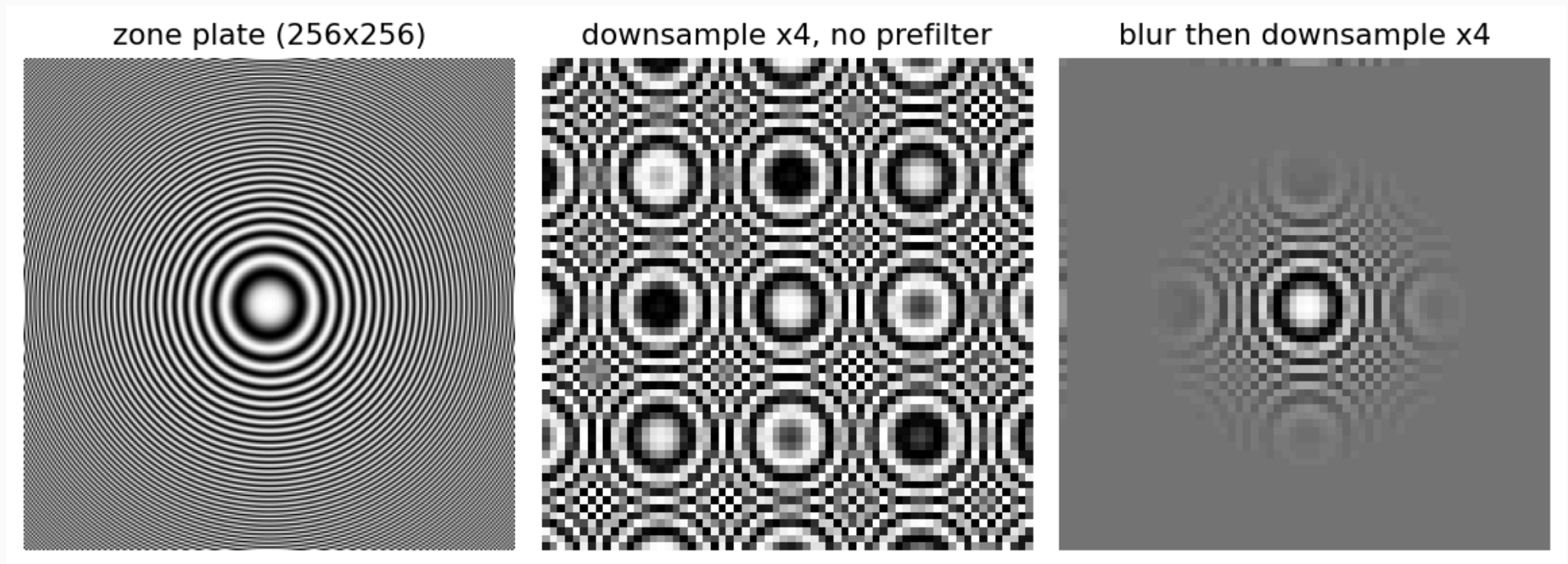


Figure 12: A zone plate downsampled by 4x with and without a pre-filter, note the spurious low-frequency rings without prefiltering

Scale-Space





Figure 13: Source: visionbook

Multi-Scale Representation

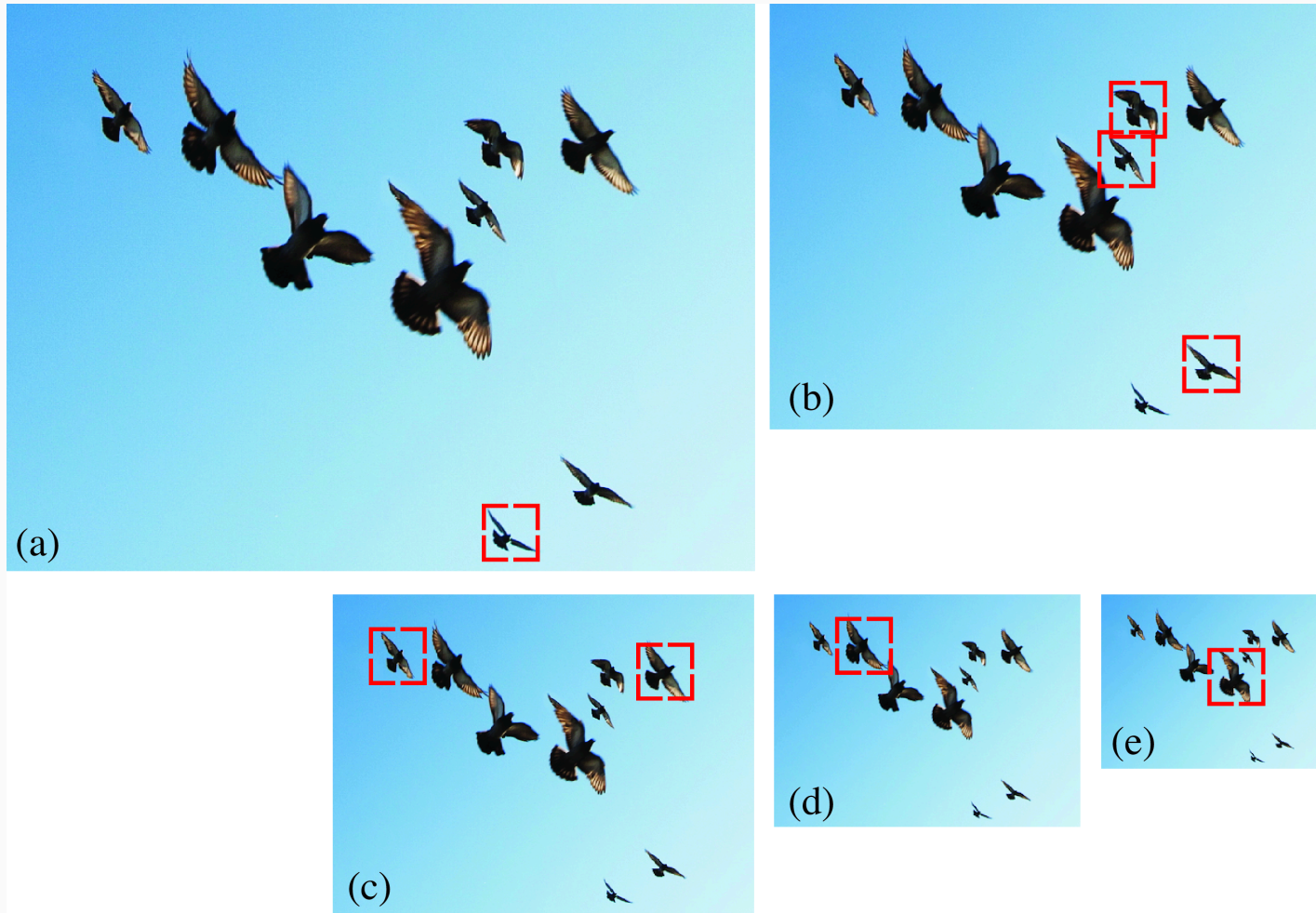


Figure 14: If we find the correct zoom level, every bird has the same size and we can use the same template to detect them. Source: visionbook

- Many CV tasks need to be solved at multiple scales (e.g. detect both small and large faces in an image)
- Two ways to search across scales:
 1. Fix the image, vary the **filter size** (g_σ for many σ)
 - this is the scale-space representation
 - expensive: a conv per scale, computed at full resolution. Filters also get bigger for bigger σ
 2. Fix the filter, vary the **image size** (repeatedly blur + downsample)
 - this is an **image pyramid**
 - the same small filter is reused at every level. Each level is cheaper than the previous one (smaller image)
- Pyramids trade some flexibility (discrete set of scales, related by a fixed factor) for large speedups

- Recall: choosing σ for Gaussian derivatives = choosing a **scale of analysis**
- we can define the **scale-space representation** of an image as the family

$$L(x, y; \sigma) = (g_\sigma * l)(x, y), \quad \sigma \geq 0$$

- $L(\cdot, \cdot; 0) = l$ (original image); increasing σ progressively removes fine detail
- A single image becomes a **continuum of images**, indexed by scale, structures appear/disappear/merge as σ grows

- The Gaussian is the Green's function of the **diffusion equation** (Lecture 5)
- $L(x, y; \sigma)$ satisfies the PDE (with $t = \sigma^2/2$):

$$\frac{\partial L}{\partial \sigma} = \nabla^2 L$$

- Increasing σ = letting the image “diffuse”, intuitively, intensity flows from brighter to darker regions, smoothing out structure monotonically

Consequence: **no new structure (extrema) is created** as σ increases

The Gaussian kernel can be demonstrated to be the unique choice that satisfies all the scale-space axioms

- **LSI**: linear and shift-invariant
- **rotational symmetry**
- **“composable”**: $g(g(l, \sigma_1), \sigma_2) = g(l, \sigma_1 + \sigma_2)$ is still a Gaussian
- doesn't add new structure (e.g. edges)

plus others we don't care about...



Figure 15: Image at increasing scales (sigma = 0, 1, 2, 4, 8)

- Fine texture disappears first; large-scale structures (face, silhouette) persist longest
- Different image structures have different “natural” scales

- $L(x, y; \sigma) = g_\sigma * l$ is defined for **every** $\sigma \geq 0$ on the **full-resolution** grid
- Computing and storing $L(\cdot, \cdot; \sigma)$ for many σ at full resolution is wasteful: at large σ , the image is very smooth and most of its information is already captured at much lower resolution
- **Idea**: once an image has been smoothed enough to remove detail at a given scale, we can safely **subsample** it without losing information

Image Pyramids

- A **pyramid** is a stack of images $\{I_0, I_1, I_2, \dots\}$ where I_0 is the original image and each I_{l+1} is a smoothed, downsampled version of I_l
- Each level represents the image at a coarser scale **and** a coarser sampling grid, doubling the effective σ at each level corresponds to halving the resolution

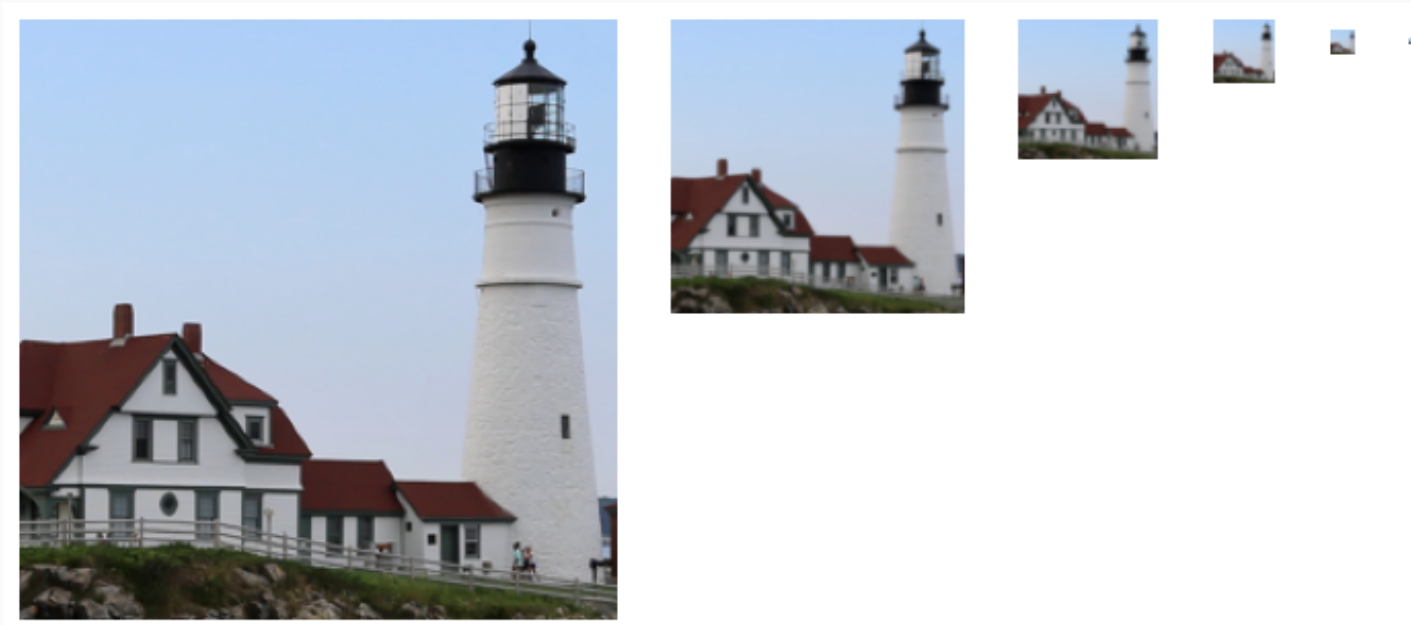


Figure 16: source: visionbook

- A halving of resolution (and corresponding doubling of σ) is called an **octave**, by analogy with frequency doubling in music/audio
- Level l has resolution $\frac{W}{2^l} \times \frac{H}{2^l}$ and effective scale $\sigma_l = 2^l \sigma_0$
- Total size of a pyramid with L levels (each $\frac{1}{4}$ the pixels of the previous, in 2D):

$$\sum_{l=0}^{L-1} \frac{WH}{4^l} \leq \frac{4}{3} WH$$

- A **full pyramid** (down to 1×1) costs only **33% more memory** than the original image
- Their efficiency is a major reason pyramids are ubiquitous

Gaussian Pyramid

- The **Gaussian pyramid** is the simplest pyramid: at each level, blur with a fixed small kernel, then downsample by 2 in each dimension

$$I_{l+1}[m, n] = (b * I_l)[2m, 2n]$$

- b is a small, fixed blur kernel (independent of level), applying it repeatedly to progressively smaller images approximates the effect of larger and larger Gaussians on the original image (repeated convolution is approximately Gaussian)

Reminder: Without the blur, downsampling would result in aliasing instead of highlighting structure at different scales

- The **binomial filter** is the standard choice for b : cheap, integer, separable, exact composition rule $b_n \circ b_n = b_{2n}$
- A common choice is the 5×5 binomial-derived kernel (a.k.a. “Burt-Adelson” kernel), separable as $\frac{1}{16} [1, 4, 6, 4, 1]$ along each axis

- all the operations are linear
- in matrix form, for a 1D signal g_0 :

$$g_{k+1} = D_k B_k g_k = G_k g_k$$

- D_k is the downsampling operator
- B_k the blur operator

$$G_0 = D_0 B_0 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \frac{1}{16} \begin{pmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{pmatrix}$$

Example: Gaussian Pyramid



Figure 17: Gaussian pyramid: each level is blurred and downsampled by 2 from the previous
Coarse levels highlight global features, useful for object detection, which requires also global-scale features.

- Total memory: $\leq \frac{4}{3}$ the original image
- Each level halves both the spatial resolution **and** the relevant frequency content, by a factor of 2 in each dimension
- Coarse levels: cheap to process, capture global/large-scale structure, miss fine detail
- Fine levels: expensive, capture detail and texture, but small-scale structure is hard to detect reliably

Laplacian Pyramid

- a Gaussian pyramid gradually loses information at each level
- we can complement this representation with another pyramid encoding the difference between two consecutive levels of the pyramid

Laplacian pyramid:

- given g_k and g_{k+1} , two consecutive levels of the Gaussian pyramid
- rescale g_{k+1} to have the same size as g_k
- compute their difference

Why is it called Laplacian if we don't use the LoG (mexican hat)? The Difference of Gaussians (DoG) is an approximation of the LoG.

Laplacian pyramid:

- given g_k and g_{k+1} , two consecutive levels of the Gaussian pyramid
- rescale g_{k+1} to have the same size as g_k
- compute their difference

$$l_k = g_k - F_k g_{k+1} = (I_k - F_k G_k) g_k = L_k g_k$$

- $F_k = B_k U_k$ combines upsampling U_k and blur B_k
- upsampling U_k is the naive upsampling that just inserts zeros between samples
- B_k is the same blur we used for the Gaussian pyramid (e.g. fourth binomial filter b_4)



Figure 18: source: visionbook

By construction, we can reconstruct the original image given the Laplacian pyramid and the last level of the Gaussian pyramid (g_{k+1})

Laplacian pyramid:

$$l_k = g_k - F_k g_{k+1} = (I_k - F_k G_k) g_k = L_k g_k$$

Inversion: $g_k = l_k + F_k g_{k+1}$

Applications

- Many vision problems involve searching over a large parameter space (e.g. all possible displacements, all possible scales)
- **Coarse-to-fine** strategy: solve the (cheap, low-resolution) problem at the coarsest pyramid level first, then use that solution to **restrict the search range** at the next finer level, and repeat
- Massively reduces computation: instead of an exhaustive search at full resolution, only a small local search is needed at each level after the first

- Large displacements are hard to estimate directly at full resolution (a pixel may have moved far outside a small search window)
- At a coarse pyramid level, the same displacement (in pixels) is much smaller, easy to estimate
- Estimate flow at the coarsest level, upsample and use as an initial guess for the next level, refine, repeat down to full resolution
- Common strategy in classical (Lucas-Kanade) and modern (RAFT) optical flow methods

- Naive template matching (slide a template over the image, compute similarity at every position) is $O(\text{image size} \times \text{template size})$
- Coarse-to-fine: match at the coarsest level (small image, small template, cheap), keep only the top candidate locations, refine only those locations at the next finer level
- Avoids exhaustively searching the full-resolution image, while still finding the globally best match (assuming the coarse match is a good enough starting point)

- Image alignment / registration also benefits from coarse-to-fine: estimate a rough transformation at low resolution, then refine
- Combines naturally with feature-based matching: detect features at multiple pyramid levels, match across images, estimate a transformation robust to the scale at which features were found

- Goal: combine two images A and B into one, with a smooth transition at the boundary instead of a hard cut
- Naive solution: pick a boundary curve, copy A on one side and B on the other
- Problem: even if A and B are well-aligned, differences in exposure, color, or fine detail create a visible seam at the boundary
- Need a blending strategy that hides the seam while preserving detail from both images

Multi-band blending (Burt & Adelson, 1983):

- Build Laplacian pyramids LP^A , LP^B for both images
- Build a Gaussian pyramid GP^α for the blend mask α (the mask itself gets smoothed/downsampled at each level)
- Blend each band separately: $LP_l^I = GP_l^\alpha LP_l^A + (1 - GP_l^\alpha) LP_l^B$
- Reconstruct I from the blended Laplacian pyramid

Each frequency band is blended with a transition width proportional to its scale

- A CNN with strided convolutions/pooling produces a sequence of feature maps at decreasing spatial resolution, **conceptually a learned pyramid**
- Early layers (high resolution): fine detail, small receptive field, analogous to fine pyramid levels
- Late layers (low resolution): coarse, large receptive field (large effective σ), analogous to coarse pyramid levels
- The same scale/efficiency tradeoffs apply: coarse features are cheap and capture large-scale context, fine features are expensive and capture detail

- Object detection/segmentation needs to find objects at **many scales** (e.g. a person near the camera vs. far away)
- A plain CNN's last layer is too coarse for small objects, but earlier layers, while high-resolution, lack the high-level semantic information of later layers
- **FPN** (Lin et al., 2017): augment the CNN's feature pyramid with a **top-down pathway** that upsamples coarse, semantically-rich features and merges them (by addition) with finer, high-resolution features via lateral connections, directly analogous to the Laplacian pyramid's upsample-and-combine structure
- Result: every level of the pyramid is both high-resolution and semantically rich, used for detection at the appropriate scale

Conclusions

- **Aliasing** is a key issue in most vision pipelines. Frequencies above Nyquist are aliased and introduce new “structure” in the image
- **Downsampling** requires a blur to avoid aliasing
- **Scale-Invariance** is achieved via multi-scale representations (Gaussian pyramid, Laplacian Pyramid, or learned CNNs)

- MIT Vision Book, Ch. 20, 21, 23: <https://visionbook.mit.edu/series.html>

- **Lecture 7:** Features and descriptors, Harris corners, SIFT, and learned local features