Image processing operators (2)

Bing Gong
（巩冰）
gongbing@shnu.edu.cn

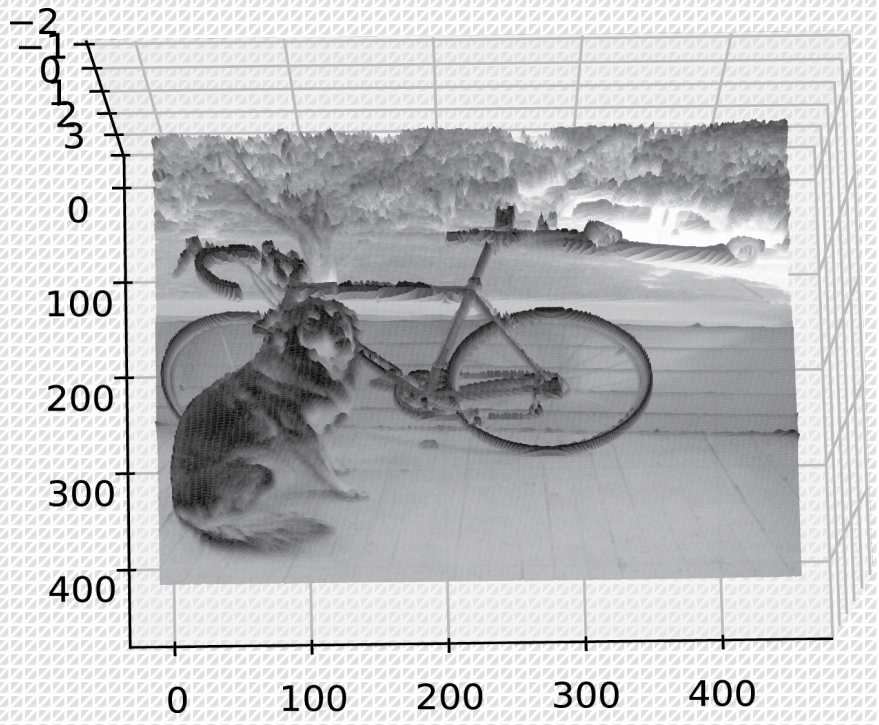
Methods of image transform

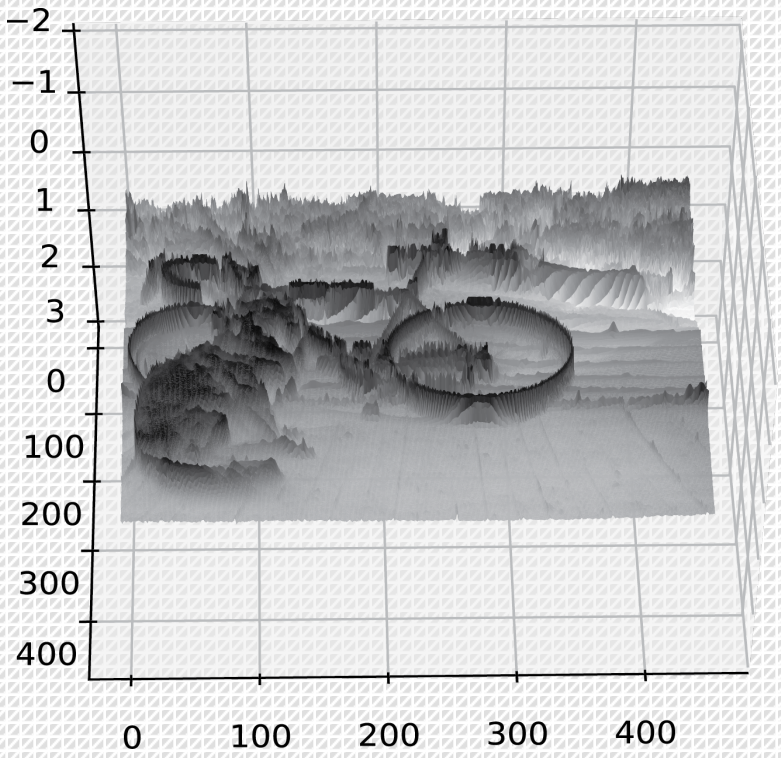
- *point* operators or *point processes:* the simplest kind of image transforms, namely those that manipulate each pixel independently of its neighbors.
- ***neighborhood* (area-based) operators:** each new pixel's value depends on a small number of neighboring input values.
- *global operators:* *geometric transformations,* such as rotations, shears, and perspective deformations.
- *global optimization:* involve the minimization of an energy functional or, equivalently, optimal estimation using Bayesian *Markov random field* models.

Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- What is an “edge”?

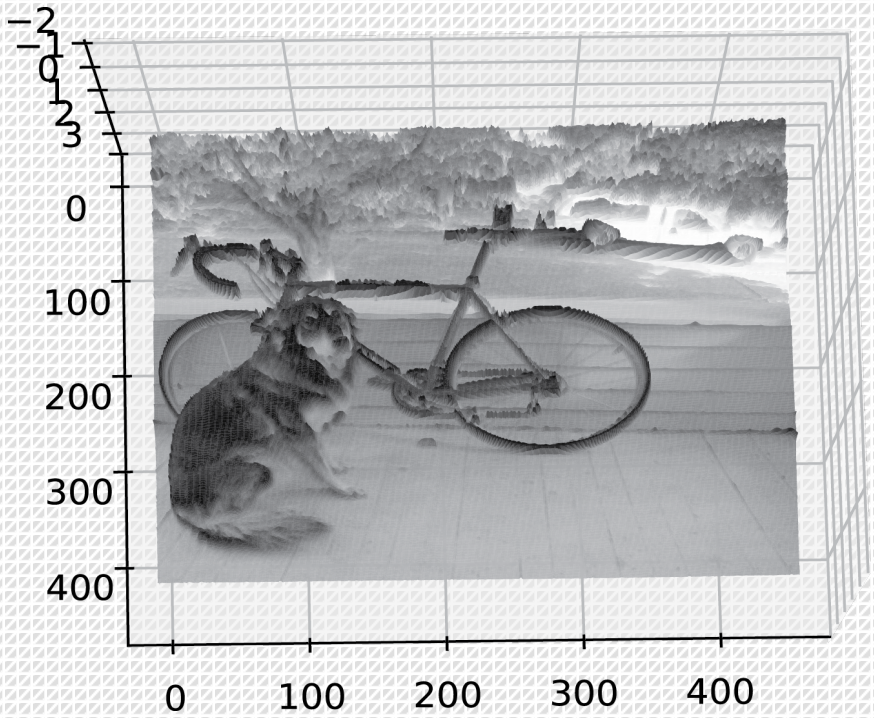


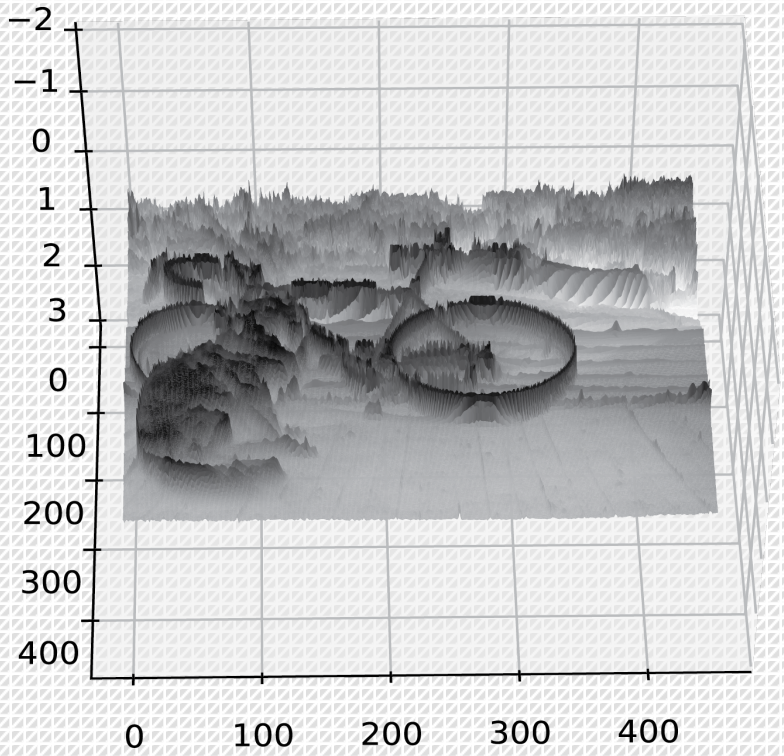


Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Image is a function.
- Think of the gray tones as HEIGHTS.
- Edges are rapid changes in this function

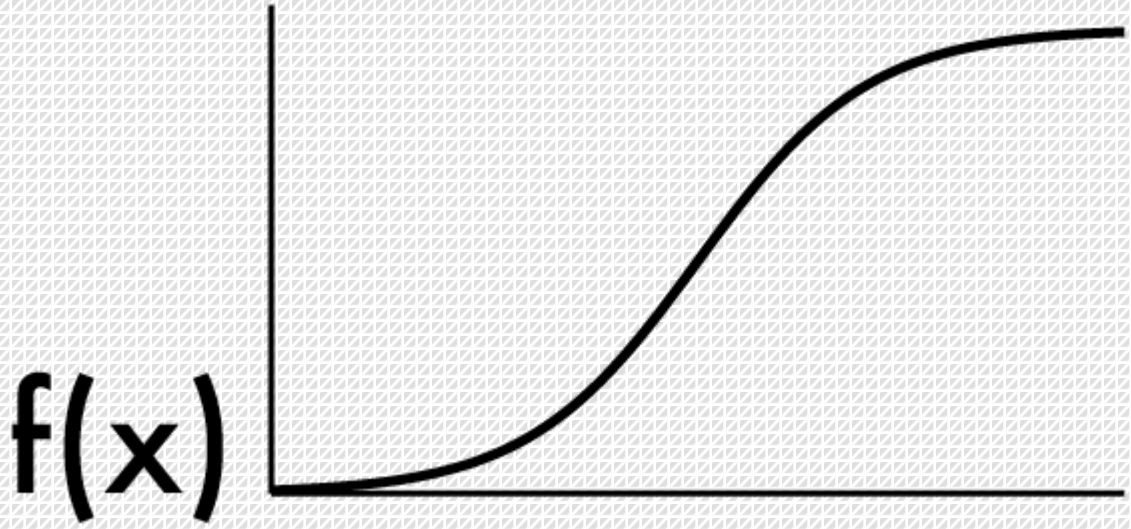




Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

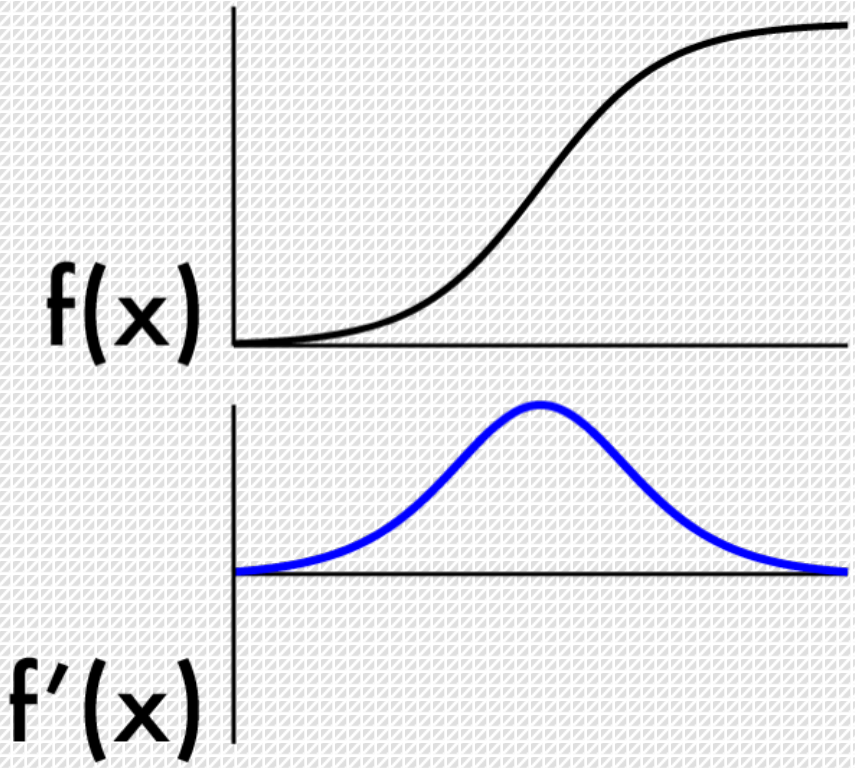
- Image is a function
- Edges are rapid changes in this function



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Image is a function
- Edges are rapid changes in this function



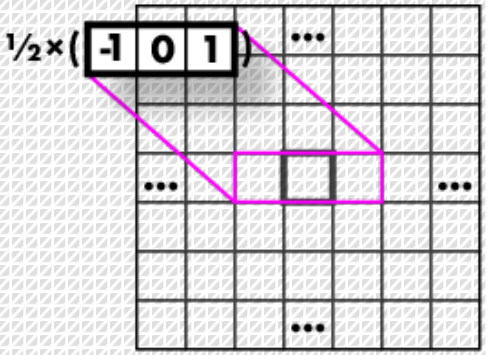
Edges

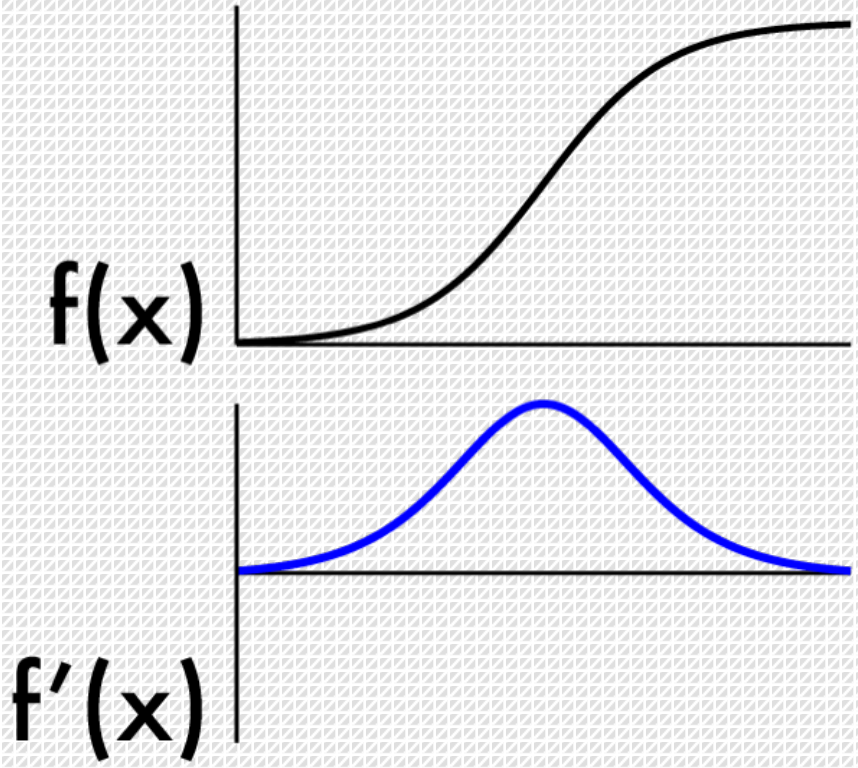
Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Recall:

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}.$$

- We don't have an "actual" function, must estimate
- Possibility: set h = 2
- What will that look like?

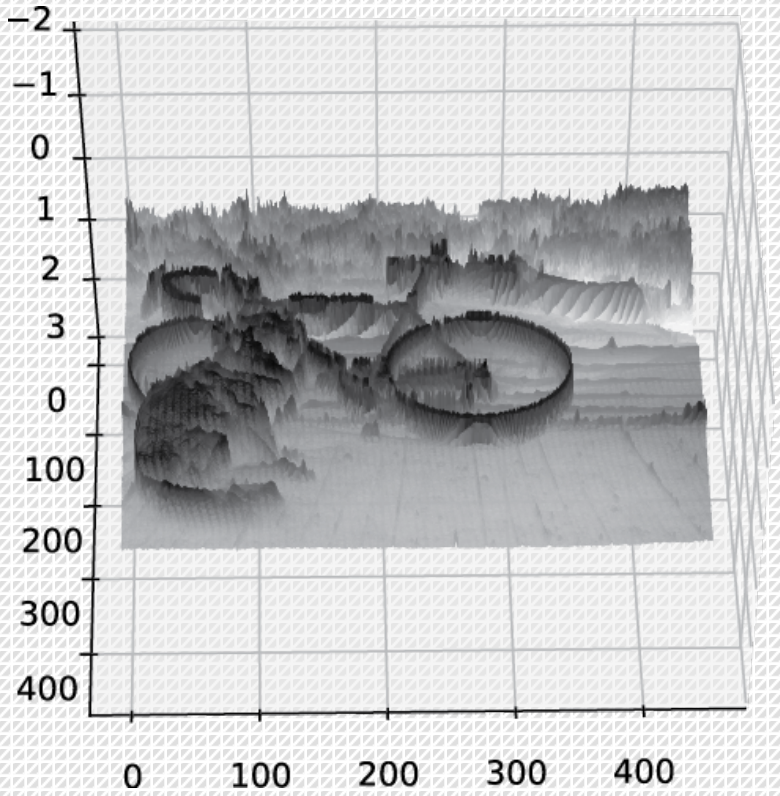


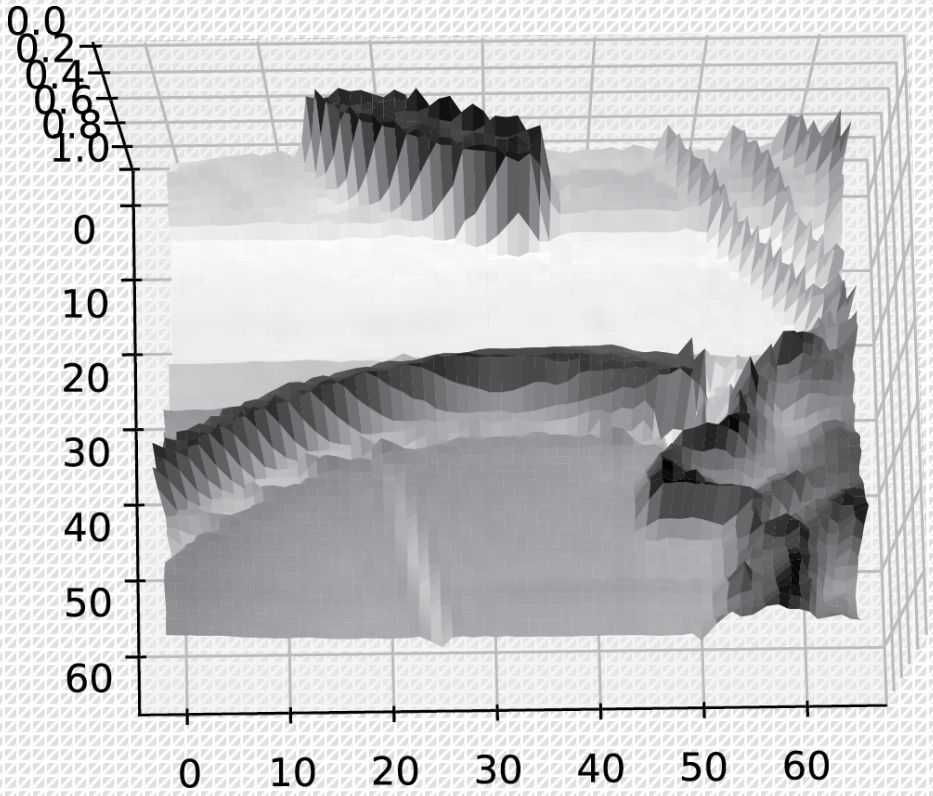


Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Images are noisy!

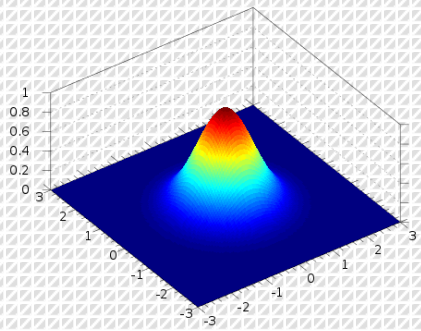


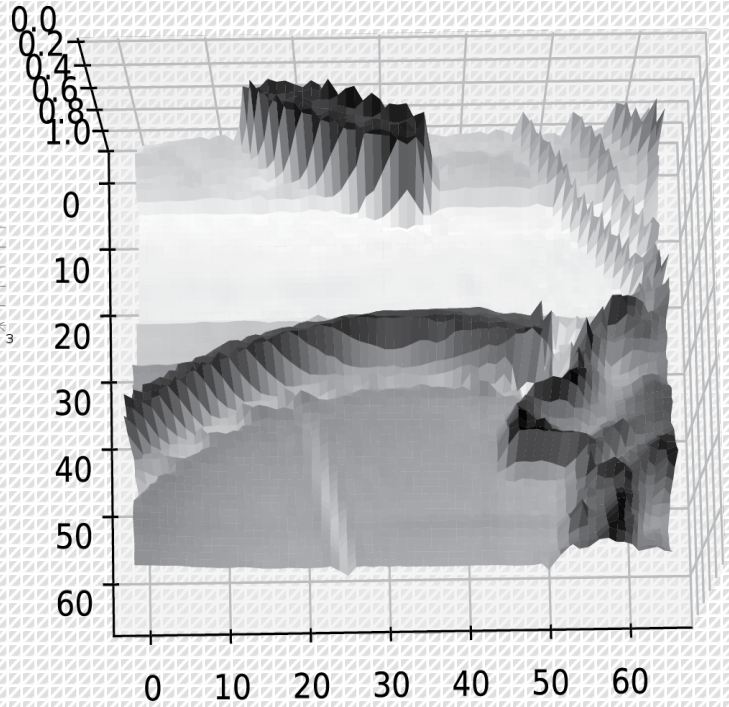


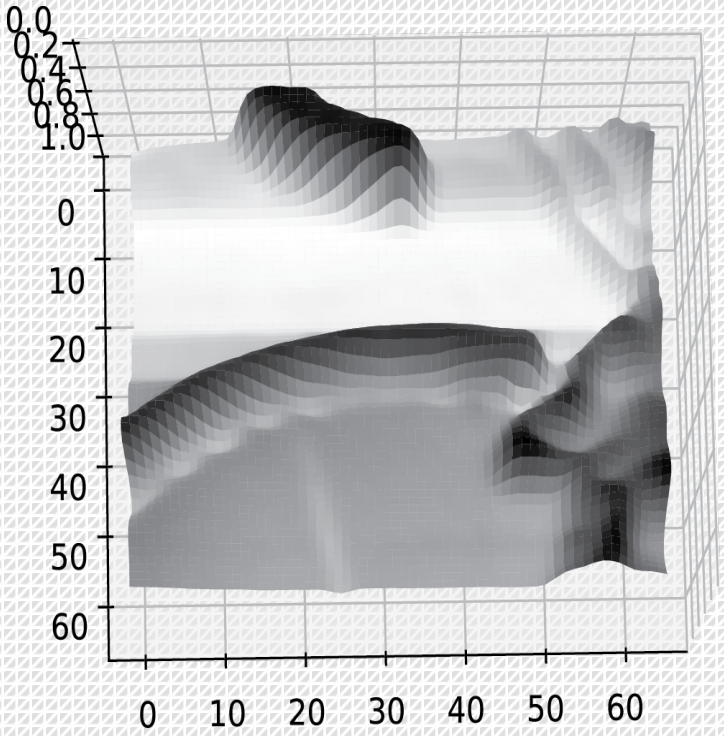
Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Images are noisy!



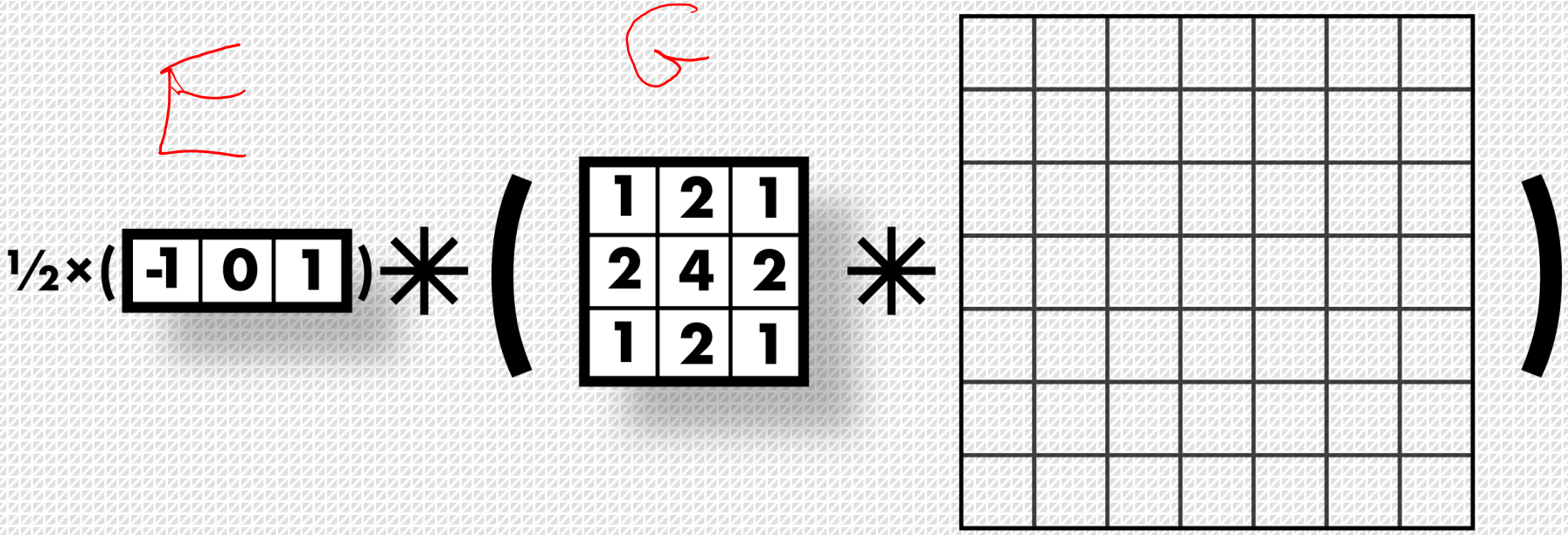




Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

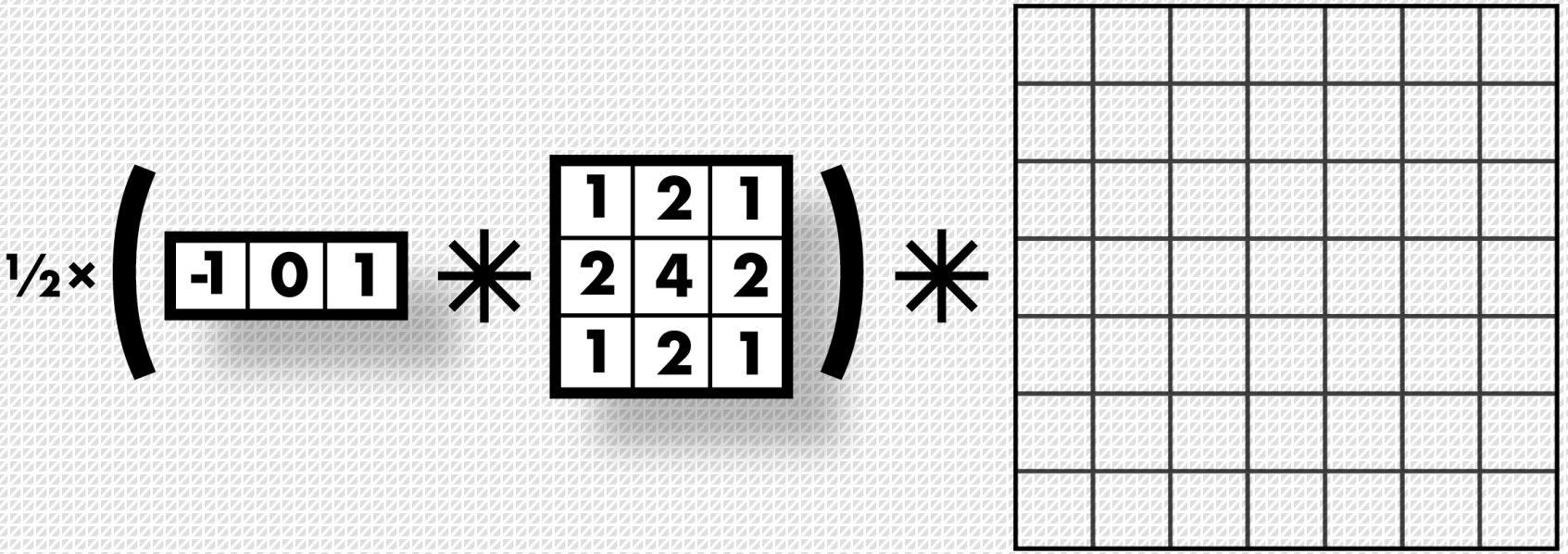
- Images are noisy!



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

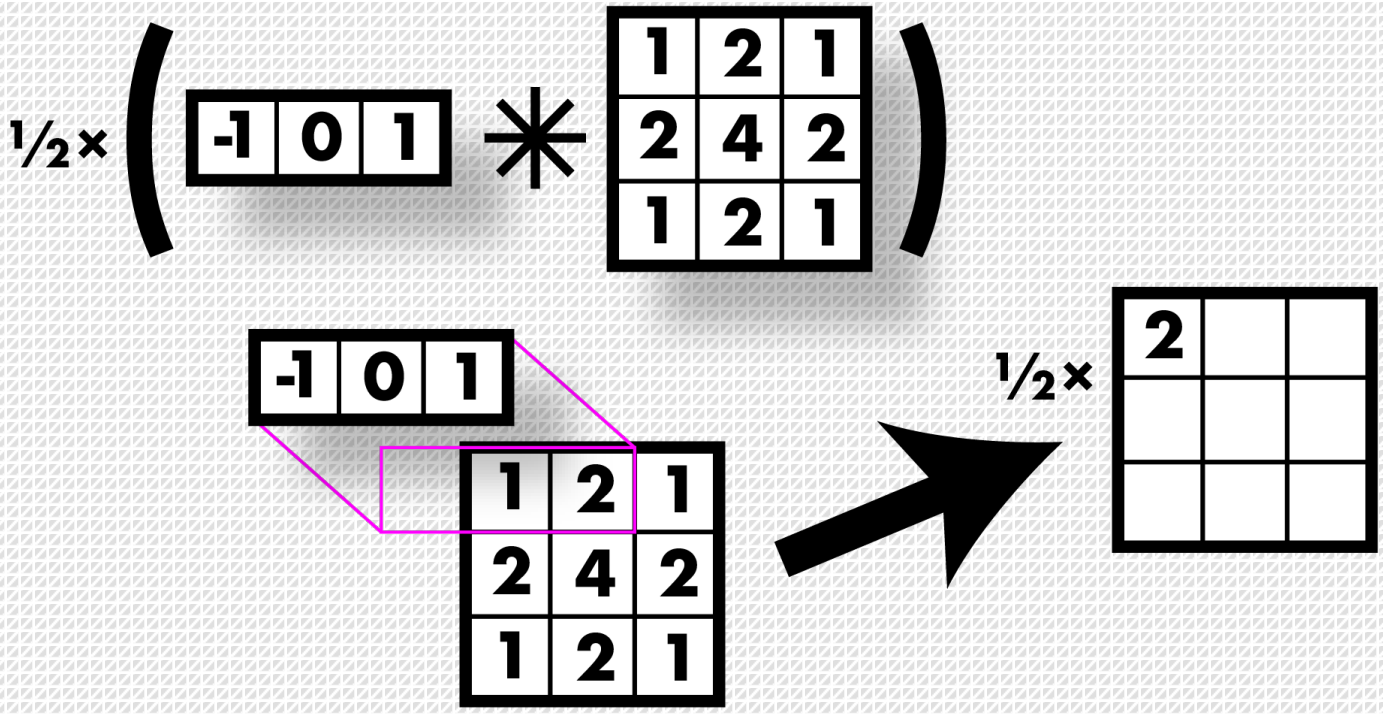
- Smooth first, then derivative



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

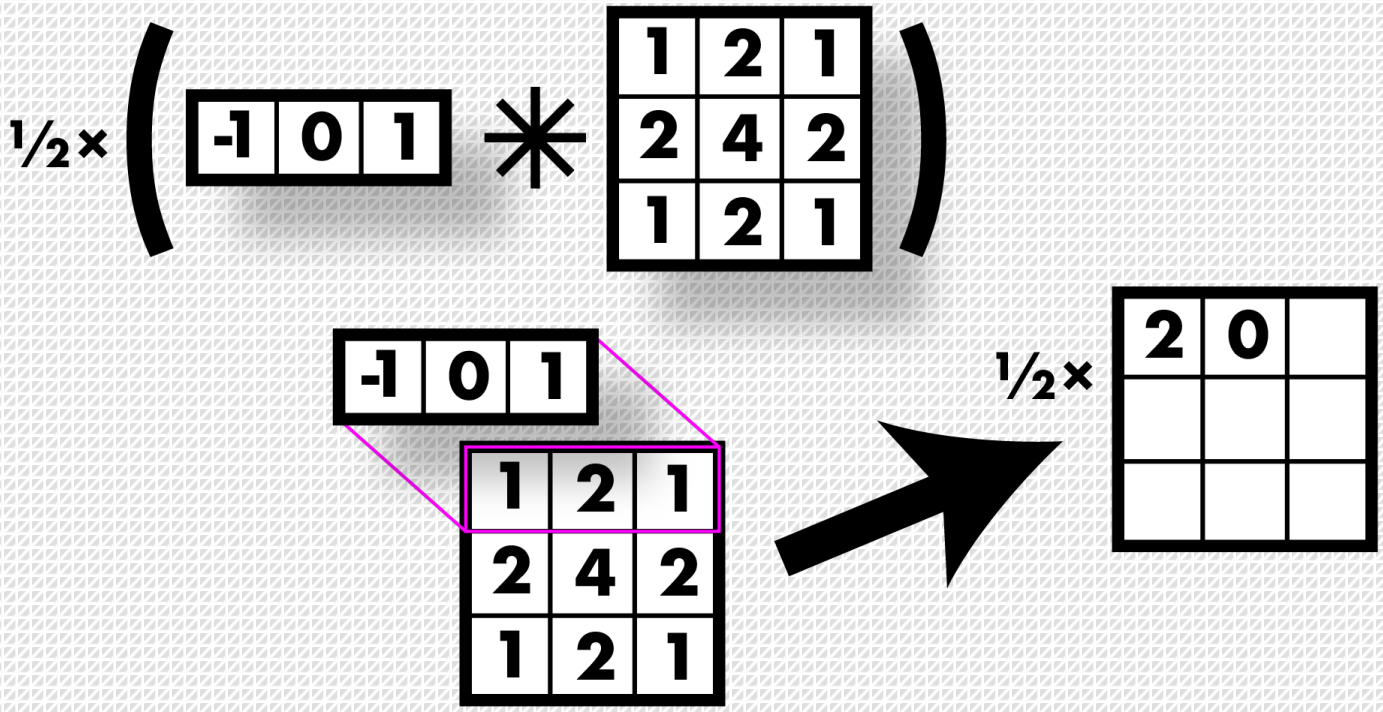
- Smooth first, then derivative



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

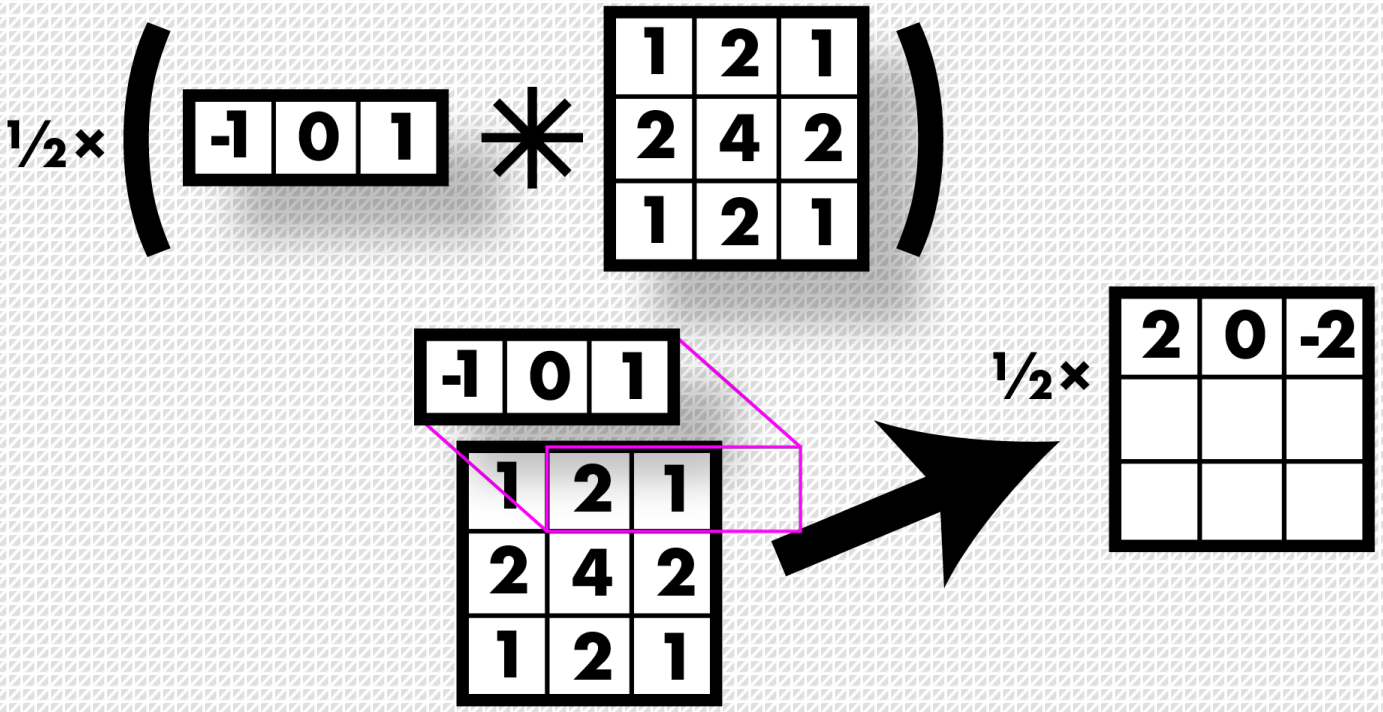
- Smooth first, then derivative



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

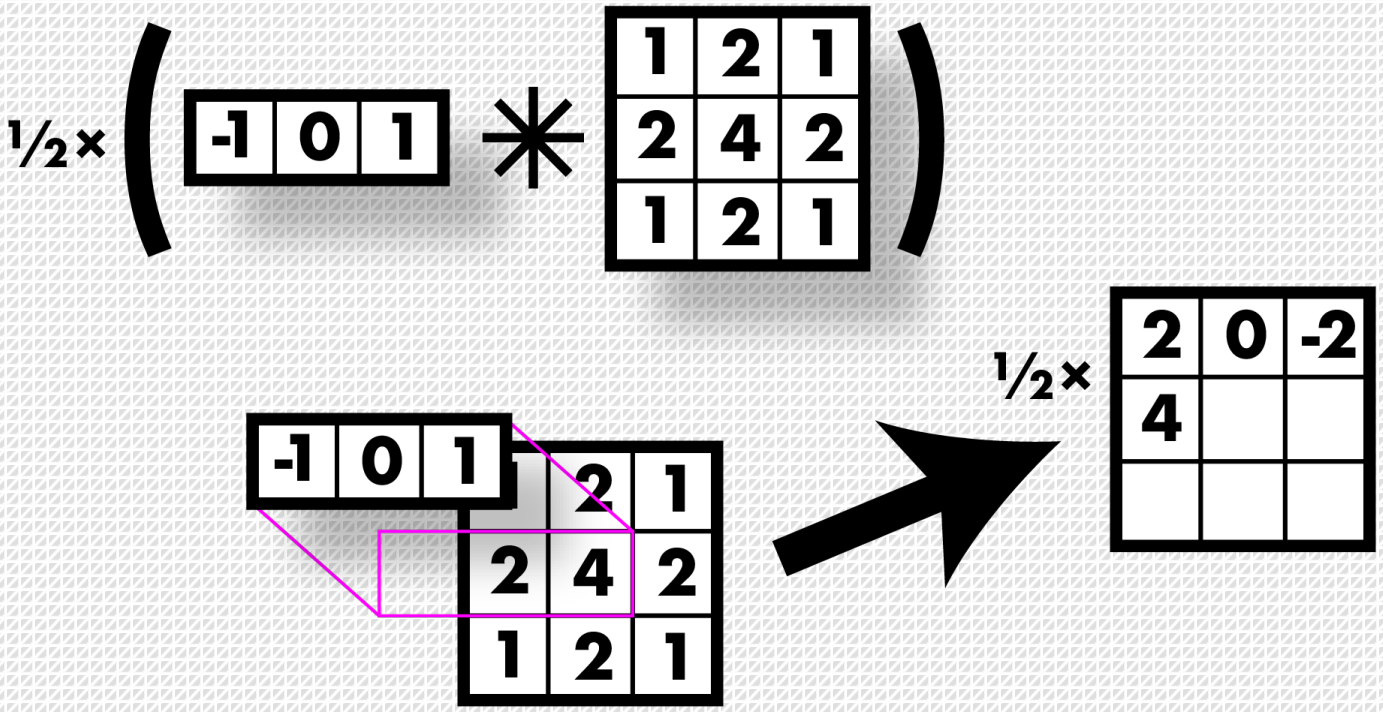
- Smooth first, then derivative



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

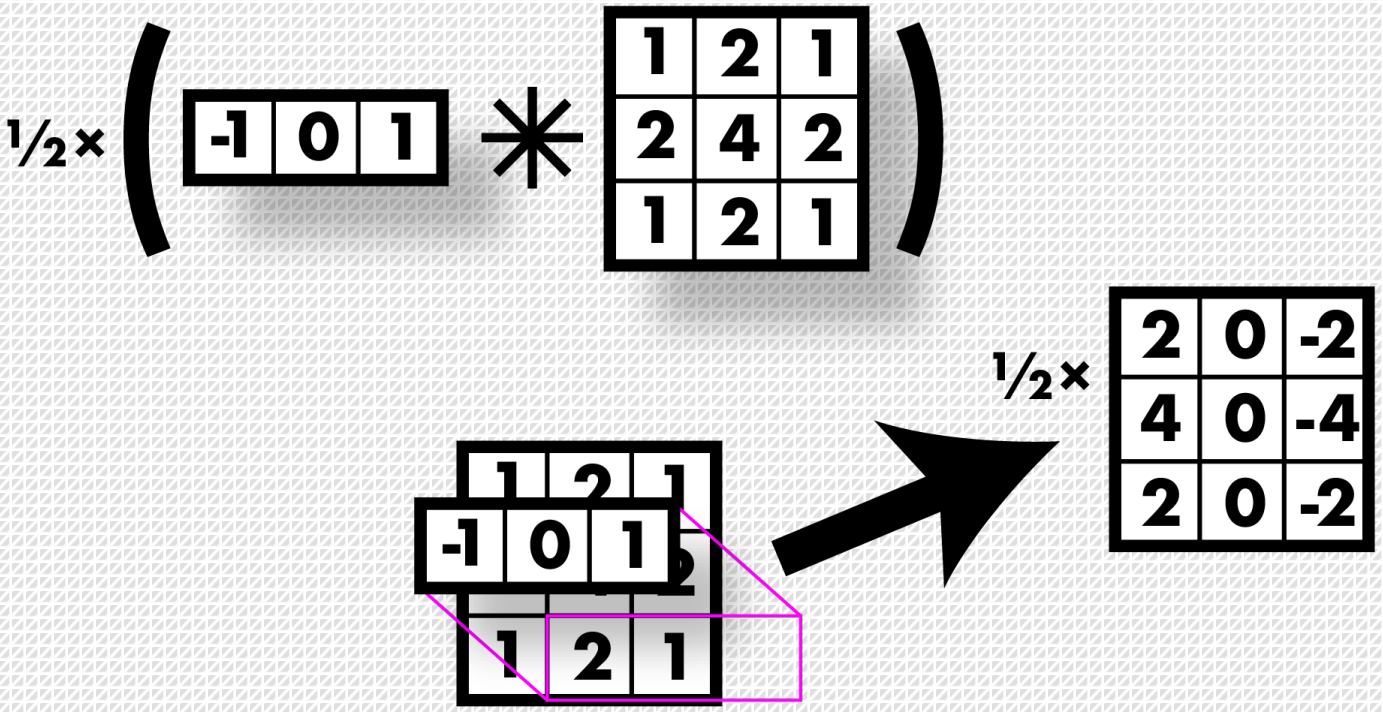
- Smooth first, then derivative



Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Smooth first, then derivative

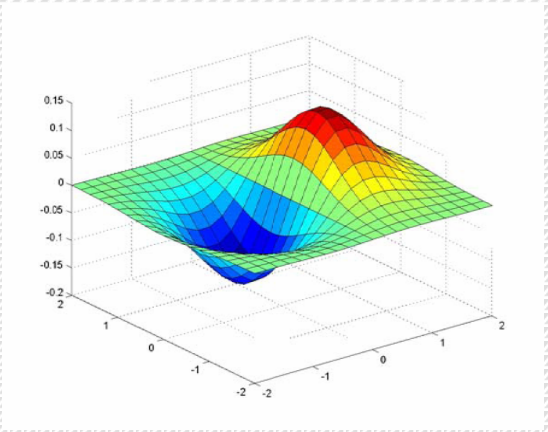


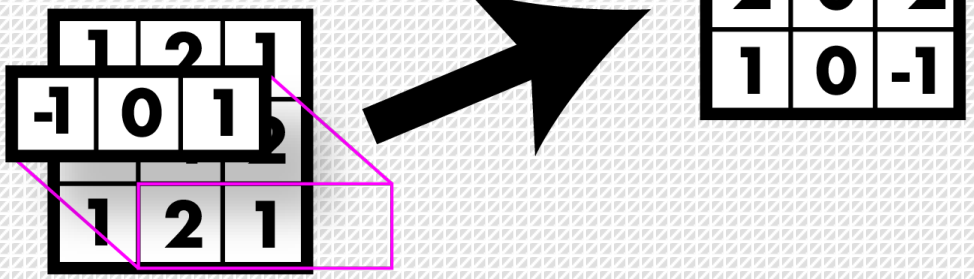
Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Smooth first, then derivative → Sobel filter!（索伯算子）

$$\frac{1}{2} \times \left(\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \right)$$





Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Smooth first, then derivative

$\frac{1}{K^2}$

1	1	⋯	1
1	1	⋯	1
⋮	⋮	1	⋮
1	1	⋯	1

$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

$\frac{1}{256}$

1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

$\frac{1}{8}$

−1	0	1
−2	0	2
−1	0	1

$\frac{1}{4}$

1	−2	1
−2	4	−2
1	−2	1

$\frac{1}{K}$ | 1 | 1 | ⋯ | 1 |

$\frac{1}{4}$ | 1 | 2 | 1 |

$\frac{1}{16}$ | 1 | 4 | 6 | 4 | 1 |

$\frac{1}{2}$ | −1 | 0 | 1 |

$\frac{1}{2}$ | 1 | −2 | 1 |

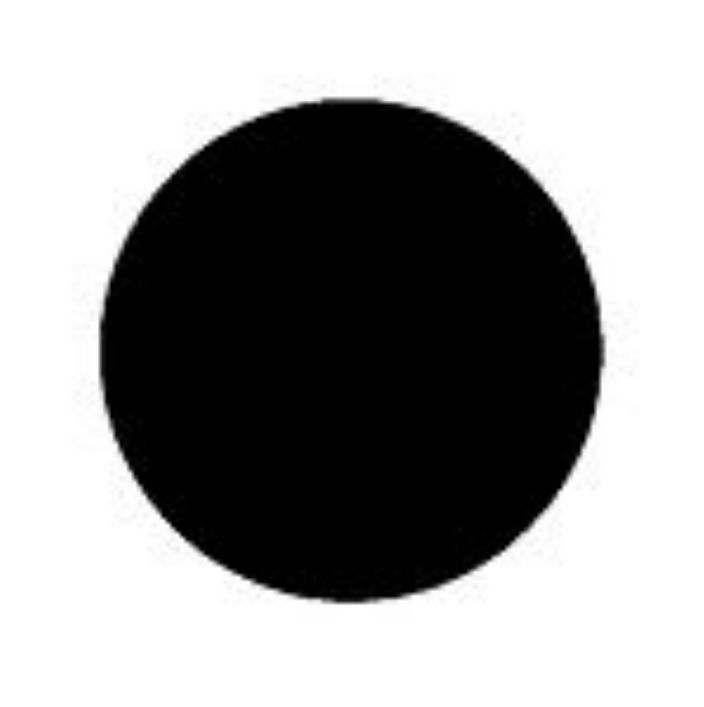
(a) box, $K = 5$ (b) bilinear (c) "Gaussian" (d) Sobel (e) corner

Figure 3.14 Separable linear filters: For each image (a)–(e), we show the 2D filter kernel (top), the corresponding horizontal 1D kernel (middle), and the filtered image (bottom). The filtered Sobel and corner images are signed, scaled up by 2× and 4×, respectively, and added to a gray offset before display.

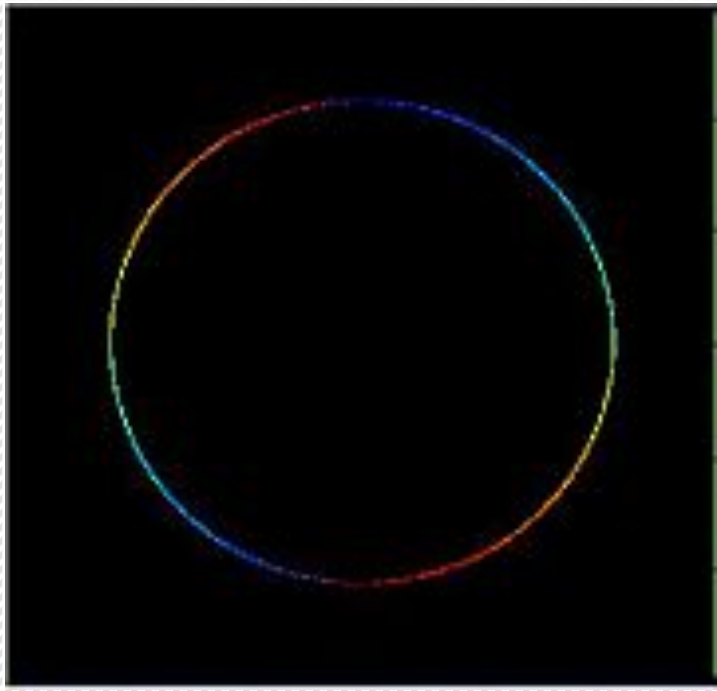
Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Could take derivative
- Find high responses
- Sobel filters!
- But…
  - Edges go both ways
  - Want to find extrema



黑色圆圈与白底的灰阶影像

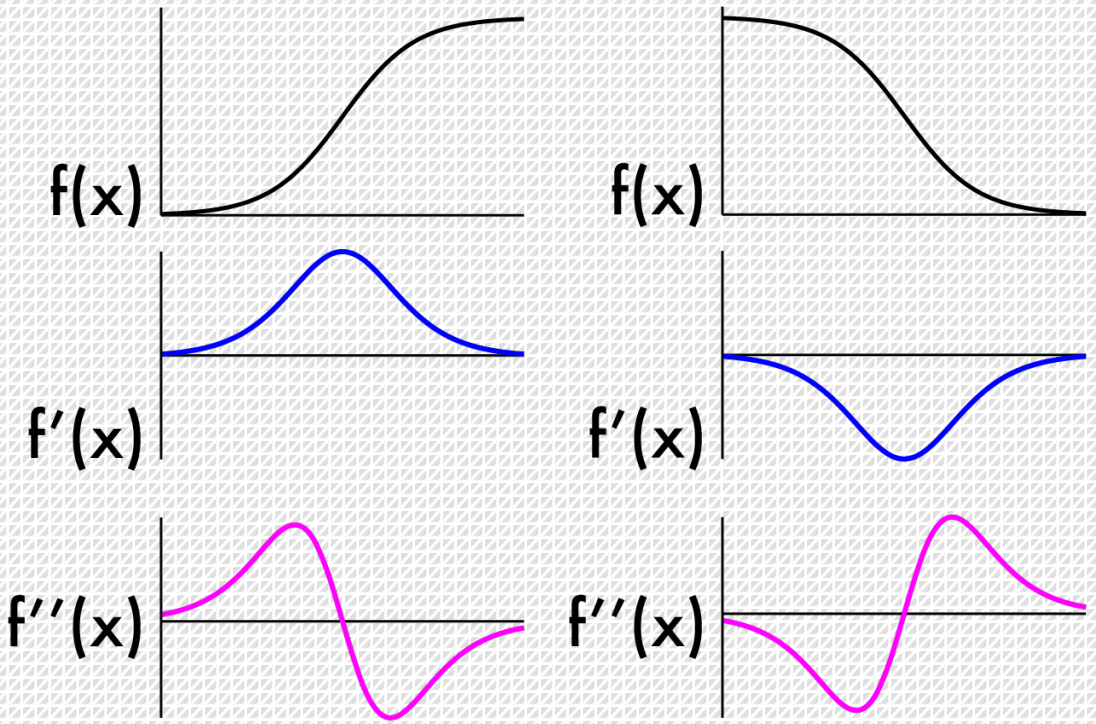


梯度方向结果

Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Could take derivative
- Find high responses
- Sobel filters!
- But…
- Edges go both ways
- Want to find extrema



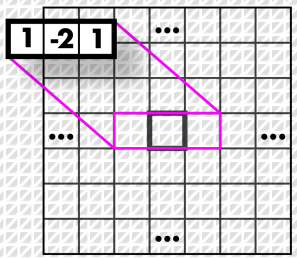
Edges

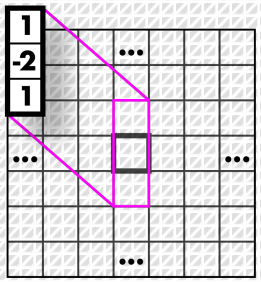
Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

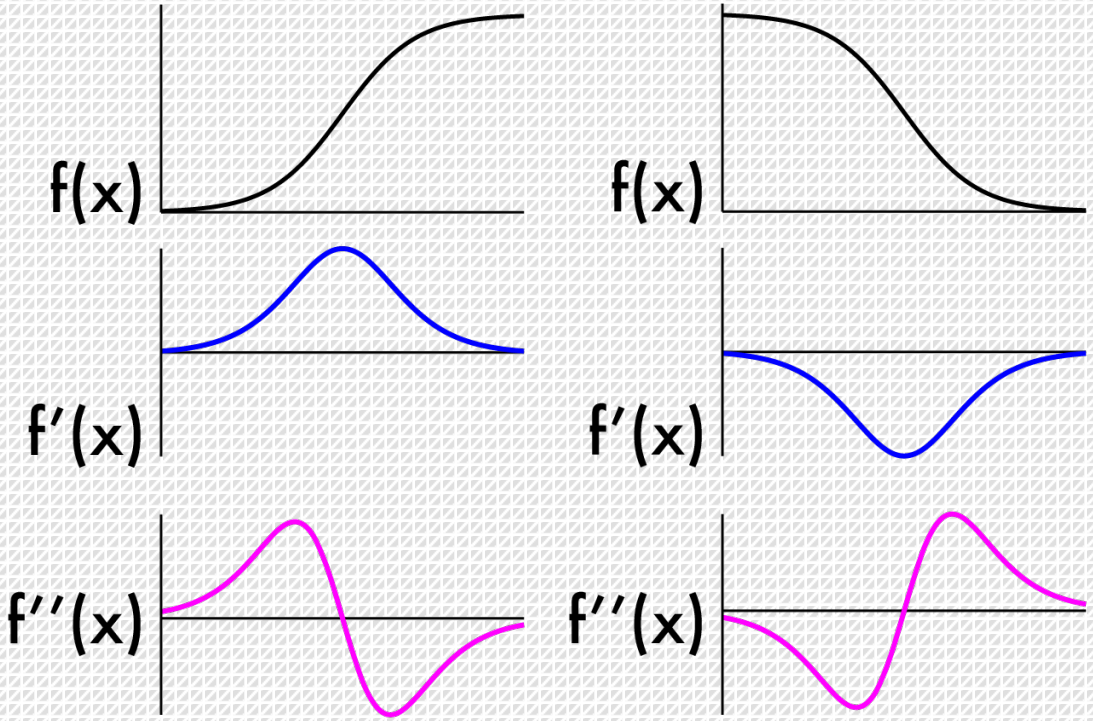
- Crosses zero at extrema
- Recall:
  - $$f''(x) = \lim_{h \to 0} \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}.$$
- Laplacian:

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- Again, have to estimate f’’(x):







Laplacian filter computes the second-order derivation for the better intensity of an image. It is a smoothening filter that reduces noise and projects the image more specifically than the Sobel filter.

Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Laplacians

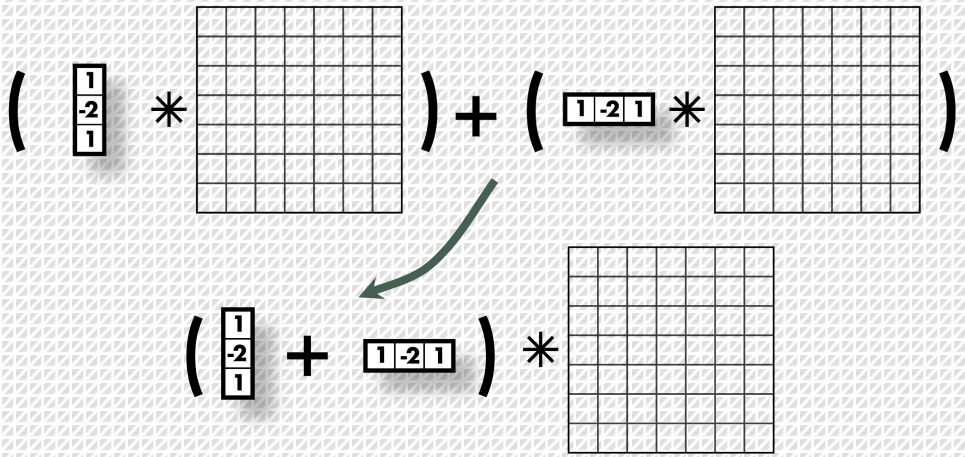
$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Laplacians

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

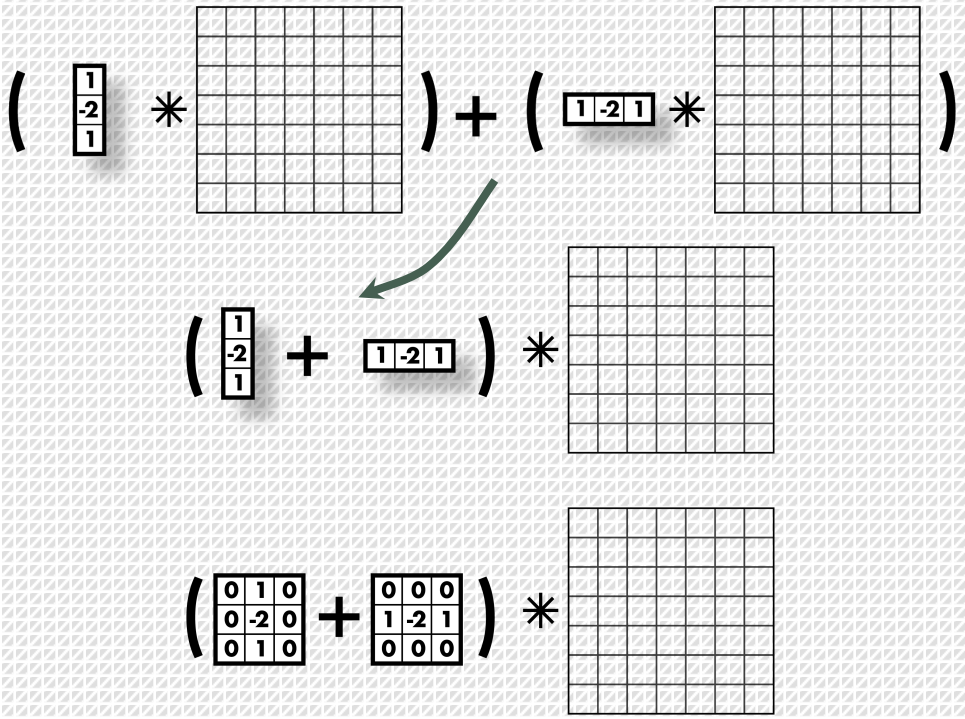


Edges

Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

- Laplacians

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

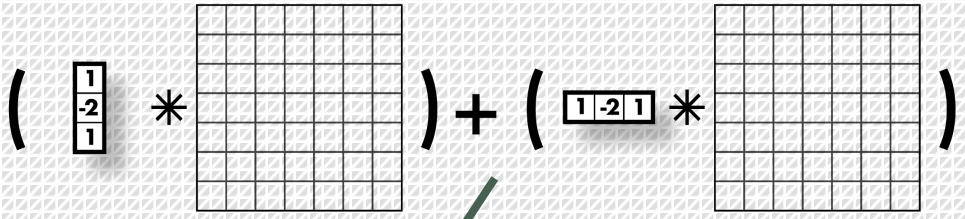


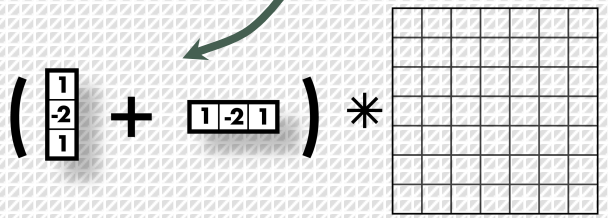
Edges

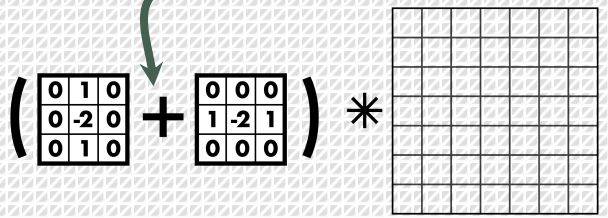
Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

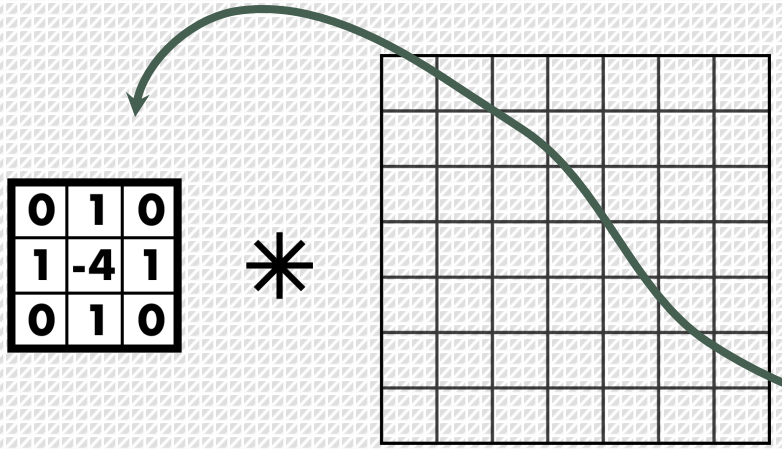
- Laplacians

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$







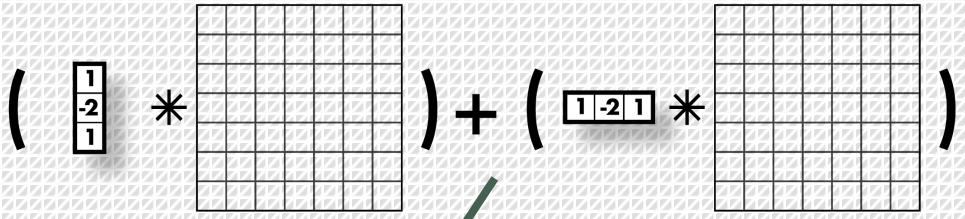


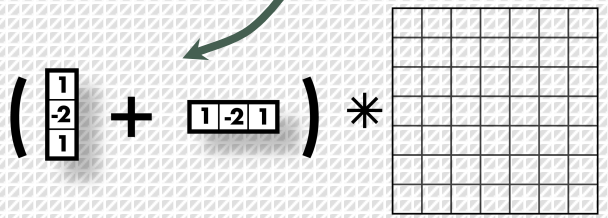
Edges

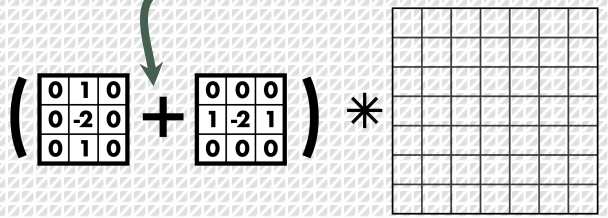
Source: https://courses.cs.washington.edu/courses/cse576/23sp/notes.html

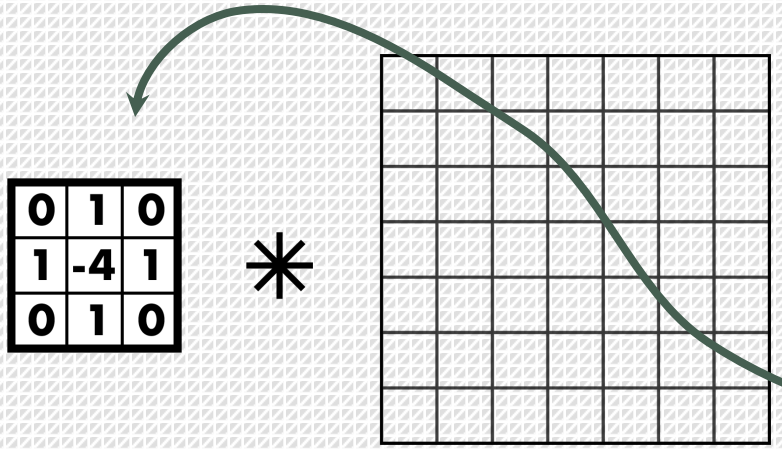
- Laplacians

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

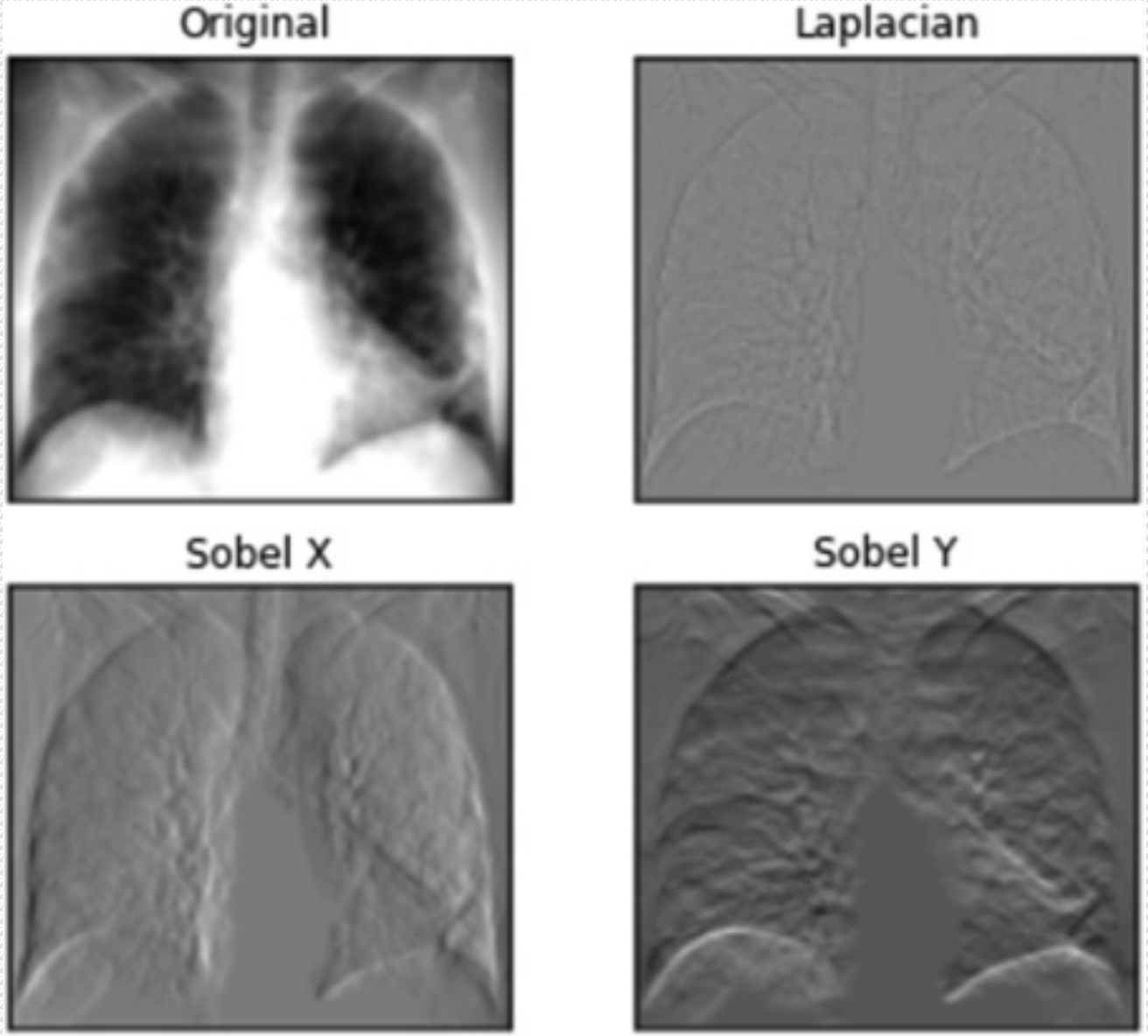








Edges



Other linear filters

- Summed area tables (*Integral Image*)
  - … is used for face detection to compute simple multi-scale low-level features

3	2	7	2	3
1	5	1	3	4
5	1	**3**	5	1
4	3	2	1	6
2	4	1	4	8

(a) S = 24

3	5	12	14	17
4	*11*	**19**	24	31
9	**17**	28	38	46
13	24	37	48	62
15	30	44	59	81

(b) s = 28

3	5	12	*14*	17
4	11	19	24	31
9	17	28	38	46
13	24	37	**48**	62
15	30	44	59	81

(c) S = 24

Figure 3.17 Summed area tables: (a) original image; (b) summed area table; (c) computation of area sum. Each value in the summed area table $s(i, j)$ (red) is computed recursively from its three adjacent (blue) neighbors (3.31). Area sums S (green) are computed by combining the four values at the rectangle corners (purple) (3.32). Positive values are shown in **bold** and negative values in *italics*.

Non-linear filter

- Why we need non-linear filter?
- Regular blurring with a Gaussian filter fails to remove the noisy pixels and instead turns them into softer (but still visible) spots

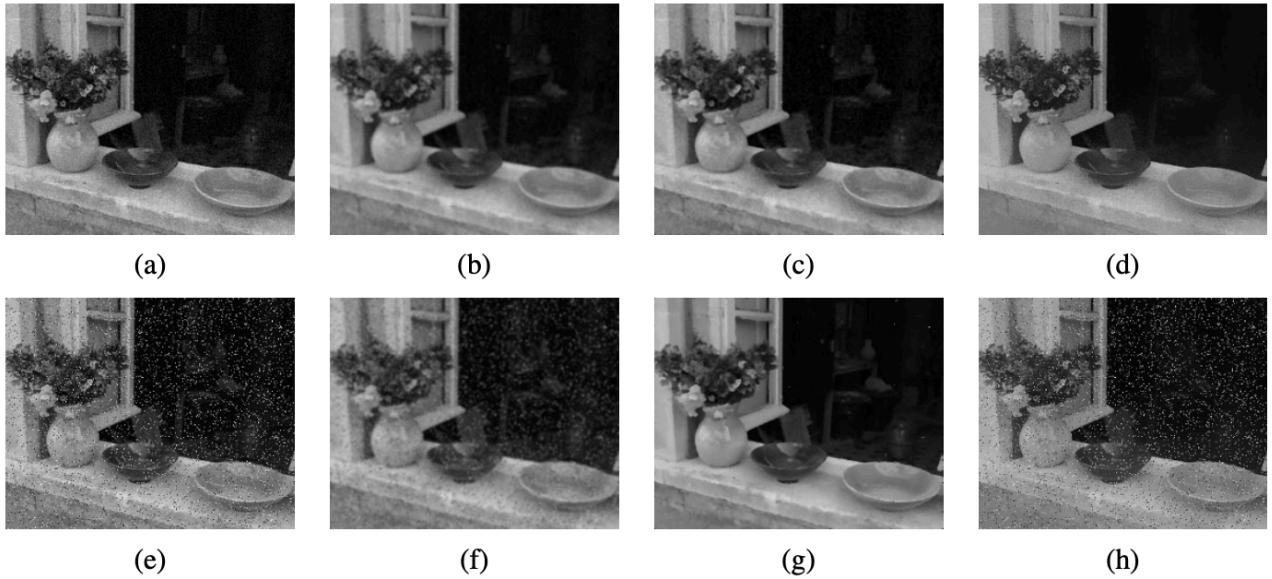


Figure 3.18 Median and bilateral filtering: (a) original image with Gaussian noise; (b) Gaussian filtered; (c) median filtered; (d) bilaterally filtered; (e) original image with shot noise; (f) Gaussian filtered; (g) median filtered; (h) bilaterally filtered. Note that the bilateral filter fails to remove the shot noise because the noisy pixels are too different from their neighbors.

Non-linear filter

- Median Filtering:

…selects the median value from each pixel's neighborhood.

…since the shot noise value usually lies well outside the true values in the neighborhood, the median filter is able to filter away such bad pixels.

1	2	1	2	4
2	1	3	5	8
1	3	7	6	9
3	4	8	6	7
4	5	7	8	9

(a) median = 4

Non-linear filter

- Median Filtering (α-trimmed mean):

….averages together all of the pixels except for the α fraction that are the smallest and the largest

1	2	1	2	4
2	1	3	5	8
1	3	7	6	9
3	4	8	6	7
4	5	7	8	9

(b) α-mean= 4.6

Non-linear filter

- Median Filtering (weighted median):
- …. each pixel is used a number of times depending on its distance from the center

Non-linear filter

- Bilateral filtering :
  - What if we were to combine the idea of a weighted filter kernel with a better version of outlier rejection?
  - What if instead of rejecting a fixed percentage α, we simply reject (in a soft way) pixels whose *values* differ too much from the central pixel value?

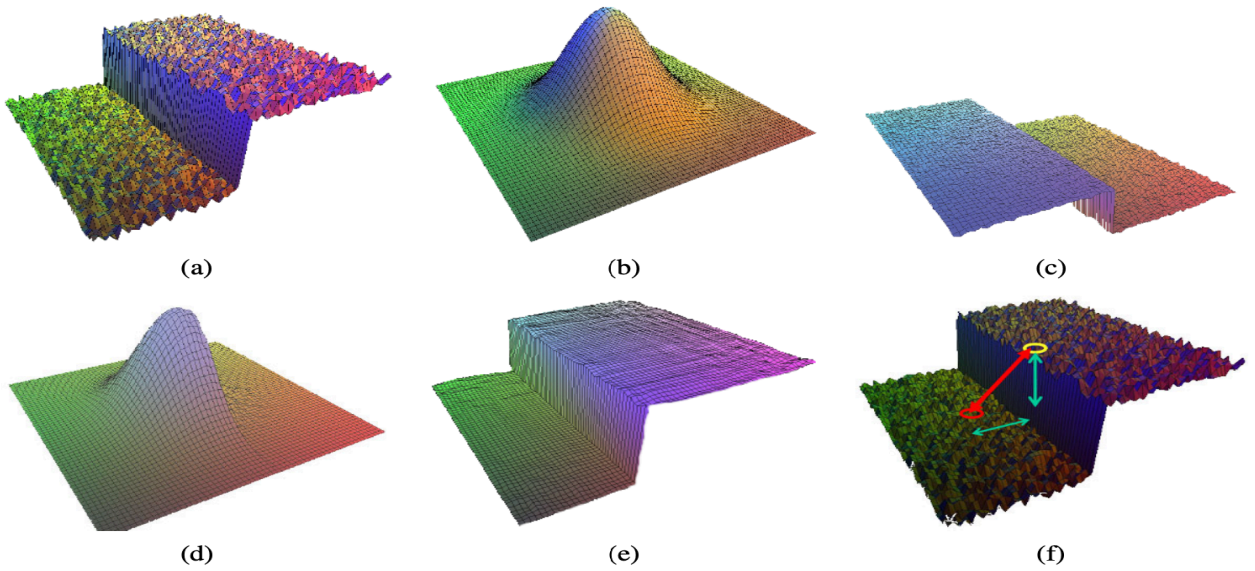
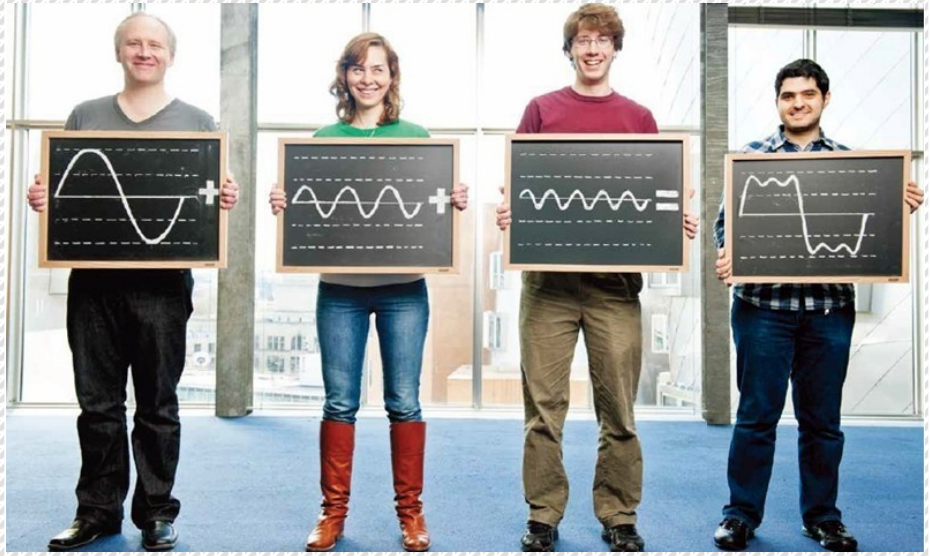
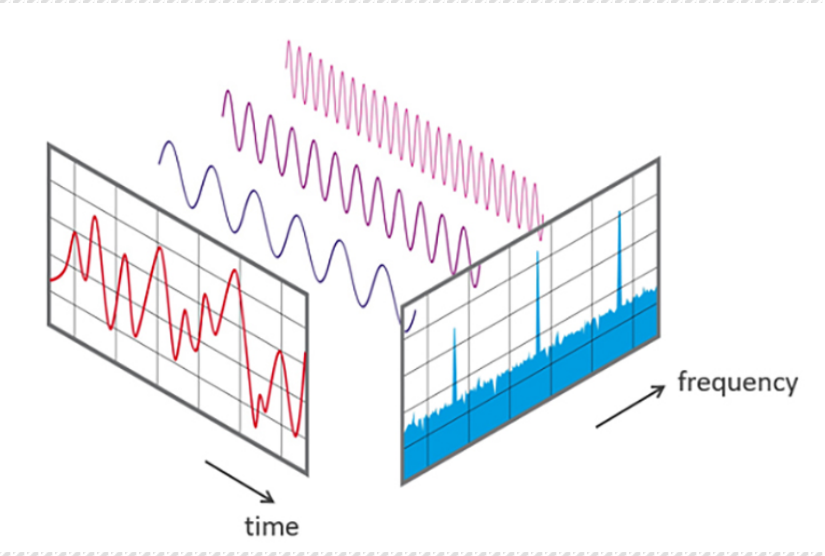


Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.

Fourier transforms

- Spatial domain → Frequency domain

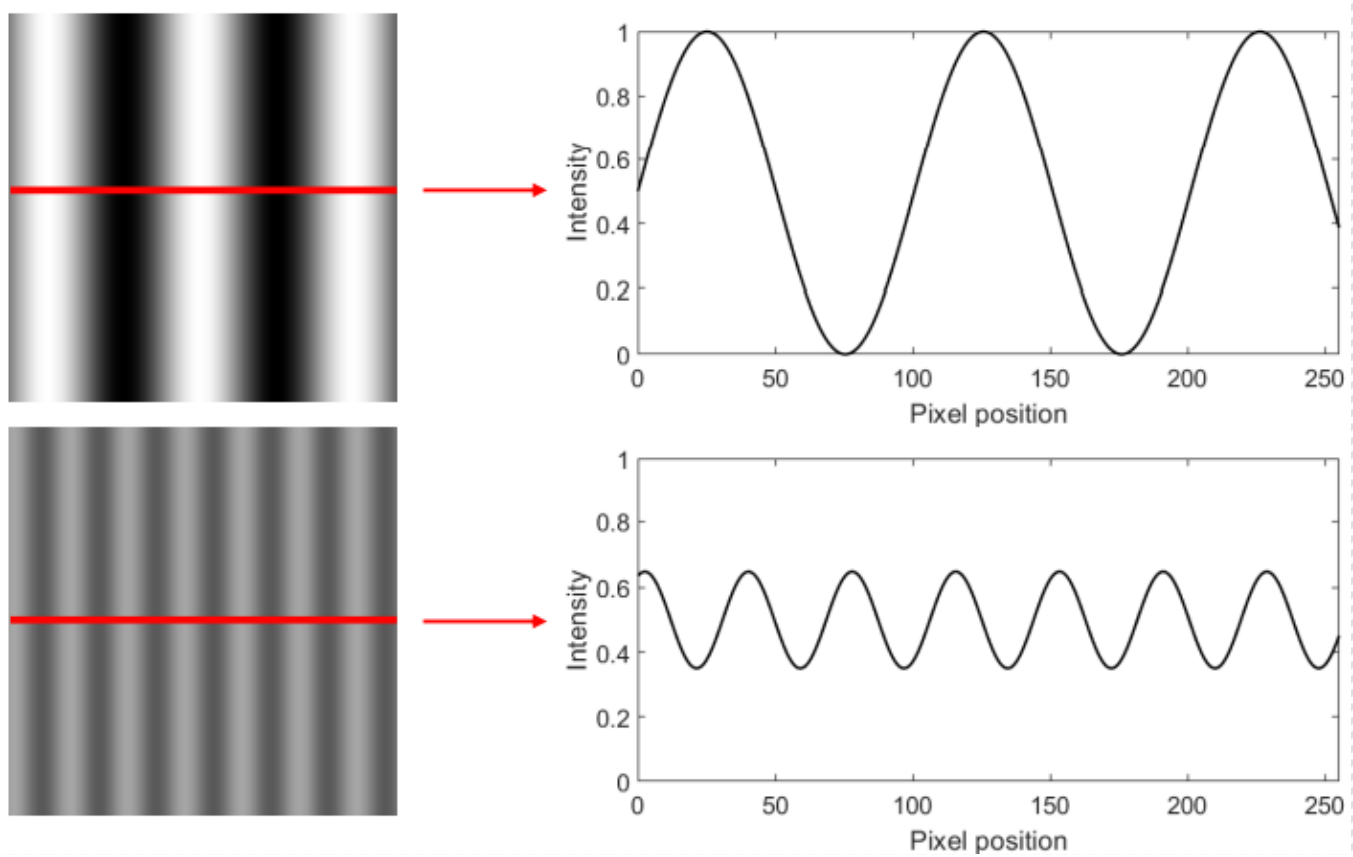




https://www.bilibili.com/video/BV1uY411z7uk/?spm_id_from=333.788.recommend_more_video.0&vd_source=1d98d45f728c64dff48447f704f66bae

Fourier transforms

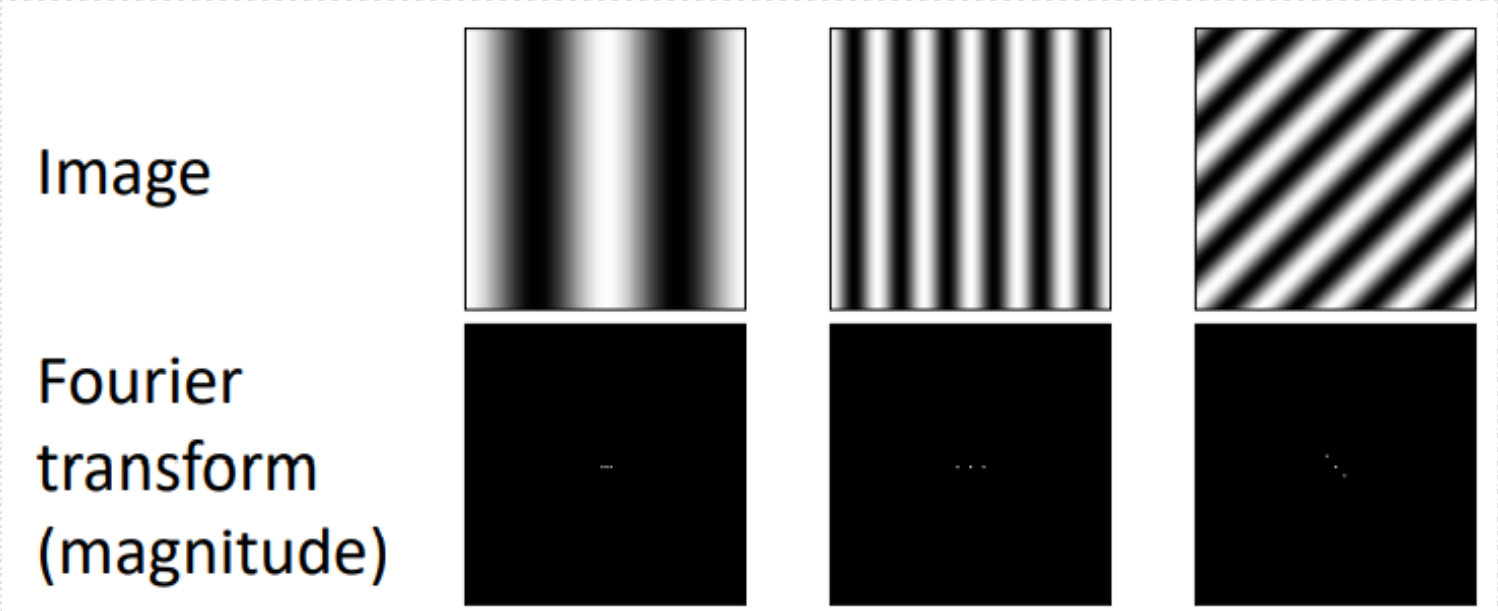
- Spatial domain → Frequency domain



Fourier transforms

- Spatial domain → Frequency domain

- 逆傅里叶变换将频域转化为空间域
- 频率域处理是先将图像变换到频率域，然后在频率域对图像进行处理，最后通过反变换将图像变为空间域。



Pyramids and wavelets

- What if we wish to change the resolution of an image before proceeding further ?
  - Case 1: Interpolate a small image to make its resolution match that of the output printer or computer screen.
  - Case 2: Reduce the size of an image to speed up the execution of an algorithm or to save on storage space or transmission time.

Pyramids and wavelets

- What if we wish to change the resolution of an image before proceeding further ?
  - We do not know the scale at which the face will appear, we need to generate a whole *pyramid* of differently sized images and scan each one for possible faces.

Pyramids and wavelets

- What if we wish to change the resolution of an image before proceeding further ?
  - Such a pyramid can also be very helpful in accelerating the search for an object by first finding a smaller instance of that object at a coarser level of the pyramid and then looking for the full resolution object only in the vicinity of coarse-level detections

Pyramids and wavelets

- What if we wish to change the resolution of an image before proceeding further ?
  - Such a pyramid can also be very helpful in accelerating the search for an object by first finding a smaller instance of that object at a coarser level of the pyramid and then looking for the full resolution object only in the vicinity of coarse-level detections.
  - Image pyramids are extremely useful for performing multi-scale editing operations such as blending images while maintaining details.

Pyramids and wavelets

- What if we wish to change the resolution of an image before proceeding further ?
  - Such a pyramid can also be very helpful in accelerating the search for an object by first finding a smaller instance of that object at a coarser level of the pyramid and then looking for the full resolution object only in the vicinity of coarse-level detections.
  - Image pyramids are extremely useful for performing multi-scale editing operations such as blending images while maintaining details.

Pyramids and wavelets

- Change image resolution by filters:
  - upsampling (*interpolation*)
  - downsampling (*decimation*).
  - multi-resolution pyramids, which can be used to create a complete hierarchy of differently sized images and to enable a variety of applications.
  - *wavelets*, which are a special kind of pyramid with higher frequency selectivity and other useful properties

Pyramids and wavelets

- Upsampling (*interpolation*)

$$g(i,j) = f(k,l)h(i - rk, j - rl).$$

r is the upsampling rate

Pyramids and wavelets

- Upsampling (*interpolation*)

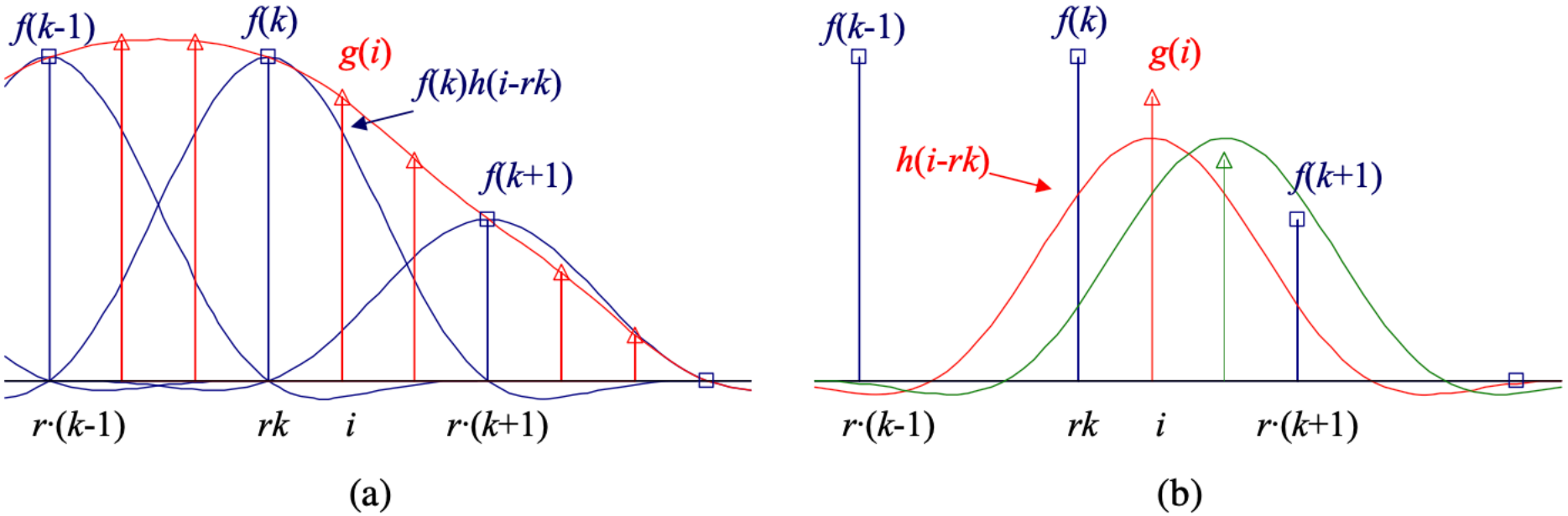


Figure 3.27 Signal interpolation, $g(i) = \sum_k f(k)h(i - rk)$: (a) weighted summation of input values; (b) polyphase filter interpretation.

Pyramids and wavelets

- Upsampling (*interpolation*)

  What kinds of kernel make good interpolators?

Name	Signal		Transform
impulse	$\delta(x)$	$\Leftrightarrow$	1
shifted impulse	$\delta(x-u)$	$\Leftrightarrow$	$e^{-j\omega u}$
box filter	$\text{box}(x/a)$	$\Leftrightarrow$	$a\text{sinc}(a\omega)$
tent	$\text{tent}(x/a)$	$\Leftrightarrow$	$a\text{sinc}^2(a\omega)$
Gaussian	$G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega;\sigma^{-1})$
Laplacian of Gaussian	$(\frac{x^2}{\sigma^4}-\frac{1}{\sigma^2})G(x;\sigma)$	$\Leftrightarrow$	$-\frac{\sqrt{2\pi}}{\sigma}\omega^2 G(\omega;\sigma^{-1})$
Gabor	$\cos(\omega_0 x)G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega \pm \omega_0;\sigma^{-1})$
unsharp mask	$(1+\gamma)\delta(x)$ $-\gamma G(x;\sigma)$	$\Leftrightarrow$	$(1+\gamma)-$ $\frac{\sqrt{2\pi}\gamma}{\sigma}G(\omega;\sigma^{-1})$
windowed sinc	$\text{rcos}(x/(aW))$ $\text{sinc}(x/a)$	$\Leftrightarrow$	(see Figure 3.29)

Pyramids and wavelets

- Upsampling (*interpolation*)

  What kinds of kernel make good interpolators?

  →depends on the application and the computation time involved.

Pyramids and wavelets

- Upsampling (*interpolation*)
  - *Linear* interpolator(corresponding to the tent kernel) produces interpolating piecewise linear curves.
  - cubic B-spline, whose discrete 1/2-pixel sampling appears as the *binomial kernel*
  - bilinear kernel
  - *bicubic* interpolation

Pyramids and wavelets

- Upsampling (*interpolation*)
  - cubic interpolant is a C1 (derivative-continuous) piecewise-cubic *spline* (the term "spline" is synonymous with "piecewise-polynomial")

$$h(x) = \begin{cases} 1 - (a+3)x^2 + (a+2)|x|^3 & \text{if } |x| < 1 \\ a(|x|-1)(|x|-2)^2 & \text{if } 1 \le |x| < 2 \\ 0 & \text{otherwise,} \end{cases}$$

a is often set to −1, but it also introduces a small amount of sharpening, which can be visually appealing. Unfortunately, this choice does not linearly interpolate straight lines (intensity ramps), so some visible ringing may occur. A better choice for large amounts of interpolation is probably a = −0.5, which produces a *quadratic reproducing* spline; it interpolates linear and quadratic functions exactly

Pyramids and wavelets

- Upsampling (*interpolation*)

Visible ringing artifact: In signal processing, particularly digital image processing, **ringing artifacts** are artifacts that appear as spurious signals near sharp transitions in a signal. Visually, they appear as bands or "ghosts" near edges; audibly, they appear as "echos" near transients, particularly sounds from percussion_instruments; most noticeable are the pre-echos.

Image showing ringing artifacts. 3 levels on each side of transition: overshoot, first ring, and (faint) second ring.

Same image without ringing artifacts.

Pyramids and wavelets

- Upsar

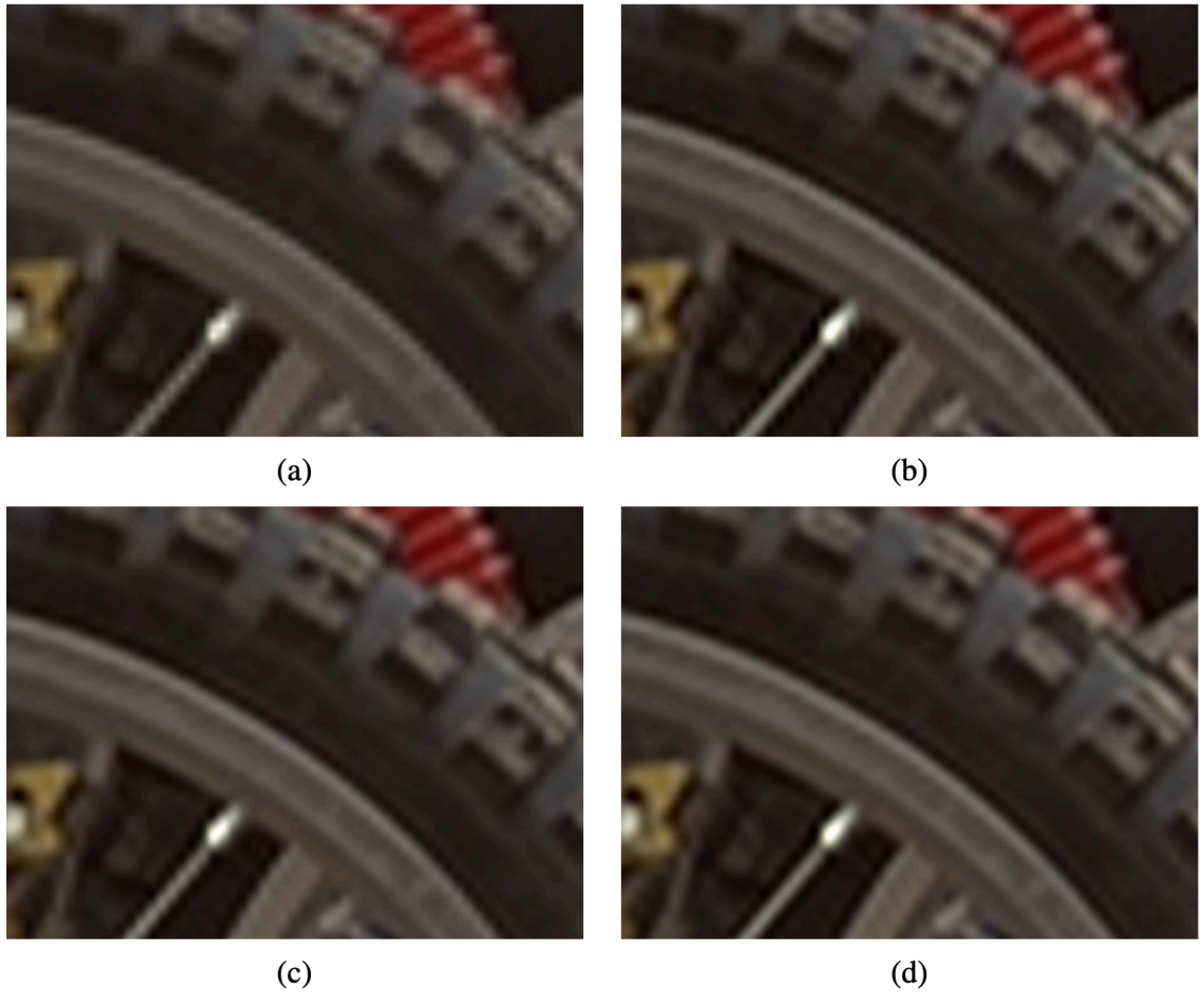


Figure 3.28 Two-dimensional image interpolation: (a) bilinear; (b) bicubic ($a = -1$); (c) bicubic ($a = -0.5$); (d) windowed sinc (nine taps).

Pyramids and wavelets

- Decimation (*downsmpling*)

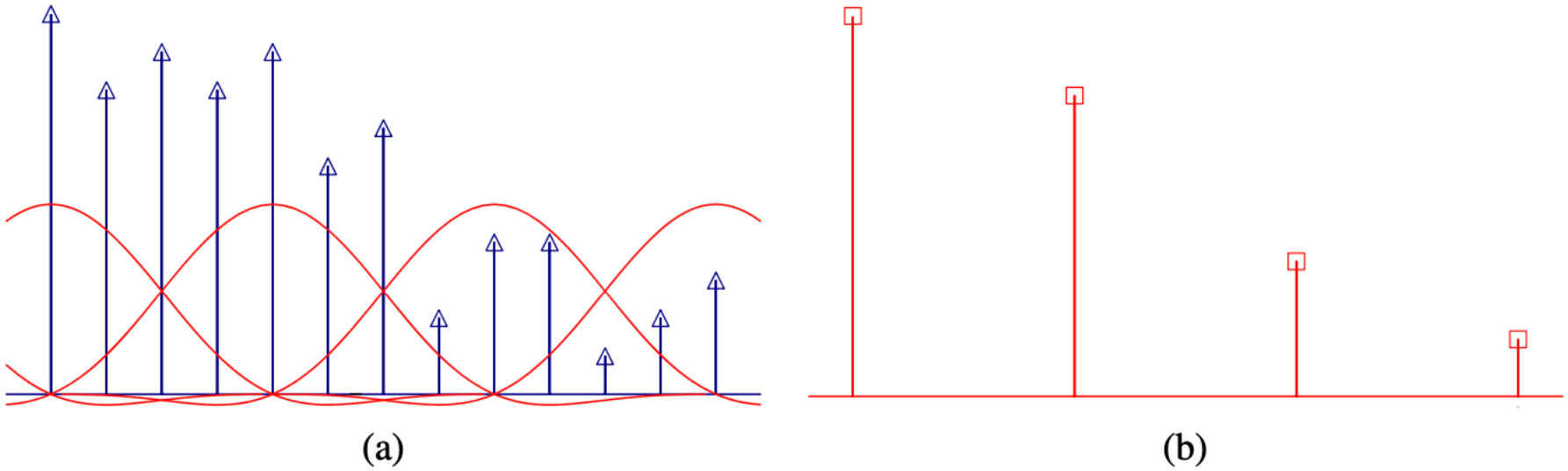


Figure 3.30 Signal decimation: (a) the original samples are (b) convolved with a low-pass filter before being downsampled.

Pyramids and wavelets

- Multi-resolution representations：
  - pyramids can be used to accelerate coarse-to-fine search algorithms, to look for objects or patterns at different scales, and to perform multi-resolution blending operations.

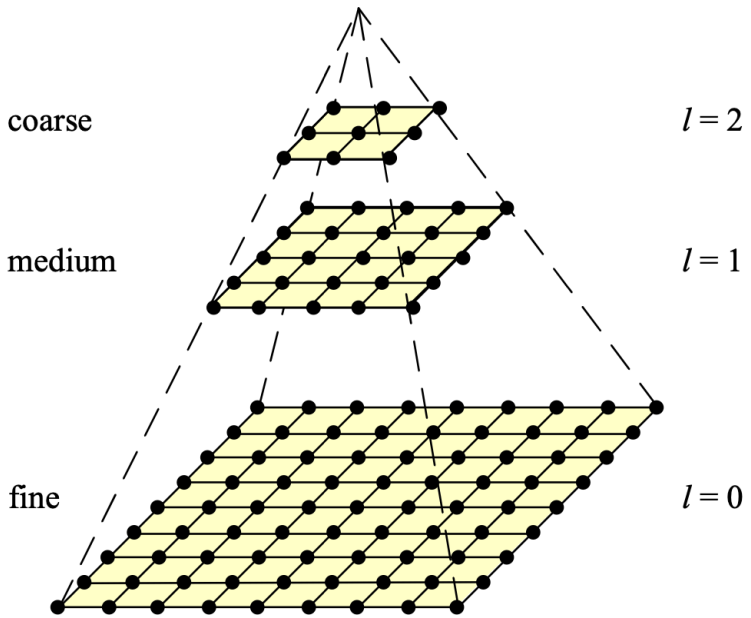


Figure 3.32 A traditional image pyramid: each level has half the resolution (width and height), and hence a quarter of the pixels, of its parent level.

Pyramids ar

- Multi-resolution
  - Laplacian pyram
    and store this in

factor of two

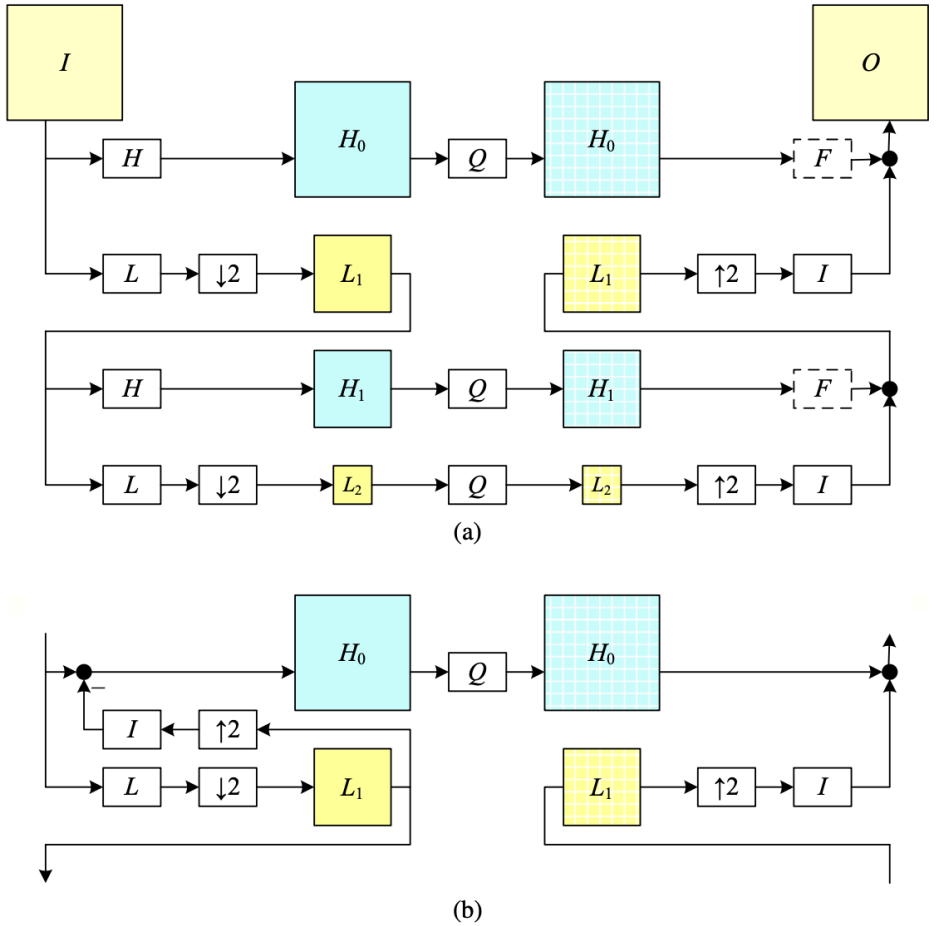


Figure 3.34 The Laplacian pyramid: (a) The conceptual flow of images through processing stages: images are high-pass and low-pass filtered, and the low-pass filtered images are processed in the next stage of the pyramid. During reconstruction, the interpolated image and the (optionally filtered) high-pass image are added back together. The Q box indicates quantization or some other pyramid processing, e.g., noise removal by *coring* (setting small wavelet values to 0). (b) The actual computation of the high-pass filter involves first interpolating the downsampled low-pass image and then subtracting it. This results in perfect reconstruction when Q is the identity. The high-pass (or band-pass) images are typically called *Laplacian* images, while the low-pass images are called *Gaussian* images.

Pyramids and wavelets

- Wavelet decompositions as an alternative for Pyramids：
  - Wavelets are filters that localize a signal in both space and frequency and are defined over a hierarchy of scales.

Pyramids and wavelets

- Wavelet decompositions as an alternative for Pyramids：
  - Wavelets are filters that localize a signal in both space and frequency and are defined over a hierarchy of scales.
- But what are the difference between these two filters?

Pyramids and wavelets

- Wavelet decompositions as an alternative for Pyramids：
  - Wavelets are filters that localize a signal in both space and frequency and are defined over a hierarchy of scales.
- But what are the difference between these two filters?

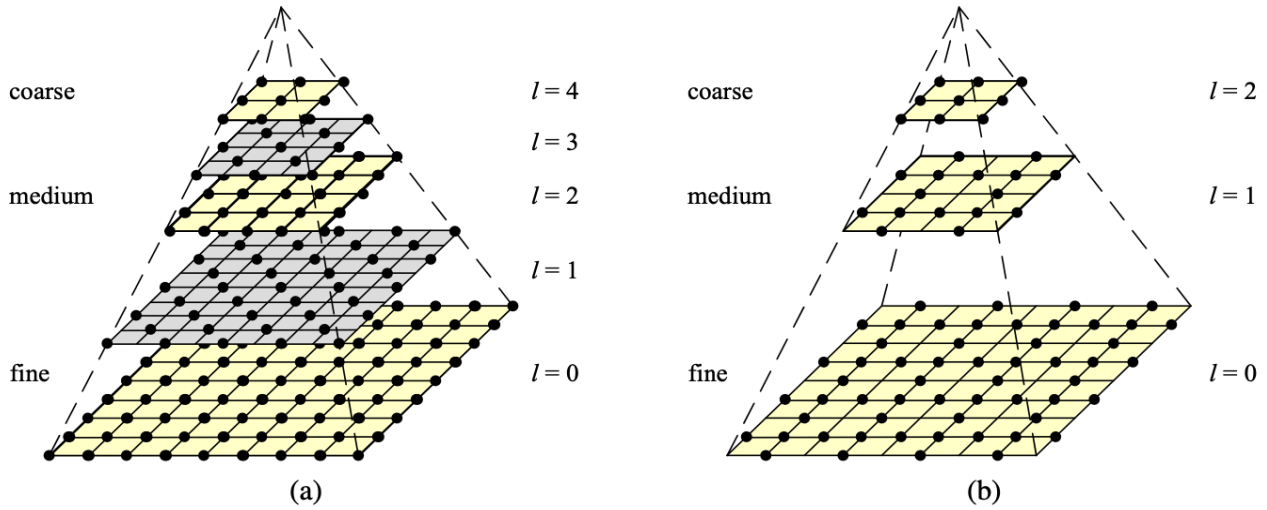


Figure 3.36 Multiresolution pyramids: (a) pyramid with half-octave (*quincunx*) sampling (odd levels are colored gray for clarity). (b) wavelet pyramid—each wavelet level stores 3/4 of the original pixels (usually the horizontal, vertical, and mixed gradients), so that the total number of wavelet coefficients and original pixels is the same.

Pyramids and wavelets

- Wavelet decompositions as an alternative for Pyramids：
  - Wavelets are filters that localize a signal in both space and frequency and are defined over a hierarchy of scales.
- But what are the difference between these two filters?
  - The usual answer is that traditional pyramids are *overcomplete*, i.e., they use more pixels than the original image to represent the decomposition, whereas wavelets provide a *tight frame*, i.e., they keep the size of the decomposition the same as the image

Pyramids and wavelets

- How are two-dimensional wavelets constructed?

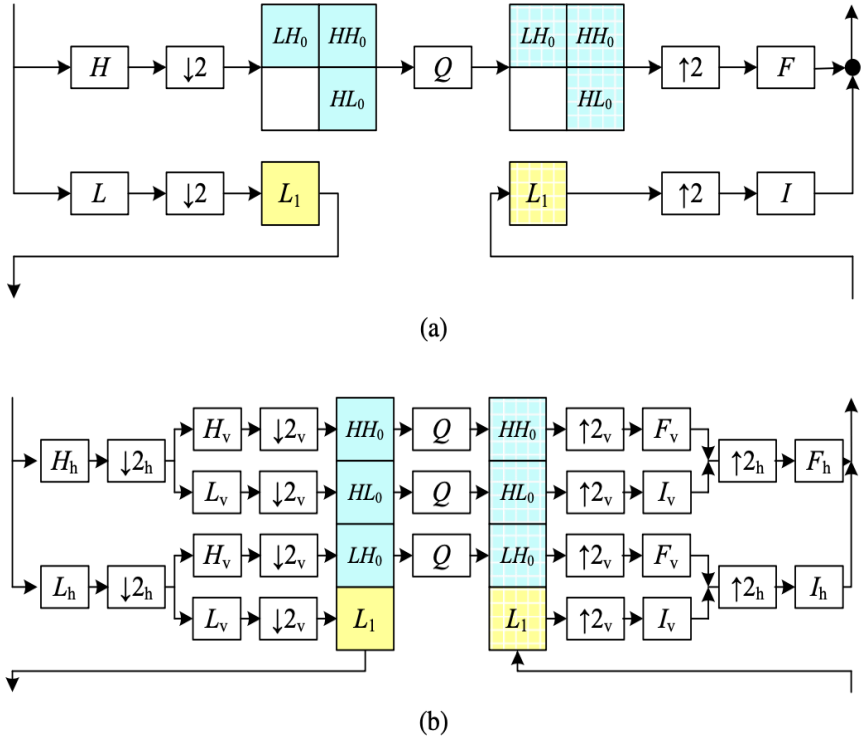


Figure 3.37 Two-dimensional wavelet decomposition: (a) high-level diagram showing the low-pass and high-pass transforms as single boxes; (b) separable implementation, which involves first performing the wavelet transform horizontally and then vertically. The I and F boxes are the interpolation and filtering boxes required to re-synthesize the image from its wavelet components.

High-level diagram of one stage of the (recursive) coarse-to-fine construction (analysis) pipeline alongside the complementary re-construction (synthesis) stage

Pyramids and wavelets

- A “separable” approach to wavelet construction → steerable pyramids（导航金字塔）

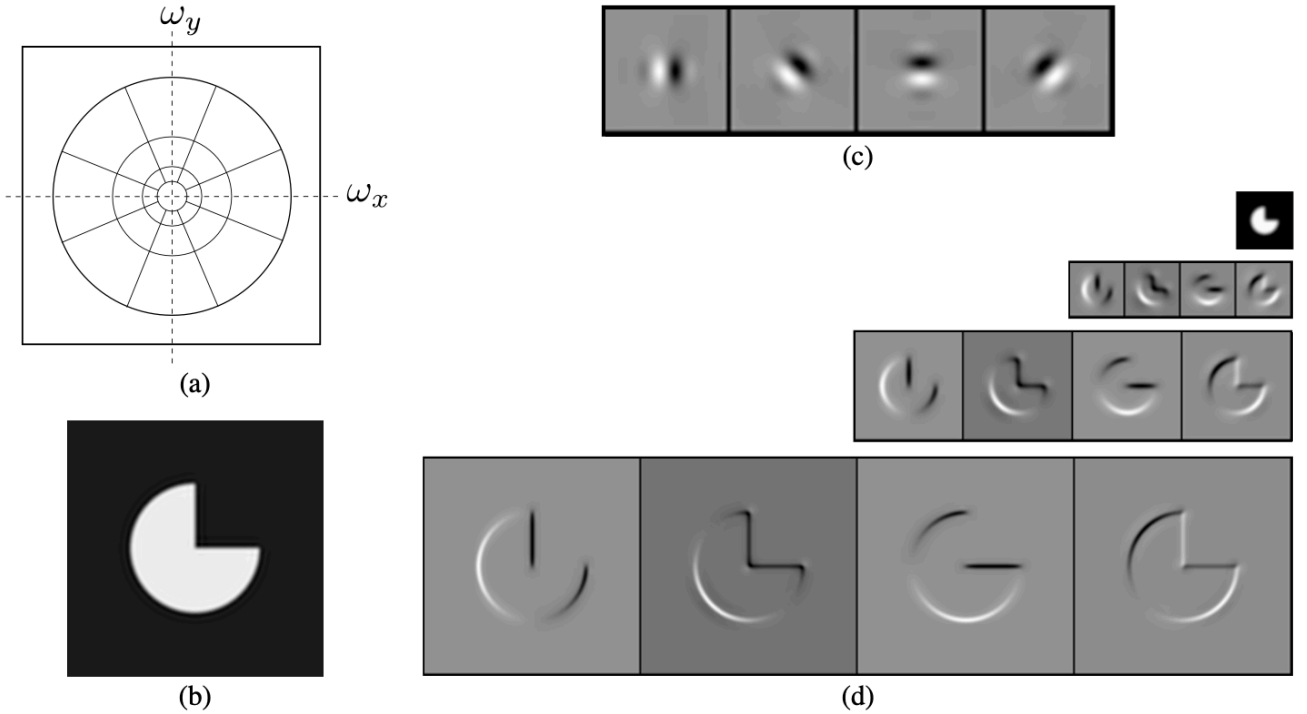
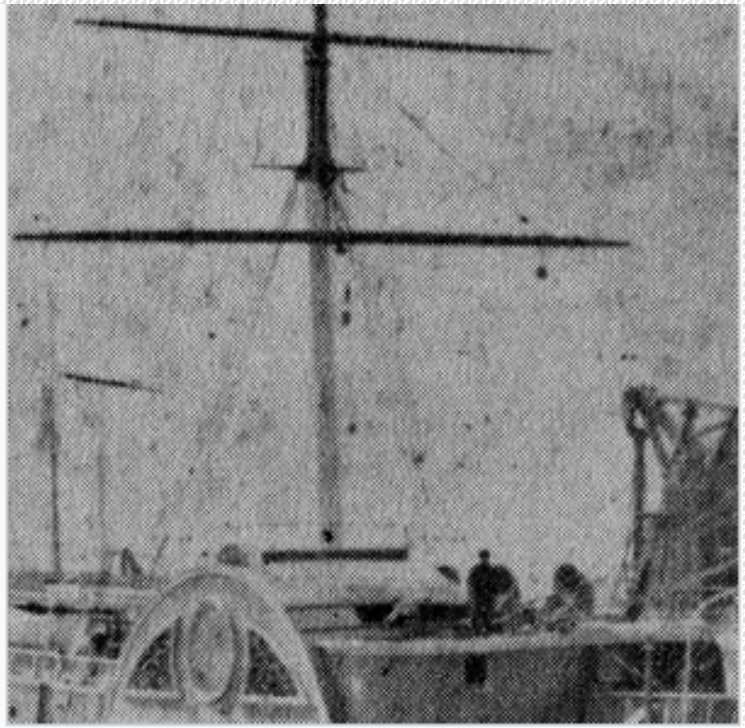


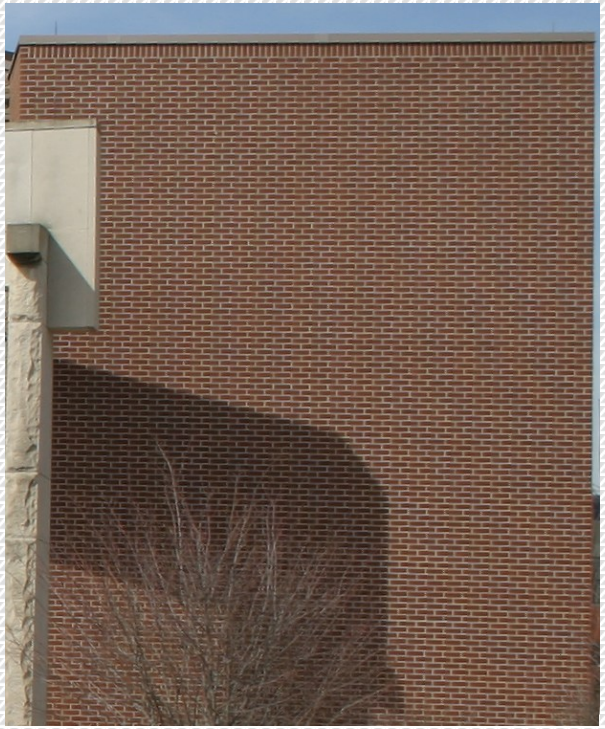
Figure 3.40 Steerable shiftable multiscale transforms (Simoncelli, Freeman, Adelson *et al.* 1992) © 1992 IEEE: (a) radial multi-scale frequency domain decomposition; (b) original image; (c) a set of four steerable filters; (d) the radial multi-scale wavelet decomposition.

Pyramids and wavelets

- A “separable” approach to wavelet construction → steerable pyramids（导航金字塔）→ avoid aliasing problem (混叠问题)



Dots in the sky due to spatial aliasing caused by halftone resized to a lower resolution

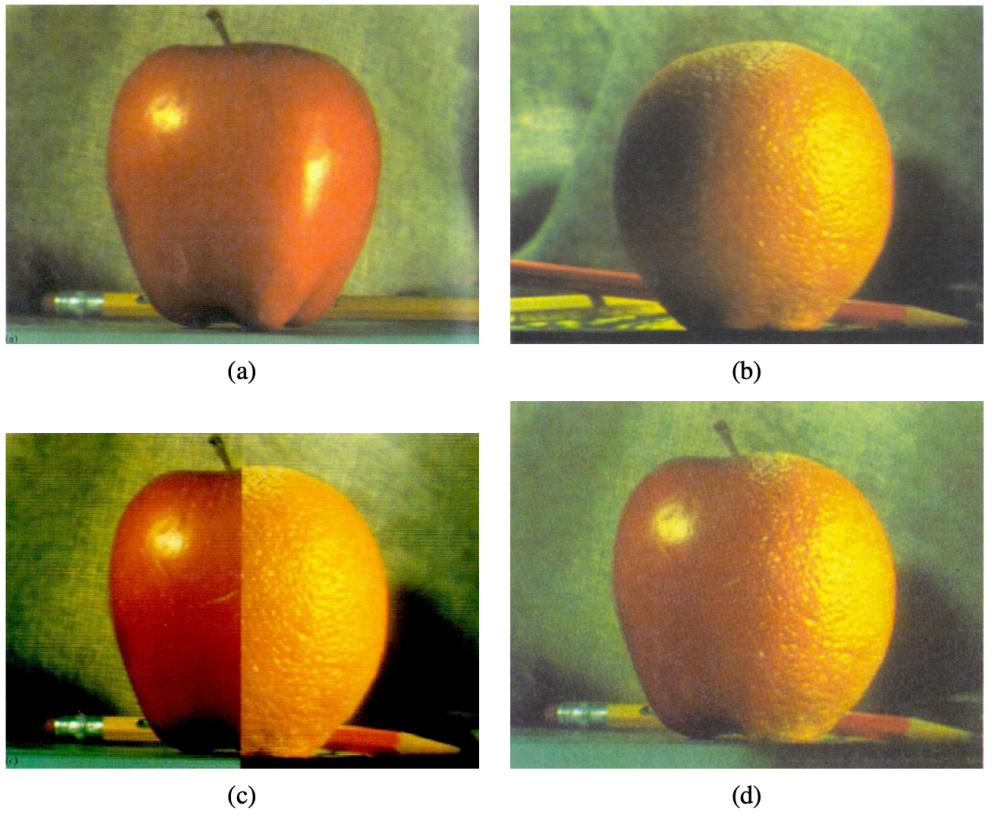


This full-sized image shows what a properly sampled image of a brick wall should look like with a screen of sufficient resolution.

When the resolution is reduced, aliasing appears in the form of a moiré pattern.

Pyramids and wavelets

- Application: Image blending



(a) (b) (c) (d)

Figure 3.41 Laplacian pyramid blending (Burt and Adelson 1983b) © 1983 ACM: (a) original image of apple, (b) original image of orange, (c) regular splice, (d) pyramid blend.

Pyramids and wavelets

- Application: Image blending

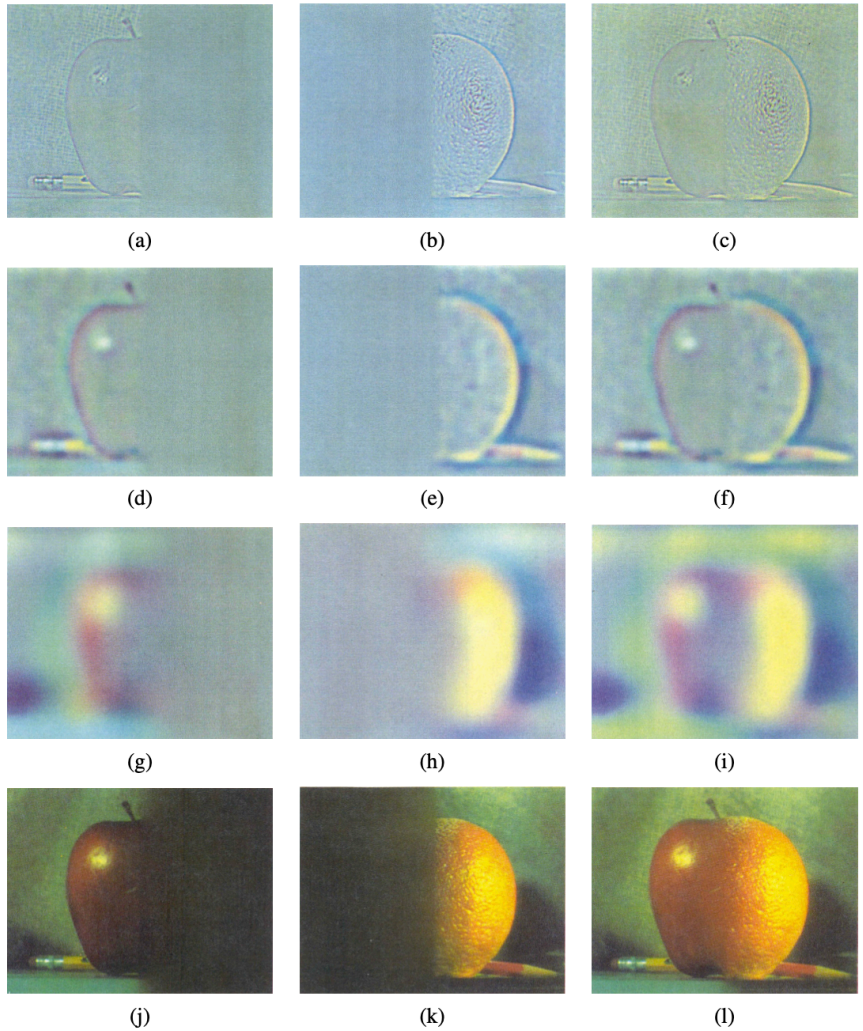


Figure 3.42 Laplacian pyramid blending details (Burt and Adelson 1983b) © 1983 ACM. The first three rows show the high, medium, and low frequency parts of the Laplacian pyramid (taken from levels 0, 2, and 4). The left and middle columns show the original apple and orange images weighted by the smooth interpolation functions, while the right column shows the averaged contributions.

Methods of image transform

- *point* operators or *point processes:* the simplest kind of image transforms, namely those that manipulate each pixel independently of its neighbors.
- *neighborhood* (area-based) operators: each new pixel's value depends on a small number of neighboring input values.
- **global operators:** *geometric transformations,* such as rotations, shears, and perspective deformations.
- *global optimization:* involve the minimization of an energy functional or, equivalently, optimal estimation using Bayesian *Markov random field* models.

Geometric transformations

- We look at functions that transform the *domain:*

$$g(x) = f(h(x))$$

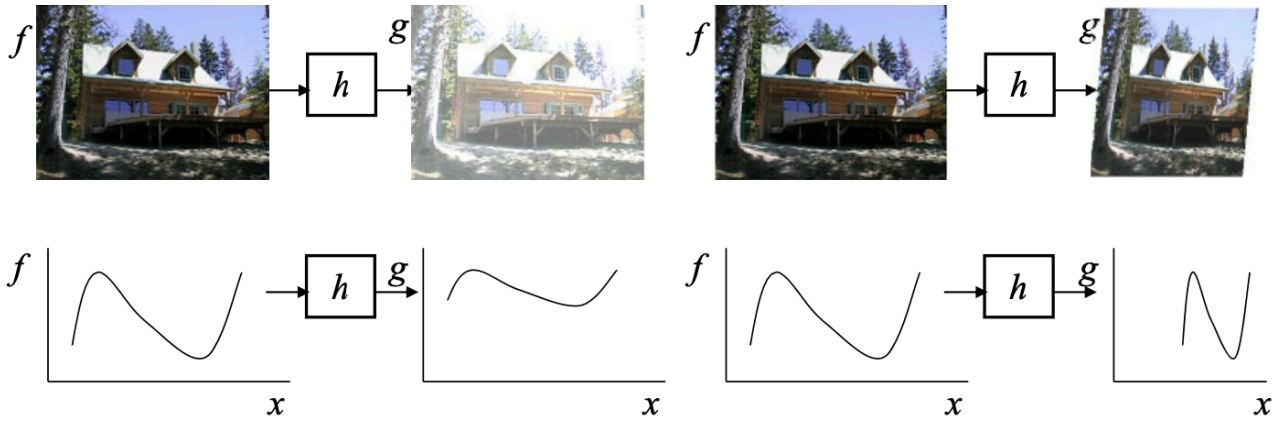


Figure 3.44 Image warping involves modifying the *domain* of an image function rather than its *range*.

Geometric transformations

- We look at functions that transform the *domain:*
  - global *parametric* 2D transformation（全局参数二维变换）
  - local general deformations
    - *morphs* (in-between animations) (图像变形)

Geometric transformations

- Global *parametric* 2D transformation :

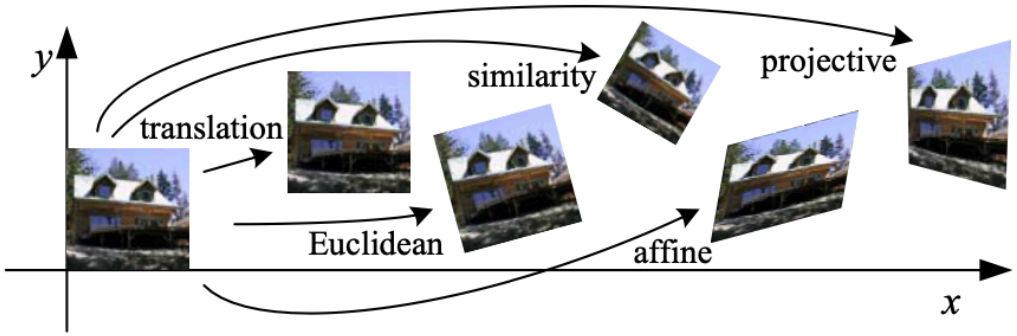


Figure 3.45 Basic set of 2D geometric image transformations.

Geometric transformations

- Global *parametric* 2D transformation :

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	2	orientation	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	3	lengths	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	4	angles	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times3}$	6	parallelism	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times3}$	8	straight lines	

Table 3.5 Hierarchy of 2D coordinate transformations. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The 2×3 matrices are extended with a third $[\mathbf{0}^T\ 1]$ row to form a full 3×3 matrix for homogeneous coordinate transformations.

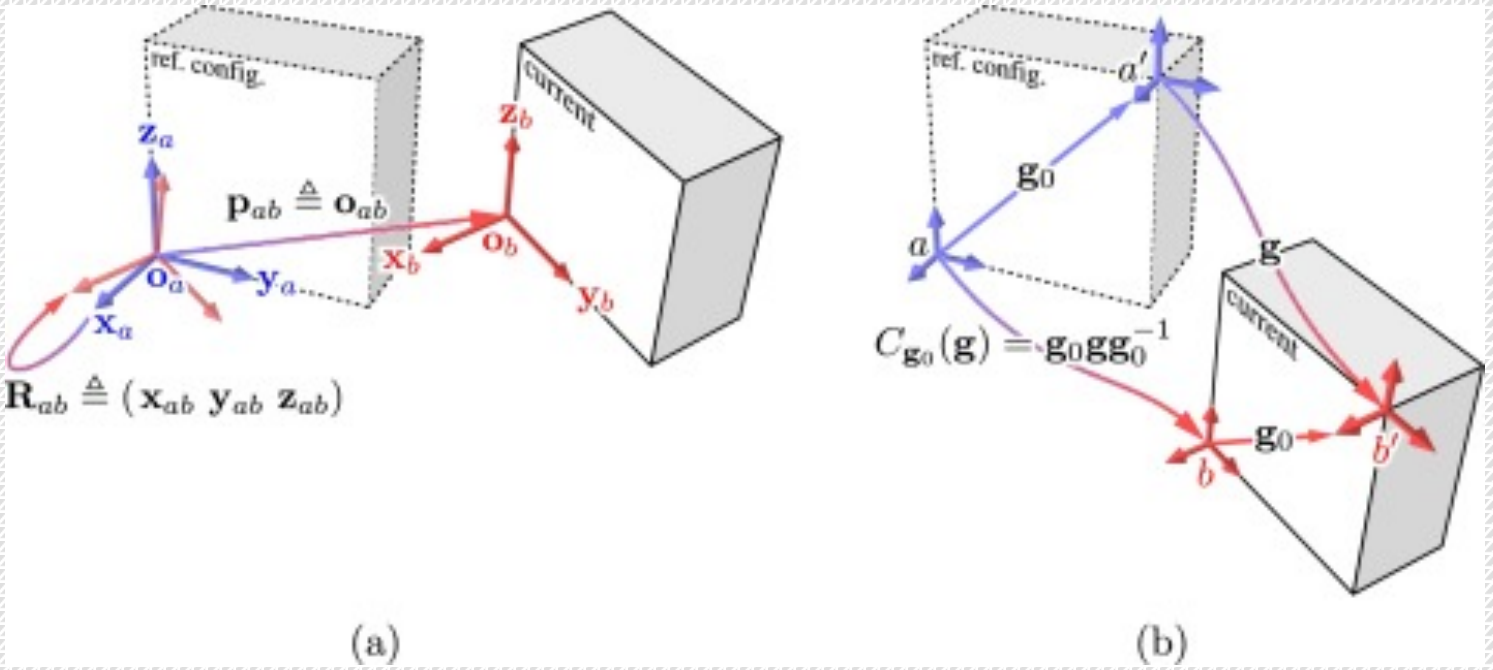
Geometric transformations

- Global *parametric* 2D transformation : Image translation （平移转换）

Translates a source image into a target image while preserving certain visual properties of the original image.

Geometric transformations

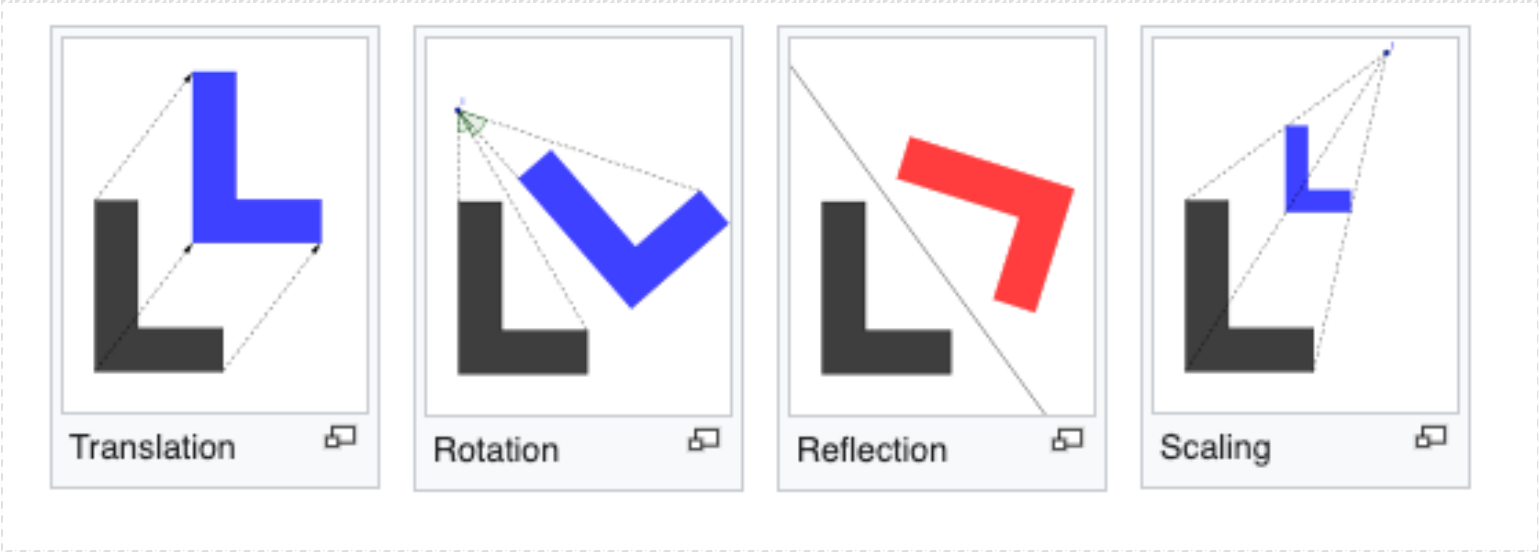
- Global *parametric* 2D transformation : Rigid transformation (Euclidean)（刚体变换）

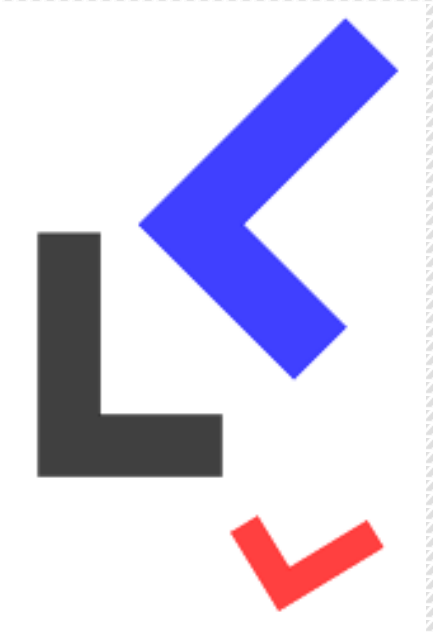


A rigid transformation (also called Euclidean transformation or Euclidean isometry) is a geometric transformation of a Euclidean space that preserves the Euclidean distance between every pair of points

Geometric transformations

- Global *parametric* 2D transformation : Similarity transformation

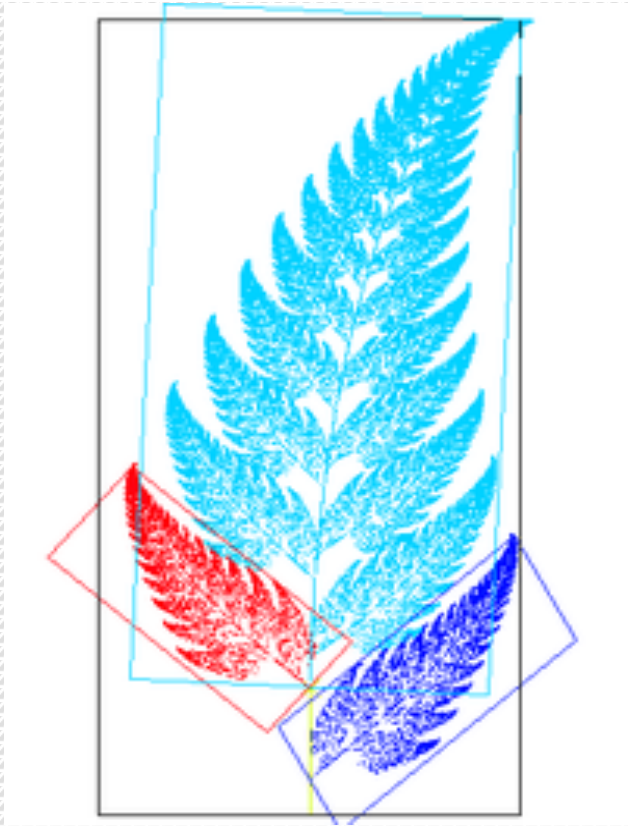




Similar figures

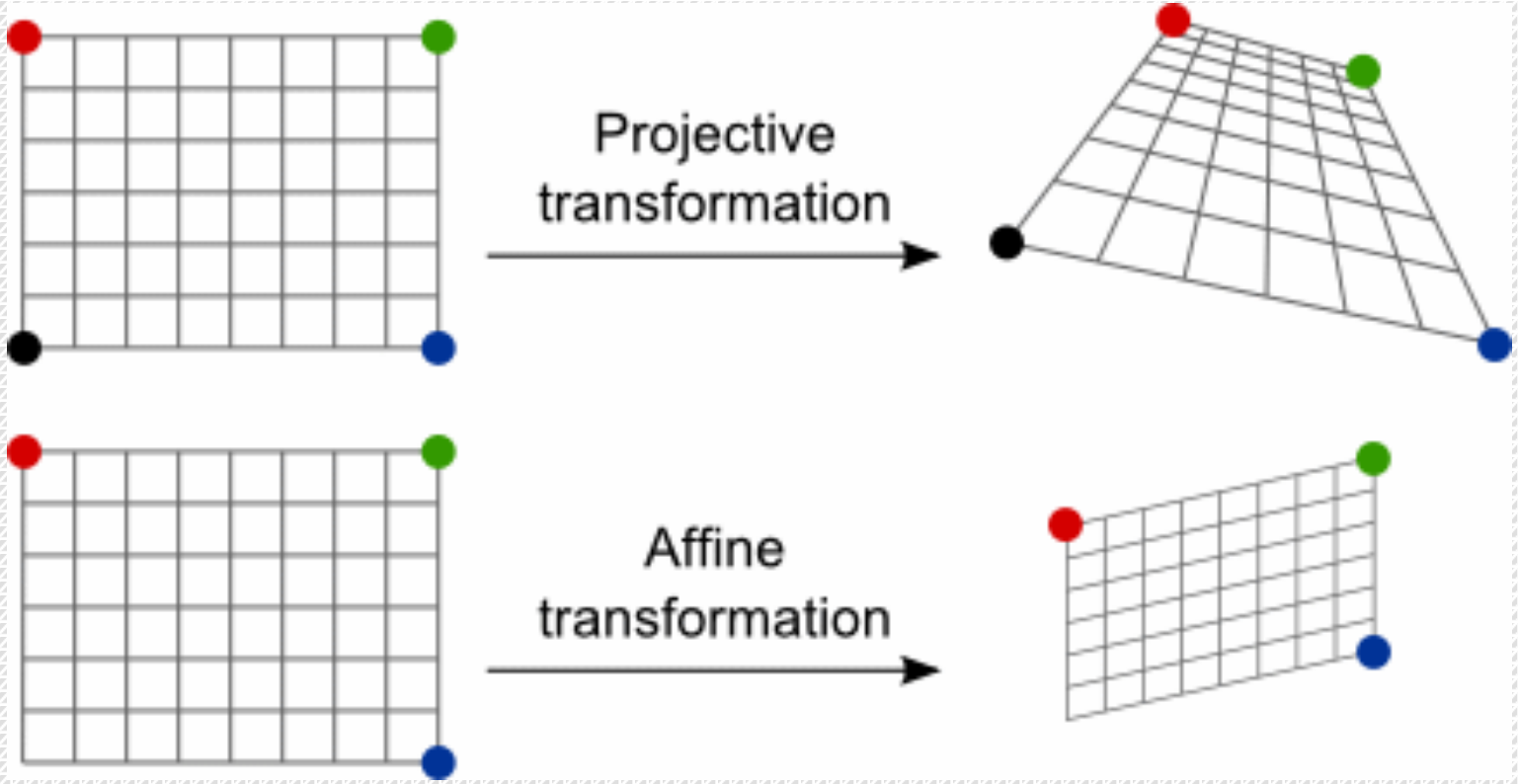
Geometric transformations

- Global *parametric* 2D transformation : Affine transformation（ 仿射变换 ）
  - A linear mapping method that preserves points, straight lines, and planes



Geometric transformations

- Global *parametric* 2D transformation : projective transformation（射影变换）

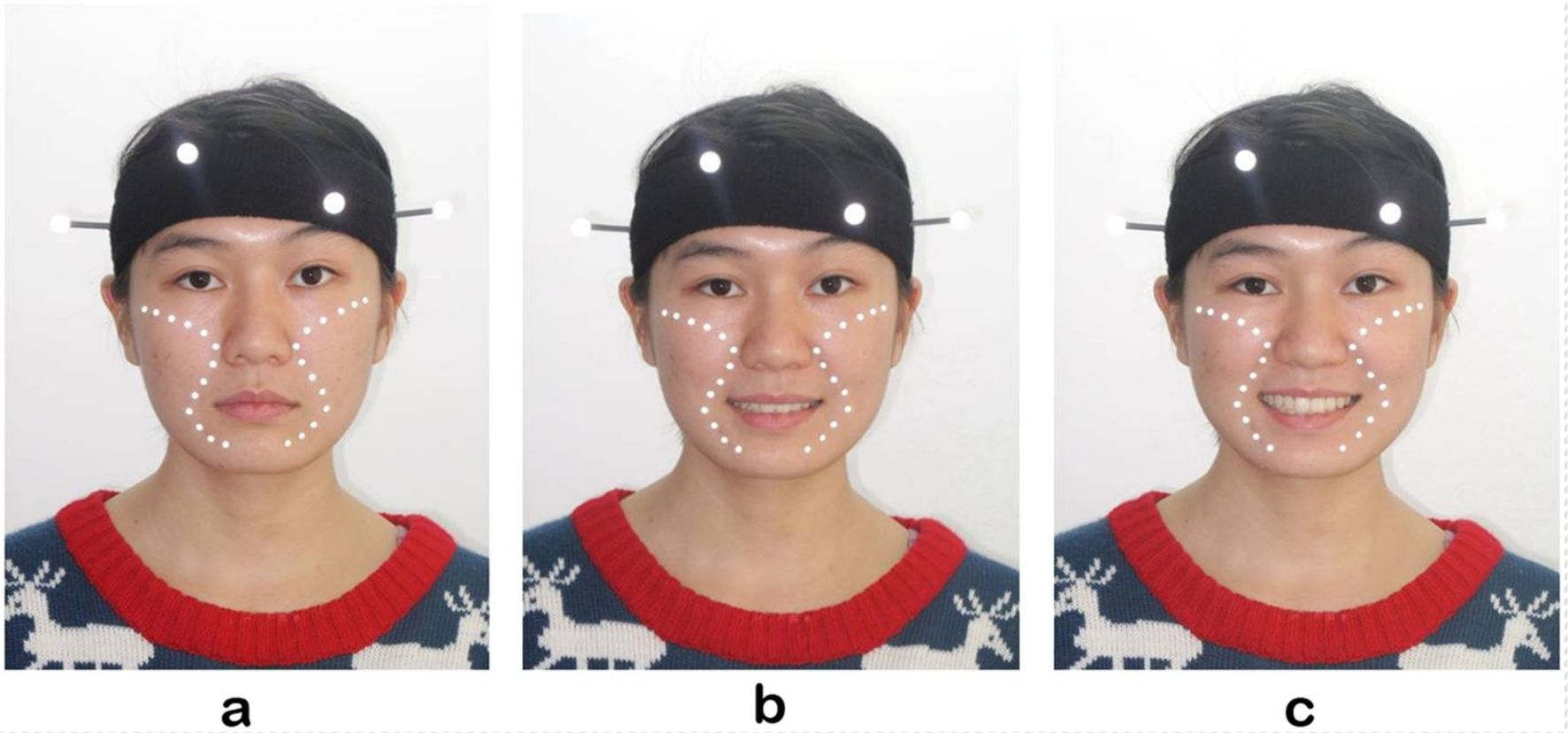


Geometric transformations

- We look at functions that transform the *domain:*
  - global *parametric* 2D transformation
  - local general deformations
    - *morphs* (in-between animations) (图像变形)

Geometric transformations

- local general deformations：Mesh-based warping（基于网格的扭曲方法)
  - If you only want to change the appearance of a face from a frown to smile?



Geometric transformations

- local general deformations：Mesh-based warping (基于网格的扭曲方法)

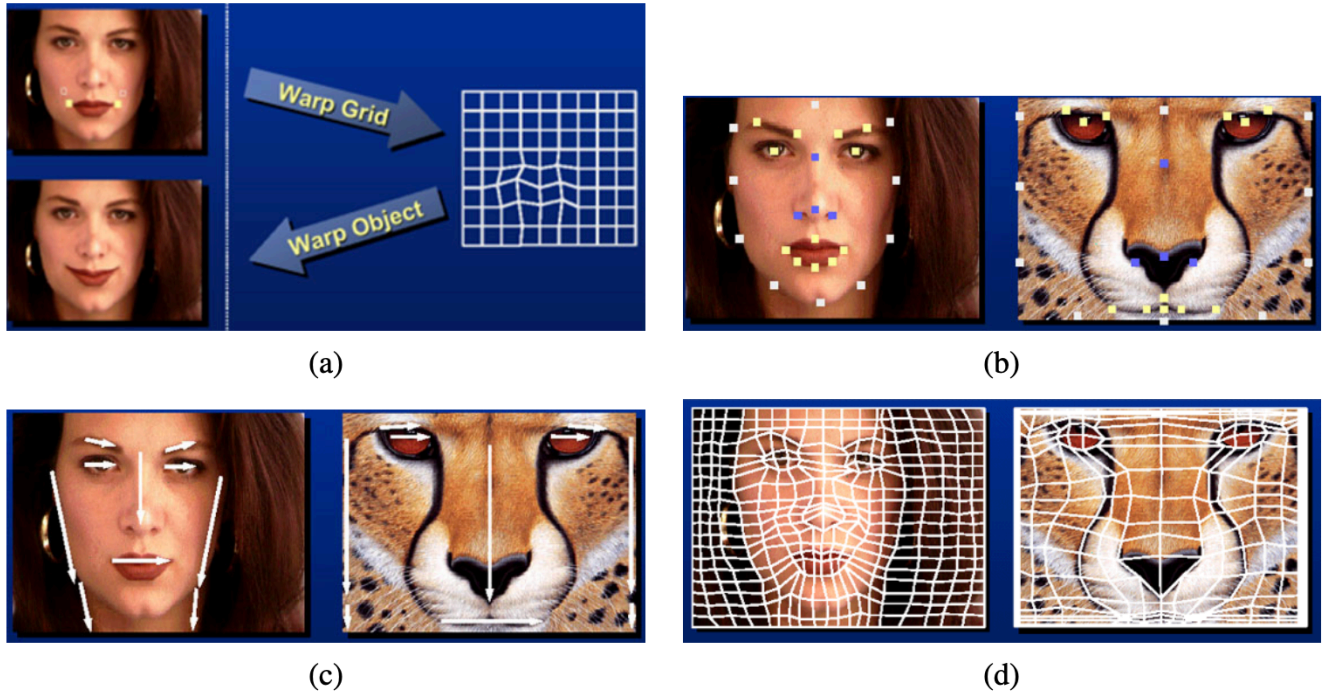


Figure 3.51 Image warping alternatives (Gomes, Darsa, Costa *et al.* 1999) © 1999 Morgan Kaufmann: (a) sparse control points ⟶ deformation grid; (b) denser set of control point correspondences; (c) oriented line correspondences; (d) uniform quadrilateral grid.

Methods of image transform

- *point* operators or *point processes:* the simplest kind of image transforms, namely those that manipulate each pixel independently of its neighbors.
- *neighborhood* (area-based) operators: each new pixel's value depends on a small number of neighboring input values.
- *global operators:* *geometric transformations,* such as rotations, shears, and perspective deformations.
- ***global optimization:*** involve the minimization of an energy functional or, equivalently, optimal estimation using Bayesian *Markov random field* models. **(Homework, Read Page 152 - 171)**

Talk from the top scientist

李开复：AI 如何拯救人性