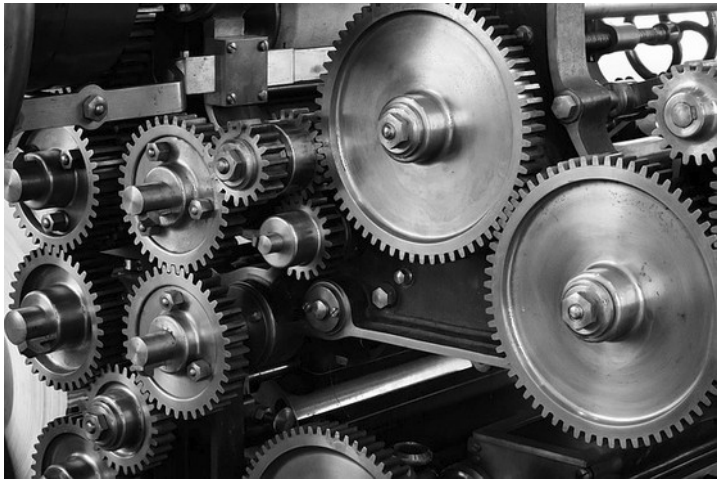


Processing

Last time ...

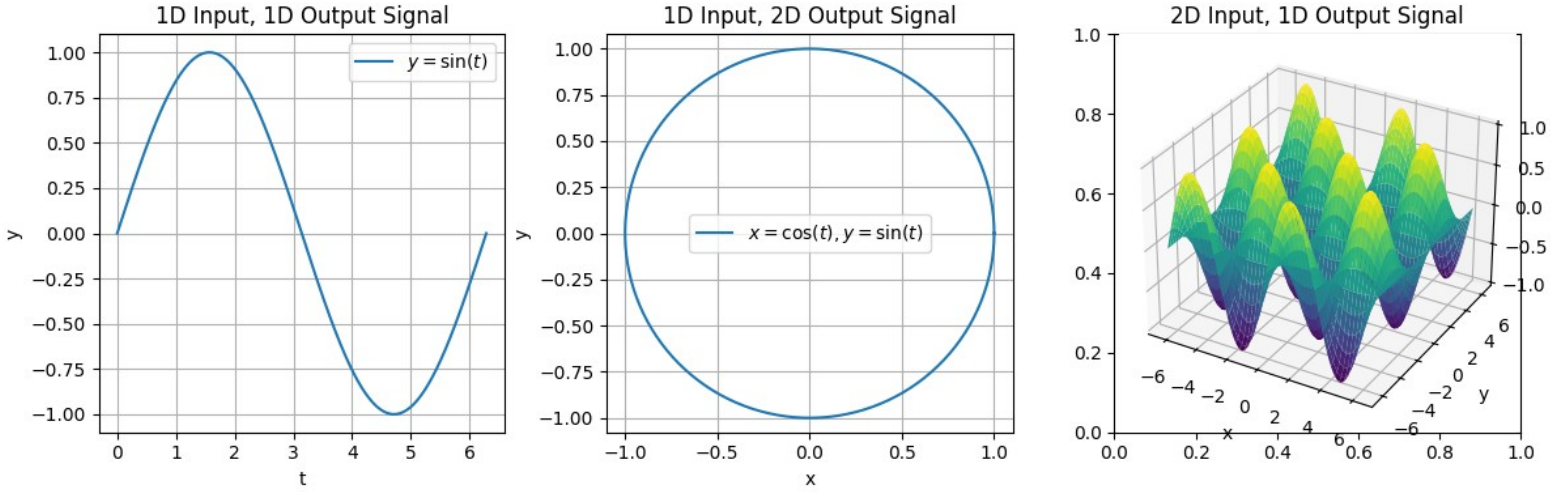
- Introduction to Multimedia Systems
  - Bridge between user and sensory data
- Human perception
  - Reality is subjective
  - Focus on sight and hearing
- Multiple ways of representing data
  - Different abstractions

What is a signal?

- Function that conveys information
  - Audio, video, speech, image, sonar, radar
  - Information content (entropy)
- Universal concept in multimedia
  - Representation of observable pheonomena
  - Transmission, storage

Functions

- Mathematical representation
  - One/more input variables (domain)
  - Output variables (codomain)

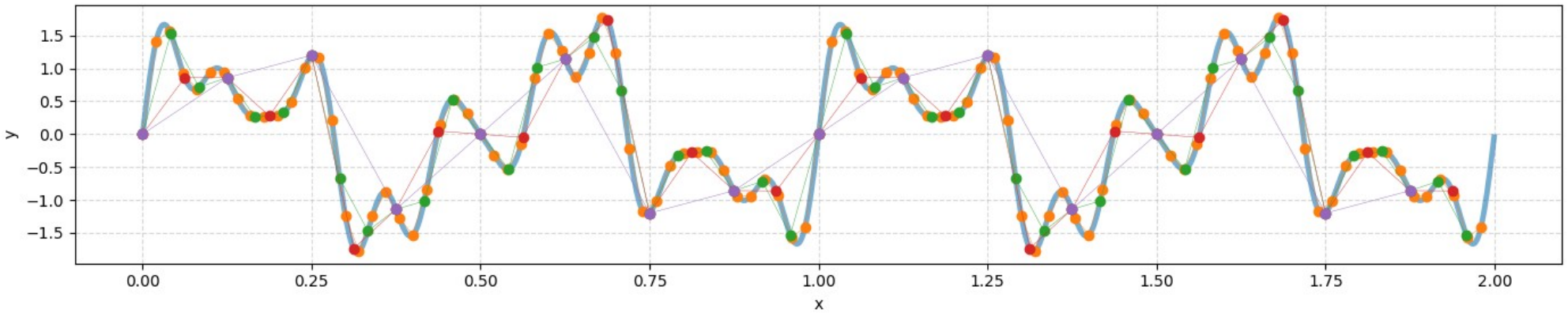


Analog Signal

- Signals in nature are analog
  - Electrical, hydraulic, pneumatic, mechanical
  - Continious input and/or output
- Digital devices require digital signal
  - Sampling (input)
  - Quantization (output)

Continious and Discrete Signals

- Sampling: measure at regular intervals
- Resolution trade-off
  - Finer sampling/quantization = better quality, larger data
  - Nyquist theorem (aliasing)



Approximating Values

- Quantization: round to a finite set of values
  - Available range
  - Resolution
- Introduces noise

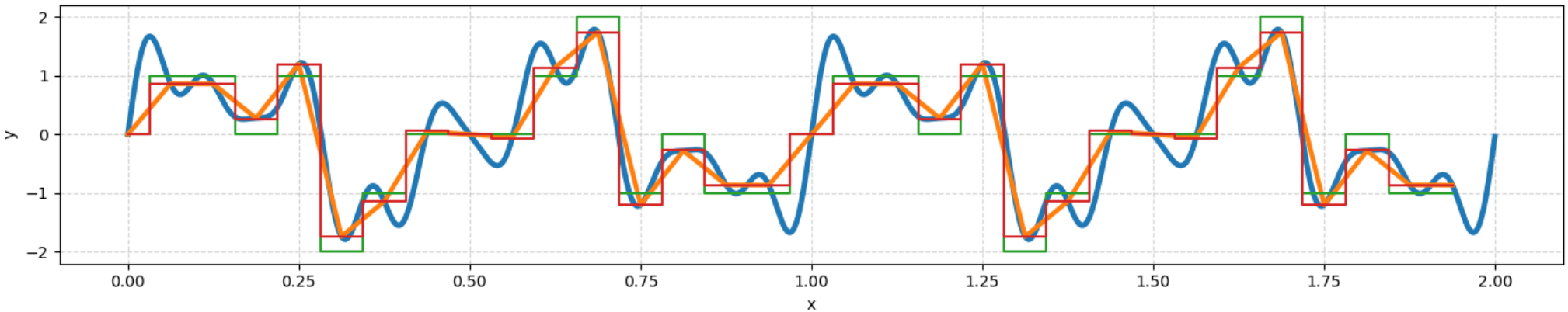
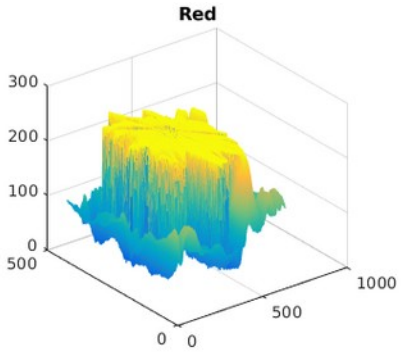
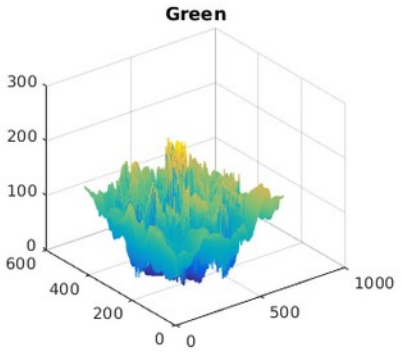
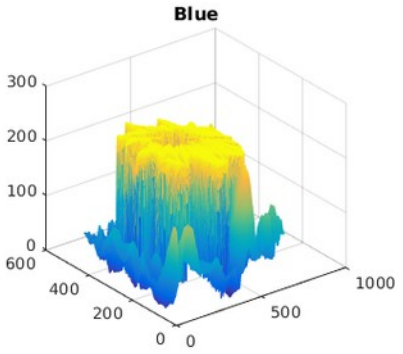


Image as a Function

- Brightness or color signal $f : \mathcal{R}^2 \rightarrow \mathcal{R}(\mathcal{R}^3)$
- A function of a 2D location
  - Defined over a region (rectangle) $x \in [0, W], y \in [0, H]$
  - Has a finite (limited) output range $f(x, y) \in [0, N]$
- Analog / digital image



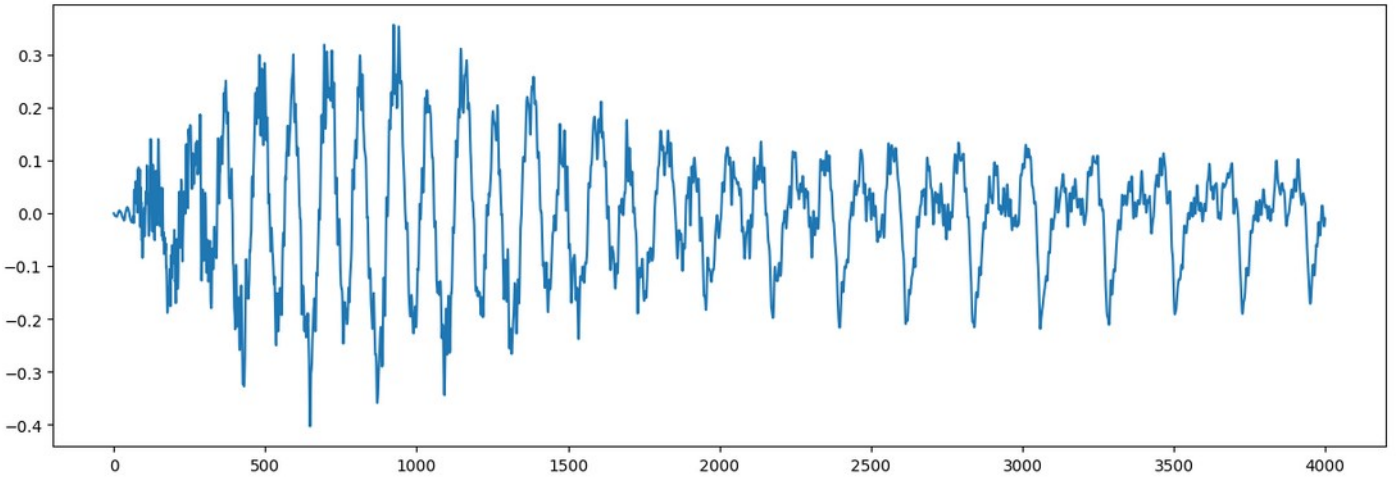




Audio as a Function

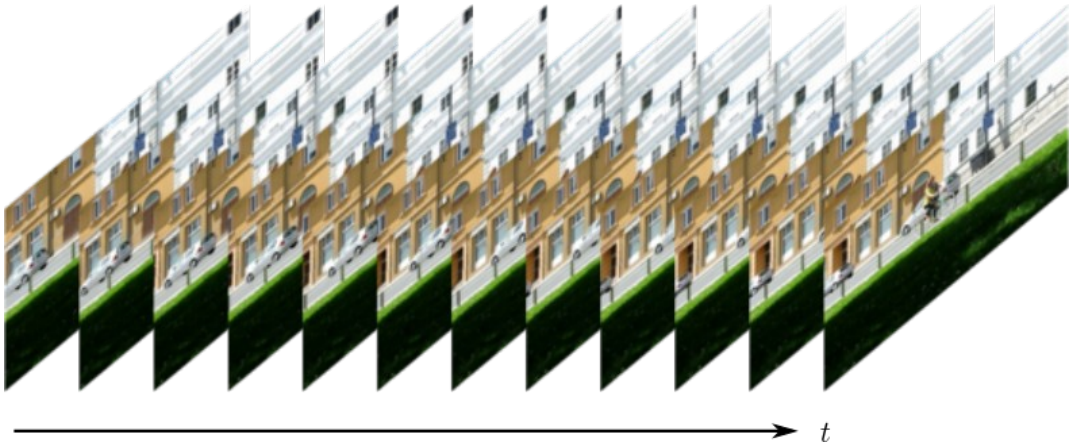
- Pressure/voltage changes over time
- Only change is meanigful
- Multiple measurements – spatial context

$$f : \mathcal{R} \rightarrow \mathcal{R}$$



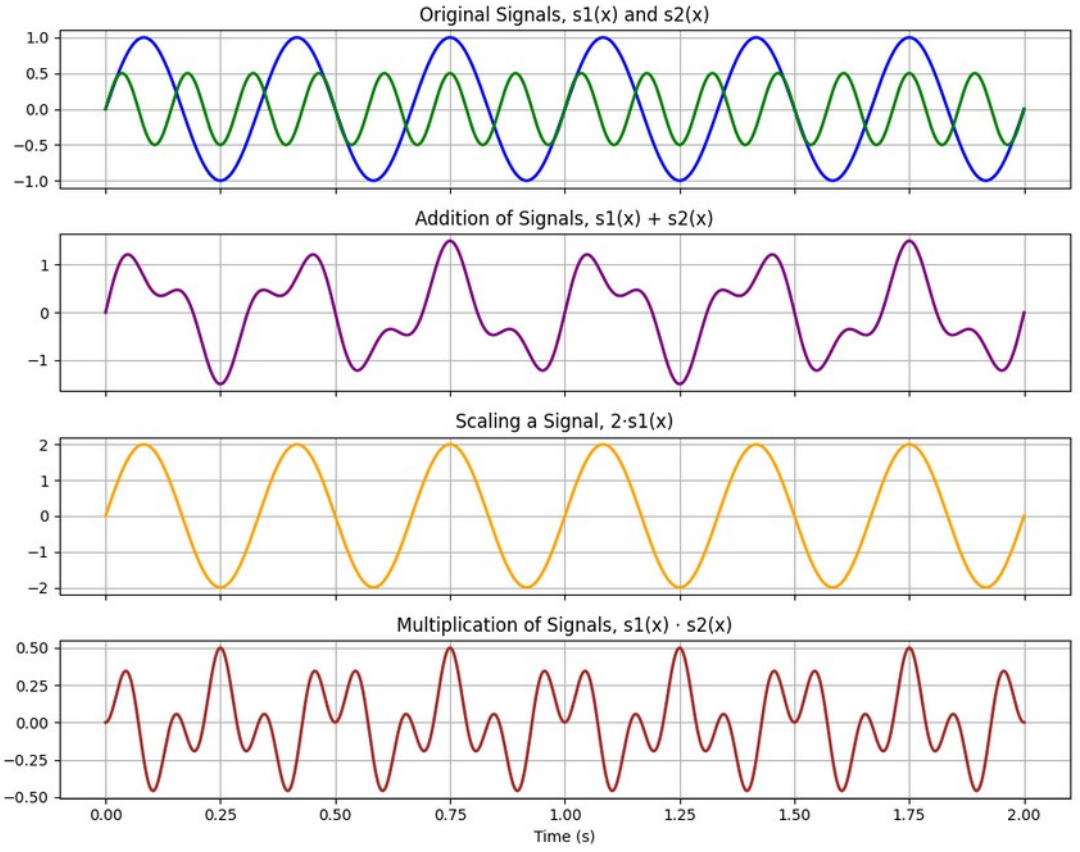
Video as a Function

- Sequence of mutiple images
  - Extension to time domain
  - Frame rate
- Temporal correlation



Signal Arithmetic

- Addition
  - Superposition
  - Mixing of sources
- Scaling
  - Amplification/attenuation
  - Gain control in audio
  - Brightness/contrast in images
- Multiplication
  - Modulation
  - Applying masks/windows



Filters

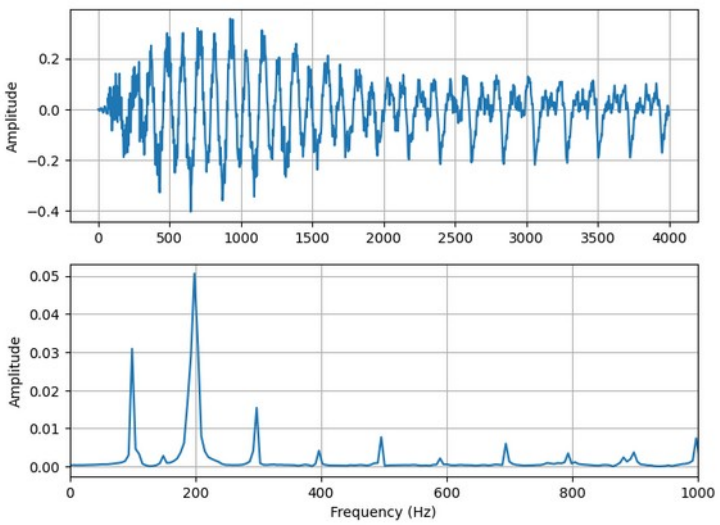
- Memoryless Filters
  - Output at time T depends only on the input at T
  - Image processing: point (or pixel-wise)
- Memory Filters
  - Use context (other data points)
  - Convolution

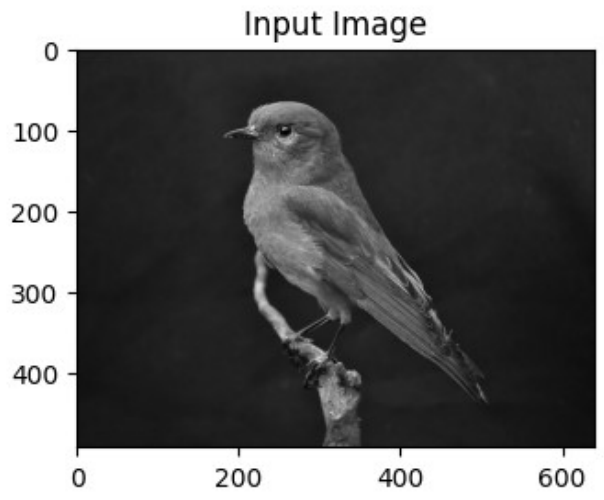
Time and Frequency Domains

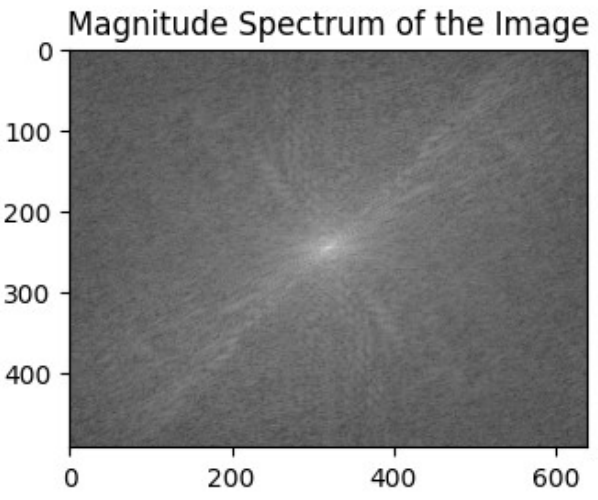
- Time/space domain
  - How values change directly
- Frequency domain
  - What kinds of oscillations are present
  - Low frequencies = smooth variations
  - High frequencies = rapid changes

Frequency spectrum

- Linear combination of basis functions
  - Sinusoidal (sine and cosine) – repeatability
  - Coefficients – presence of individual basis functions

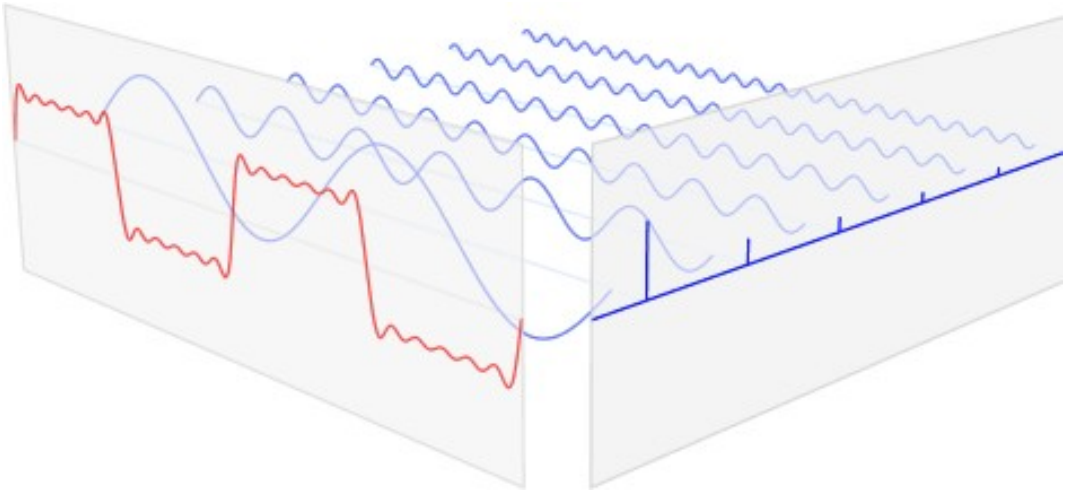






Fourier analysis

- General functions represented/approximated by sums of simpler trigonometric functions
- Decomposing signal into base sine waves
  - Frequency distribution
  - Fourier transform
  - Inverse transform



Discrete Fourier Transform

- N point signal described with N coefficients

$$\{x_0, x_1, \ldots, x_{N-1}\} \rightarrow \{X_0, X_1, \ldots, X_{N-1}\} \qquad X_k = \sum_{n=0}^{N-1} x_n \cdot [\cos(2\pi kn/N) - i \cdot \sin(2\pi kn/N)]$$

- Fast implementation (FFT) $O(n^2) \rightarrow O(n \log n)$

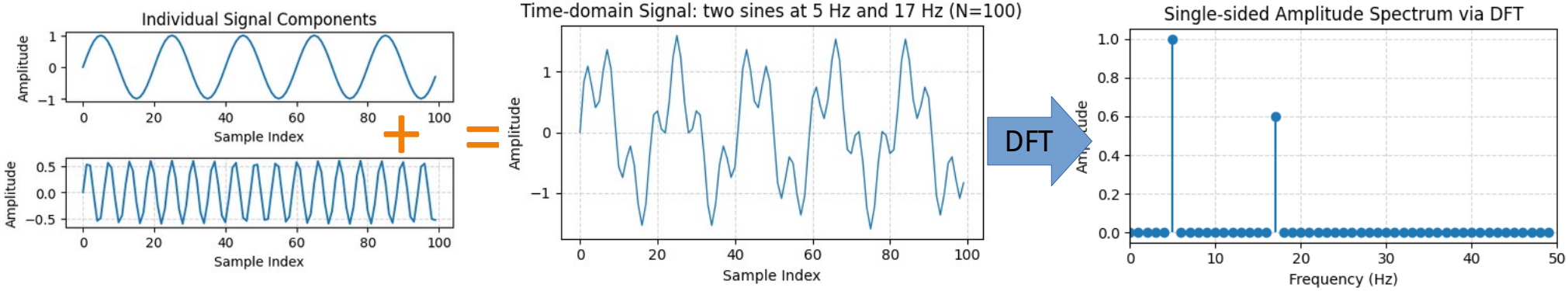


Image Processing

About Image Processing

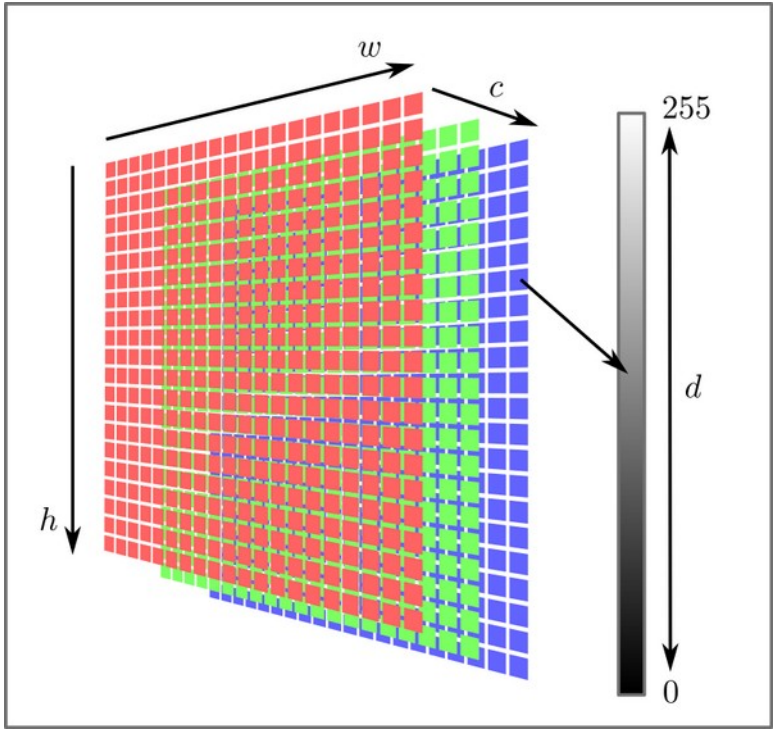
- Broad field that includes low-level operations as well as complex high-level algorithms
  - Low-level image processing
  - Computer vision
  - Computational photography
- Several procedures and concepts of that are frequently used in the context of multimedia systems
- Can also be applies to videos (frame by frame)

Image operations

- Pixel-wise transformations
  - Color spaces
  - Linear Filters
  - Non-linear Filters
- Geometric transformations
  - Linear
  - Non-linear
- Complex operations
  - Context-aware resizing
  - Compositing

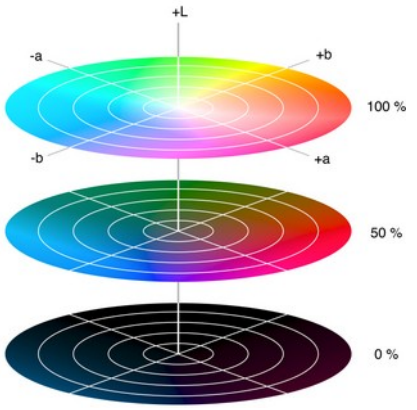
Image as a matrix

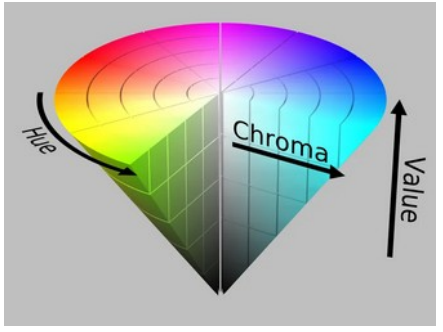
- 2D array (width x height)
- Channels
  - RGB = 3 channels
  - Sampling resolution
- Dimension ordering
  - HxWxC
  - WxHxC
  - CxHxW

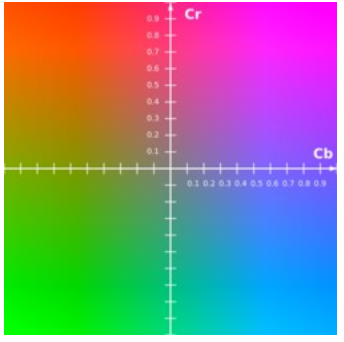


Other color spaces

- Colors can be represented in different ways
  - **RGB** – hardware, reproduction
  - **CIE Lab** – perceptually uniform
  - **YCbCr** – compression
  - **HSV** – human intuition, HCI





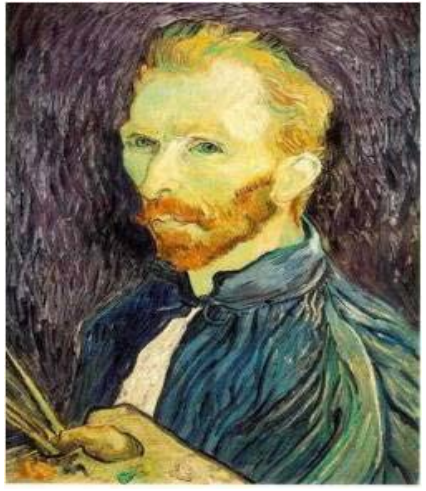


Chromatic Separation

- Matches the characteristics of human vision
  - **Compression efficiency –** reduce the resolution of chromatic channels
  - **Image processing –** avoid unnatural color shifts (contrast adjustment, histogram equalization, filtering)
  - **Device/medium independence** - perceptually uniform colorspaces

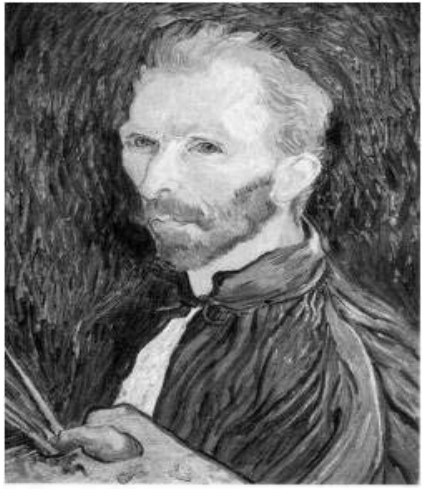
Conversion to grayscale

- From RGB: (weighted) averaging of channels



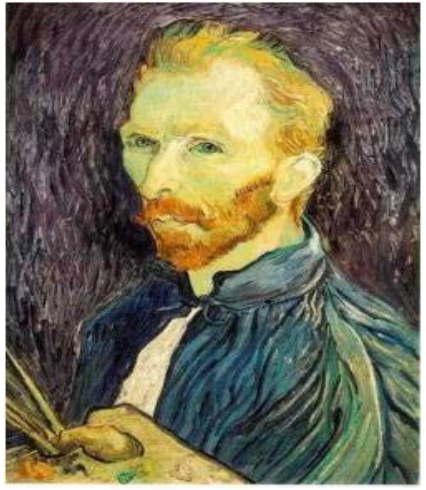
$$V = (R + G + B) / 3$$

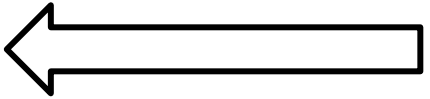
$$V = 0.299\ R + 0.587\ G + 0.144\ B$$



Conversion from grayscale

- How is gray represented in RGB color space?





Brightness and Contrast

- **Brightness** – absolute value of all pixels
- **Contrast** – difference between minimum and maximum pixel

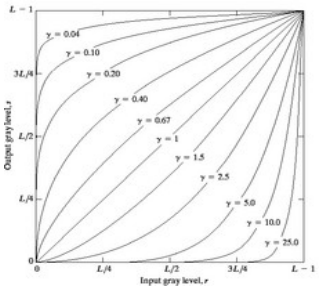
$$\mathrm{f(x)} = \alpha x + \beta$$

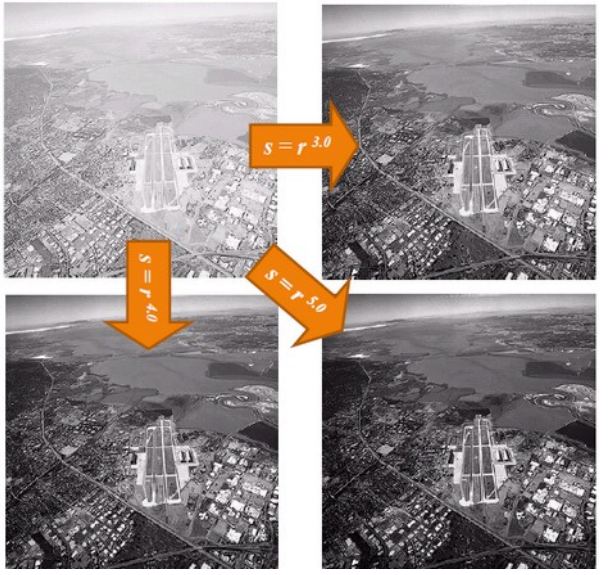
$$\mathrm{f(x)} = \alpha(x - 128) + 128 + \beta$$

Nonlinear Intensity Transformations

- Parametric function that maps source values r to destination values s.
- Exponential function family
- Parameter c is usually 1
- Parameter r is in [0, 1]

$$s = cr^{\gamma}$$



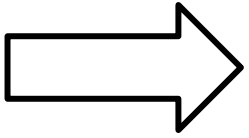


Negation

8-bit intensities are defined on interval from 0 to 255

Image negation of image A is B = (255 – A)

A



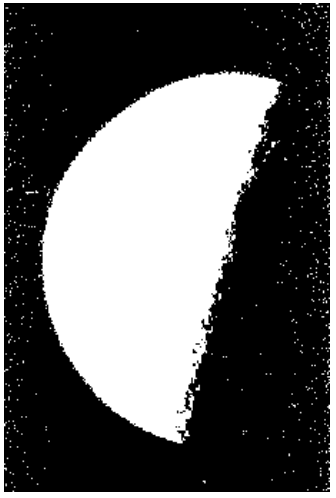
B

Image Threshold

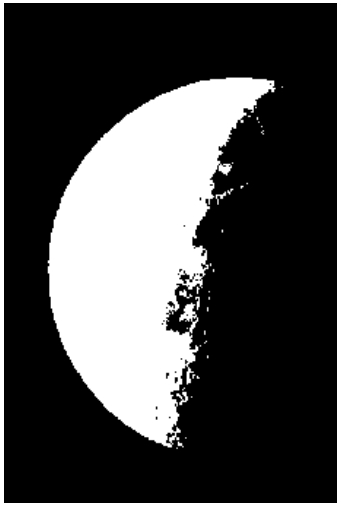
Pixel values higher than value T are set to 1 (max), others to 0

I

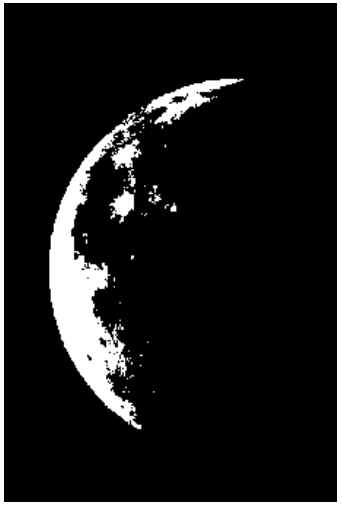
I>10



I>100



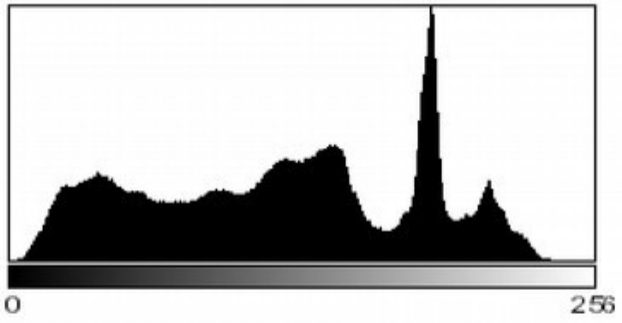
I>200



Object segmentation: determining a suitable threshold is not trivial

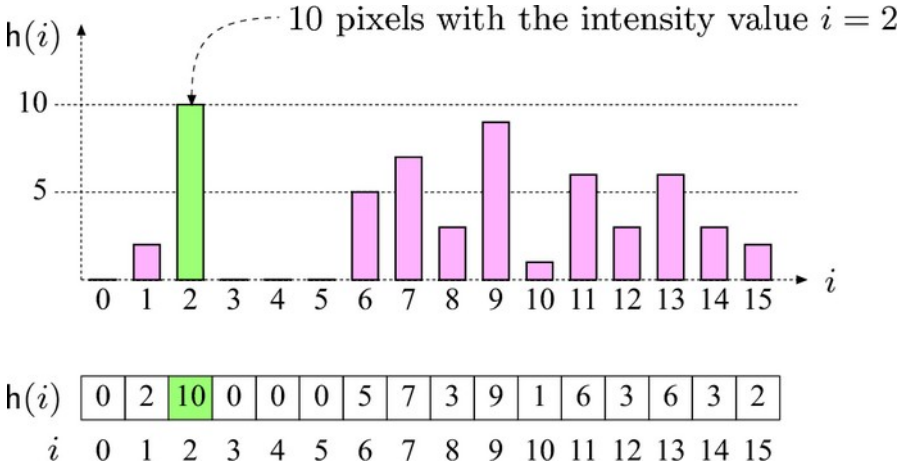
Distribution of values in images

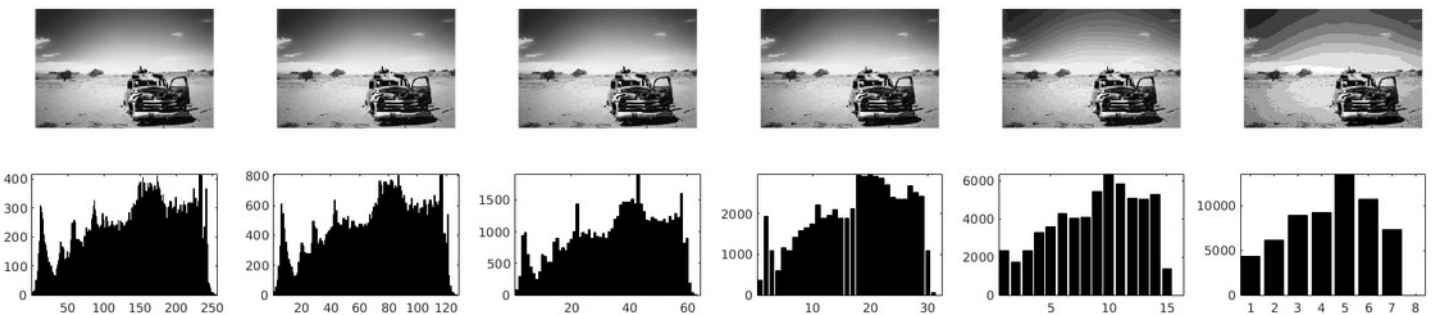
- How to adjust values based on the image?
- Use image-specific statistics - histograms



Histogram

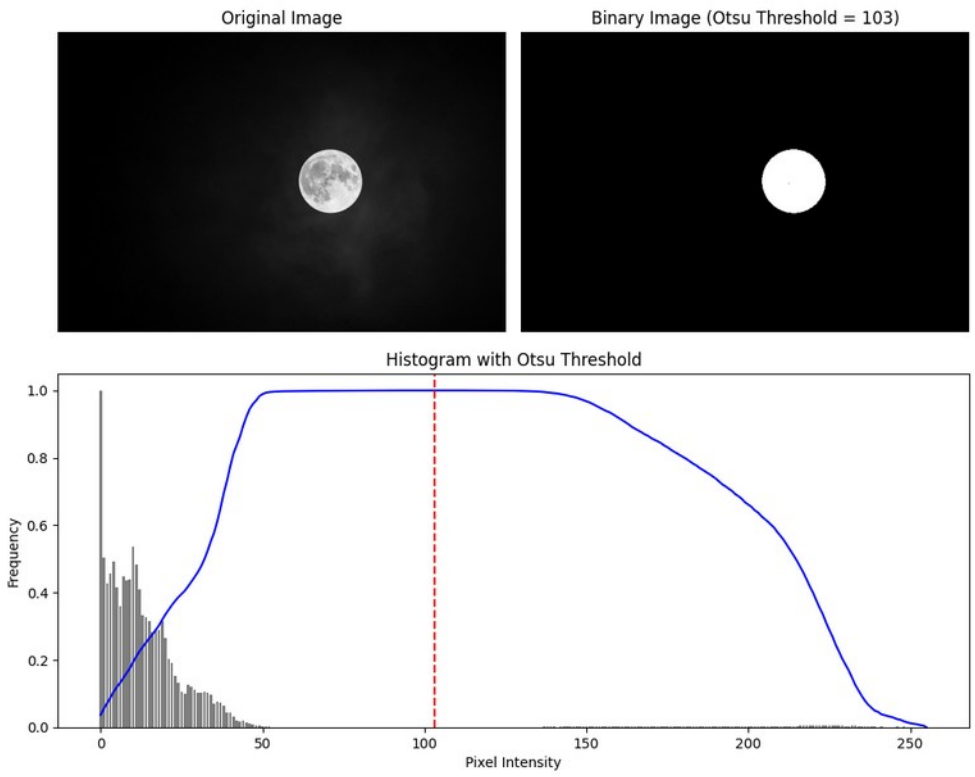
- Not a filter but a descriptor
- Frequency of different pixel values
  - How often they occur in image
  - Sub-sampling into cells/buckets
- Robust description
  - Rotation
  - Translation
  - Scale





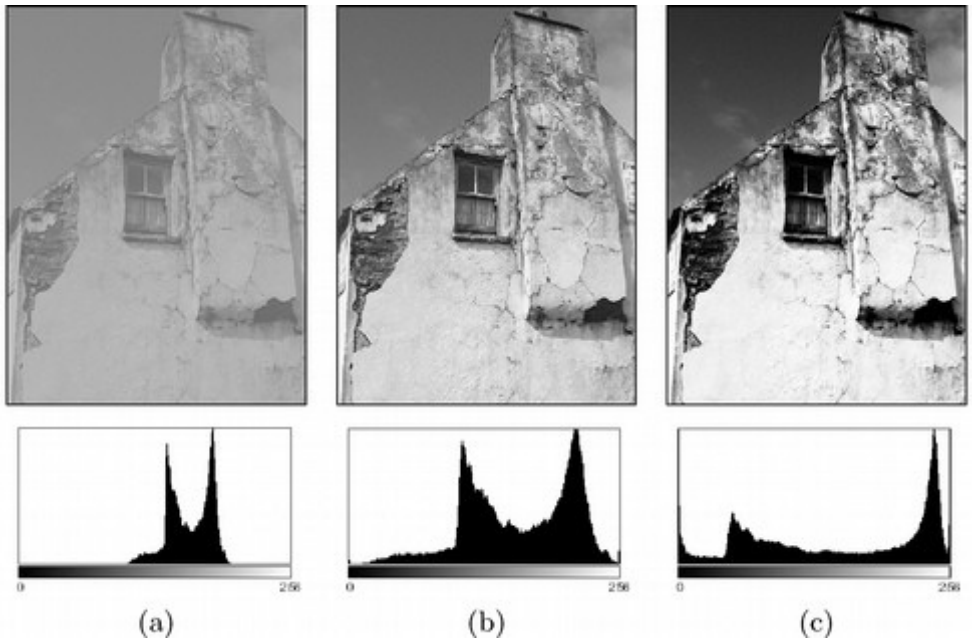
Histogram and Thresholds

- Analyze histogram to find good threshold
- Bi-modal histogram
- Otsu method
  - Minimize variance of foreground and background



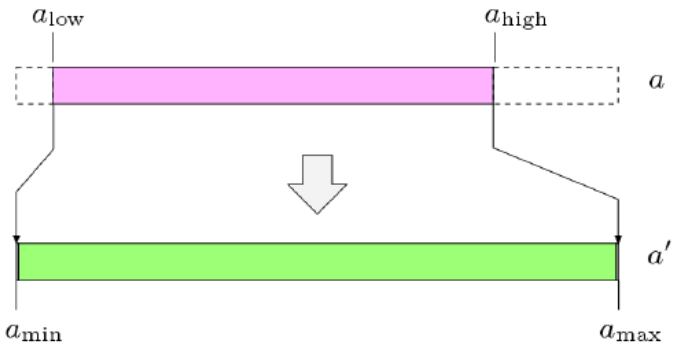
Histogram and image quality

- Increase/reduce brightness:
  - Histogram shifts left/right
- Increase/reduce contrast:
  - Histogram is shrinking/stretching

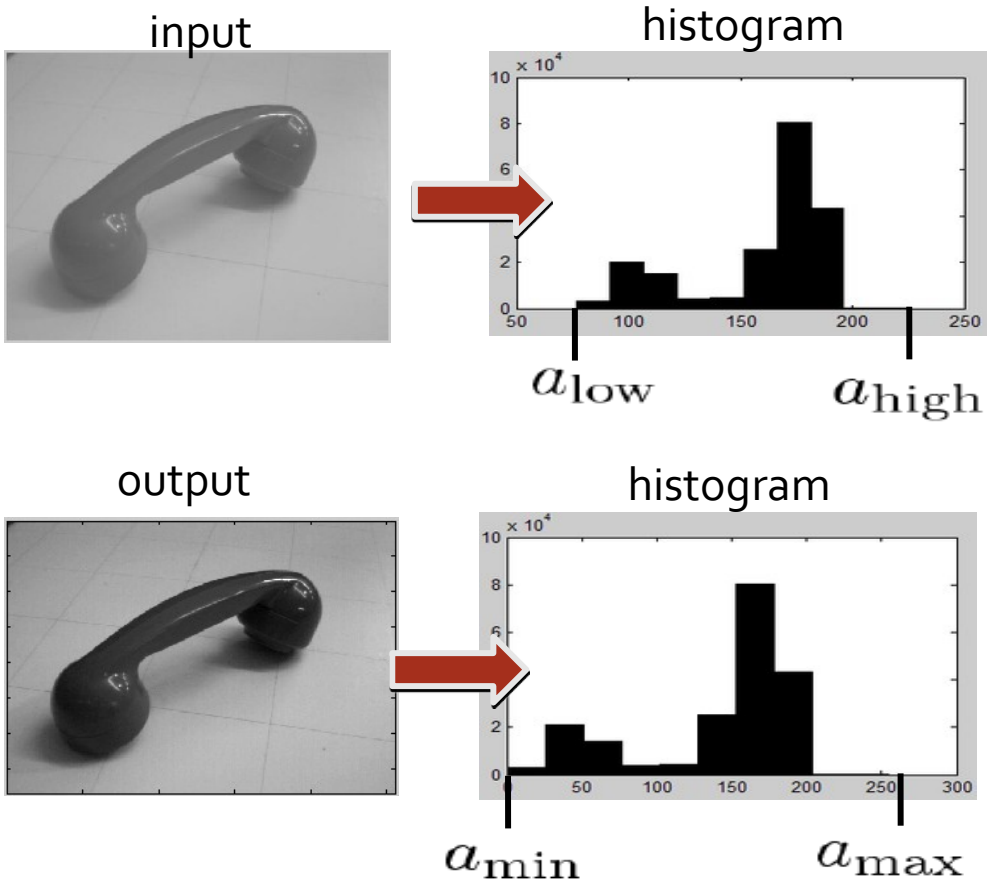


Histogram Stretching

$$f_{\mathrm{ac}}(a) = a_{\min} + (a - a_{\mathrm{low}}) \cdot \frac{a_{\max} - a_{\min}}{a_{\mathrm{high}} - a_{\mathrm{low}}}$$



Operation performed on each pixel individually.

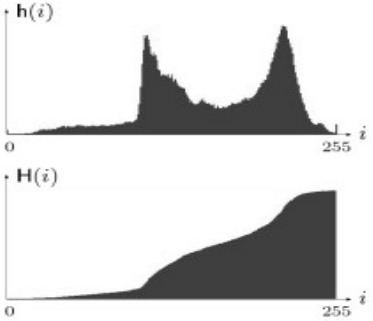


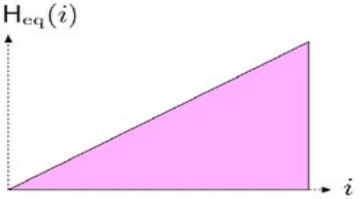
Histogram Equalization

- Cumulative histogram - value dynamics
- Desired dynamics is uniform

$$H(i) = \begin{cases} h(0) & \text{for } i = 0 \\ H(i-1) + h(i) & \text{for } 0 < i < K \end{cases}$$

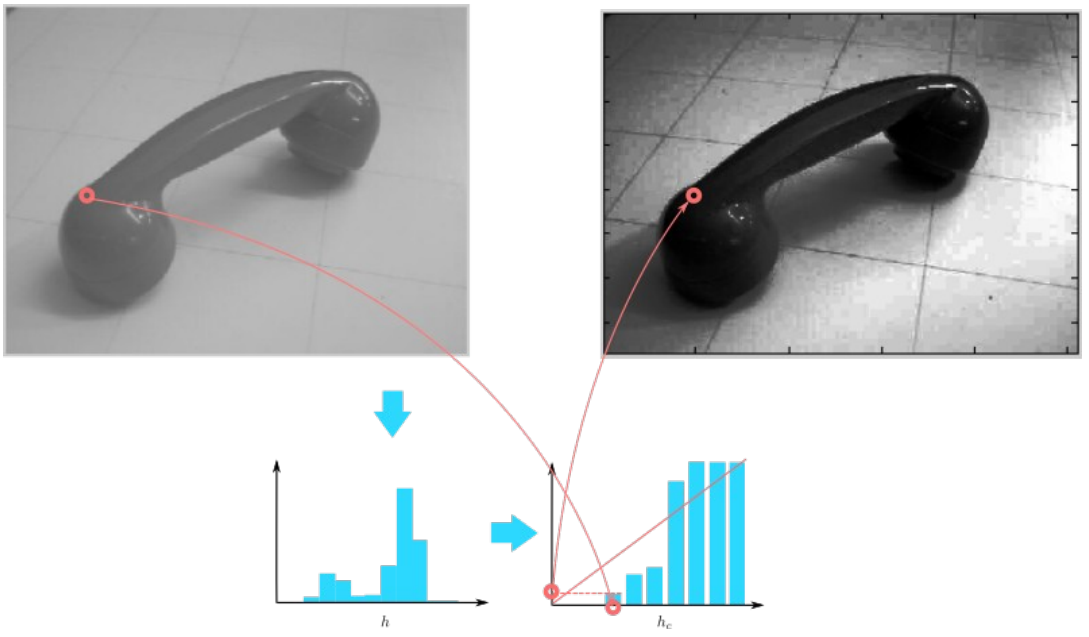
- Transform image values so that the cumulative histogram is diagonal.





Equalization Algorithm

- Compute 256-bin histogram of image I (h).
- Compute cumulative histogram (hc).
- Normalize (hc) with maximum value, multiply by 255 (hnc)
- Use (hnc) as a lookup table to transform individual pixels.



Histogram Equalization in Color Images

Original

RGB Equalized Independently

Because each channel is transformed independently, the resulting color changes as well.

Original

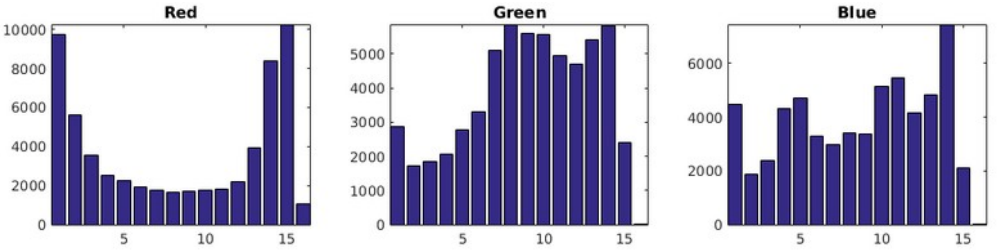
Luminance Equalization

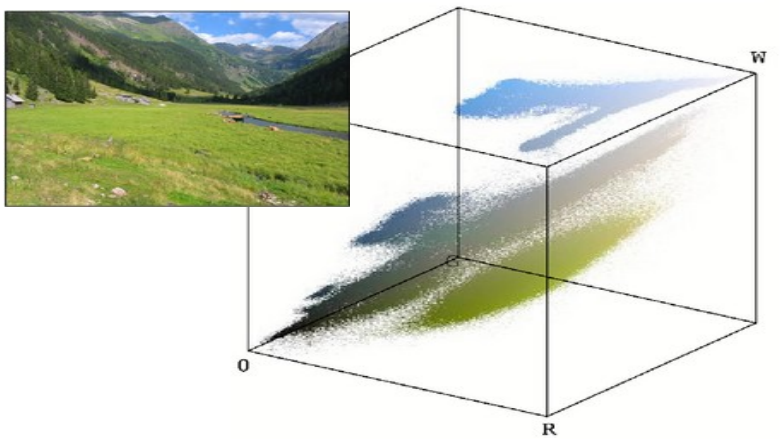
Transform to color space with separate luminance channel, equalize only intensity

This technique can be used in many operations

Color Histogram

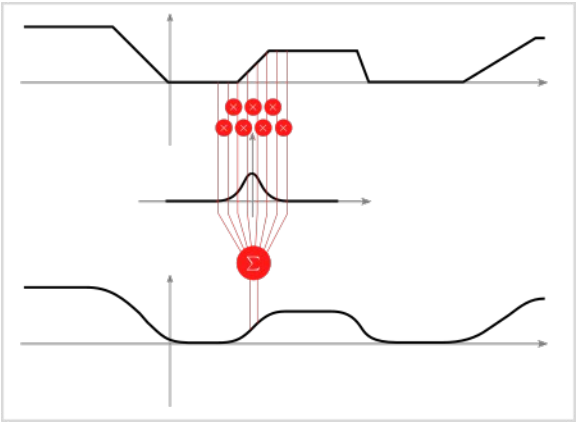
- 3 histograms
  - Each component separate
  - No correlation
  - Less space
- 3D histogram
  - Image color is a 3D index
  - More specific
  - More space

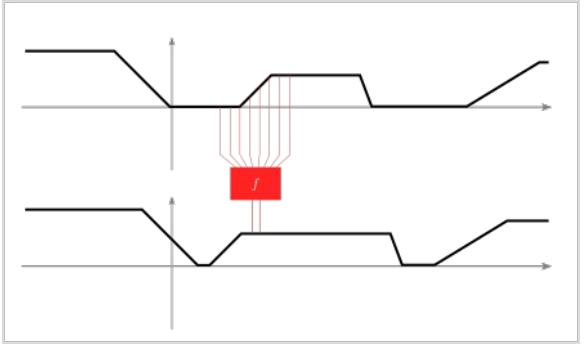




Neighborhood Filtering

- Resulting value dependent on the neighborhood
- Linear filters
  - Convolution / correlation
  - Kernel function (parameters)
  - Associativity, separability
- Nonlinear filters
  - Arbitrary (local) operation
  - Max, min
  - Median



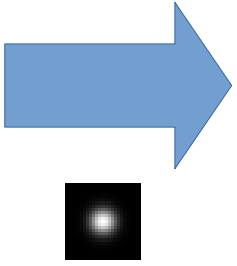


Simple, yet complex

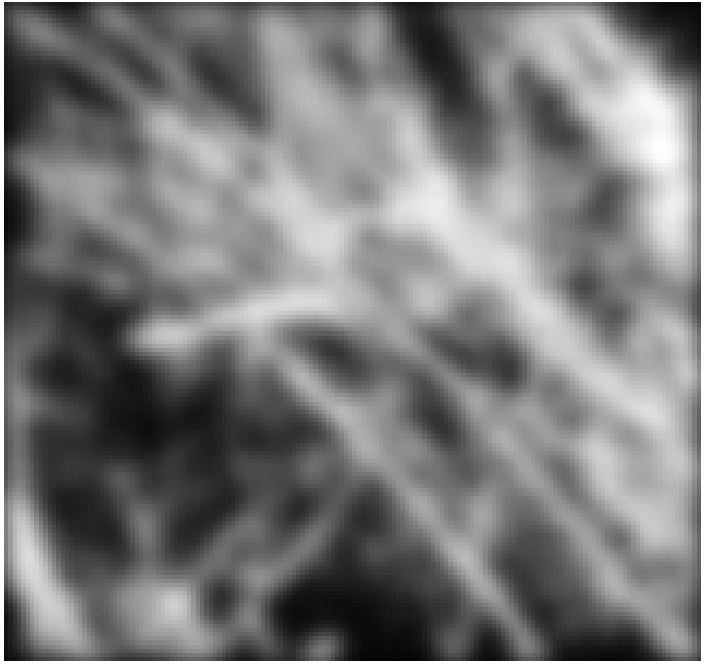
- Single linear filter
  - Blur, directional blur
  - Edges, blobs
- Multiple linear filters that are learned
  - Extract specific properties
  - Data-driven
  - Convolutional neural networks

Linear filters

image



kernel

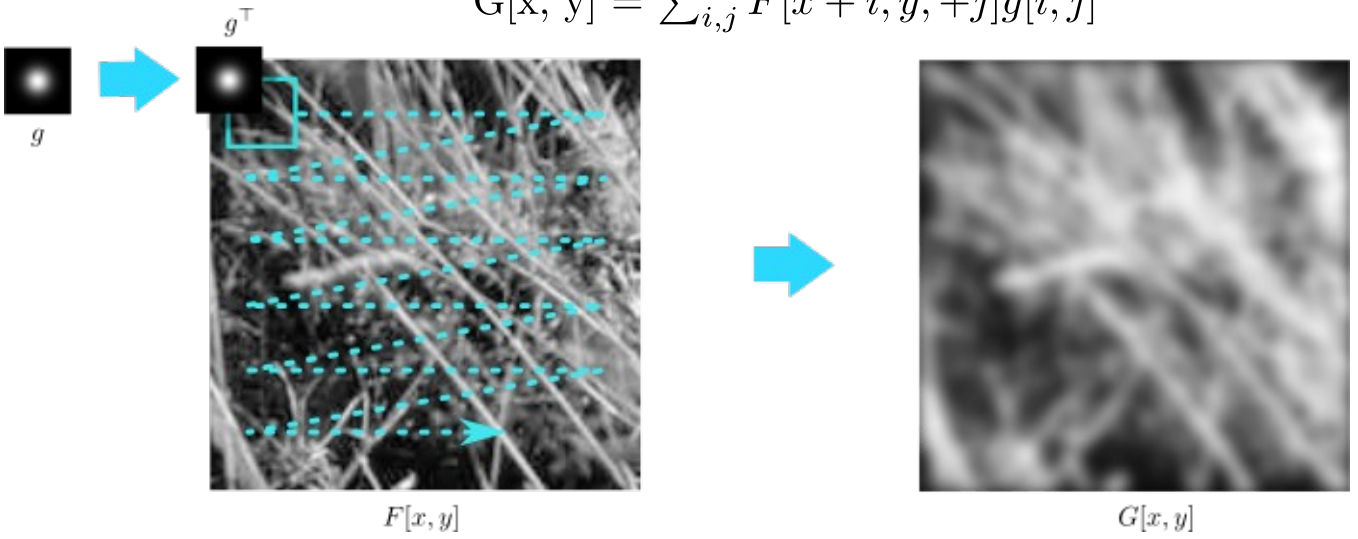


result

In a nutshell

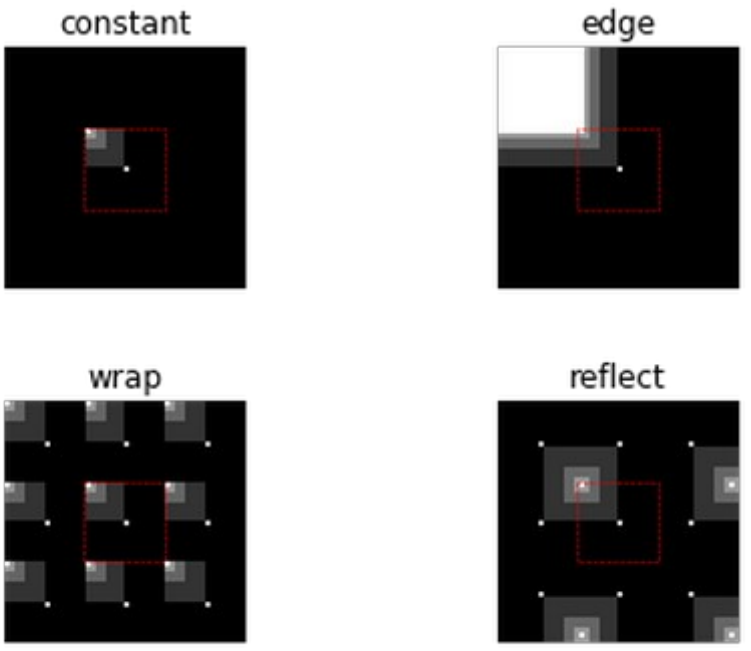
- How to compute filter response in individual pixel?
  - Transpose kernel (convolution) and align its center with the pixel
  - Multiply corresponding elements and sum together

$$\mathrm{G[x,\ y]} = \sum_{i,j} F[x+i, y, +j]g[i,j]$$



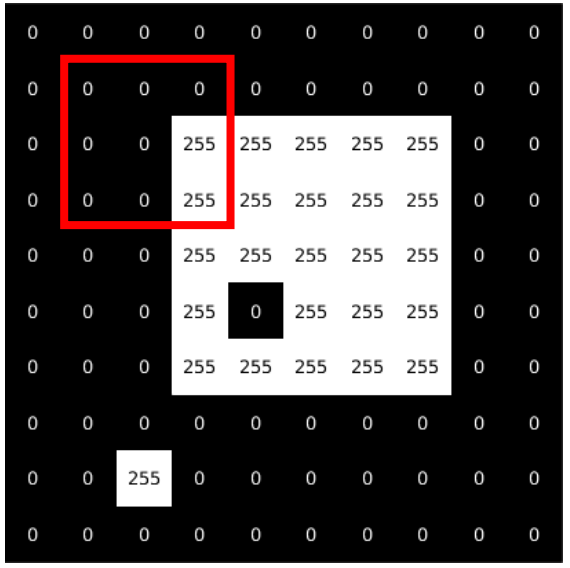
What to do with borders?

- Image is a finite signal
  - Filtering
  - Interpolation
- Data out of border has to be fabricated
- Different techniques
  - Based on use-case



Weighted sum

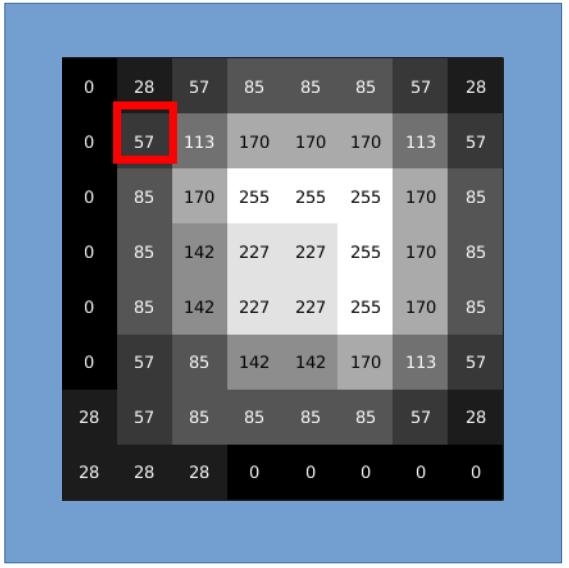
$F[x, y]$



g

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

$G[x, y]$



what do to with the border?

Identity filter

*

0	0	0
0	1	0
0	0	0

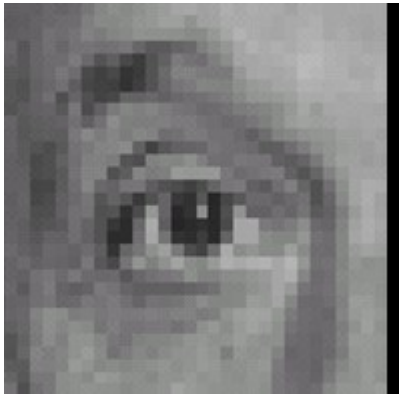
=

Shift filter

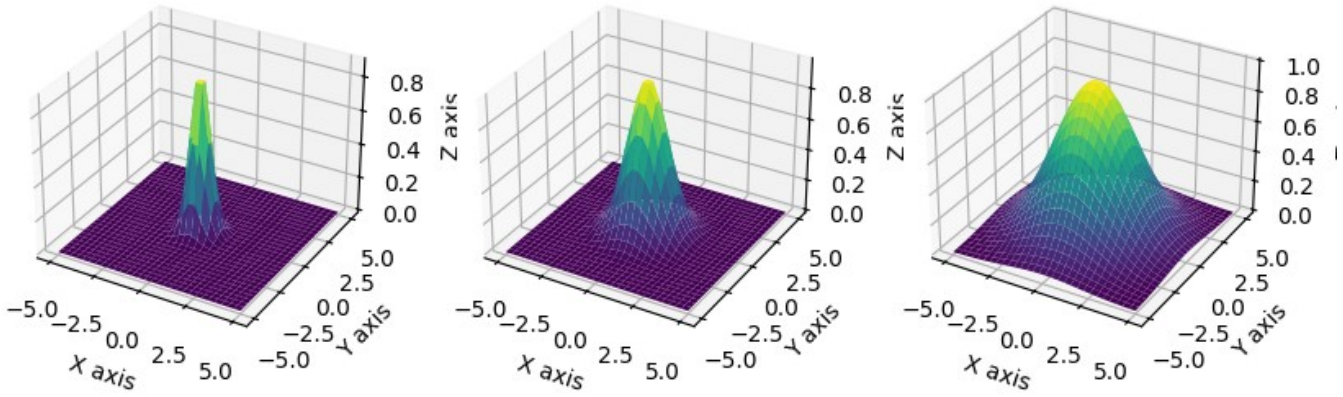
*

0	0	0
0	0	1
0	0	0

=



Gaussian Kernel



0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.00	0.00	0.01	0.02	0.01	0.00	0.00
0.00	0.01	0.06	0.10	0.06	0.01	0.00
0.00	0.02	0.10	0.16	0.10	0.02	0.00
0.00	0.01	0.06	0.10	0.06	0.01	0.00
0.00	0.00	0.01	0.02	0.01	0.00	0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00

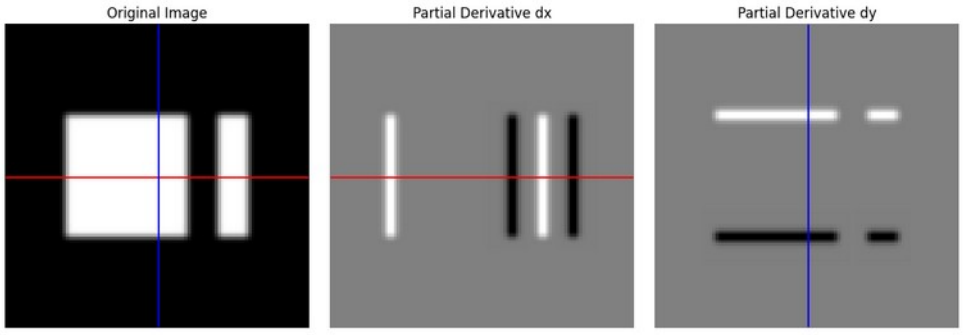
$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}}$$

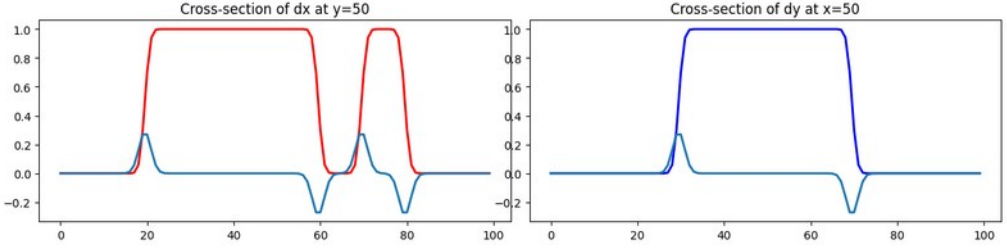
7 x 7, sum = 1

Constant before the exponential function ensures that the sum of the elements is always 1 (in continuous space).

Detecting edges

- Goal: find sudden changes in illumination in the image
- Ideal: line drawing by an artist (semantic knowledge)
- Derivatives: high pass filtering





Using convolution

- Kernel can represent approximation of image derivation
- We use separate kernels for vertical and horizontal derivation

Prewitt

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

Sobel

-1	0	1
-2	0	2
-1	0	1

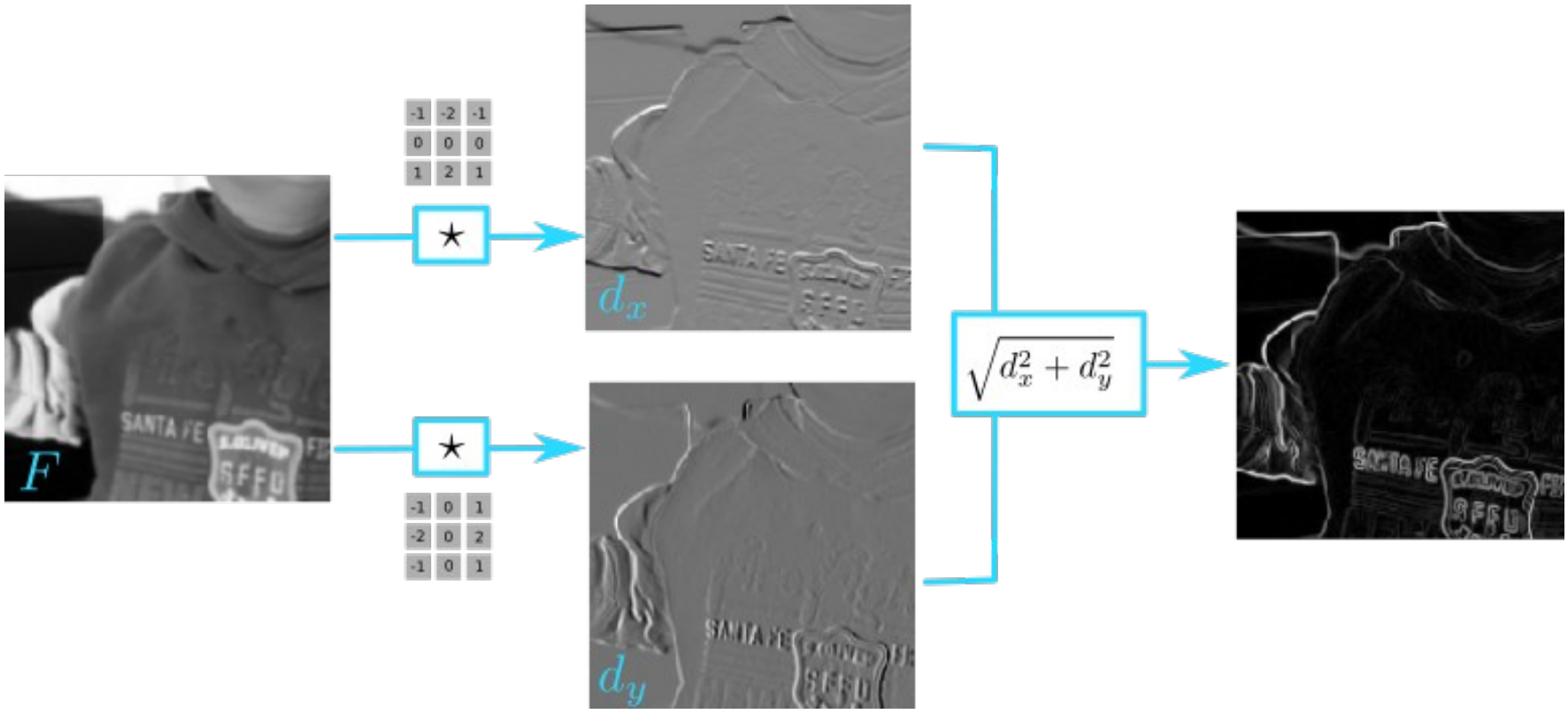
1	2	1
0	0	0
-1	-2	-1

Roberts

0	1
-1	0

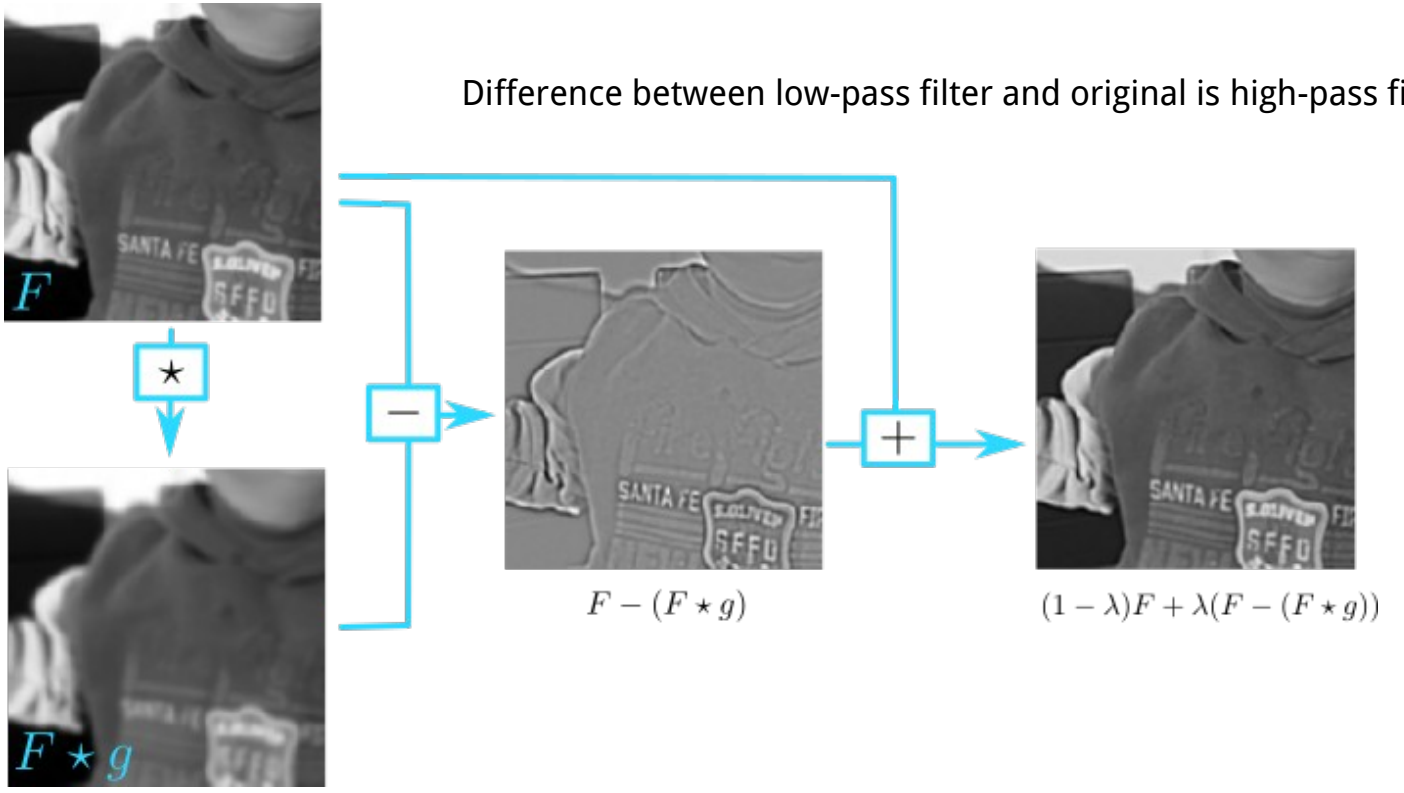
1	0
0	-1

Derivative magnitude



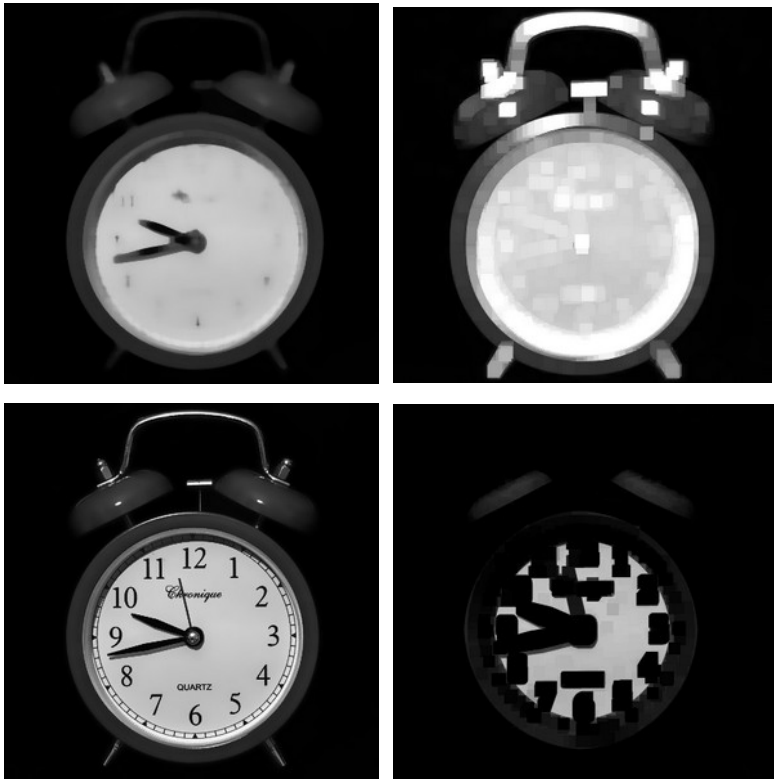
Sharpening by blurring

Difference between low-pass filter and original is high-pass filter.

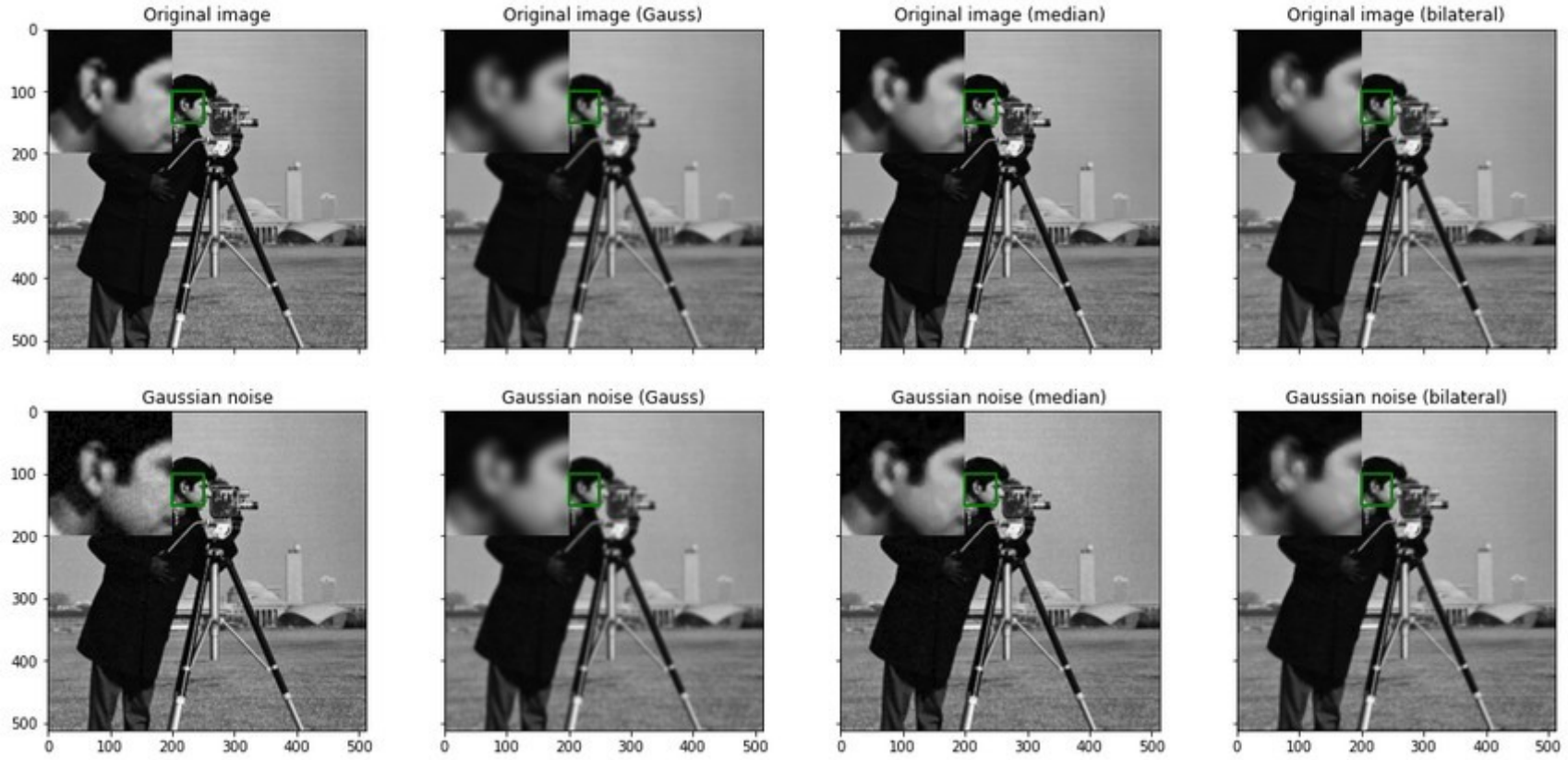


Non-Linear Filters

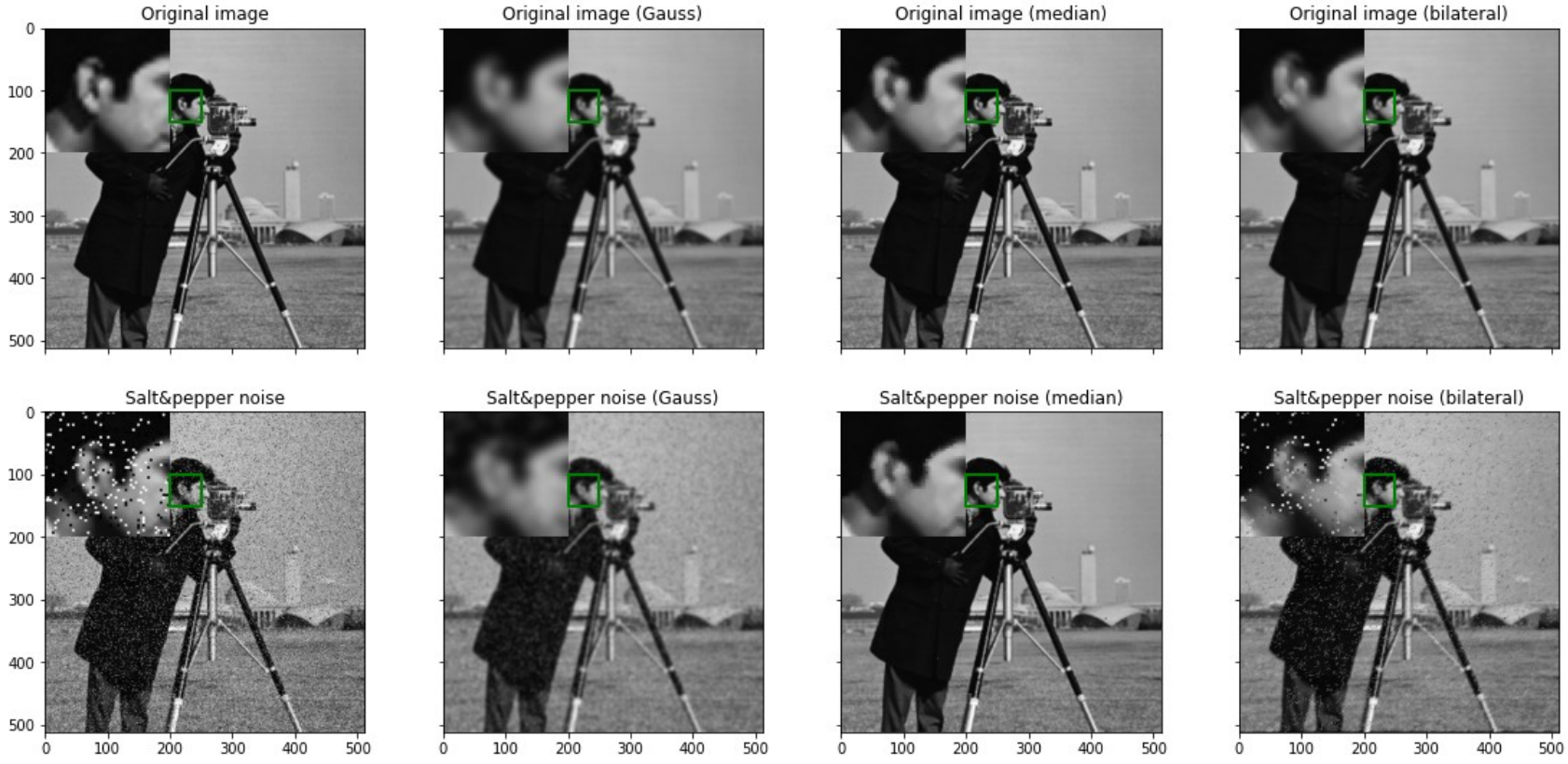
- Median: middle element by value
- Bilateral filter
  - Weights based on neighborhood
  - Preserves edges
- Morphological operations
  - Max: highest element in neighborhood
  - Min: lowest element in neighborhood



Gaussian noise removal

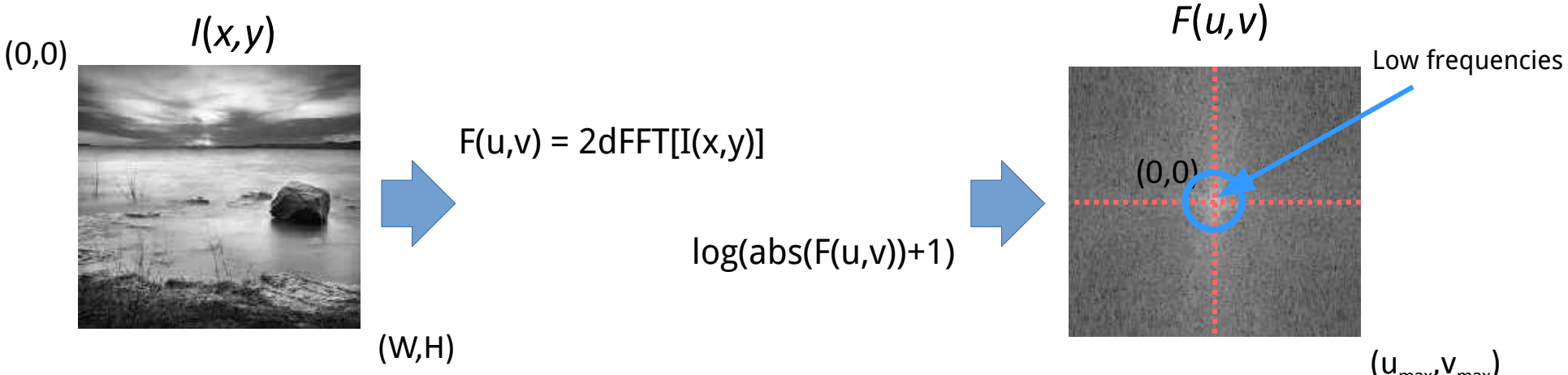


Salt&pepper noise removal

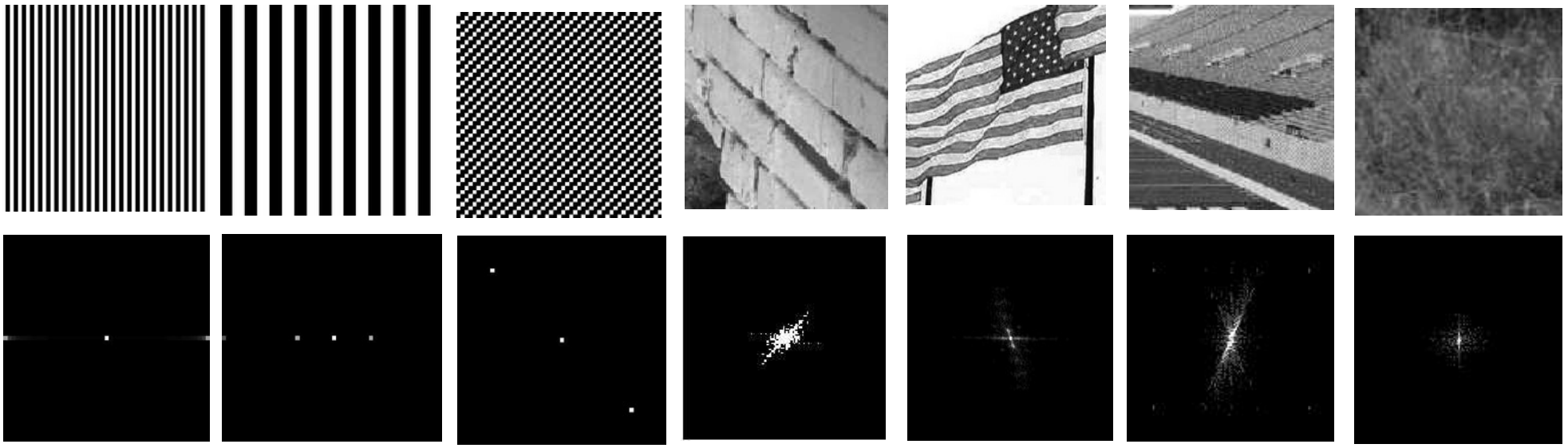


Fourier transform

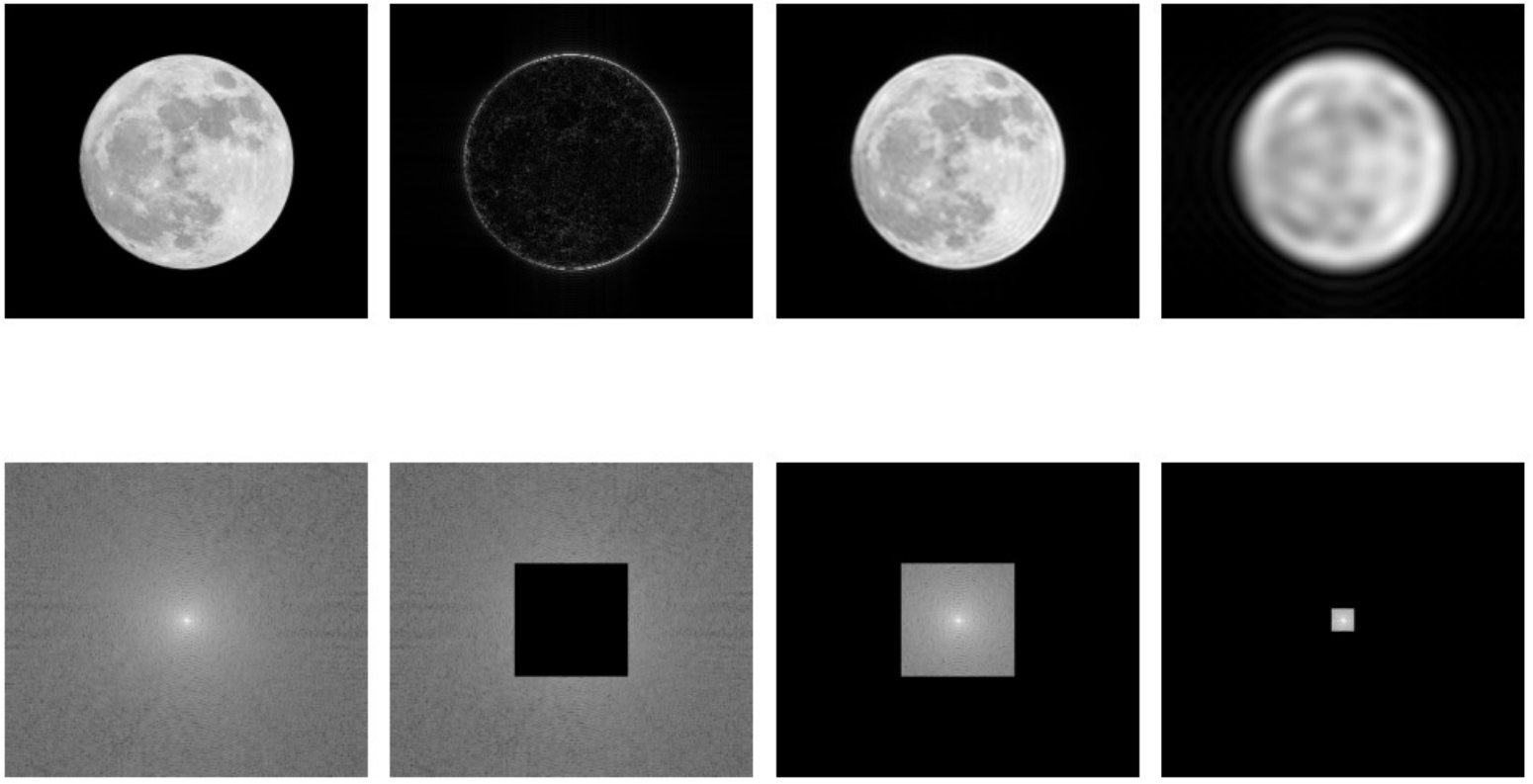
- Description of image with complex basis functions
  - Energy of spectrum: |F(u,v)|
  - If I is WxH, then F is WxH



Examples



Manipulation in Frequency Space

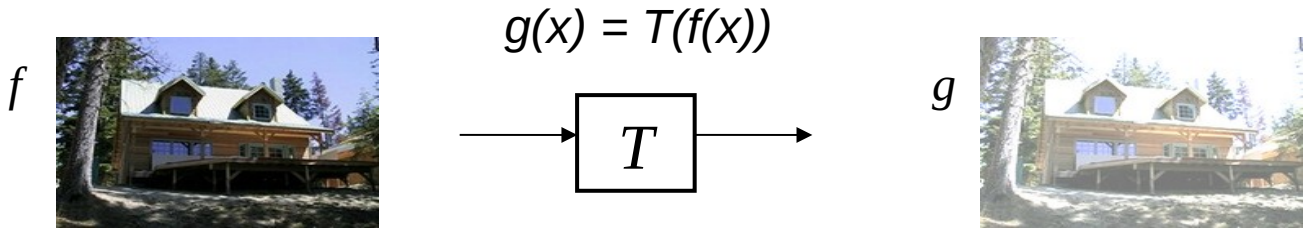


Summary

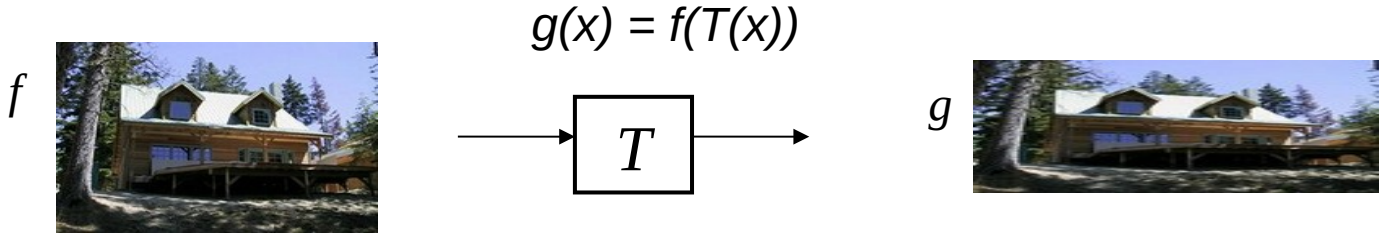
- Image is a matrix of pixels, but also a signal
  - Signal processing toolbox
  - Filtering and convolution
- Basic representation abstractions
  - Histogram
  - Frequency space

Geometry vs. Intensity

- Image filtering is an intensity transformation



- Geometry transformation changes geometry of the image



Parametric transformations

T

$\mathbf{p} = (x,y)$ $\quad \mathbf{p}' = T(\mathbf{p})$ $\quad \mathbf{p'} = (x',y')$

- Transformation T changes coordinates of pixel p
  - Linear transformation
  - Non-linear transformation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \mathbf{p}' = \mathbf{M} \cdot \mathbf{p}$$

Linear transformations

- Rotation:

$$x' = \cos\Theta x - \sin\Theta y \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\Phi & -sin\Phi \\ sin\Phi & cos\Phi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$
$$y' = \sin\Theta x + \cos\Theta y$$

- Shear:

$$x' = x + \alpha_x y \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & \alpha_x \\ \alpha_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$
$$y' = \alpha_y x + y$$

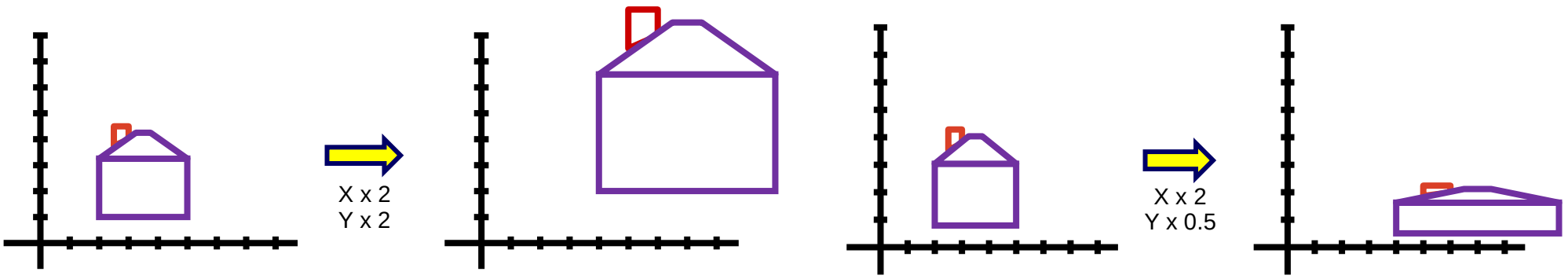
- Mirroring:

$$x' = -x \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$
$$y' = -y$$

Scaling

- Multiply coordinates with a scalar
  - **Uniform** - same scalar for all axes
  - **Non-uniform** - different scalar

$$x' = \alpha x \quad y' = \beta y \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Translation

- Translation is not homogeneous in 2D space

$$x' = x + t_x$$
$$y' = y + t_y$$

- We can use homogeneous coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \Longrightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \mathrm{M} \begin{bmatrix} x \\ y \end{bmatrix}$$

Homogeneous coordinates

- 2D space: transformation matrix of size 3x3

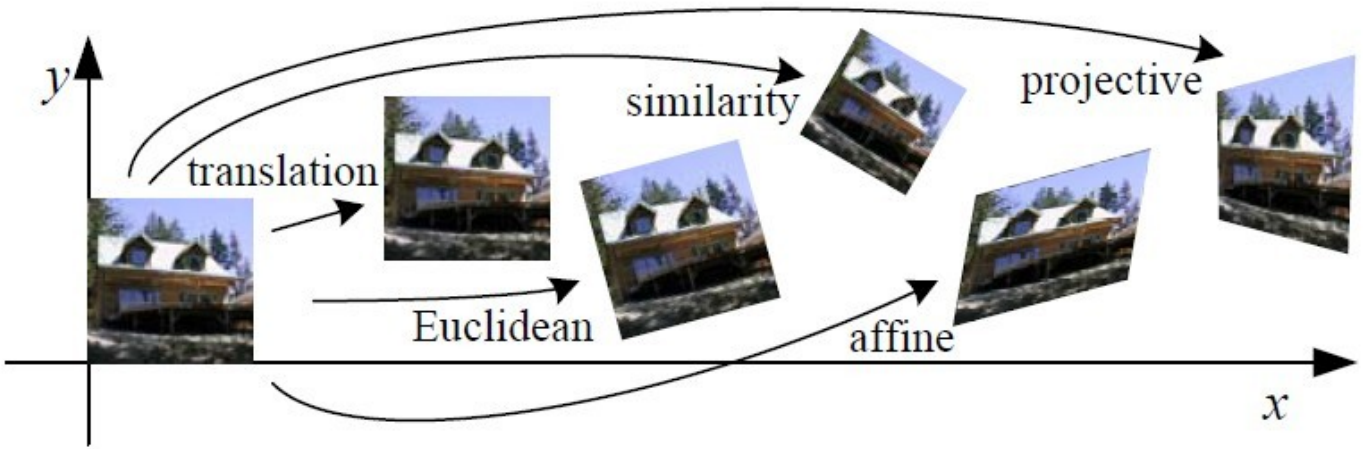
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \alpha_x & 0 \\ \alpha_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- General projective transform

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

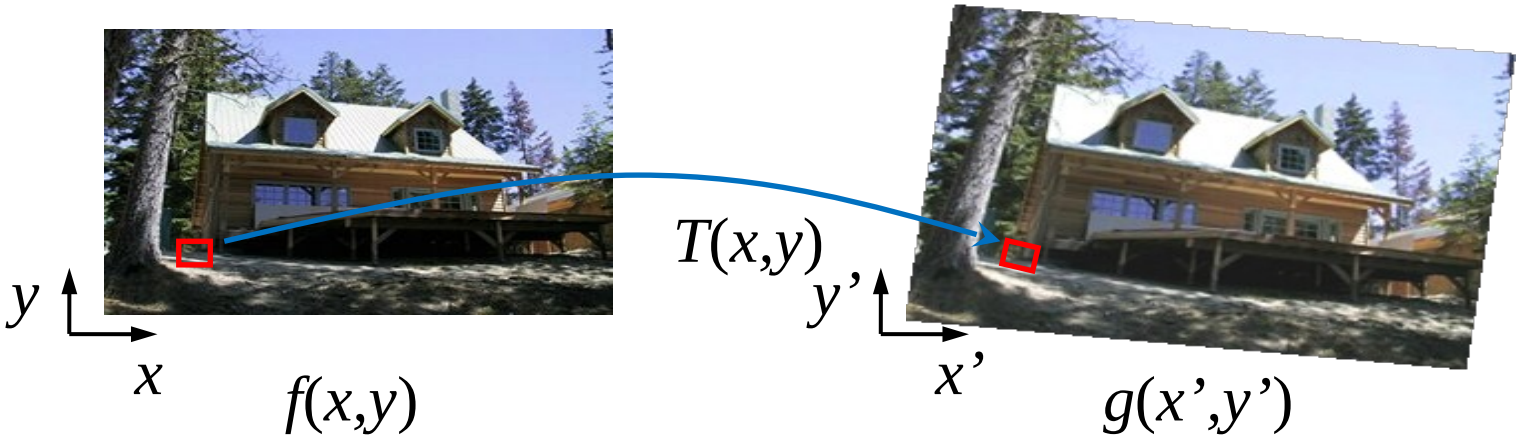
Common transformations

- Translation
- Euclidean transform (translation, rotation)
- Similarity transform (translation, rotation, scaling)
- Affine transform (maintains parallelism of straight lines)
- Projective transform



Warping

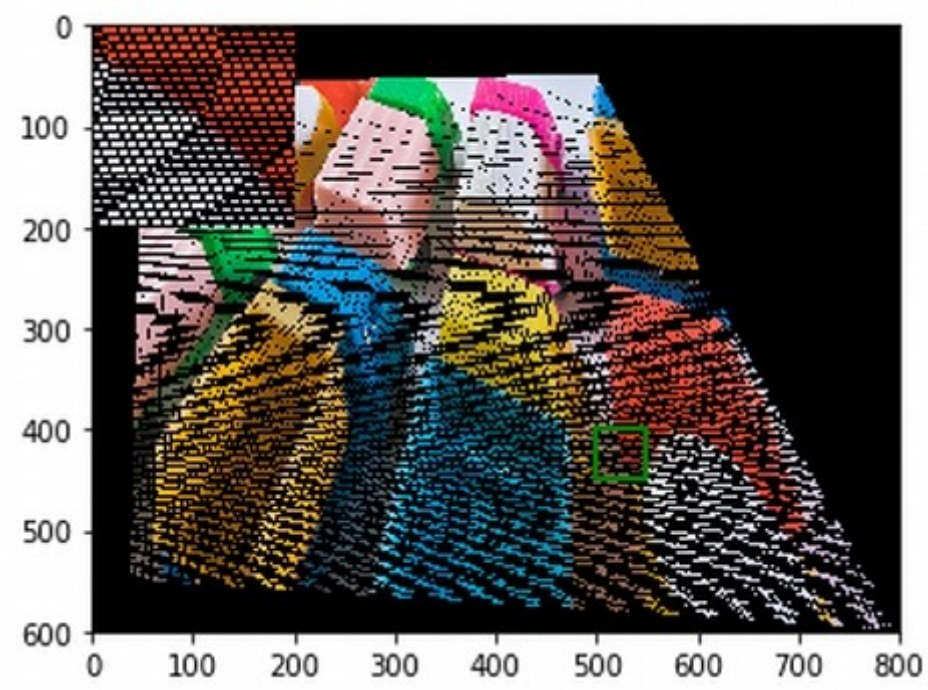
Given transform [x‘,y‘] = T(x,y) and f(x,y), how to compute g(x‘,y‘) = f(T(x,y)) ?

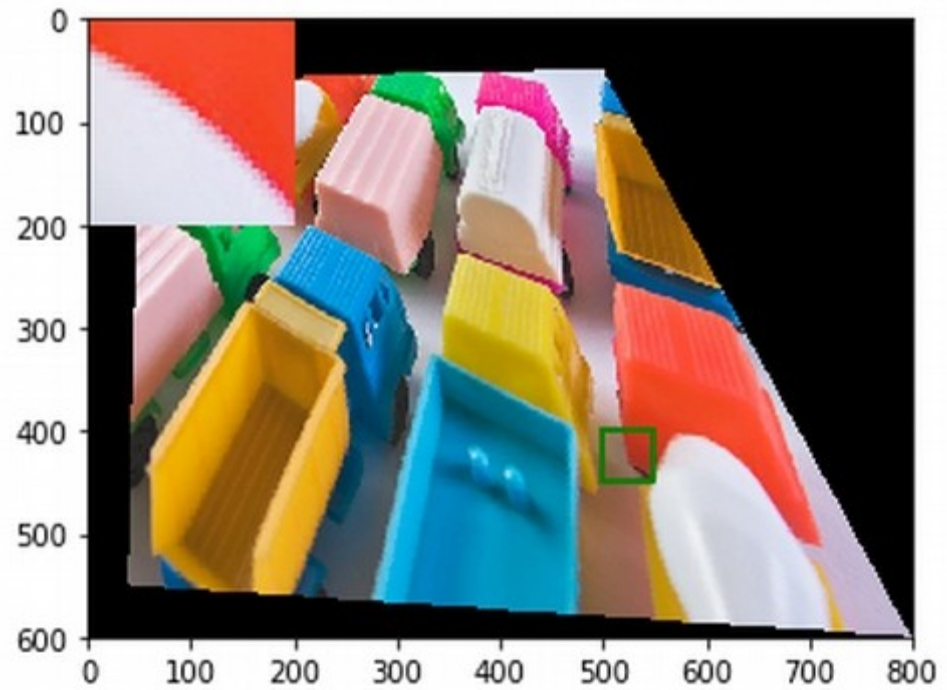


Naive approach

- For each pixel f with coordinates (x,y)
  - Compute transformed coordinates [x‘,y‘]=T(x,y).
  - Copy color of f(x,y) to new image at coordinates g(x‘,y‘)
- Why is it naive?
  - We visit all pixels in f(x,y)
  - Do we visit all pixels of g(x,y)?

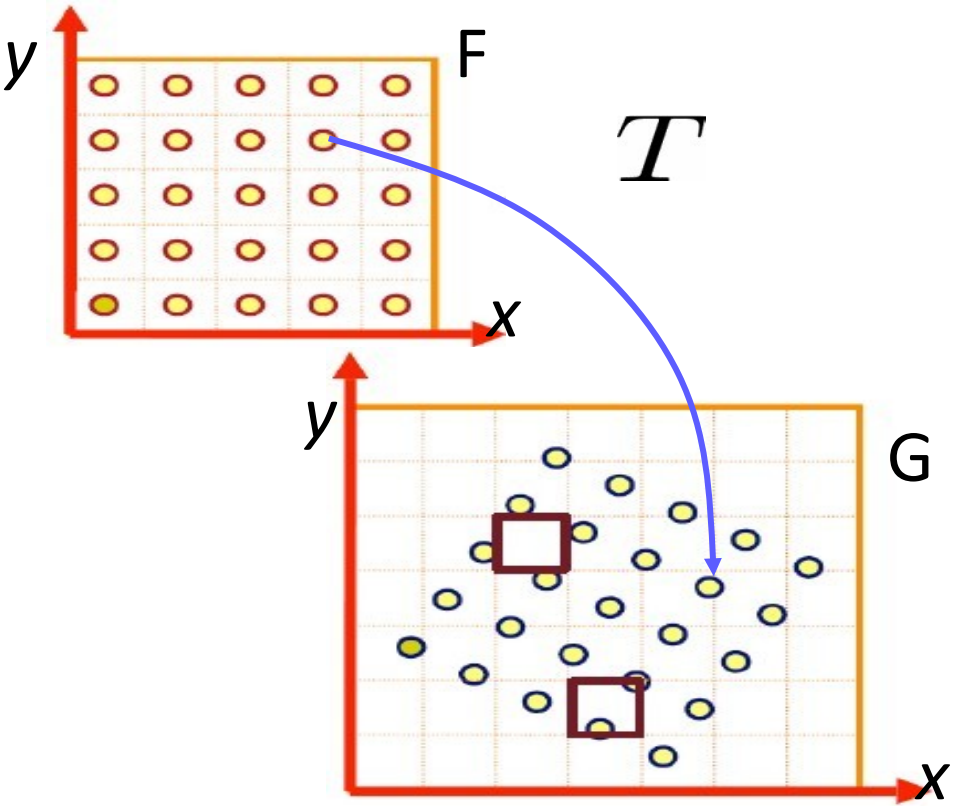
Example


0
100
200
300
400
500
600
0
100
200
300
400
500
600
700
800


0
100
200
300
400
500
600
0
100
200
300
400
500
600
700
800

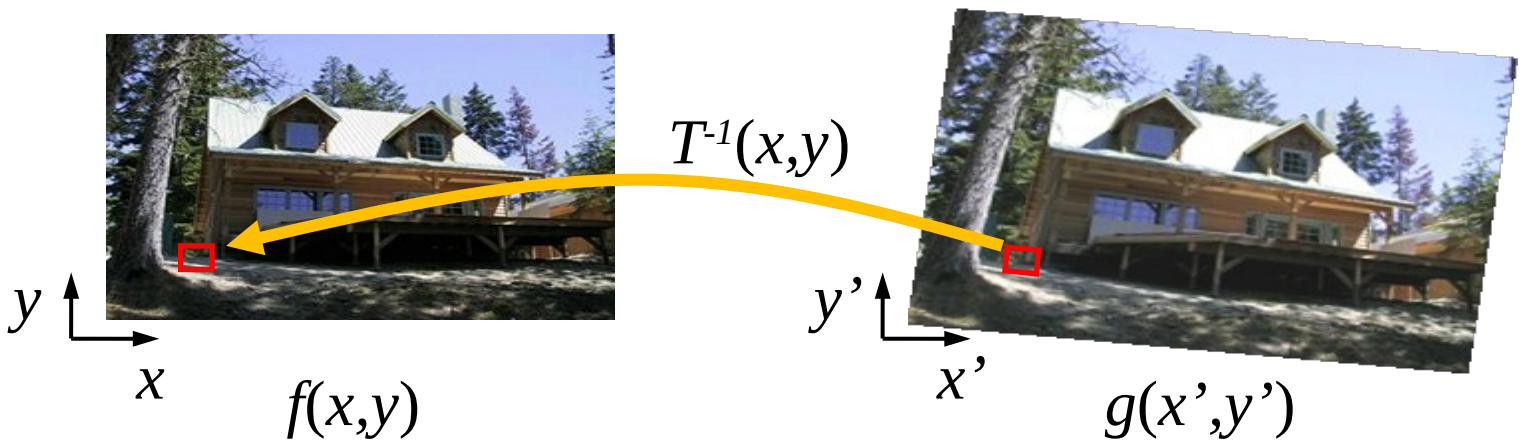
Asymmetry of Discretization

- A single pixel of F is mapped to more pixels in G.
- Pixel in F is not mapped to any pixel in G.
- Visit all pixels in G



Transform as Resampling

- For each pixel in g with coordinates (x‘,y‘)
  - Compute old coordinates using inverse transform [x,y]=T-1(x‘,y‘)
  - We copy pixel color of f(x,y) to g(x‘,y‘)
- We visit all pixels in g
- Pixel from g can transform to more than one pixel in f

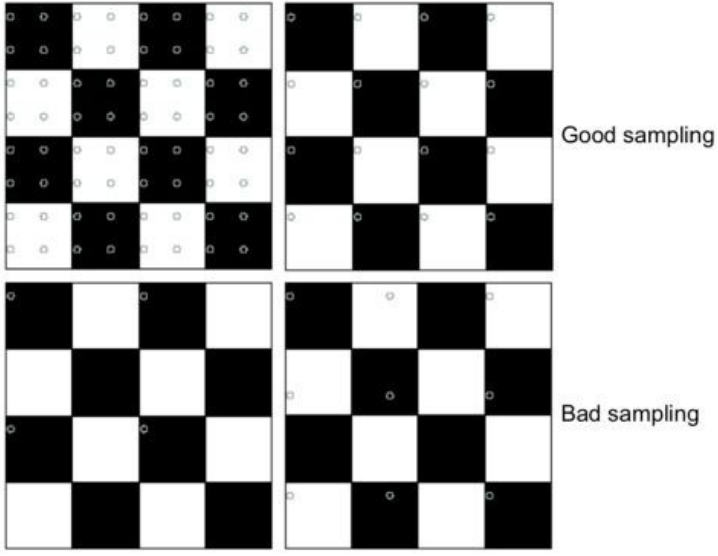


Resizing the Image

- Adjusting resolution in image – changing sampling rate
- Two general cases:
  - Reduction / decimation / down-sampling – shrinking image
  - Interpolation / up-sampling – enlarging image

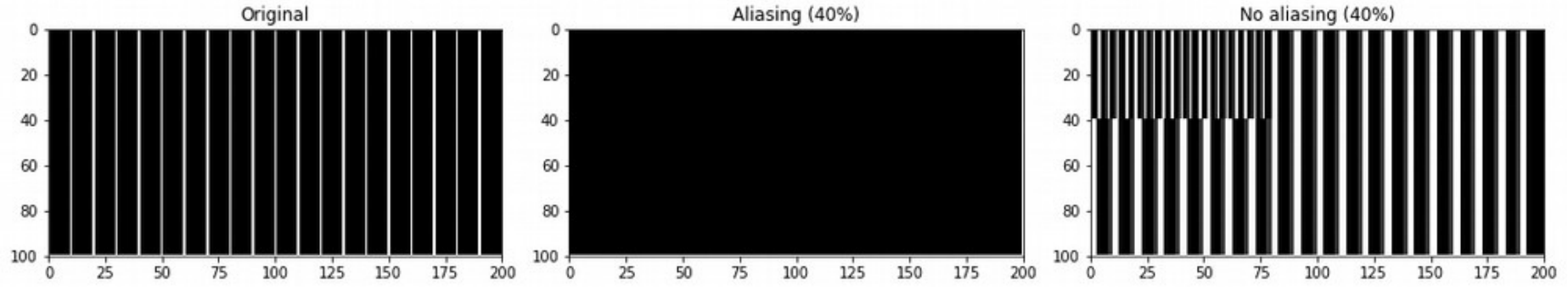
Decimation approaches

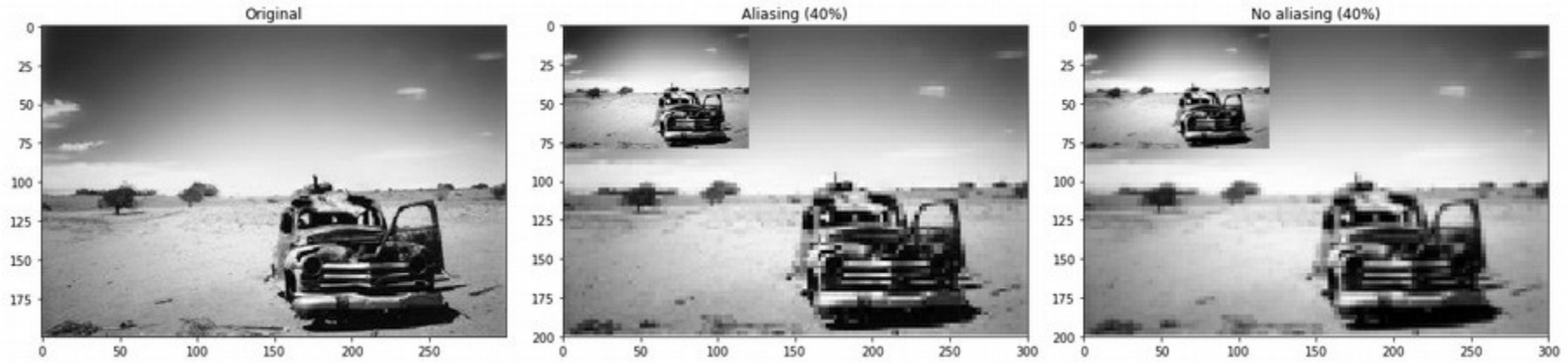
- Sampling values
  - Shannon sampling theorem $f_s \geq 2f_{max}$
  - Remove high frequencies
- Anti-aliasing
  - First remove high frequencies
  - Then subsample with appropriate frequency
- Filters used
  - Gaussian
  - Lanczos kernel



Maël Fabien

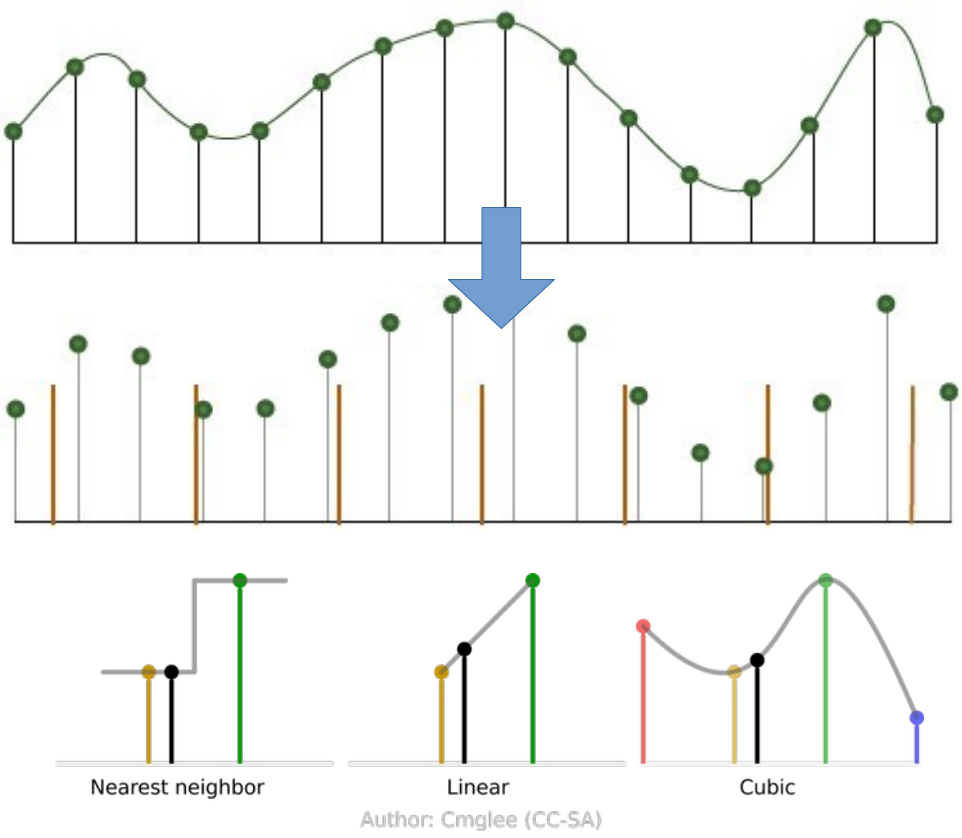
Decimation examples





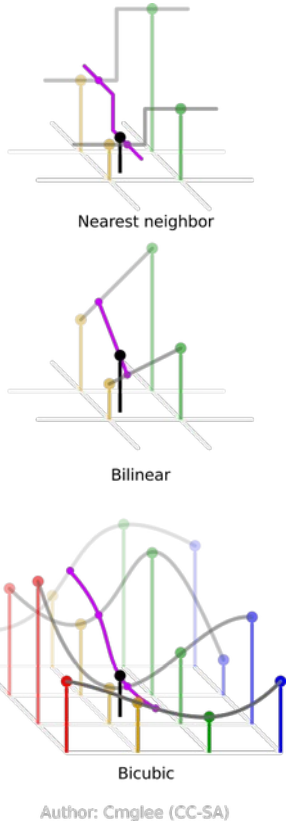
Up-sampling – guessing the values

- Nearest neighbor
  - Take value of the closest sample
- Linear interpolation
  - Use two values
  - Fit linear function
- Cubic interpolation
  - Use four values
  - Fit third-degree polynomial

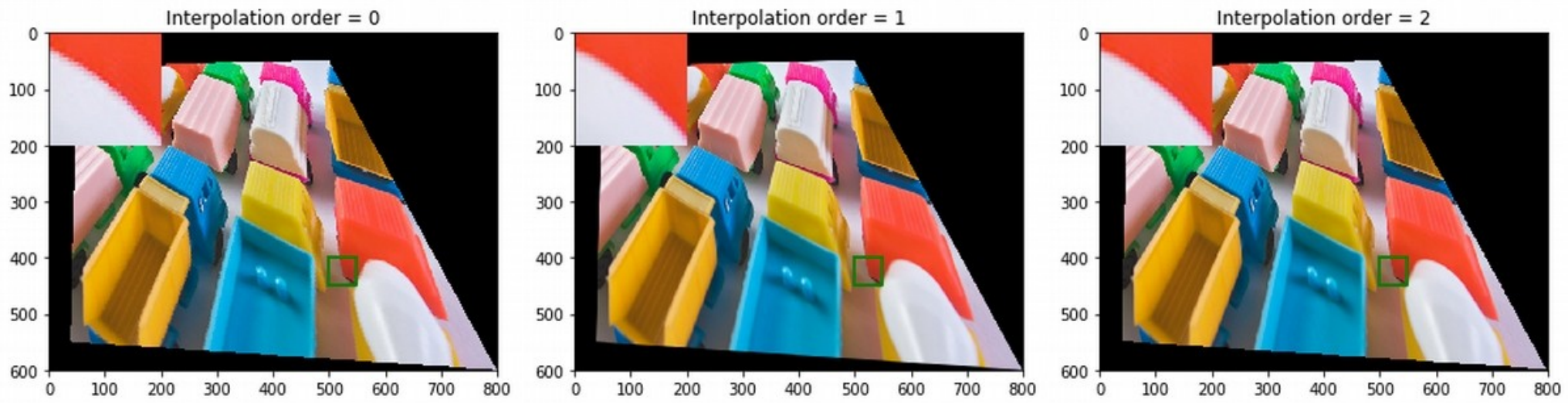


Interpolation in Images

- Nearest neighbor
  - Take pixel closest to desired coordinates
- Bi-linear
  - Use four closest pixels
  - Estimating a plane
- Bi-cubic
  - Use 16 closest pixels
  - Estimating a polynomial surface
  - Slower



Interpolation examples

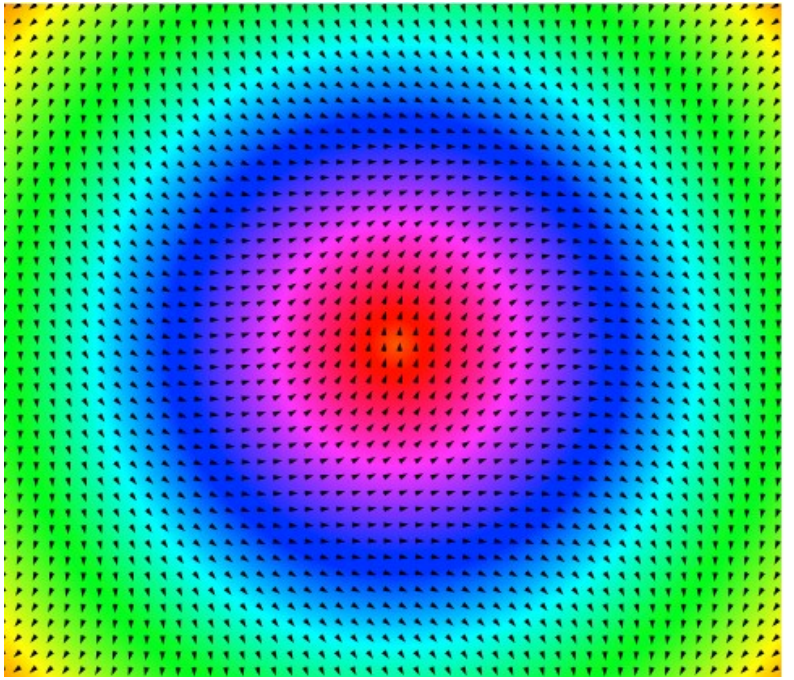


Non-Linear Transforms

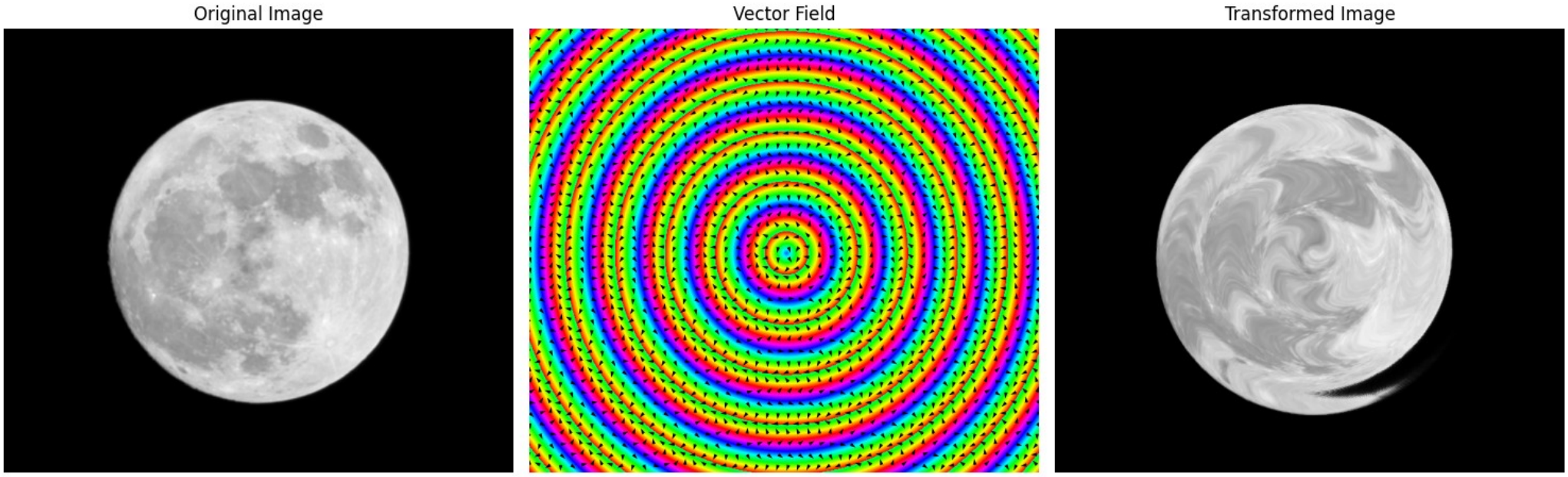
- Transform can be a general 2D to 2D function
  - If the value is between pixels, we interpolate
- Parametric: swirl, wave, ripple
- Free-form: splines, mesh
- Nonparametric: vector field

Vector Field

- 2D Matrix: V[i, j] = (u, v)
- Apply: I_V[i, j] = I[(i, j) + V[i, j]]
- Visualization
  - Field of arrows
  - Color coding (HSV)

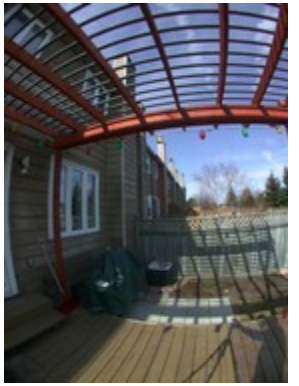


Parametric non-linear transforms



Practical Applications

- Camera rectification
  - Counter lens distortions
  - Camera calibration model
- Locally-linear transformation
  - Local regions are transformed locally
  - Control points
  - Morphing effects



Content-aware resizing

- Change size, aspect
- Automatically preserve important structures

Important vs. unimportant content

Input

Scaling

Crop

Content-aware

“less important” content

Content-aware resizing

- General ideas
  - Adhere geometric constraints (size)
  - Preserve important structures
  - Reduce image artifacts
- Weakly conditioned problem
  - What is important? / Universal importance measure?
  - Would more people agree on the process?
  - Aesthetic rating (composition, ...)?

What is important in images?

- What do people consider important?

- Fast approximation - edges

Judd et al. Learning to predict where people look. ICCV 2009

How to remove?

Optimal (pixels with least energy)

Pixels with least energy in row

Columns with least energy

slide credit: Michael Rubinstein

Image seam

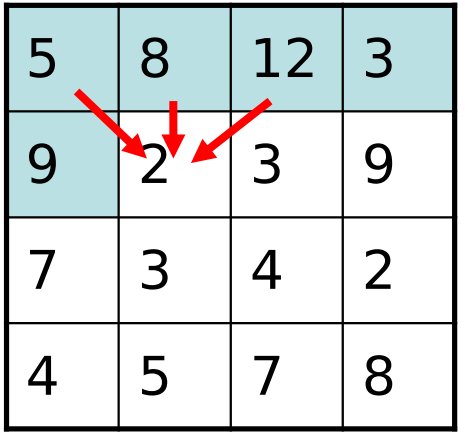
- Connected path of pixels from the top to bottom

Seam carving

- We want to shrink image in one direction
- Basic idea: remove unimportant pixels
- Unimportant = little energy = little change = little edge
- Intuition
  - Preserve strong contours
  - Human perception is more sensitive to local changes
- Simple but achieves good performance

Determining optimal seam

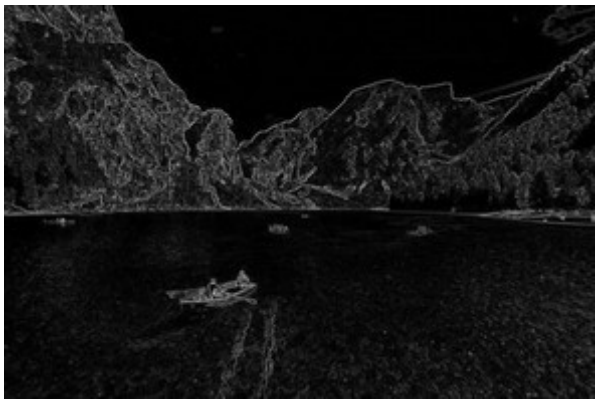
- Optimal connected path of pixels from the top to the bottom that minimizes energy
  - Dynamic programming
  - Cumulative cost
  - Backtracking

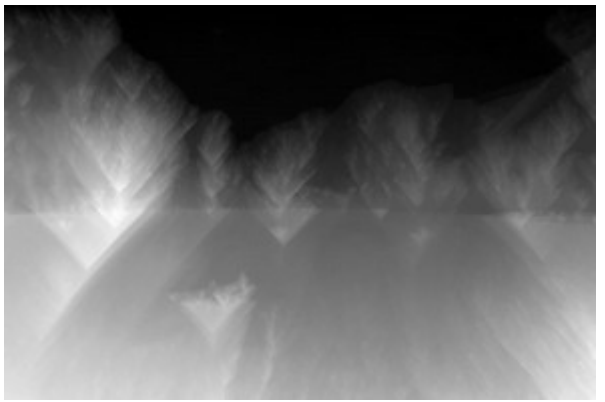


$$M(i,j) = E(i,j) + min(M(i-1,j-1), M(i-1,j), M(i-1,j+1)$$

Removing a seam

- Compute edge energy
- Compute cumulative energy
- Determine optimal seam
- Remove seam





Examples – reducing width

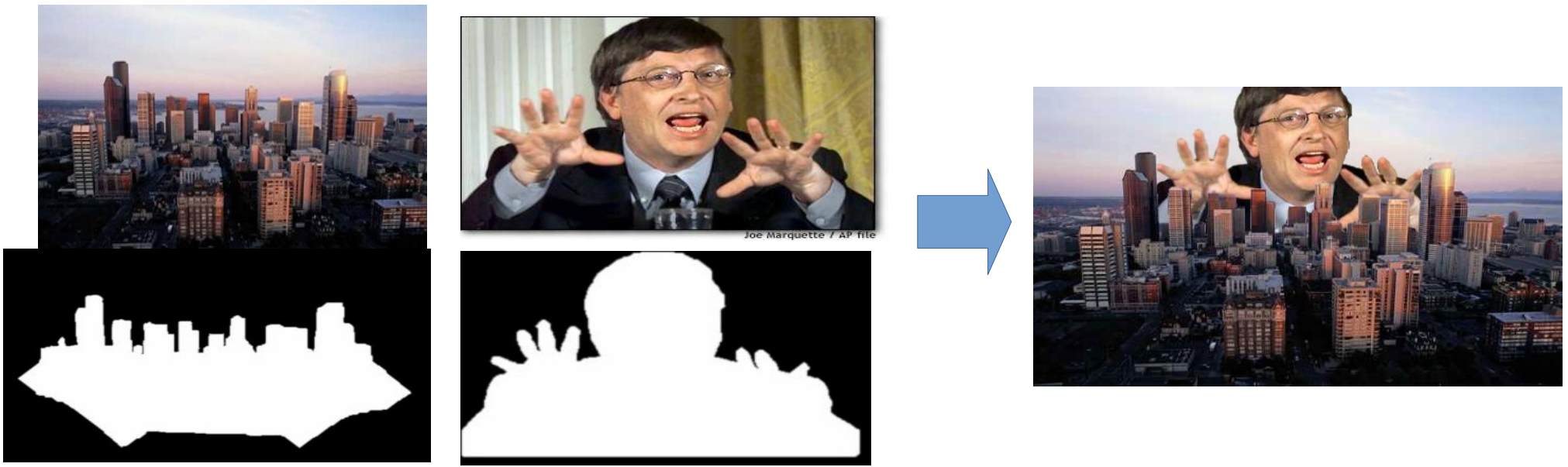
Examples – reducing height

Examples – scaling down

Interactive demo: https://www.aryan.app/seam-carving/

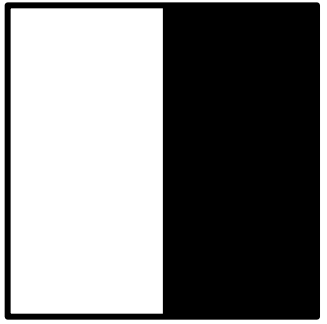
Image compositing

Combine images by taking pixels from appropriate images

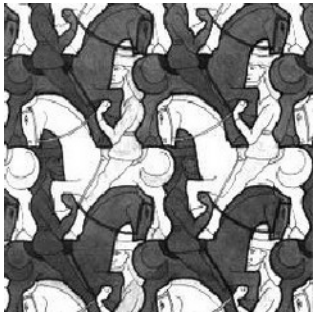


Naive combination – binary mask

A

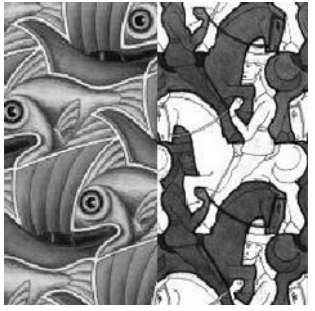


R

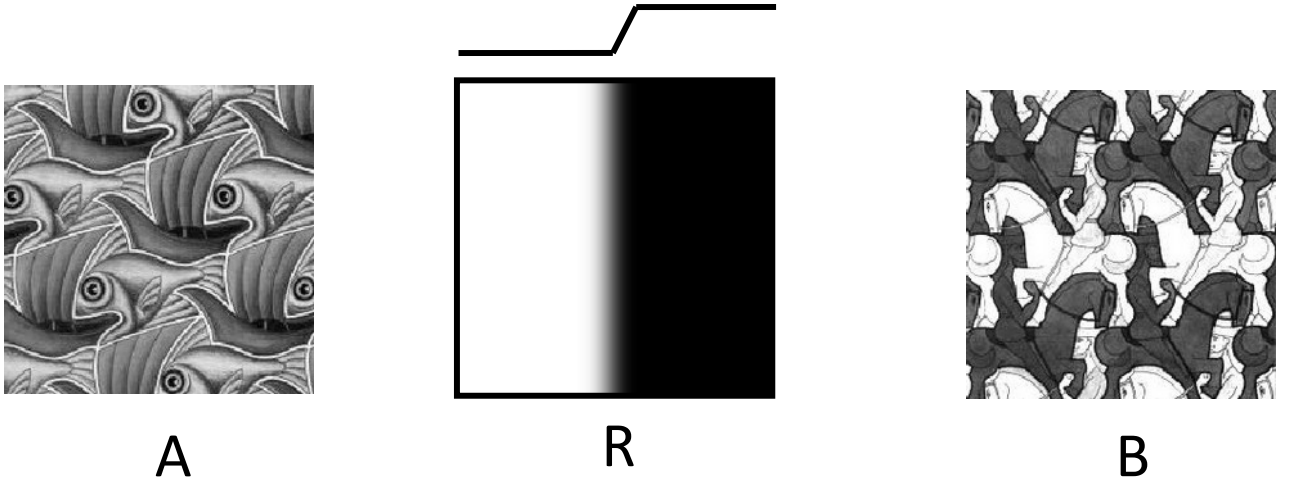


B

$$RA + (1 - R)B =$$



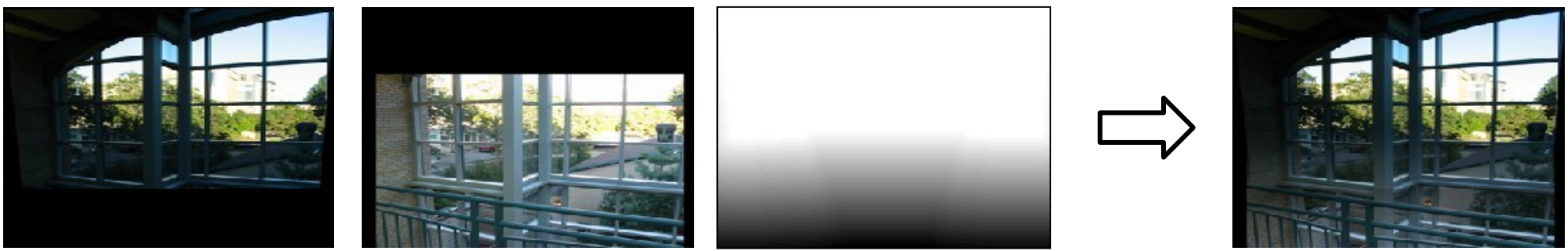
Smooth alpha channel



Example

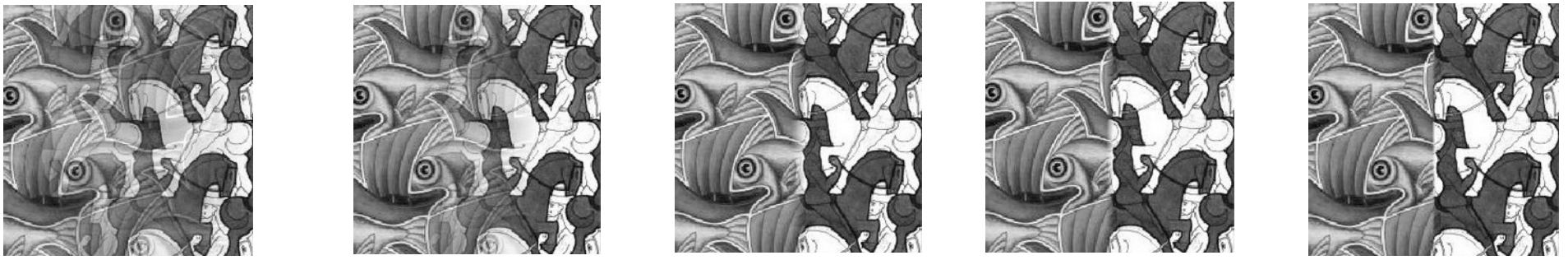
- Sharp transition – unreal image

- Smooth transition – more realistic



Smoothing influence

Very smooth transition - ghosting

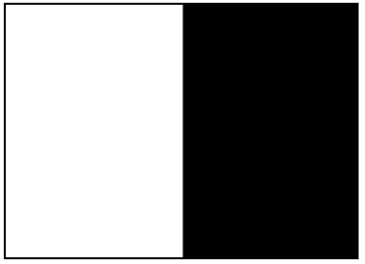
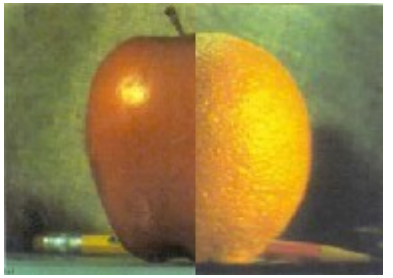


Sharp transition - cutoff

Frequency-aware blending

- Simple alpha mask blending

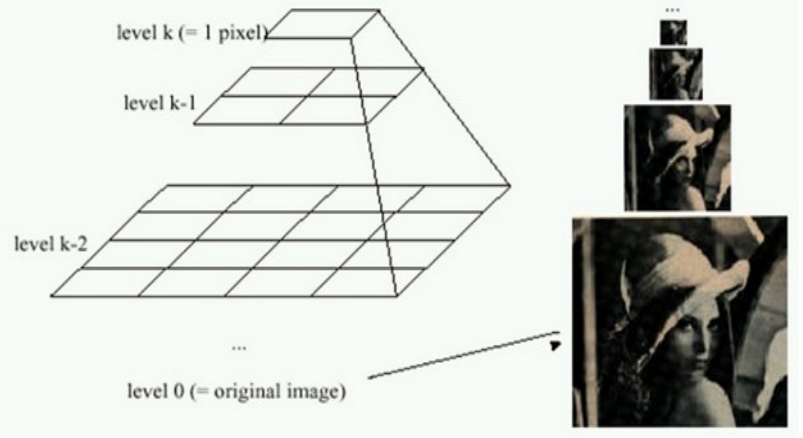


- More natural effect if we blend images by frequency bands

Image pyramids

- Multi-scale signal representation
  - Sequence of images
  - Each image only includes lower frequencies
- Gaussian pyramid
  - Smooth with Gaussian filter
  - Reduce resolution by half
  - Repeat

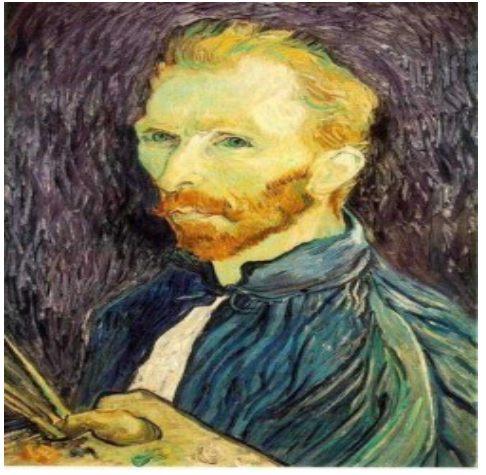


Gaussian pyramid

- Kernel size fixed
- Discard every second pixel
- Each layer removes frequency band

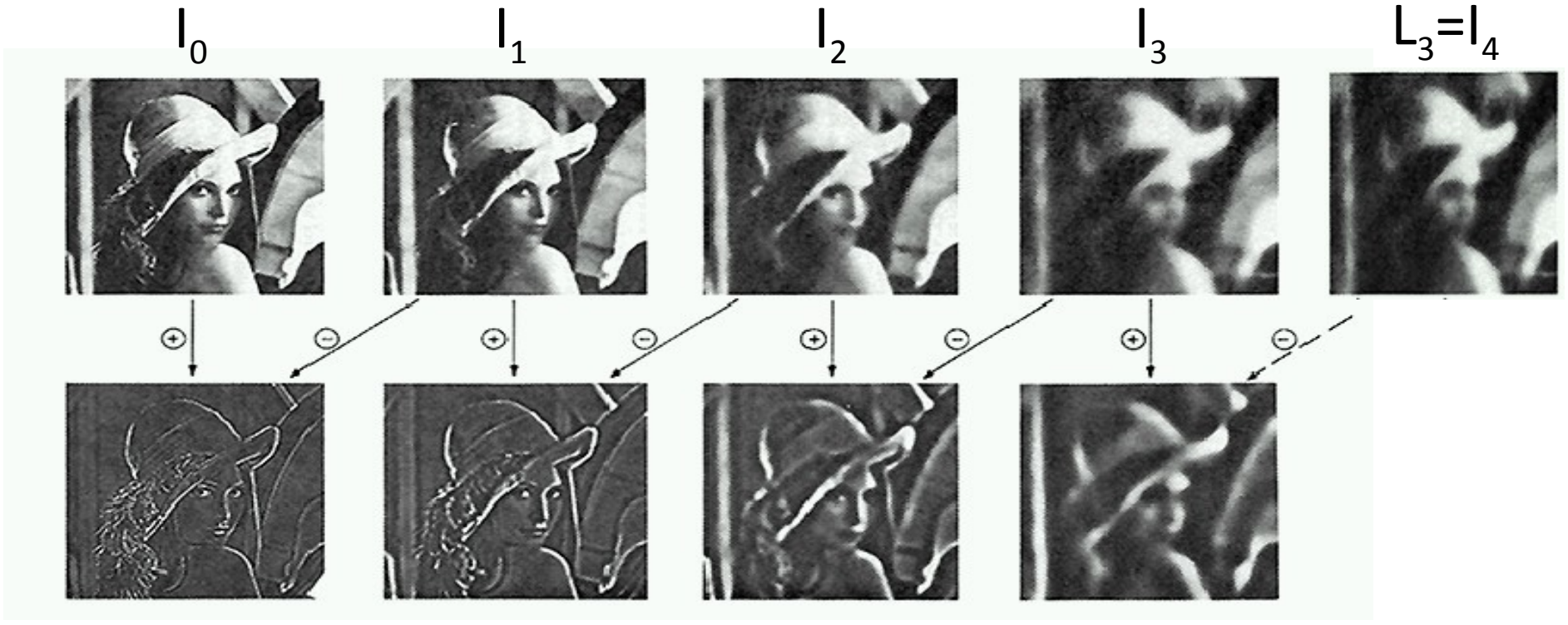
G 1/8

G 1/4



G 1/2

Laplacian pyramid

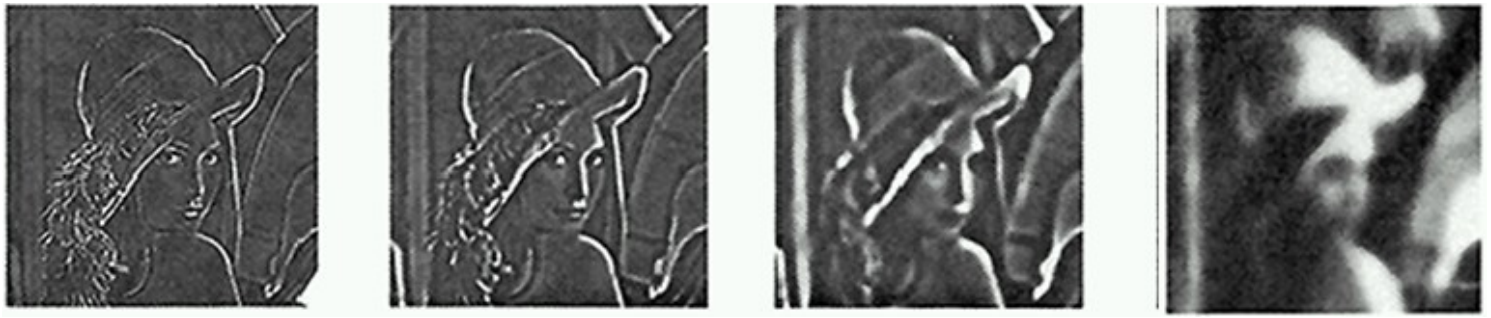


High frequencies

Medium frequencies

Low frequencies

Collapsing the pyramid



$L_0 = I_0 - I_1$ $L_1 = I_1 - I_2$ $L_2 = I_2 - I_3$ $L_3 = I_3$

Reconstruction by collapsing pyramid

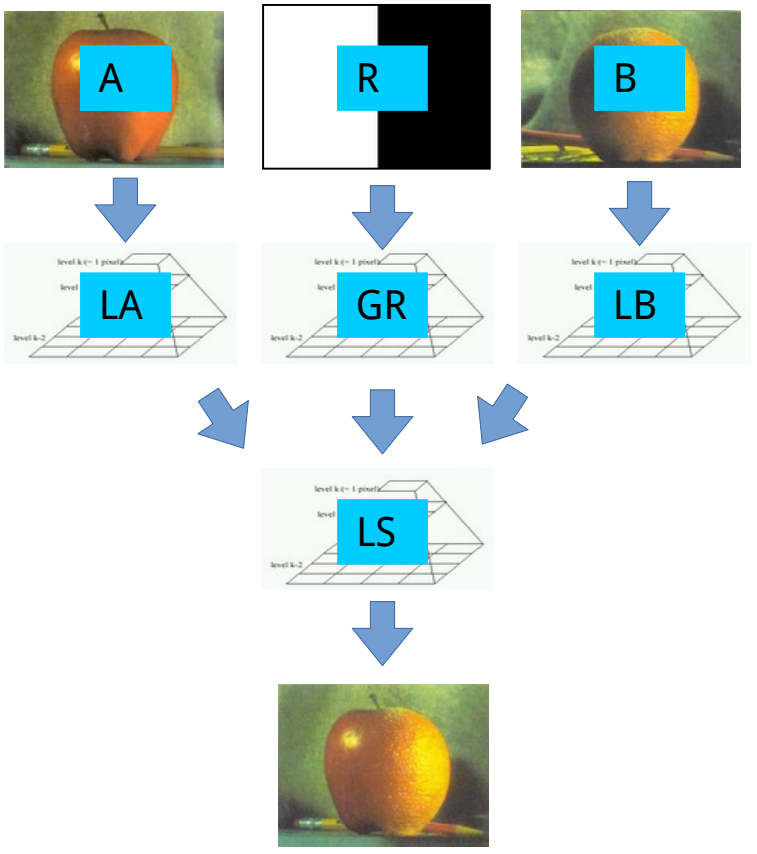
$$L_0 + L_1 + L_2 + L_3 = (I_0 - I_1) + (I_1 - I_2) + (I_2 - I_3) + I_3 = I_0$$

Laplacian blending algorithm

- Generate Laplacian pyramids LA and LB for images A and B
- Generate Gaussian pyramid GR for the alpha mask R
- Combine new Laplacian pyramid LS by combining corresponding layers from LA and LB using weights from the corresponding layer in GR:

$$LS_i = GR_i LA_i + (1 - GR_i) LB_i$$

- Collapse pyramid LS into the resulting image S



Layer 4

Layer 2

Layer 0

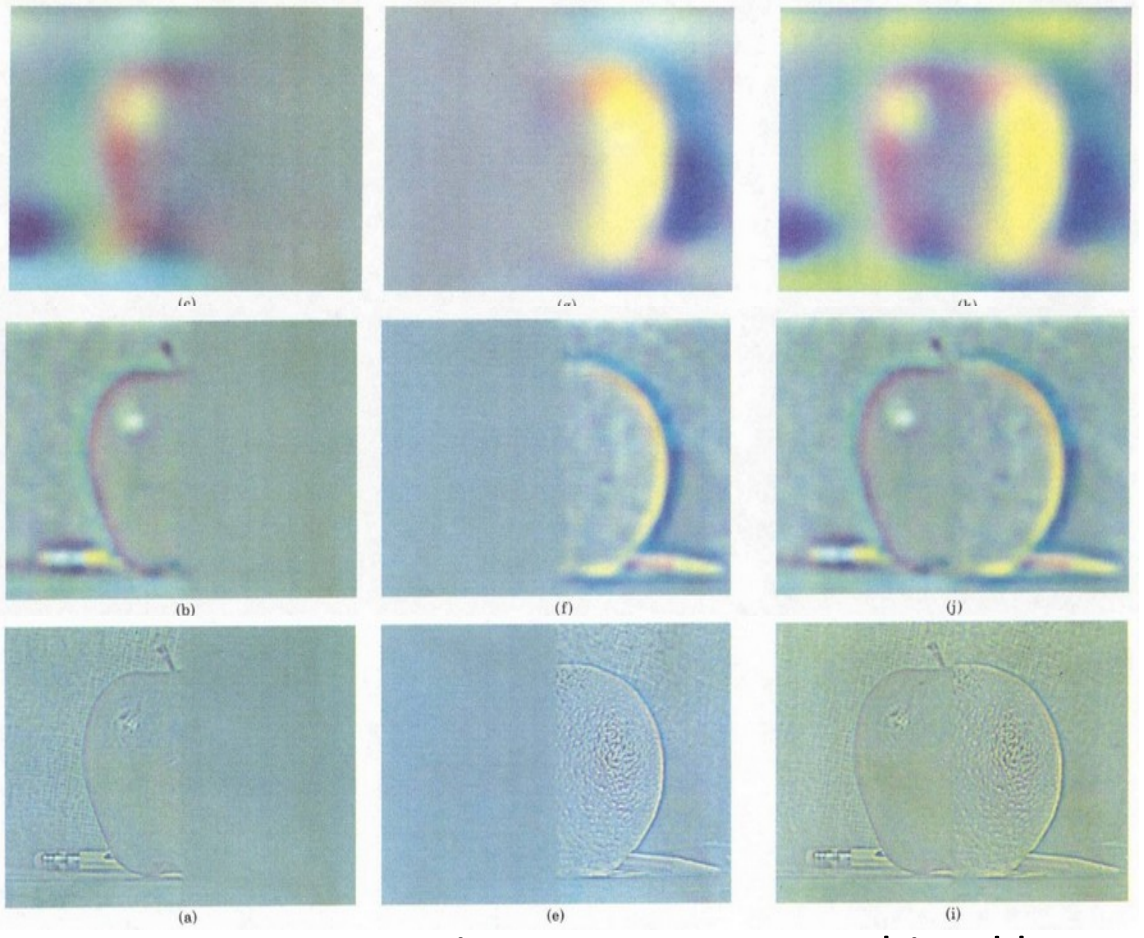
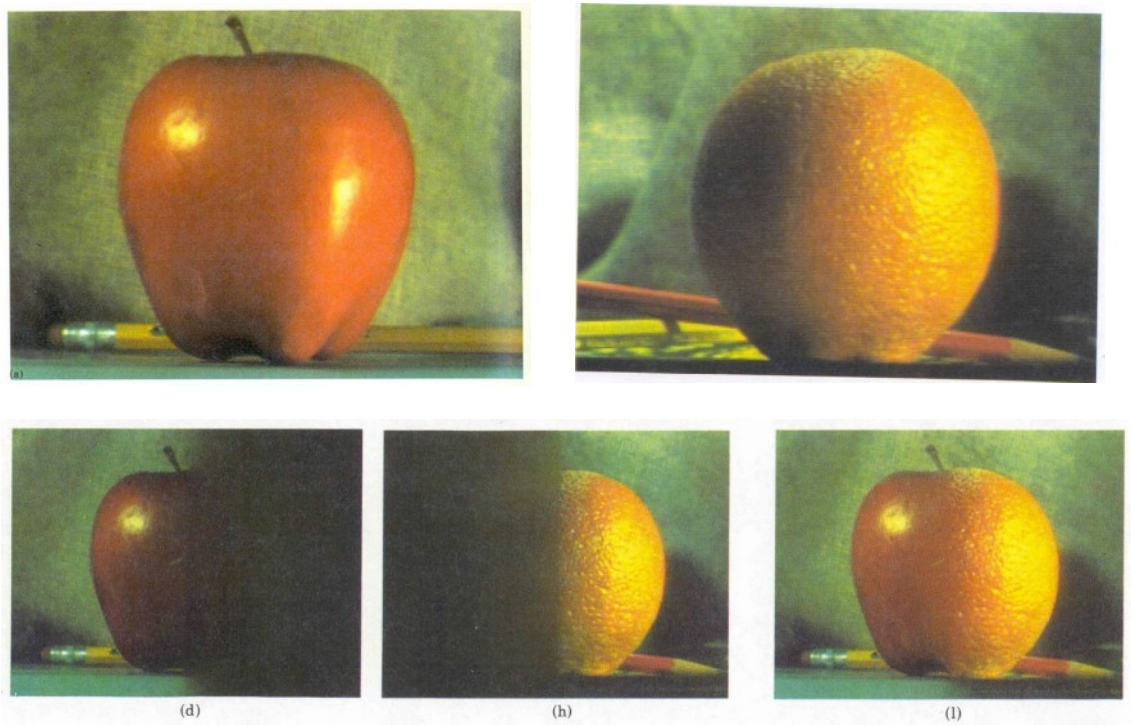


image A

image B

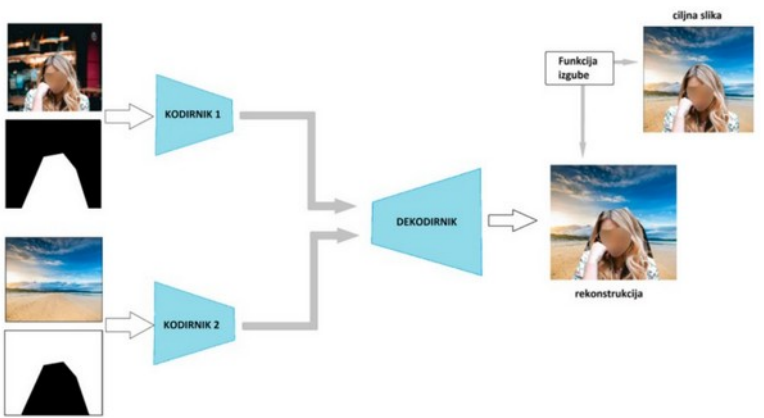
combined layer

Merging examples



Compositing using approximate masks

- Mask not following borders
  - User input
  - Low-resolution / speed
- Improve result using a deep model
  - Train using degraded masks as input
  - Compare to reference composite

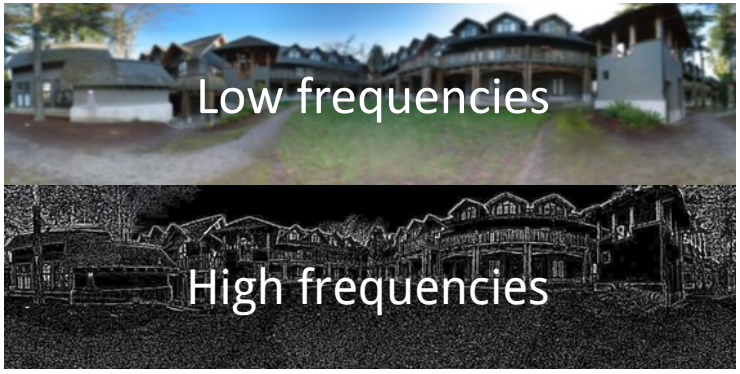


Ema Črne, Združevanje slik na podlagi nenatančnih mask, 2023

Examples

Merging examples - Autostitch

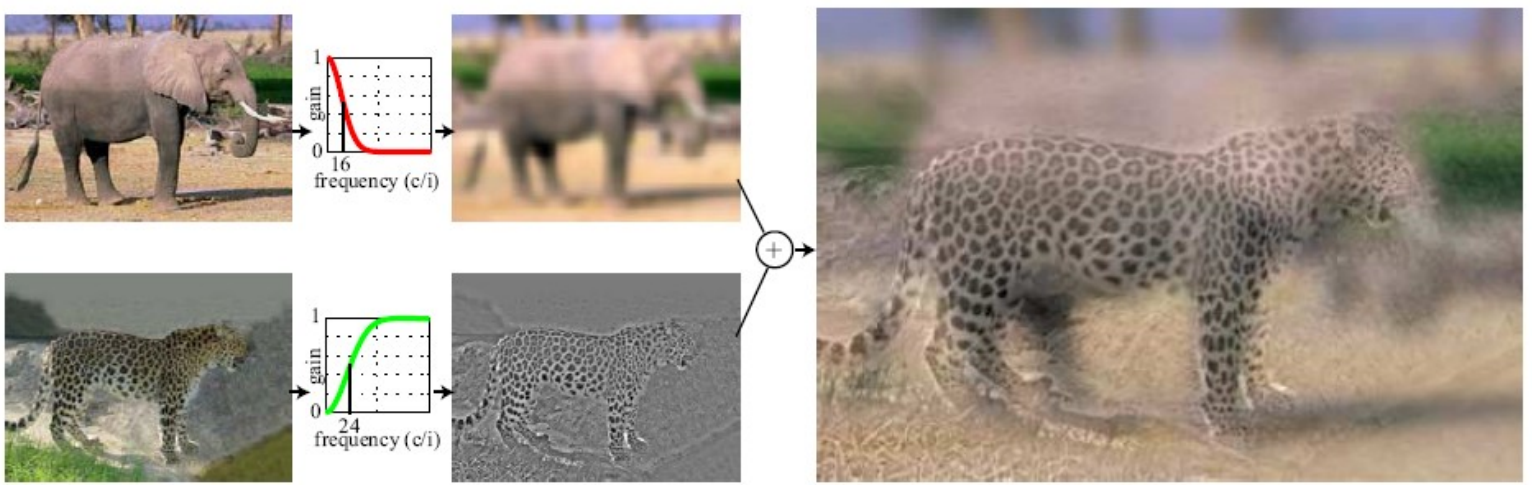
- Stitching panorama from multiple images
- Two-layer blending – high and low frequencies
- Only blend low frequencies, keep high frequencies intact



Matthew Brown and David G. Lowe, "Automatic panoramic image stitching using invariant features," International Journal of Computer Vision, 74, 1 (2007), pp. 59-73

Hybrid images

- Static images with two interpretations
  - Low frequencies – far away
  - High frequencies - nearby



A. Oliva, A. Torralba, P.G. Schyns, “Hybrid Images,” SIGGRAPH 2006

Hybrid images - examples

A. Oliva, A. Torralba, P.G. Schyns, “Hybrid Images,” SIGGRAPH 2006