컴퓨터 비전 세미나

12191584 김창수

Review

Q. What do Linear and Non-Linear mean?

- the output is directly proportional to the input
- the resulting system subjected to a complex input
  can be described as a sum of responses to simpler inputs
- Because linear systems are so much easier to analyze, predict and solve than nonlinear ones, much research is devoted to finding linear approximations of nonlinear phenomena.

Q. Why Convolutional-NN is not called Correlation-NN ?

- They are equivalent because the weights are initialized and updated in the same way.
- Using "convolution" in the name emphasizes the ties to signal processing and traditional image filtering methods.

Index

More neighborhood operators

Fourier transform

Geometric transformations

Non-linear filtering

- The filters we have looked at so far have all been linear.

- In many cases, better performance can be obtained by using a non-linear combination of neighboring pixels.

Median filtering

- selects the median value from each pixel's neighborhood

- it selects only one input pixel value to replace each output pixel, it is not as efficient at averaging away regular Gaussian noise

alpha-trimmed mean

- averages together all of the pixels except for the fraction

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

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

weighted median

- the weight is the number of times depending on its distance from the center
- robust against outliers

- Non-linear filtering is more edge preserving,
  it has less tendency to soften edges while filtering away high-frequency noise

Bilateral filtering

- instead of rejecting a fixed percentage, bilateral filtering reject pixels whose values differ too much from the central pixel value

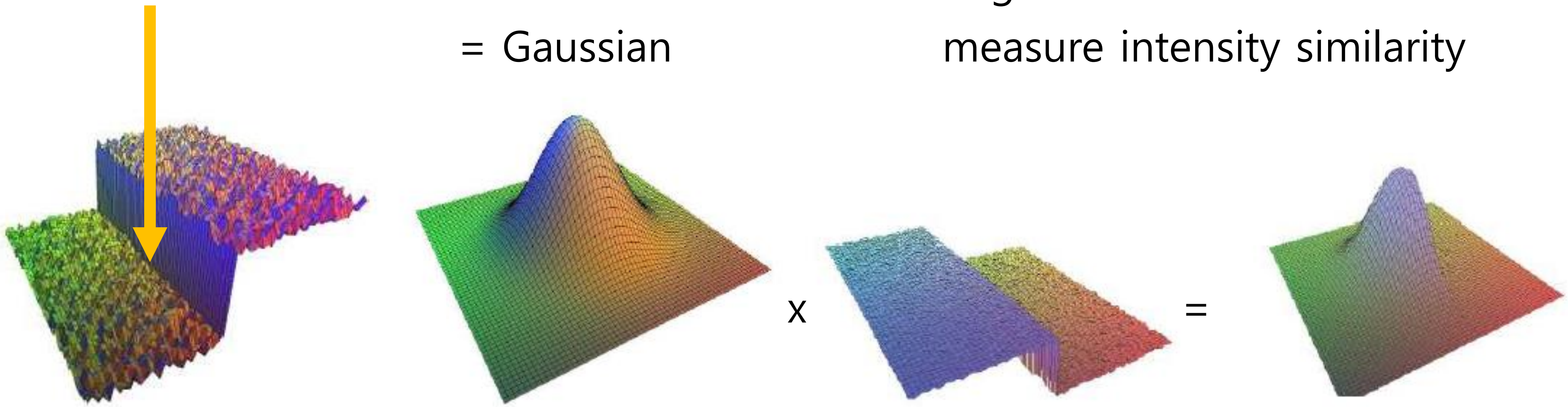
$$w(i,j,k,l) = \exp\left(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2} - \frac{\|\mathbf{f}(i,j)-\mathbf{f}(k,l)\|^2}{2\sigma_r^2}\right)$$

domain kernel

= Gaussian

range kernel

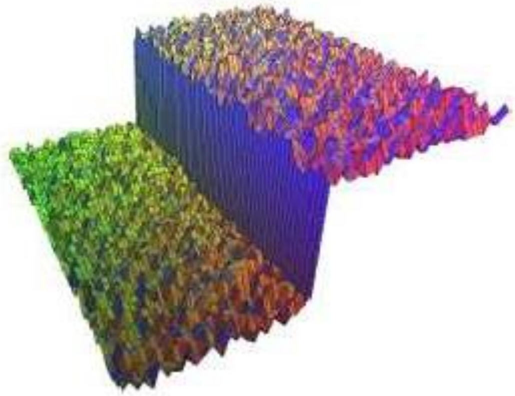
measure intensity similarity



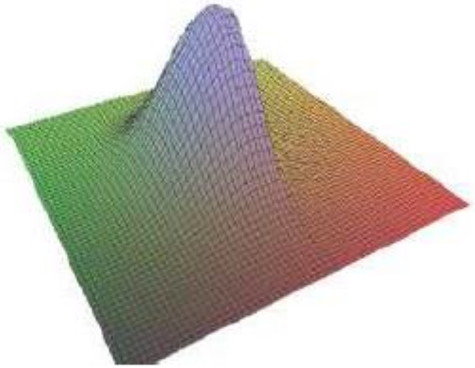
input

kernel

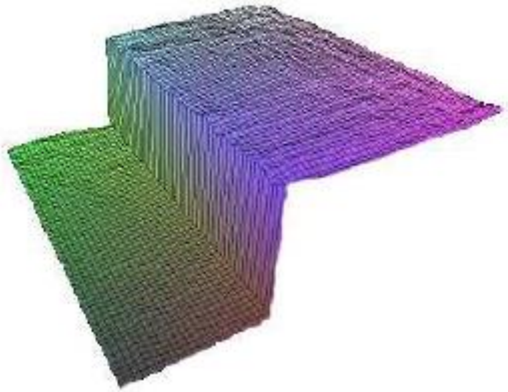
Bilateral filtering



input



kernel



output

- Since bilateral filtering is slow, a number of acceleration techniques have been developed

Binary image processing

- While non-linear filters are often used to enhance color images, they are also used extensively to process binary images.

- Such images often occur after a thresholding operation.

$$\theta(f,t) = \begin{cases} 1 & \text{if } f \geq t, \\ 0 & \text{else,} \end{cases}$$

Morphology

- Morphology changes the shape of the underlying binary objects.

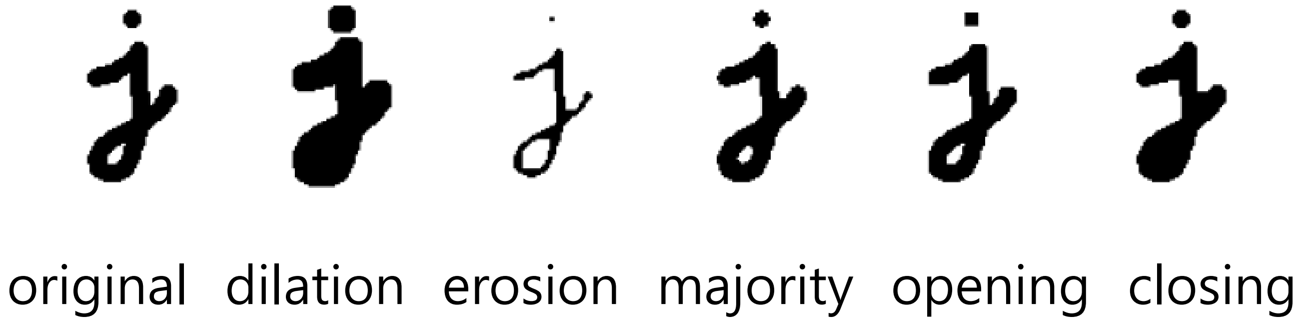
- first convolve the binary image with a binary structuring element
- then select a binary output value depending on the thresholded result of the convolution

- The structuring element can be any shape, from a simple 3 x 3 box filter, to more complicated disc structures.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Morphology

- **dilation**: $\text{dilate}(f, s) = \theta(c, 1)$;
- **erosion**: $\text{erode}(f, s) = \theta(c, S)$;
- **majority**: $\text{maj}(f, s) = \theta(c, S/2)$;
- **opening**: $\text{open}(f, s) = \text{dilate}(\text{erode}(f, s), s)$;
- **closing**: $\text{close}(f, s) = \text{erode}(\text{dilate}(f, s), s)$.



Opening fails to eliminate the dot, since it is not wide enough.

- dilation grows objects
- erosion shrinks them
- majority smooth sharp corners
- opening and closing operations tend to remove small objects or holes

Distance transforms

distance transform D(i, j) of a binary image b(i, j), d(k, l) is some distance metric

$$D(i,j) = \min_{k,l:b(k,l)=0} d(i-k, j-l),$$

it is the distance to the nearest background pixel whose value is 0.

Manhattan distance

$$d_1(k,l) = |k| + |l|$$

Euclidean distance

$$d_2(k,l) = \sqrt{k^2 + l^2}.$$

Distance transforms

0	0	0	0	1	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0
0	0	1	0	0	0	0
0	0	0	0	0	0	0

(a)

0	0	0	0	1	0	0
0	0	1	1	2	0	0
0	1	2	2	3	1	0
0	1	2	3			

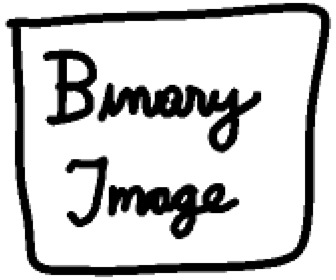
(b)

0	0	0	0	1	0	0
0	0	1	1	2	0	0
0	1	2	2	3	1	0
0	1	2	2	1	1	0
0	1	2	1	0	0	0
0	0	1	0	0	0	0
0	0	0	0	0	0	0

(c)

0	0	0	0	1	0	0
0	0	1	1	1	0	0
0	1	2	2	2	1	0
0	1	2	2	1	1	0
0	1	2	1	0	0	0
0	0	1	0	0	0	0
0	0	0	0	0	0	0

(d)



The ridges in the distance transform become the skeleton
and consist of pixels that are of equal distance to their boundaries

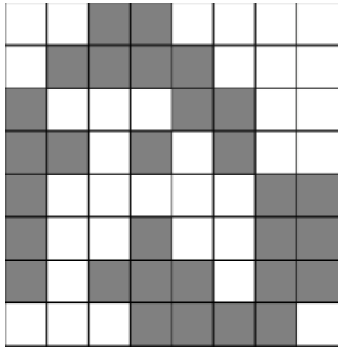
Connected components

- finding connected components, which are defined as regions of adjacent pixels that have the same input value or label

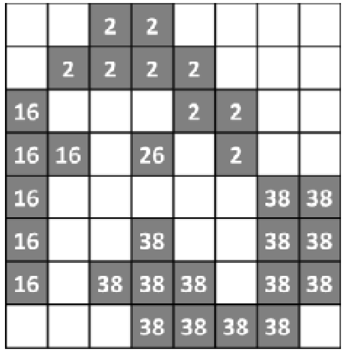
- N4 adjacent(+), N8 adjacent(ㅁ)

- it is often useful to compute the area statistics for each individual region
    - the area (number of pixels)
    - the perimeter (number of boundary pixels)
    - the centroid (average x and y values)
    - the second moments

- DFS, stack



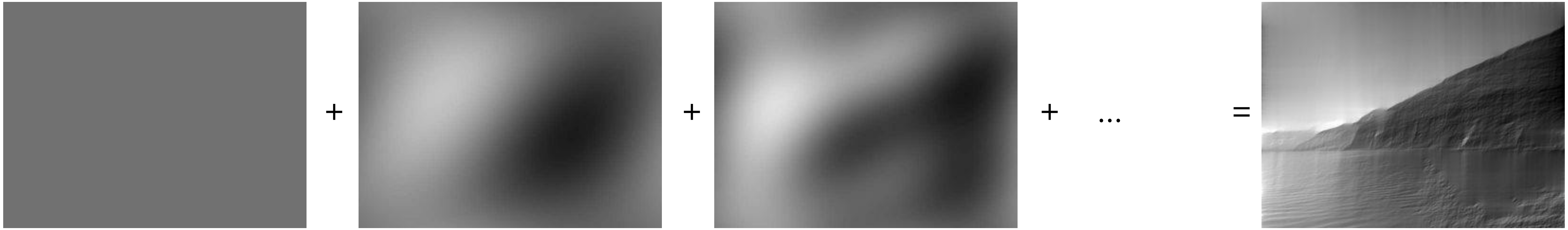
(a)



(b)

Image frequency

- The image is made up of various waves or frequencies



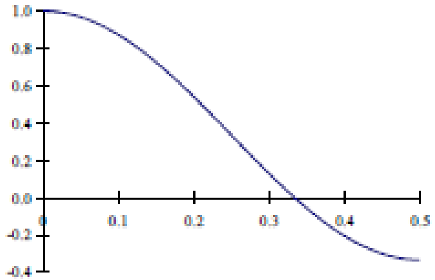
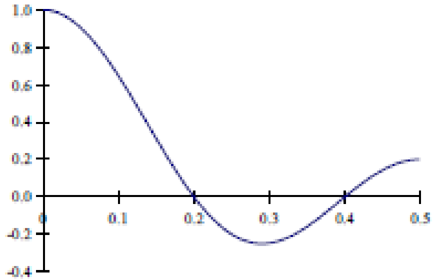
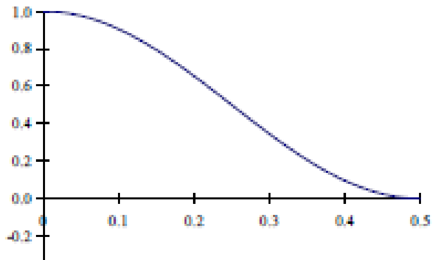
- The high frequency means rapid change of values which happens when there is sharp contrast in the image, such as edges.
- The low frequency means slow change in pixel values which corresponds to plain areas in images

- If an image has large values at high frequency components, then the data is changing rapidly on a short distance scale. e.g. a page of text, sharp image

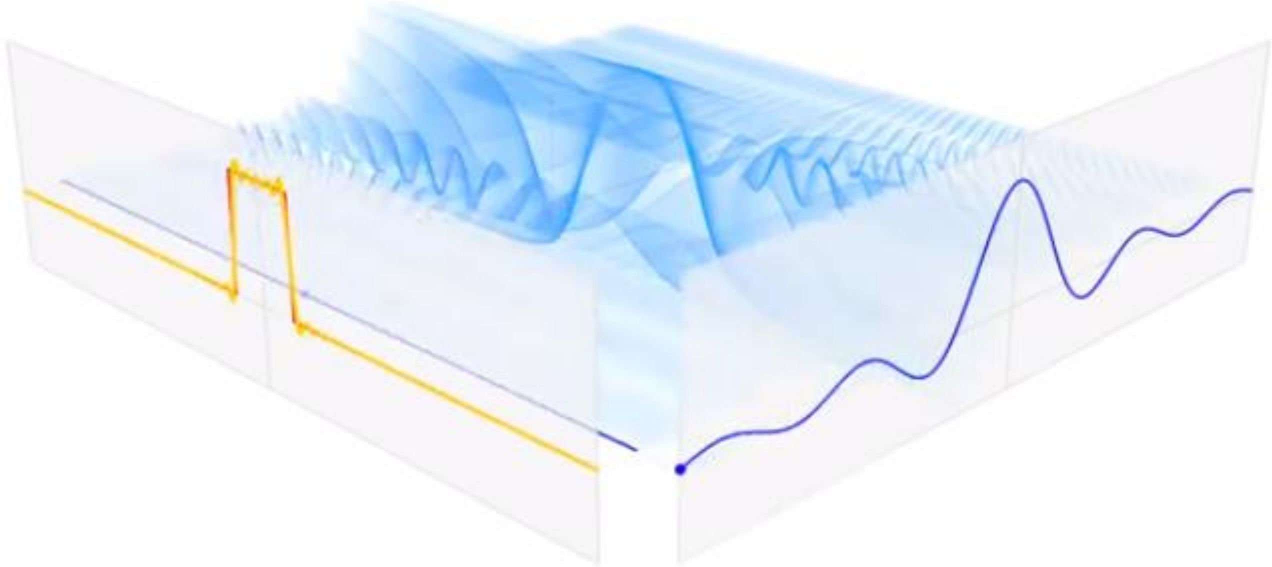
Fourier transforms

- decomposes functions depending on space(2D) or time(1D) into functions depending on spatial or temporal frequency

- any signal can be represented as a sum of periodic components with varying frequencies

box-3	$\frac{1}{3}\boxed{1 \mid 1 \mid 1}$	$\frac{1}{3}(1+2\cos\omega)$	
box-5	$\frac{1}{5}\boxed{1 \mid 1 \mid 1 \mid 1 \mid 1}$	$\frac{1}{5}(1+2\cos\omega+2\cos 2\omega)$	
linear	$\frac{1}{4}\boxed{1 \mid 2 \mid 1}$	$\frac{1}{2}(1+\cos\omega)$	

Fourier transforms



$$f \star g \overset{\text{F.T}}{\multimap} \mathcal{F} \cdot \mathcal{G}$$

Time domain or Space domain **Frequency domain**

Where $\mathcal{F} = Fourier\_Transform(\boldsymbol{f})$

- Convolving two functions corresponds to the product of them in frequency domain.

Wiener filter

- reduce the amount of noise present in a signal by comparison with an estimation of the desired noiseless signal

- The Wiener filter is based on a statistical approach, based on minimum mean square error (MMSE)

- assuming that an image is a sample from a correlated Gaussian random noise field
  combined with a statistical model of the measurement process
  yields an optimum restoration filter known as the Wiener filter

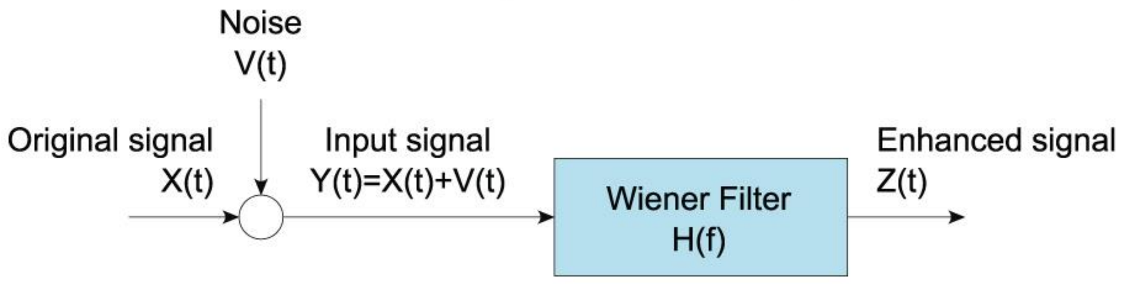
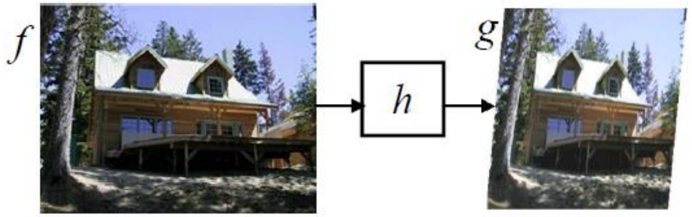
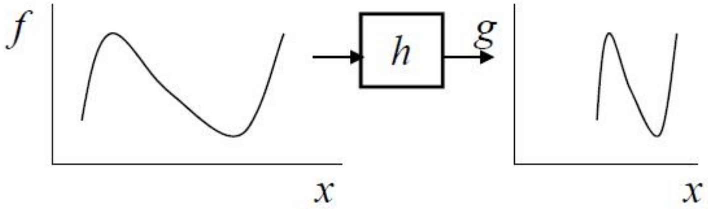


Figure 1. Optimal (Wiener) filtering signal flow

Geometric transformations

- more general transformations, such as image rotations or general warps
- In contrast to the point processes, which transform the range of the image, here we look at functions that transform the domain

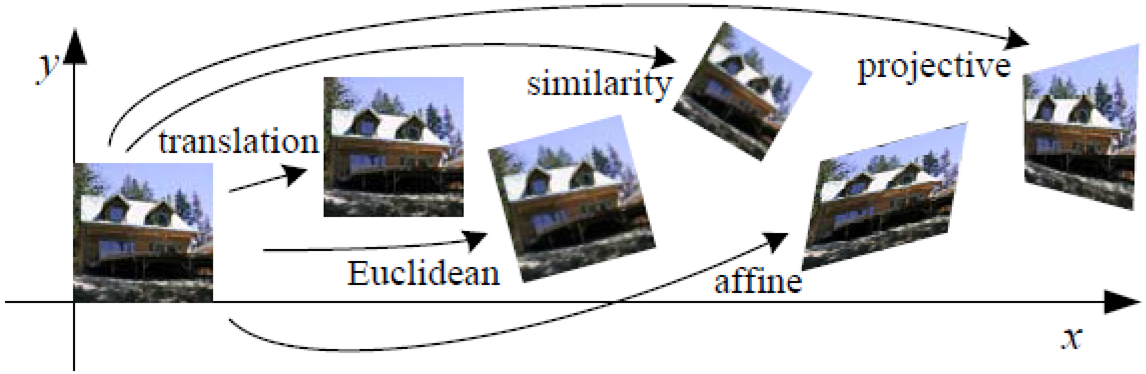




Parametric transformations

- global deformation to an image, where the behavior of the transformation is controlled by a small number of parameters

Parametric transformations



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

Parametric transformations

- translation

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$

- rigid

$$\begin{bmatrix} cos(\theta) & -sin(\theta) & t_x \\ sin(\theta) & cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

- similarity

$$\begin{bmatrix} s\ cos(\theta) & -s\ sin(\theta) & t_x \\ s\ sin(\theta) & s\ cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

- affine

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

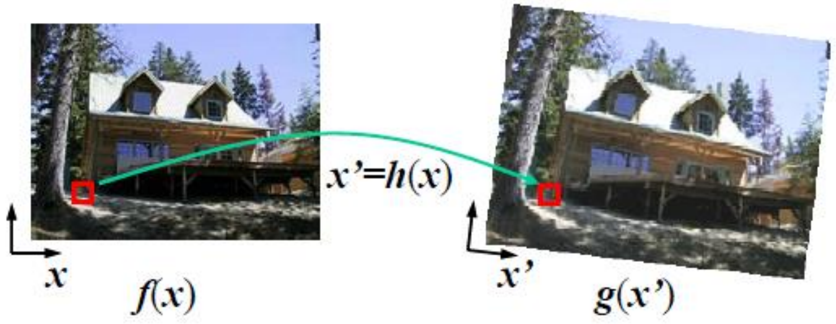
- projective

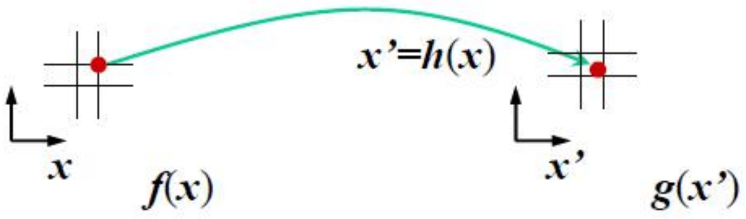
$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

- The 2 x 3 matrices are extended with a third [0^T 1] row to form a full 3 x 3 matrix for homogeneous coordinate transformations.

Forward warping

- given a transformation specified by a formula x′ = h(x)
- Compute the destination location x′ = h(x).
- Copy the pixel f(x) to g(x′).





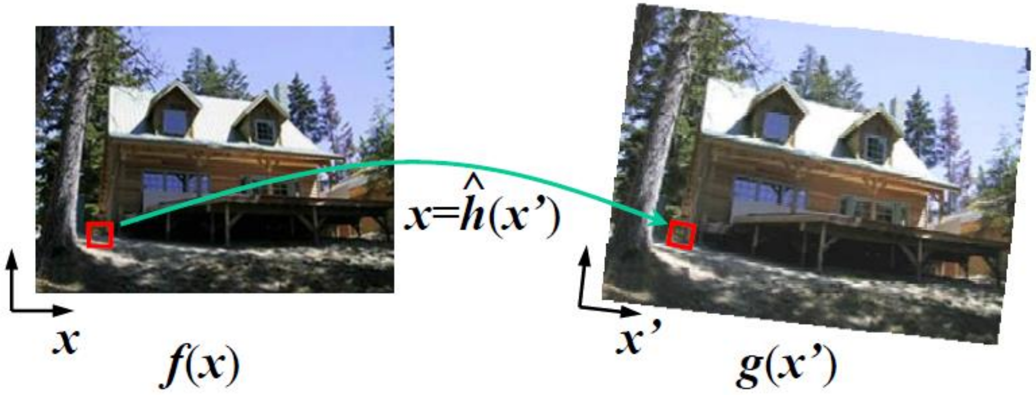
Forward warping

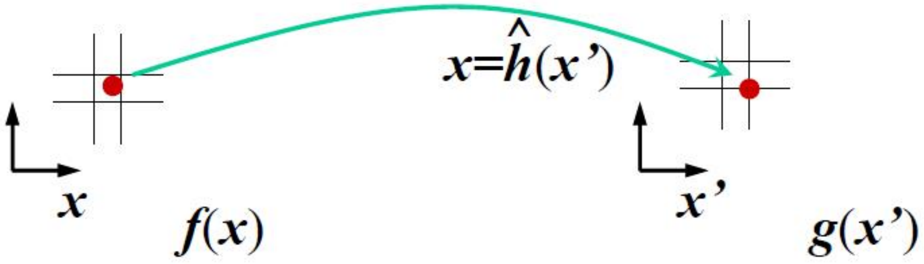
- this approach suffers from several limitations.
- copying a pixel f(x) to a location x′ in g is not well defined when x′ has a non-integer value

- round the value of x′ to the nearest integer coordinate and copy the pixel there
- distribute the value among its four nearest neighbors in a weighted fashion

Inverse warping

- ^h(x’) is presumably defined for all pixels in g(x’)
- x = ^h(x’) can simply be computed as the inverse of h(x)
- Resample f(x) at location x and copy to g(x’)





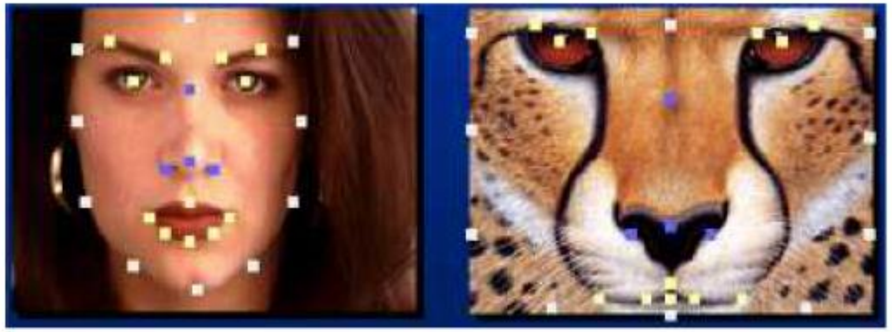
Mesh-based warping

- changing the appearance of a face from a frown to a smile

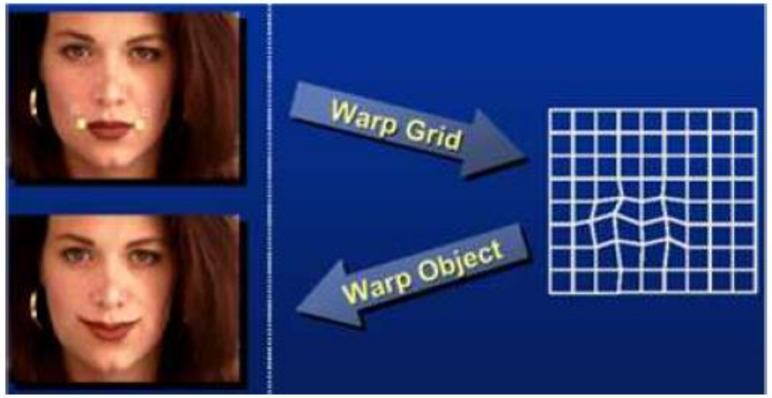
- To perform such a transformation, different amounts of motion are required in different parts of the image.

Mesh-based warping

- first approach is to specify a sparse set of corresponding points
- displacement of these points can then be interpolated to a dense displacement field

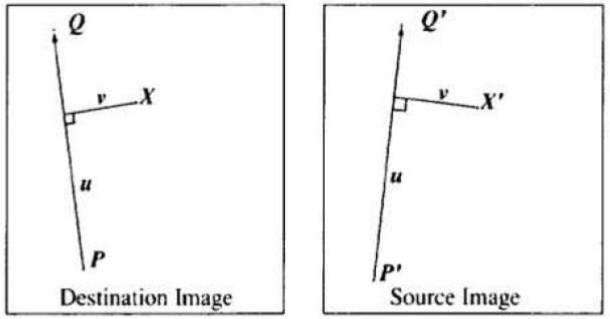


- interpolating a sparse set of displacements include moving nearby quadrilateral mesh vertices
- If quadrilateral meshes are used, it may be desirable to interpolate displacements down to individual pixel values



Mesh-based warping

- second approach is specifying displacements for local deformations is to use corresponding oriented line segments
- Each line segment correspondence specifies a translation, rotation, and scaling
- Pixels along each line segment are transferred from source to destination exactly as specified



- final approach is using a mesh specifically adapted to the underlying image content
- Once the two meshes have been specified, intermediate warps can be generated using linear interpolation and the displacements