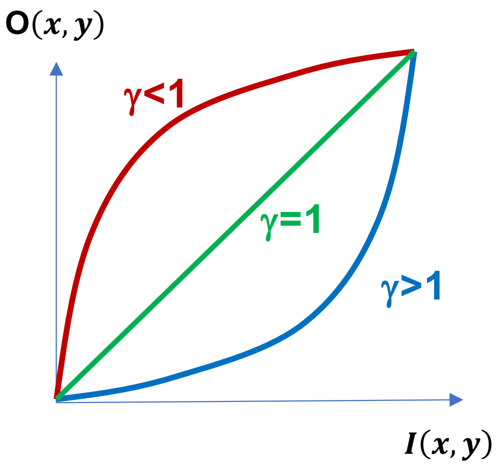
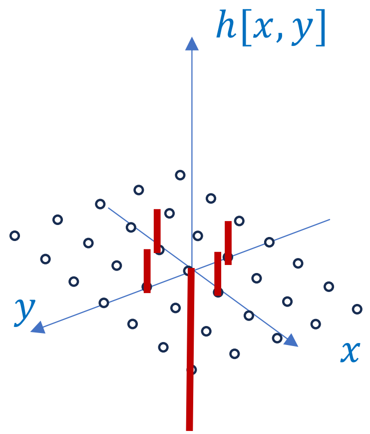
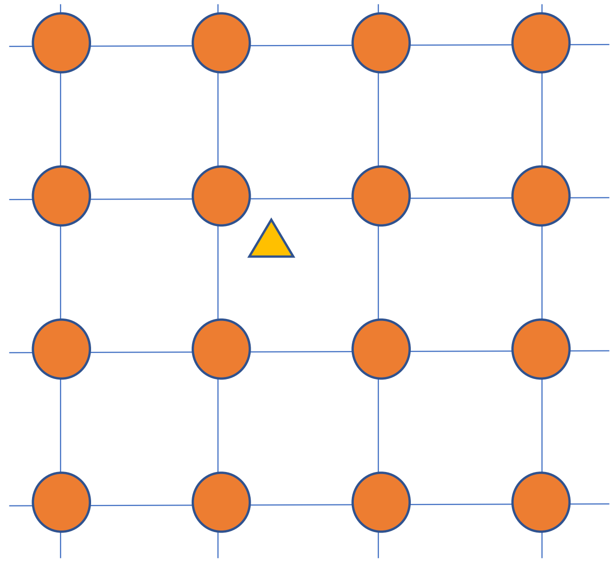




0	1	0
1	-4	1
0	1	0

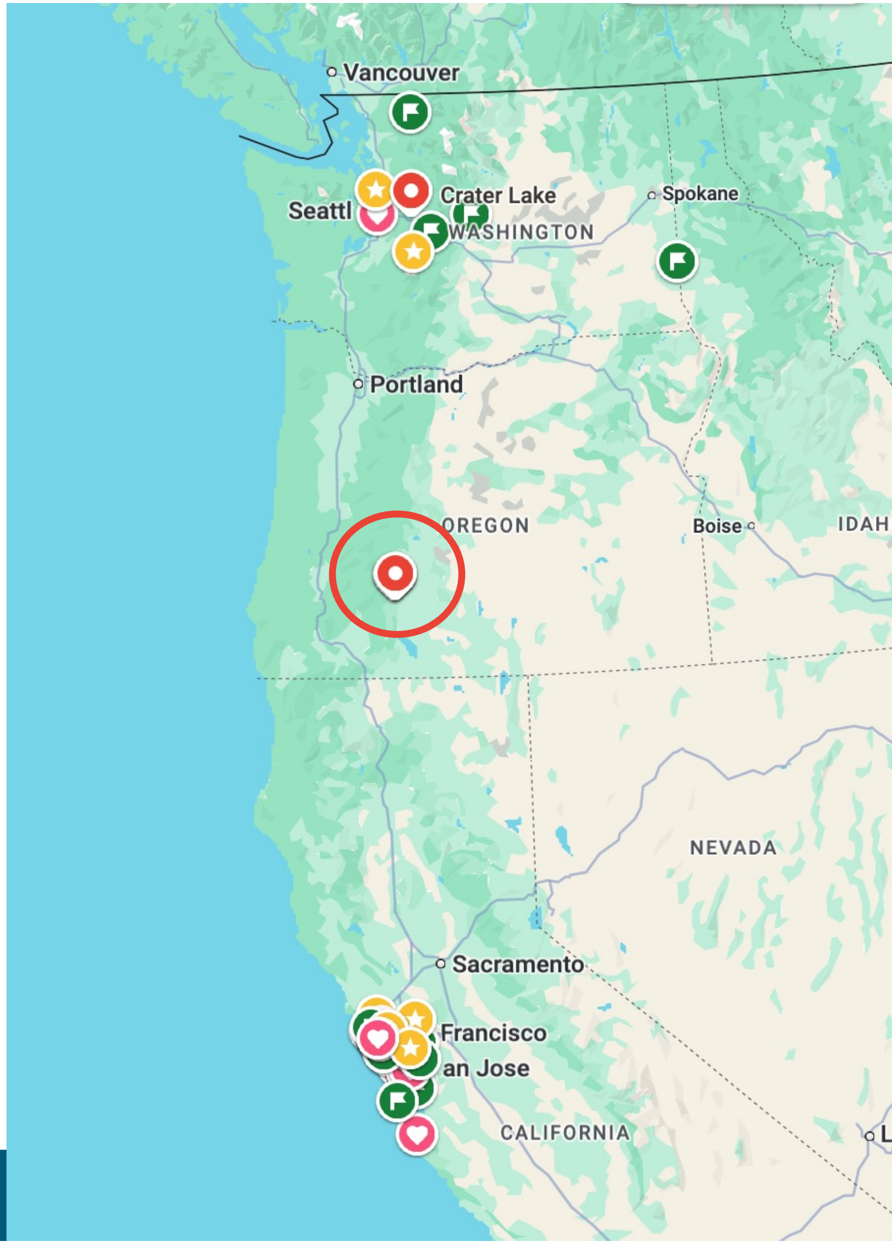


Spatial-domain Operators

Fall 2024

Yi-Ting Chen

https://app.sli.do/event/eeZ4AW8Bz8TzUUug1VTvmb



Vancouver
Seattl
Crater Lake
WASHINGTON
Spokane
Portland
OREGON
Boise
IDAH
NEVADA
Sacramento
Francisco
an Jose
CALIFORNIA

51-51 !!

What do you want to be?

Could we bring values to others?


PURDUE

https://youtu.be/8pRD9owFfkg?t=3684

“Identifiable Area of Impact”

Pick an important problem you like and become known for the work you’ve done

Popular Recent 2 questions

Anonymous
6 days ago

0

Are the demosaicing process processes on all sensors include cameras? or are only used in printer or scanners?

Anonymous
6 days ago

0

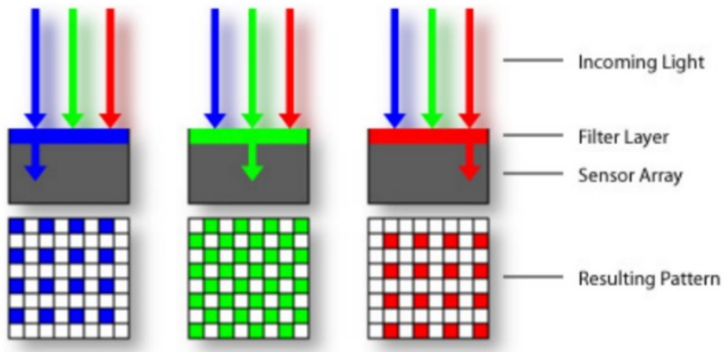
Are those simple or complex cells trained when we are baby? or it is our human talent that inherits?

Color Filter Arrays

- To measure color, pixels are arranged in a color array, e.g., Bayer RGB pattern
- Missing colors at each pixel are interpolated from the neighbors (demosaicing)

G	R	G	R
B	G	B	G
G	R	G	R
B	G	B	G

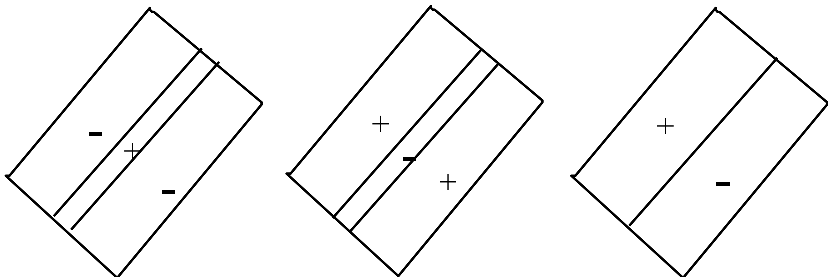
Bayer RGB Pattern

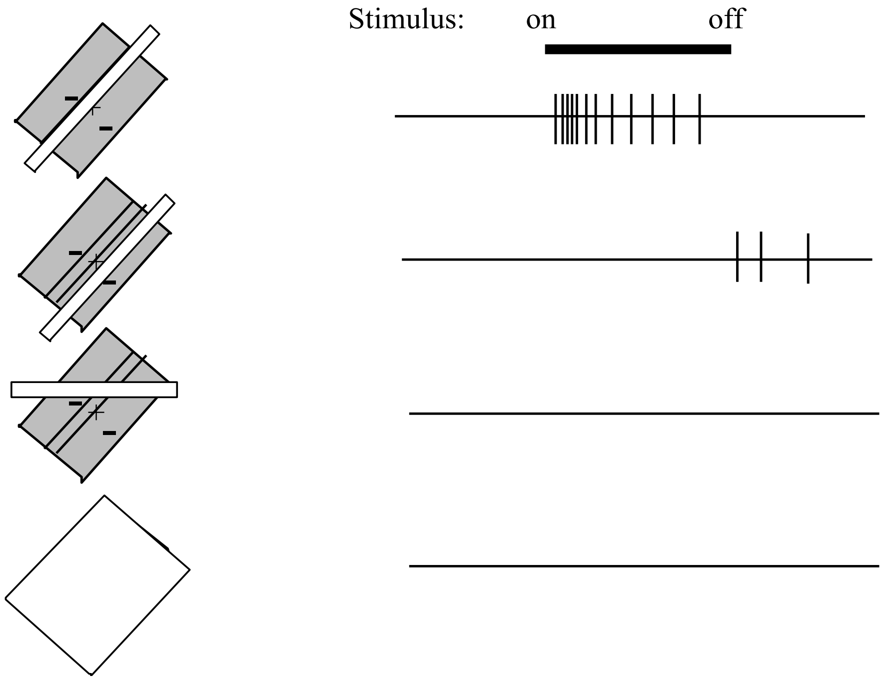


Why there are more green channels?

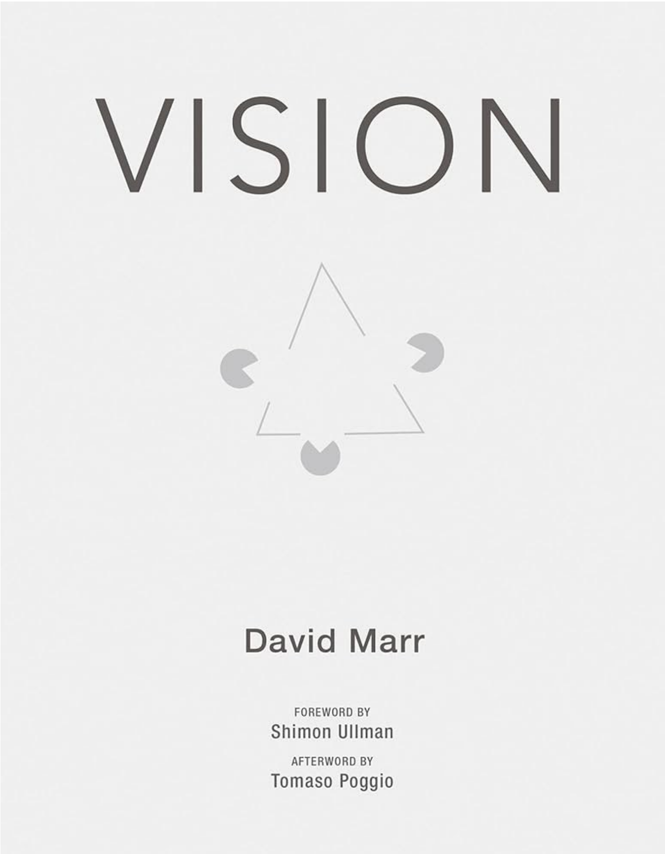
Simple Cells

- Orientation Selectivity





Vision, David Marr, 1982



Computer Vision Awards

The computer vision community gives out a variety of awards at major vision meetings. These awards are explained below, with a complete listing of winners for each following.

Conference Best Paper Awards

CVPR Best Paper Award

CVPR Best Student Paper Award

CVPR Best Paper Honorable Mention Award

ICCV Best Paper Award (Marr Prize)

ICCV Best Student Paper Award

ICCV Best Paper Honorable Mention Award

ECCV Best Paper Award

ECCV Best Paper Honorable Mention Award

https://en.wikipedia.org/wiki/David_Marr_(neuroscientist)

Vision as Information Processing System

Computational theory	Representation and algorithm	Hardware implementation
What is the goal of the computation, why is it appropriate, and what is the logic of the strategy by which it can be carried out?	How can this computational theory be implemented? In particular, what is the representation for the input and output, and what is the algorithm for the transformation?	How can the representation and algorithm be realized physically?

Figure 1–4. The three levels at which any machine carrying out an information-processing task must be understood.

Table 1–1. Representational framework for deriving shape information from images.

Name	Purpose	Primitives
Image(s)	Represents intensity.	Intensity value at each point in the image
Primal sketch	Makes explicit important information about the two-dimensional image, primarily the intensity changes there and their geometrical distribution and organization.	Zero-crossings Blobs Terminations and discontinuities Edge segments Virtual lines Groups Curvilinear organization Boundaries
2½-D sketch	Makes explicit the orientation and rough depth of the visible surfaces, and contours of discontinuities in these quantities in a viewer-centered coordinate frame.	Local surface orientation (the "needles" primitives) Distance from viewer Discontinuities in depth Discontinuities in surface orientation
3-D model representation	Describes shapes and their spatial organization in an object-centered coordinate frame, using a modular hierarchical representation that includes volumetric primitives (i.e., primitives that represent the volume of space that a shape occupies) as well as surface primitives.	3-D models arranged hierarchically, each one based on a spatial configuration of a few sticks or axes, to which volumetric or surface shape primitives are attached

Three Layer Hypothesis

- Computational level: what does the system do (e.g.: what problems does it solve or overcome) and similarly, why does it do these things
- Algorithmic level (sometimes representational level): how does the system do what it does, specifically, what representations does it use and what processes does it employ to build and manipulate the representations
- Implementational/physical level: how is the system physically realised (in the case of biological vision, what neural structures and neuronal activities implement the visual system)

Three Layer Hypothesis

- Computational level: what does the system do (e.g.: what problems does it solve or overcome) and similarly, why does it do these things
- Algorithmic level (sometimes representational level): how does the system do what it does, specifically, what representations does it use and what processes does it employ to build and manipulate the representations
- Implementational/physical level: how is the system physically realised (in the case of biological vision, what neural structures and neuronal activities implement the visual system)

Marr: 2D Image → 3D Model of the World

- Marr describe vision as processing from a two-dimensional visual array (on the retina) to a three-dimensional description of the world as output.

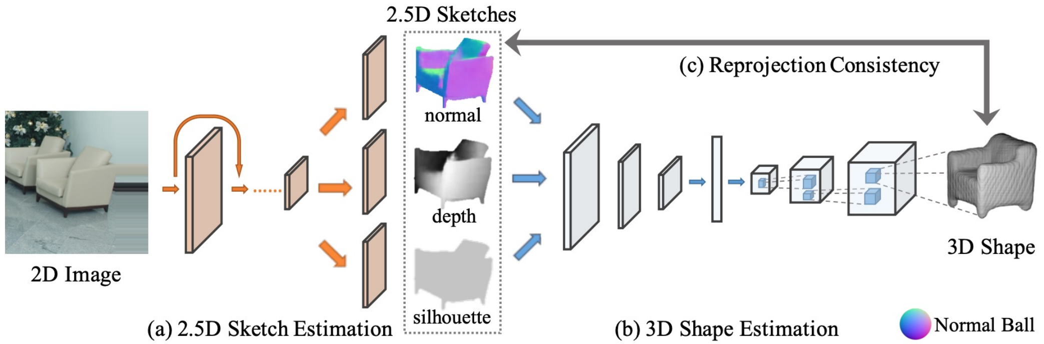


Figure 2: Our model (MarrNet) has three major components: (a) 2.5D sketch estimation, (b) 3D shape estimation, and (c) a loss function for reprojection consistency. MarrNet first recovers object normal, depth, and silhouette images from an RGB image. It then regresses the 3D shape from the 2.5D sketches. In both steps, it uses an encoding-decoding network. It finally employs a reprojection consistency loss to ensure the estimated 3D shape aligns with the 2.5D sketches. The entire framework can be trained end-to-end.

Wu et al., MarrNet: 3D Shape Reconstruction via 2.5D Sketches, NIPS 2017

What is the computation of Human Visual Systems?

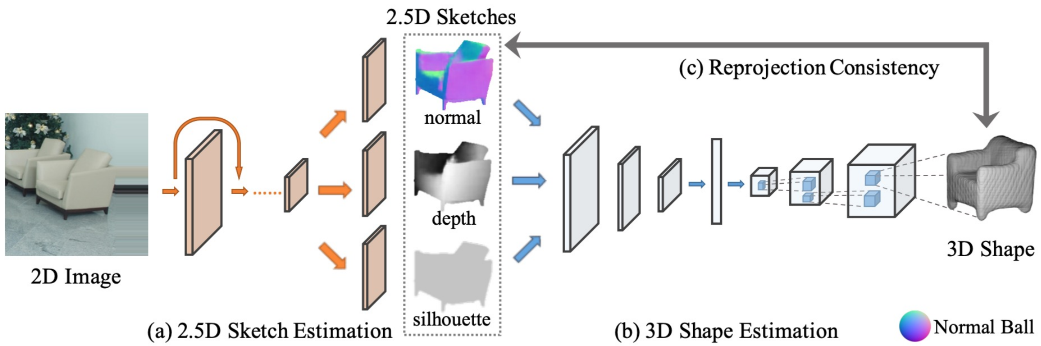
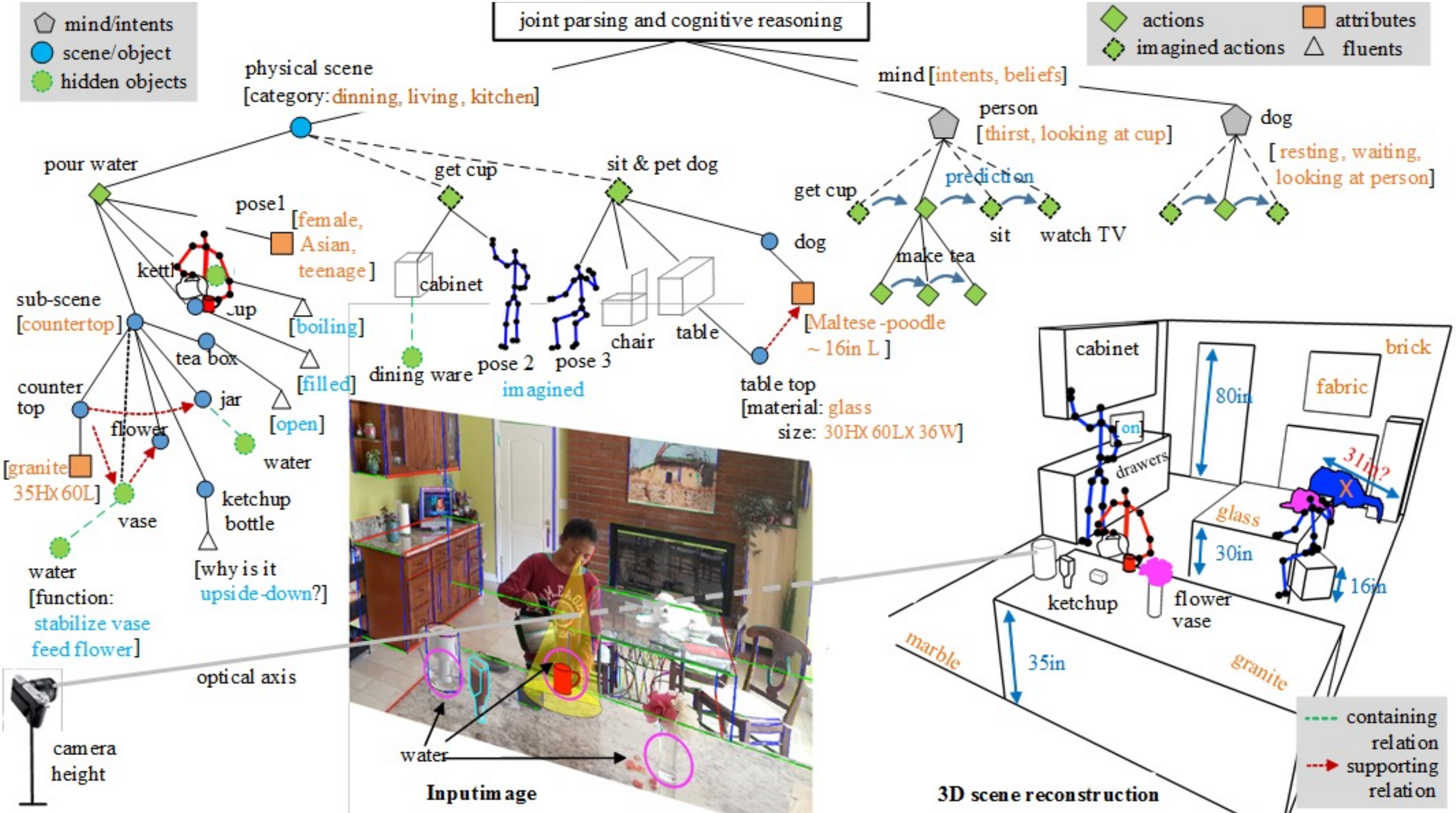


Figure 2: Our model (MarrNet) has three major components: (a) 2.5D sketch estimation, (b) 3D shape estimation, and (c) a loss function for reprojection consistency. MarrNet first recovers object normal, depth, and silhouette images from an RGB image. It then regresses the 3D shape from the 2.5D sketches. In both steps, it uses an encoding-decoding network. It finally employs a reprojection consistency loss to ensure the estimated 3D shape aligns with the 2.5D sketches. The entire framework can be trained end-to-end.

Do you see any tasks that are missing?



http://www.stat.ucla.edu/%7Esczhu/research_blog.html#VisionHistory

Visual Attention

- Spatial attention allows humans to selectively process visual information through prioritization of an area within the visual field. A region of space within the visual field is selected for attention and the information within this region then receives further processing.

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

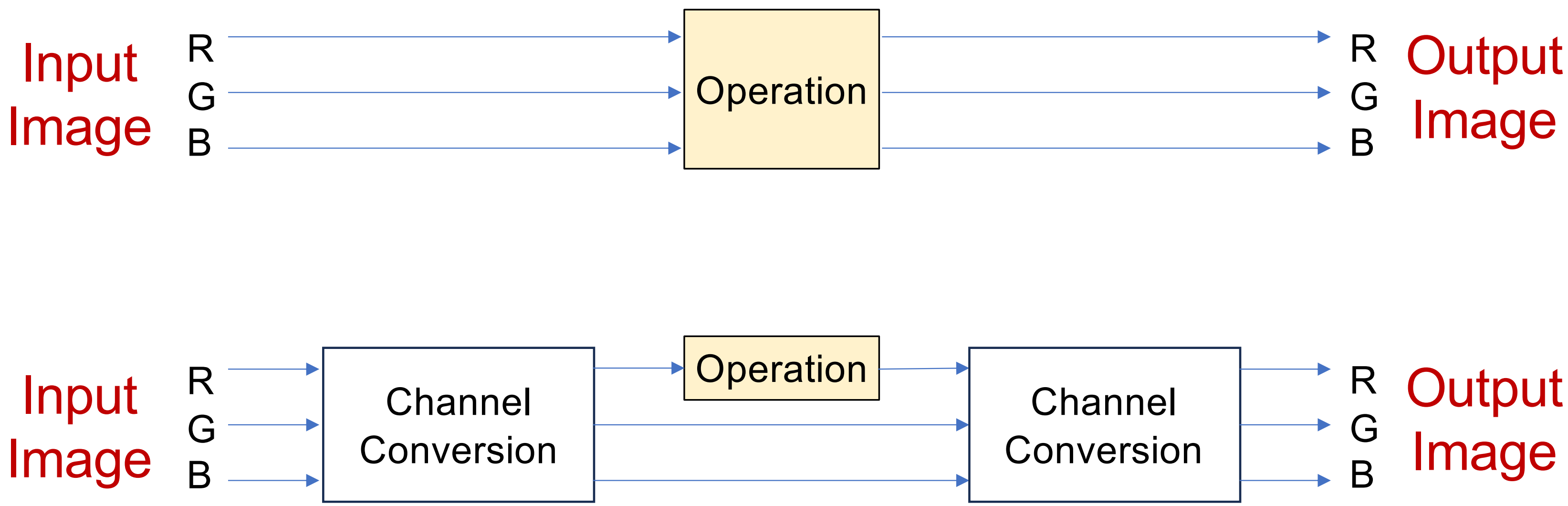
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Vaswani et al., Attention Is All You Need, NIPS 2017

Spatial-Domain Operator

Major Approaches



Commonly Used Channel Conversion

YC_rC_b

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$
$$C_r = (R - Y) \cdot 0.713 + 128$$
$$C_b = (B - Y) \cdot 0.564 + 128$$

$$R = Y + 1.403 \cdot (C_r - 128)$$
$$G = Y - 0.714 \cdot (C_r - 128) - 0.344 \cdot (C_b - 128)$$
$$B = Y + 1.773 \cdot (C_b - 128)$$

HLS

$$H = \begin{cases} \theta & \text{if } B \le G \\ 360 - \theta & \text{if } B > G \end{cases}$$

$$\theta = \cos^{-1}\left\{\frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}}\right\}$$

$$S = 1 - \frac{3}{R+G+B}[\min(R,G,B)]$$

$$I = \frac{1}{3}(R+G+B)$$

$For\ 0^\circ < H \le 120^\circ$

$$b = \frac{1-S}{3}$$
$$r = \frac{1}{3}\left(1 + \frac{ScosH}{\cos(60^\circ - H)}\right)$$
$$g = 1 - (r + b)$$

$For\ 120^\circ < H \le 240^\circ$

H= $H = H - 120^\circ$

$$r = \frac{1-S}{3}$$
$$g = \frac{1}{3}\left(1 + \frac{ScosH}{\cos(60^\circ - H)}\right)$$
$$b = 1 - (r + g)$$

$For\ 240^\circ < H \le 360^\circ$

H= $H = H - 240^\circ$

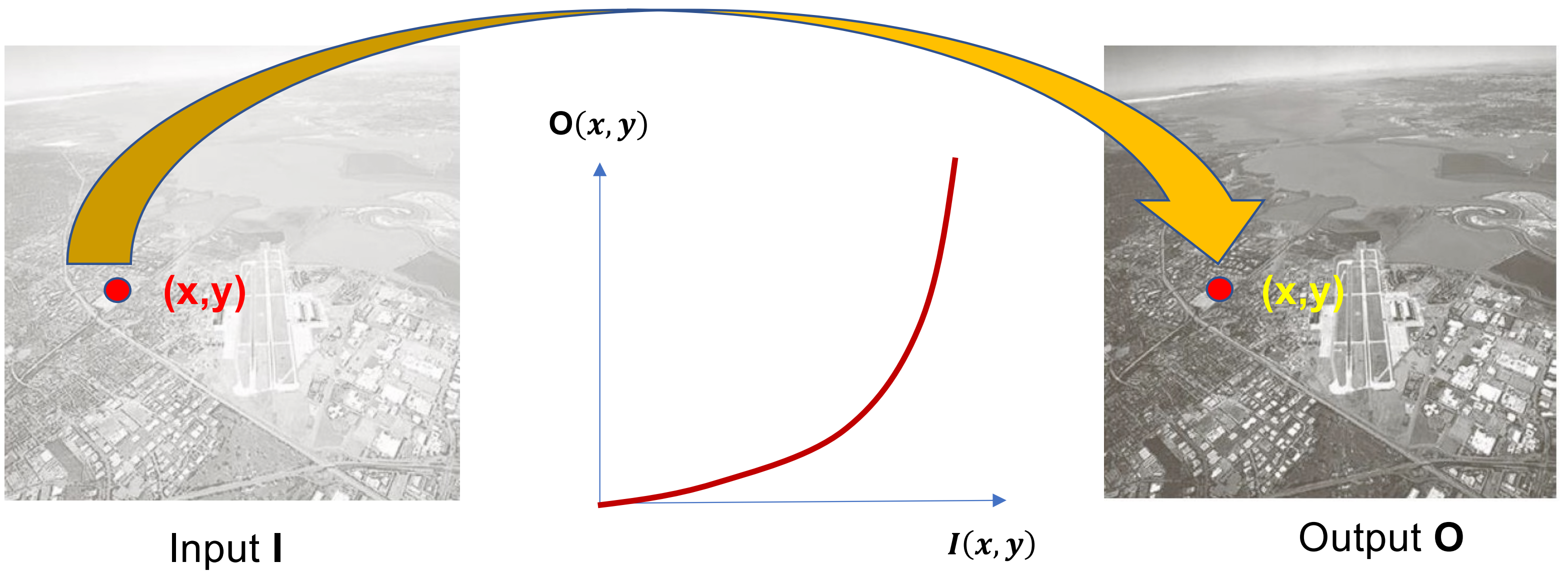
$$g = \frac{1-S}{3}$$
$$b = \frac{1}{3}\left(1 + \frac{ScosH}{\cos(60^\circ - H)}\right)$$
$$r = 1 - (b + g)$$

Types of Spatial-Domain Operators

- Point Operators
- Neighborhood Operators
  - Linear Filtering
  - Nonlinear Filtering
- Geometric Transformation

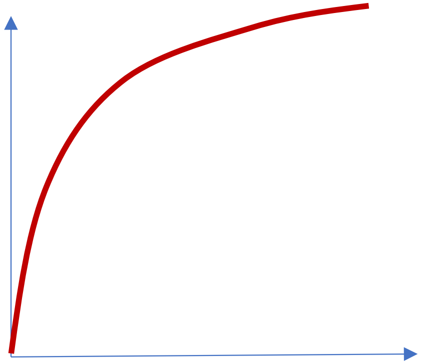
Point Operator

$$O(x, y) = f(I(x, y))$$



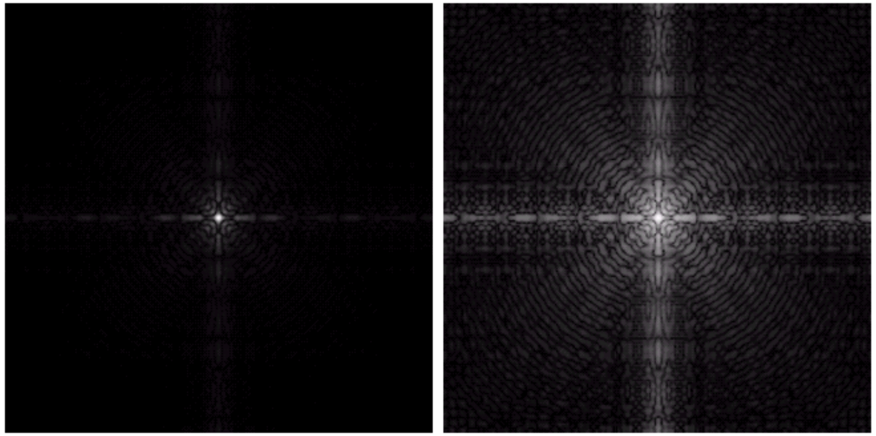
Log Transformation

- O(x,y) = c log(1 + |I(x,y)|)



a b

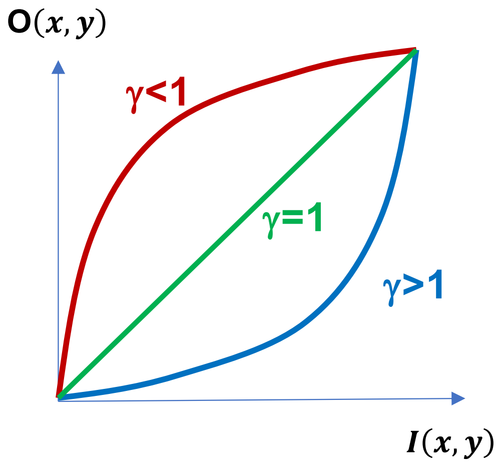
FIGURE 3.5
(a) Fourier spectrum.
(b) Result of applying the log transformation given in Eq. (3.2-2) with $c = 1$.



Ref: Gonzalez and Woods, “Digital Image Processing”

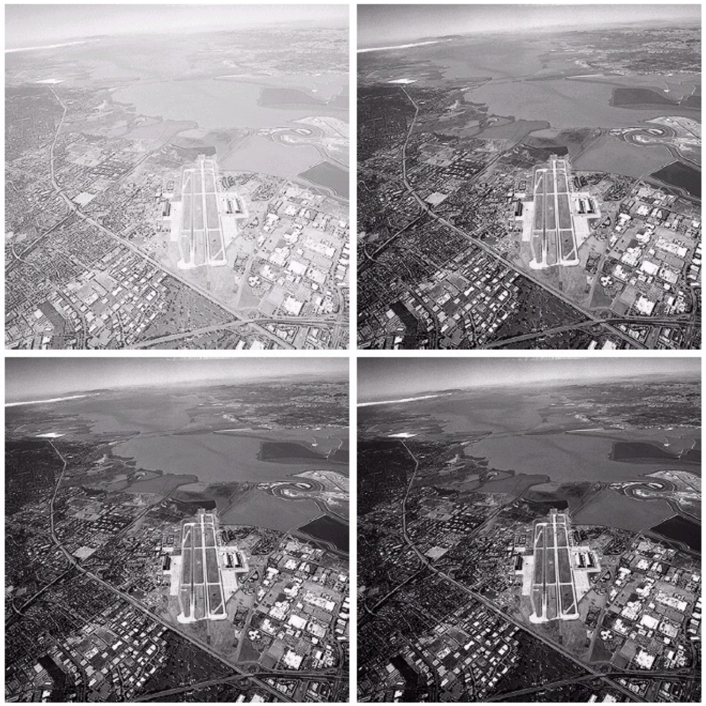
Power-law Transformation

$$O(x,y) = I_{max}\left(\frac{I(x,y)}{I_{max}}\right)^{\gamma}$$

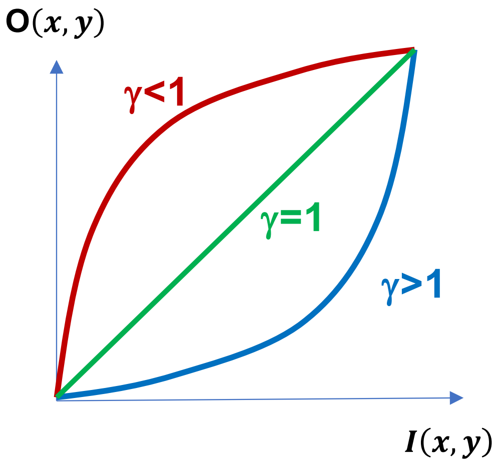


a b
c d

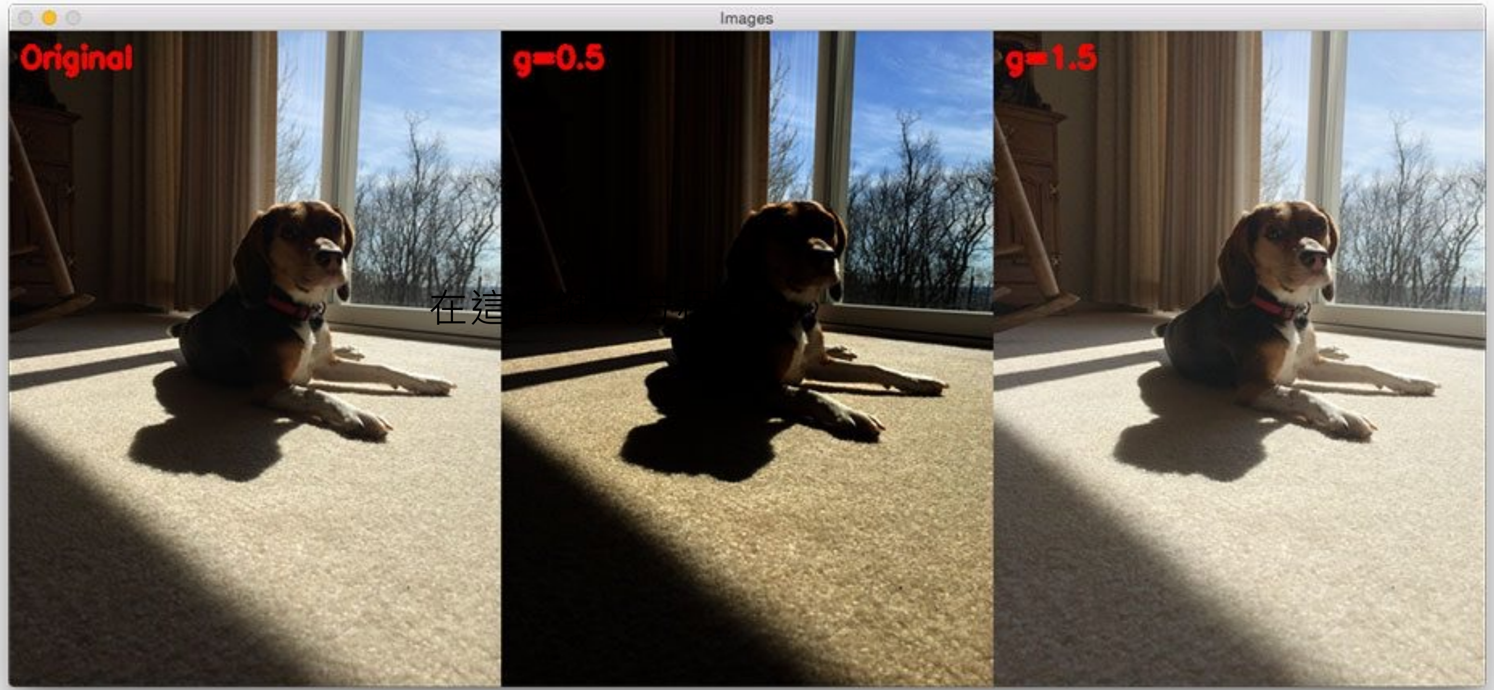
FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0, 4.0,$ and 5.0, respectively. (Original image for this example courtesy of NASA.)



Ref: Gonzalez and Woods, “Digital Image Processing”

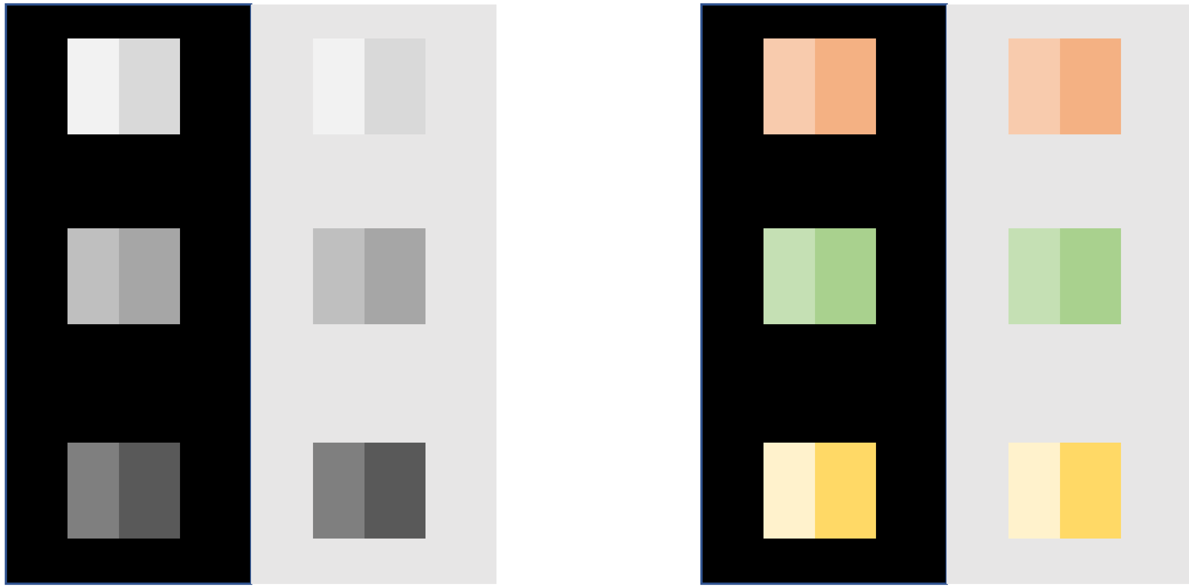


$$g = \frac{1}{\gamma}$$



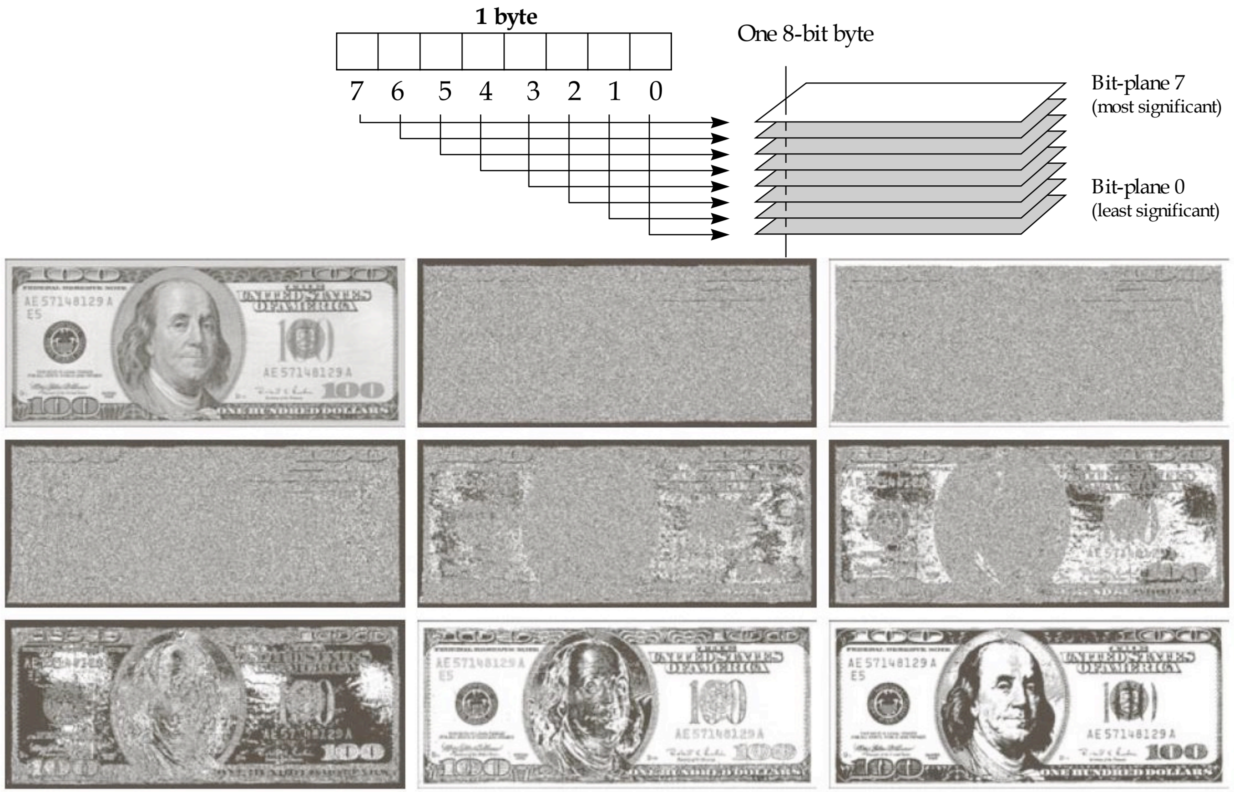
https://pyimagesearch.com/2015/10/05/opencv-gamma-correction/

Simultaneous Contrast Effect



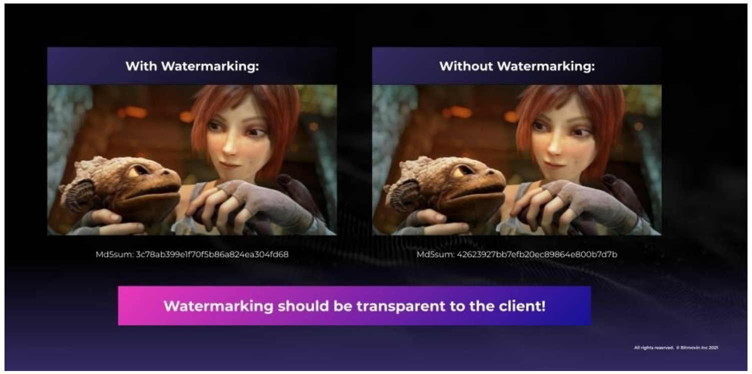
- When the region of interest is surround by darker neighborhood, the contrast appears to decrease.

Bit-Plane Slicing



Ref: Gonzalez and Woods, “Digital Image Processing”

Watermarking



Neighborhood Operator

Types of Neighborhood Operators

Linear Filtering

Nonlinear Filtering

Credit: Prof. Sheng-Jyh Wang NYCU EE

1-D Linear Time Invariant System

Linear System

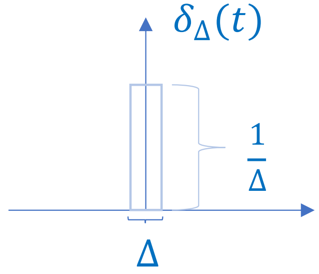
Additivity: $T\{x_1(t) + x_2(t)\} = T\{x_1(t)\} + T\{x_2(t)\} = y_1(t) + y_2(t)$

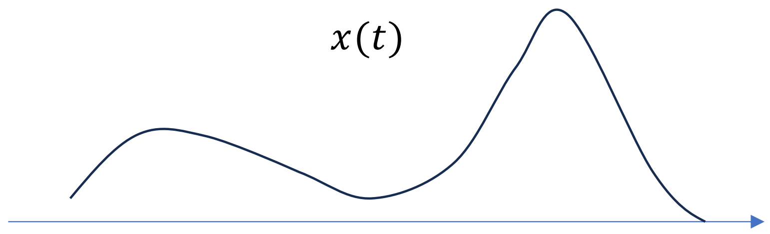
Homogeneity: $T\{ax(t)\} = aT\{x(t)\} = ay(t)$

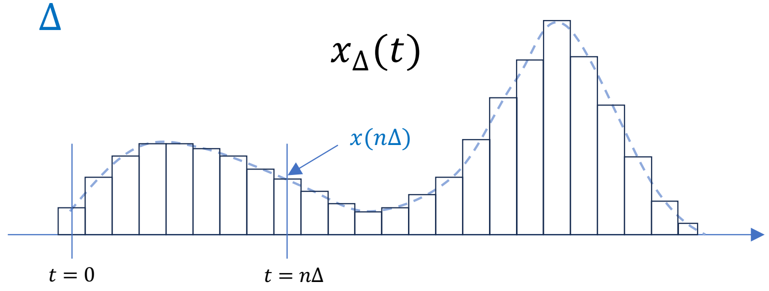
Time-Invariant System

If $y(t) = T\{x(t)\}$, then $y(t - t_0) = T\{x(t - t_0)\}$.

Signal Decomposition

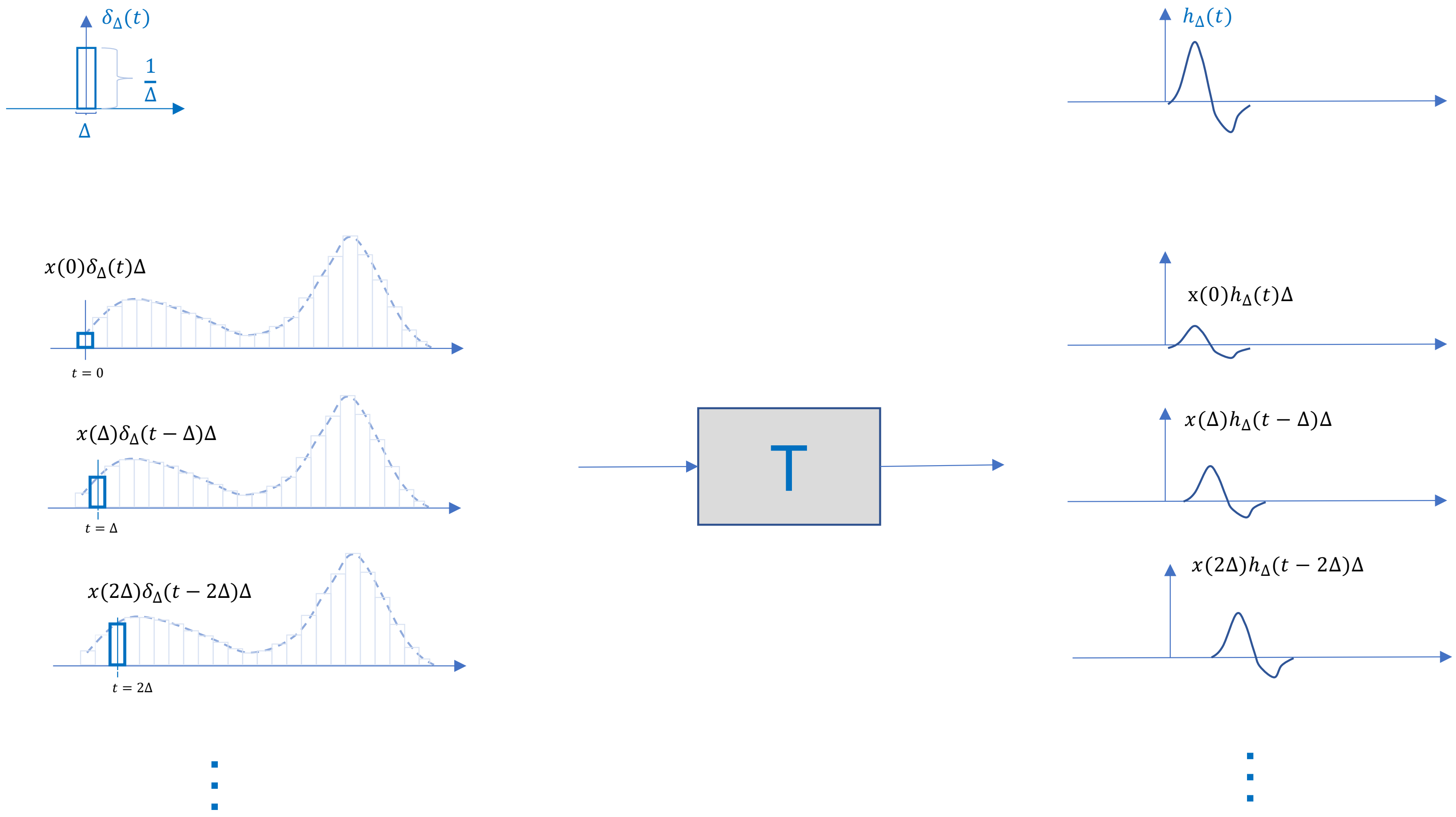




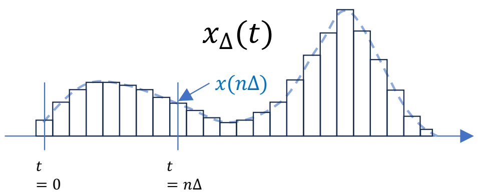


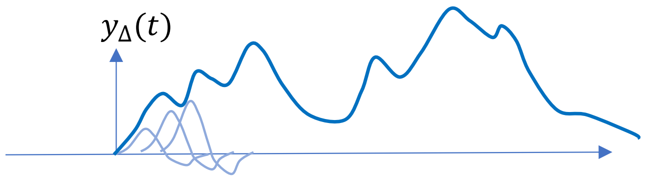
$$x_\Delta(t) = \sum_{n=-\infty}^{\infty} x(n\Delta)\delta_\Delta(t - n\Delta)\Delta \xrightarrow{\Delta \to 0} x(t) = \int_{-\infty}^{\infty} x(\tau)\delta(t-\tau)d\tau$$

$$h_\Delta(t) = T\{\delta_\Delta(t)\}$$



Superposition

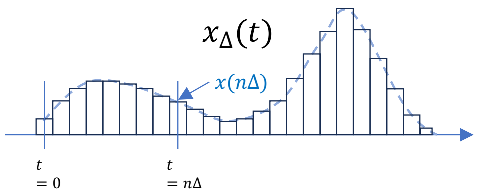




$$x_\Delta(t) = \sum_{n=-\infty}^{\infty} x(n\Delta)\delta_\Delta(t - n\Delta)\Delta$$

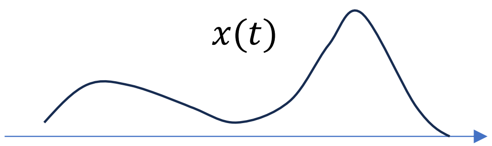
$$\begin{aligned} y_\Delta(t) &= T\{\sum_{n=-\infty}^{\infty} x(n\Delta)\delta_\Delta(t - n\Delta)\Delta\} \\ &= \sum_{n=-\infty}^{\infty} x(n\Delta)T\{\delta_\Delta(t - n\Delta)\}\Delta \\ &= \sum_{n=-\infty}^{\infty} x(n\Delta)h_\Delta(t - n\Delta)\Delta \end{aligned}$$

Superposition

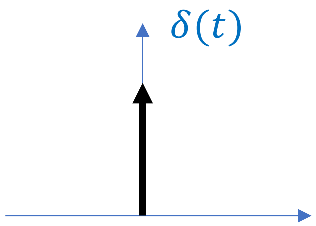


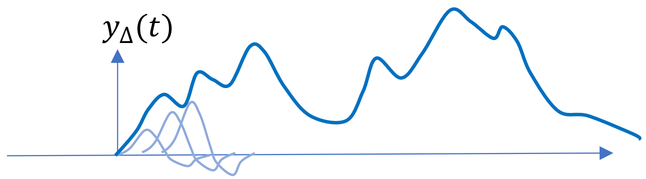
$$x_\Delta(t) = \sum_{n=-\infty}^{\infty} x(n\Delta)\delta_\Delta(t - n\Delta)\Delta$$

$\Delta \to 0$



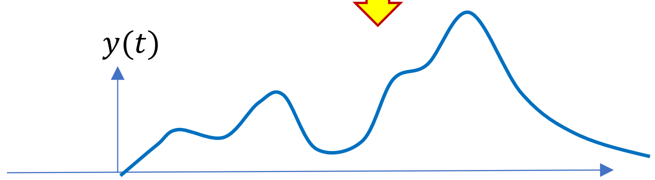
$$x(t) = \int_{-\infty}^{\infty} x(\tau)\delta(t-\tau)d\tau$$





$$y_\Delta(t) = \sum_{n=-\infty}^{\infty} x(n\Delta)h_\Delta(t - n\Delta)\Delta$$

$\Delta \to 0$



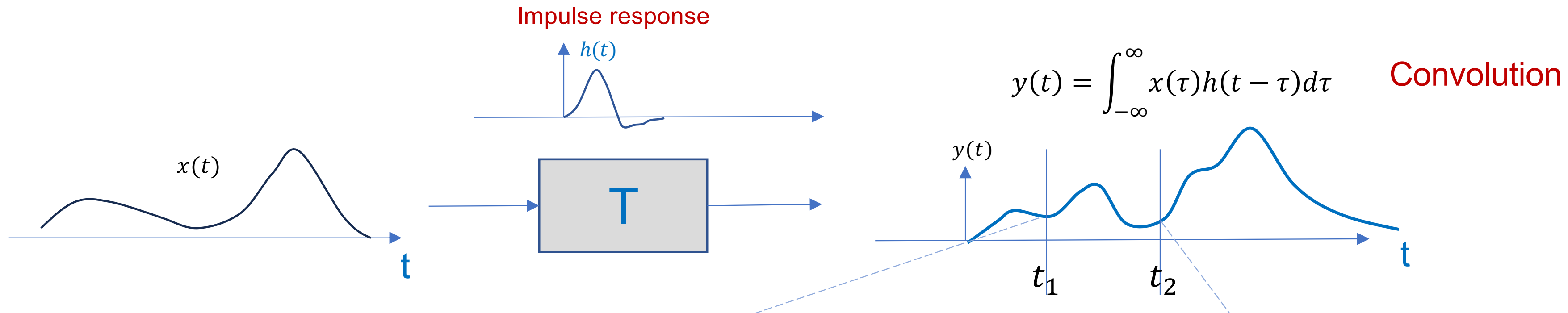
$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$ Convolution

where $h(t) = T\{\delta(t)\}$

Impulse response

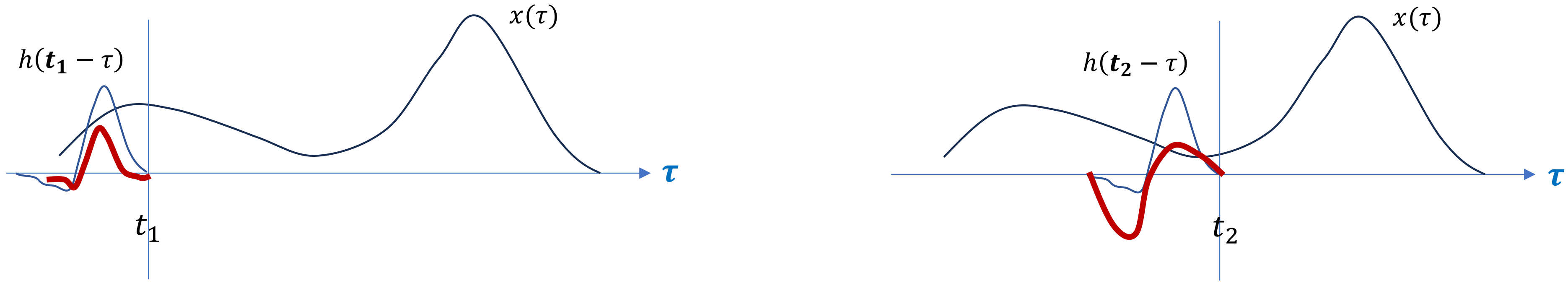
Credit: Prof. Sheng-Jyh Wang NYCU EE

1-D Convolution



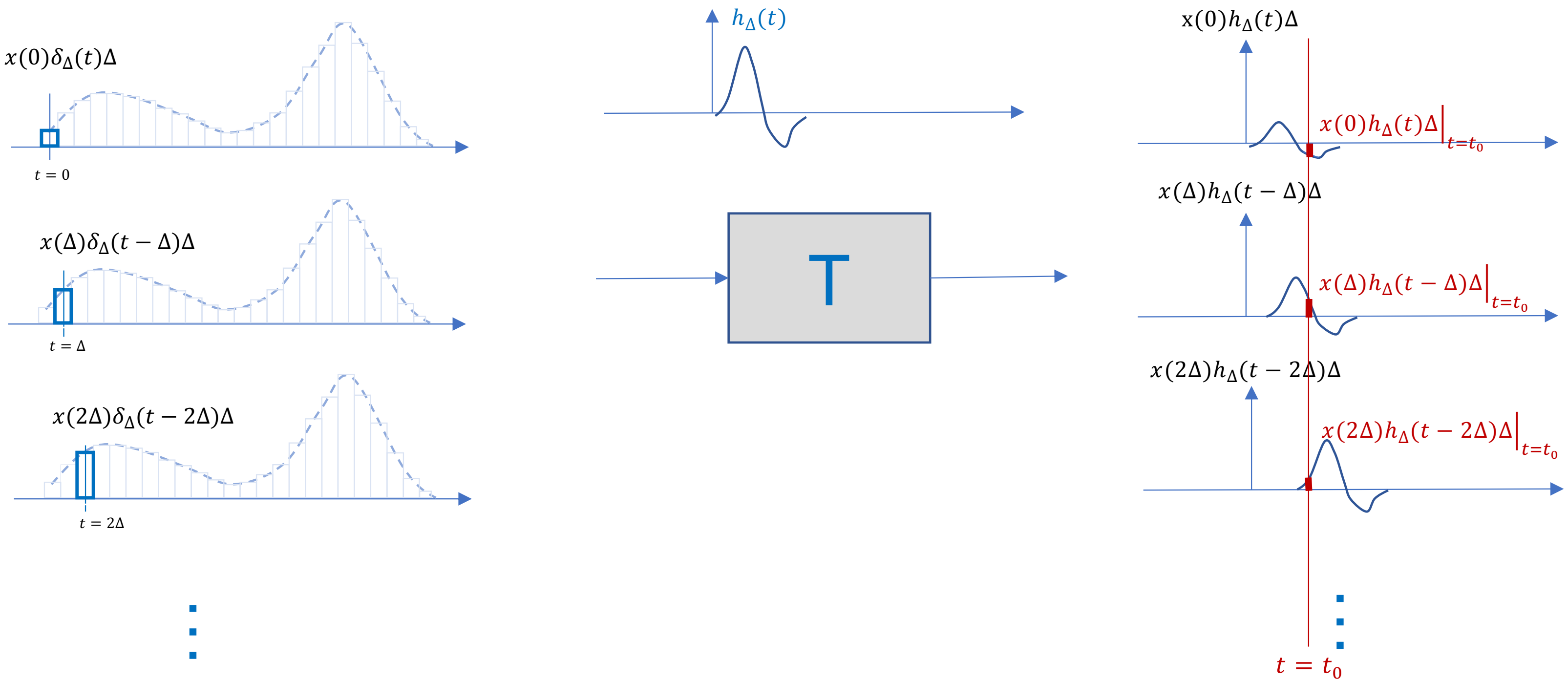
$$y(t_1) = \int_{-\infty}^{\infty} x(\tau)h(\boldsymbol{t_1} - \tau)d\tau$$

$$y(t_2) = \int_{-\infty}^{\infty} x(\tau)h(\boldsymbol{t_2} - \tau)d\tau$$

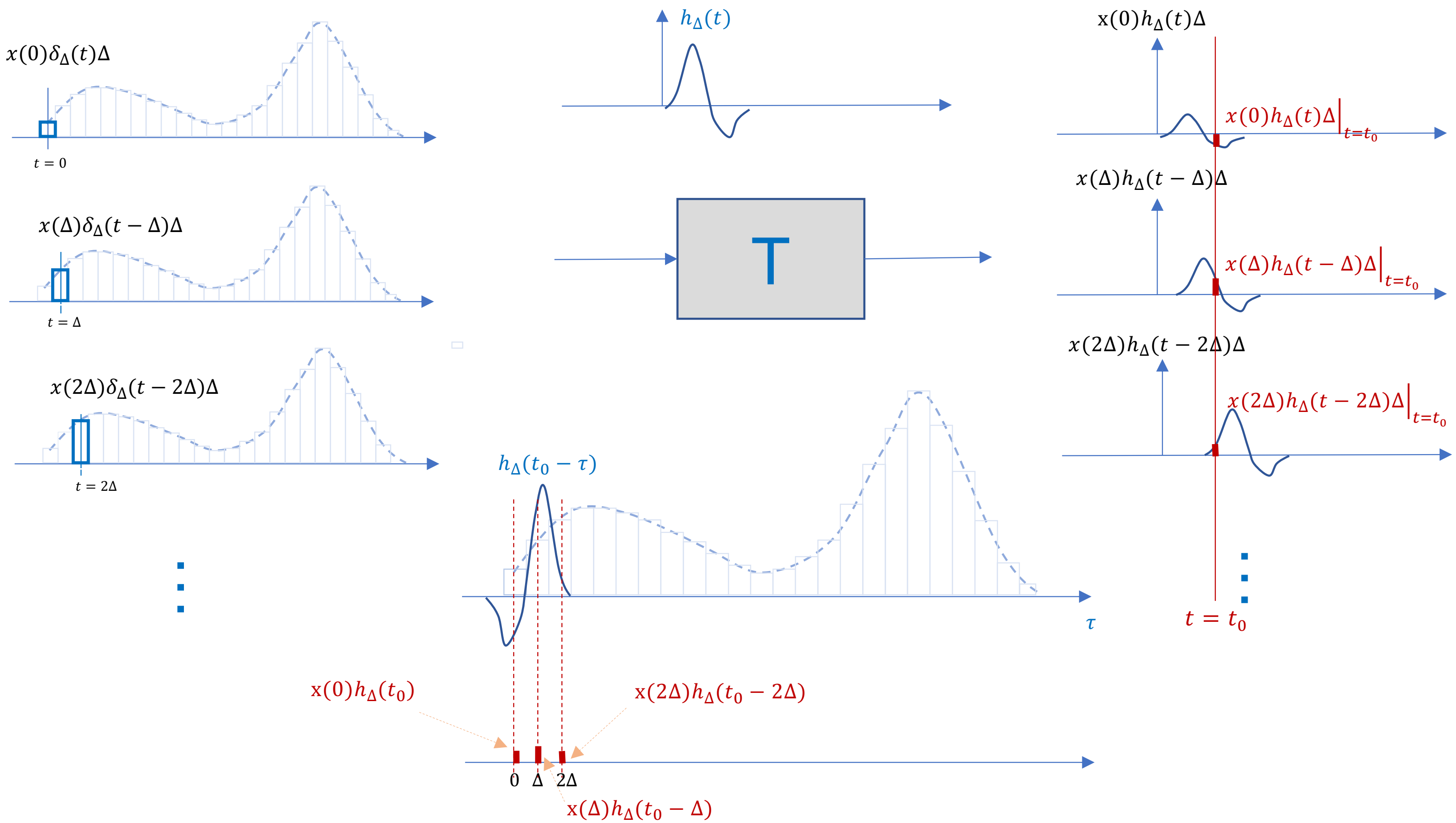


Credit: Prof. Sheng-Jyh Wang NYCU EE

1-D Convolution

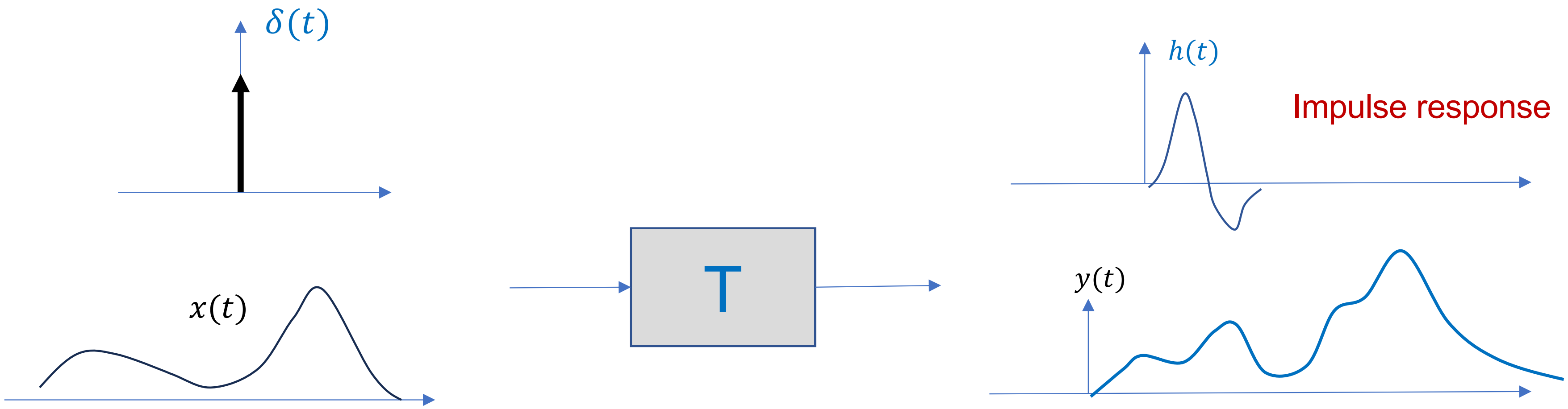


1-D Convolution



Summary

$$h(t) = T\{\delta(t)\}$$



$$x(t) = \int_{-\infty}^{\infty} x(\tau)\delta(t-\tau)d\tau$$

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$ Convolution

2D Linear Shift-Invariant System

$$I(x,y) \longrightarrow \boxed{T} \longrightarrow O(x,y)$$

Linear System

Additivity: $T\{I_1(x,y) + I_2(x,y)\} = T\{I_1(x,y)\} + T\{I_2(x,y)\} = O_1(x,y) + O_2(x,y)$

Homogeneity: $T\{aI(x,y)\} = aT\{I(x,y)\} = aO(x,y)$

Shift-Invariant System

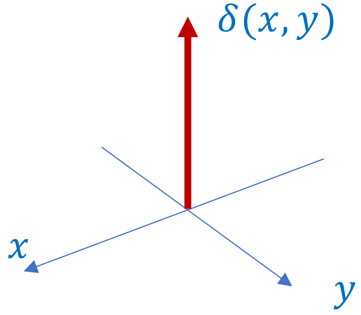
If $O(x,y) = T\{I(x,y)\}$, then $T\{I(x-x_0, y-y_0)\} = O(x-x_0, y-y_0)$

2D Convolution - Continuous

$$I(x,y) \longrightarrow \boxed{T} \longrightarrow O(x,y)$$

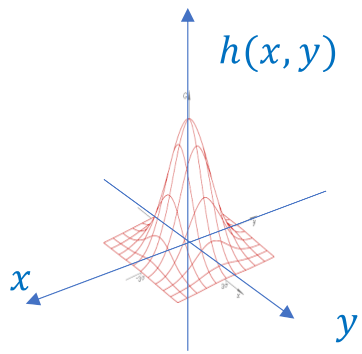
$$I(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(m,n)\,\delta(x-m,y-n)\,dm\,dn$$

$$O(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(m,n)T\{\delta(x-m,y-n)\}\,dm\,dn$$
$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(m,n)\,h(x-m,y-n)\,dm\,dn$$



$$\boldsymbol{h(x,y) = T\{\delta(x,y)\}}$$

Point Spread Function (psf)

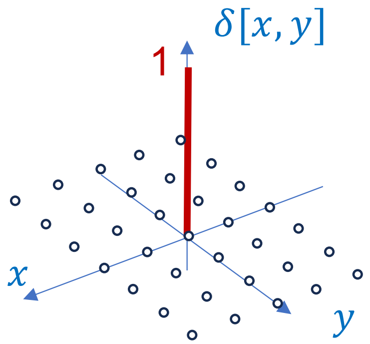


2D Convolution - Discrete

$$I[x,y] \longrightarrow \boxed{T} \longrightarrow O[x,y]$$

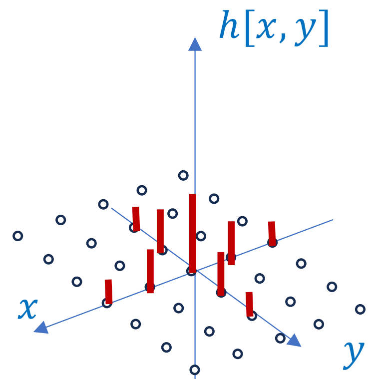
$$I[x,y] = \sum_m \sum_n I[m,n]\delta[x-m,y-n]$$

$$O[x,y] = \sum_m \sum_n I[m,n]h[x-m,y-n]$$

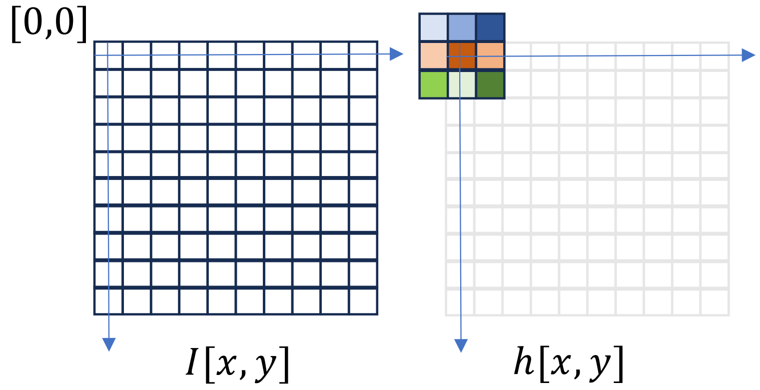


$$\boldsymbol{h[x,y] = T\{\delta[x,y]\}}$$

Point Spread Function (psf)

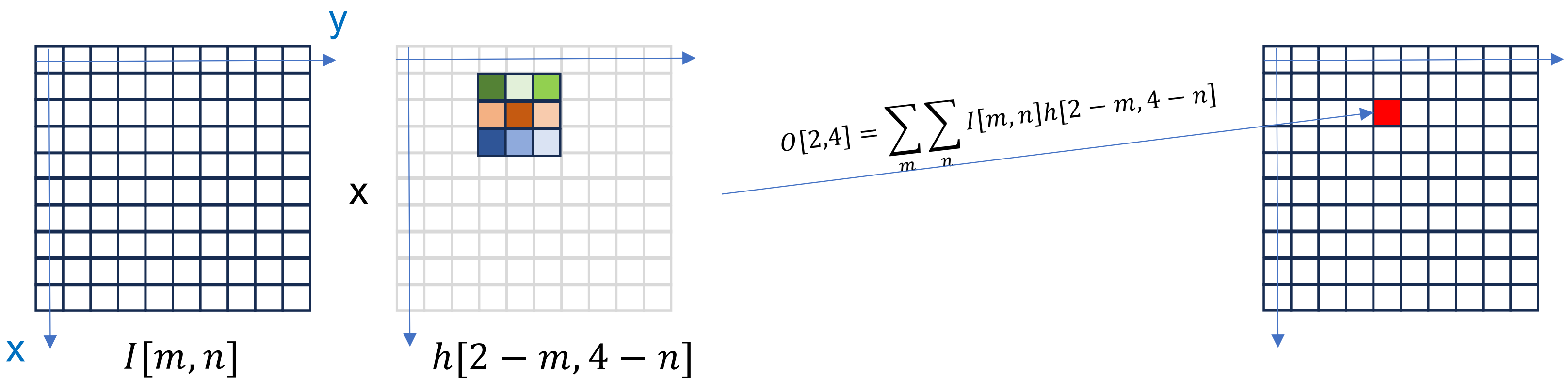


2D Convolution

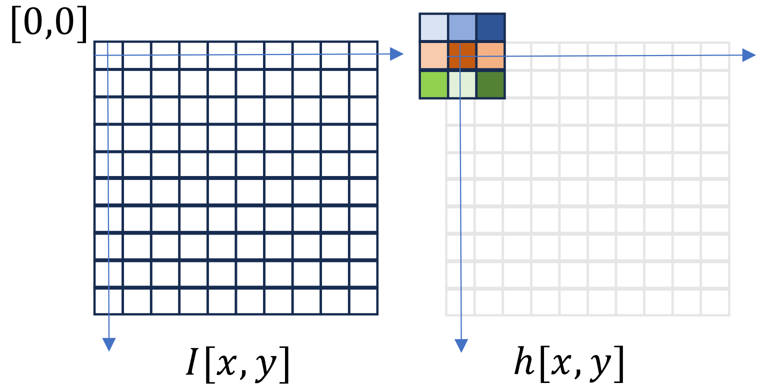


$$O[x, y] = \sum_m \sum_n I[m, n]h[x - m, y - n]$$

Example:

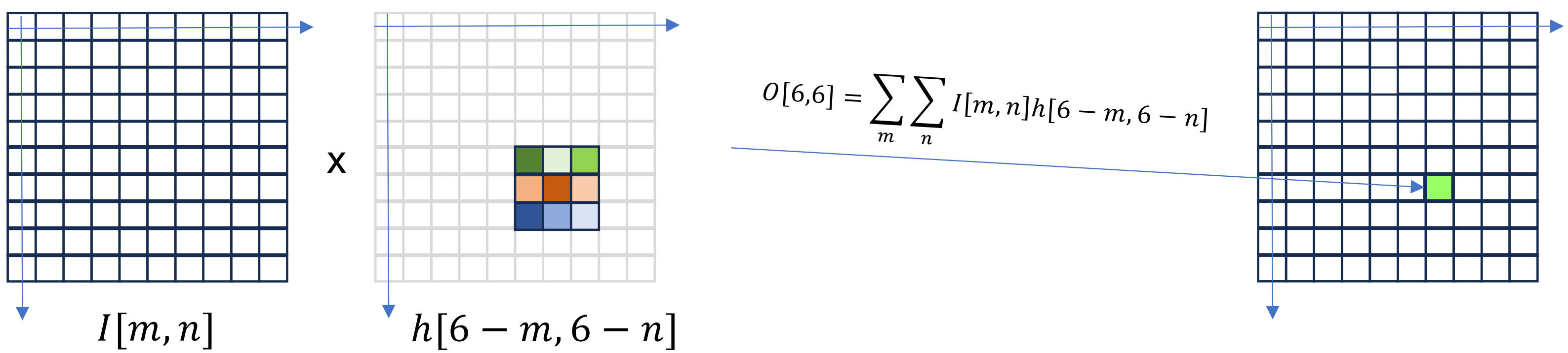


2D Convolution



$$O[x,y] = \sum_{m} \sum_{n} I[m,n] h[x-m, y-n]$$

Example:



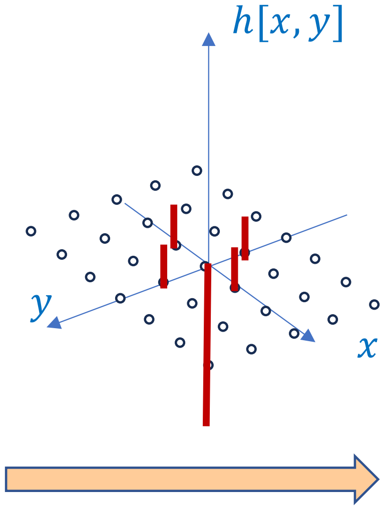
Linear Filtering

$$O(x, y) = \sum_{i=-M}^{M} \sum_{j=-N}^{N} g_{i,j} \cdot I(x + i, y + j)$$

Correlation $$O(x, y) = \sum_{i=-M}^{M} \sum_{j=-N}^{N} W(i, j) \cdot I(x + i, y + j)$$

Convolution $$O(x, y) = \sum_{i=-M}^{M} \sum_{j=-N}^{N} W(i, j) \cdot I(x - i, y - j) \equiv W(x, y) * I(x, y)$$

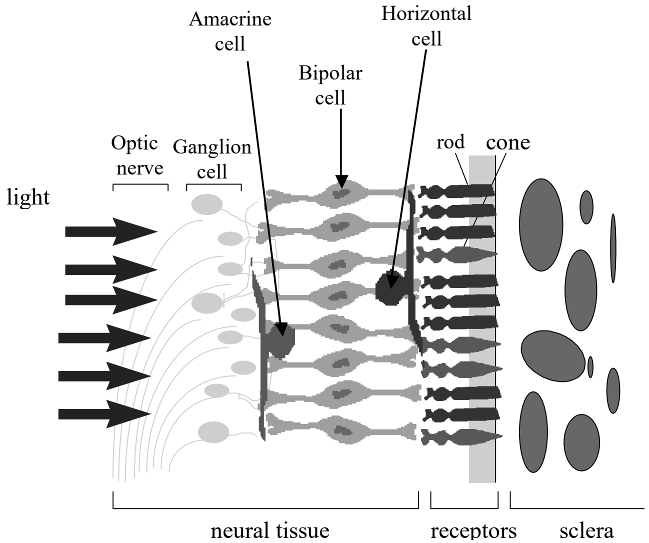
Laplacian Operator

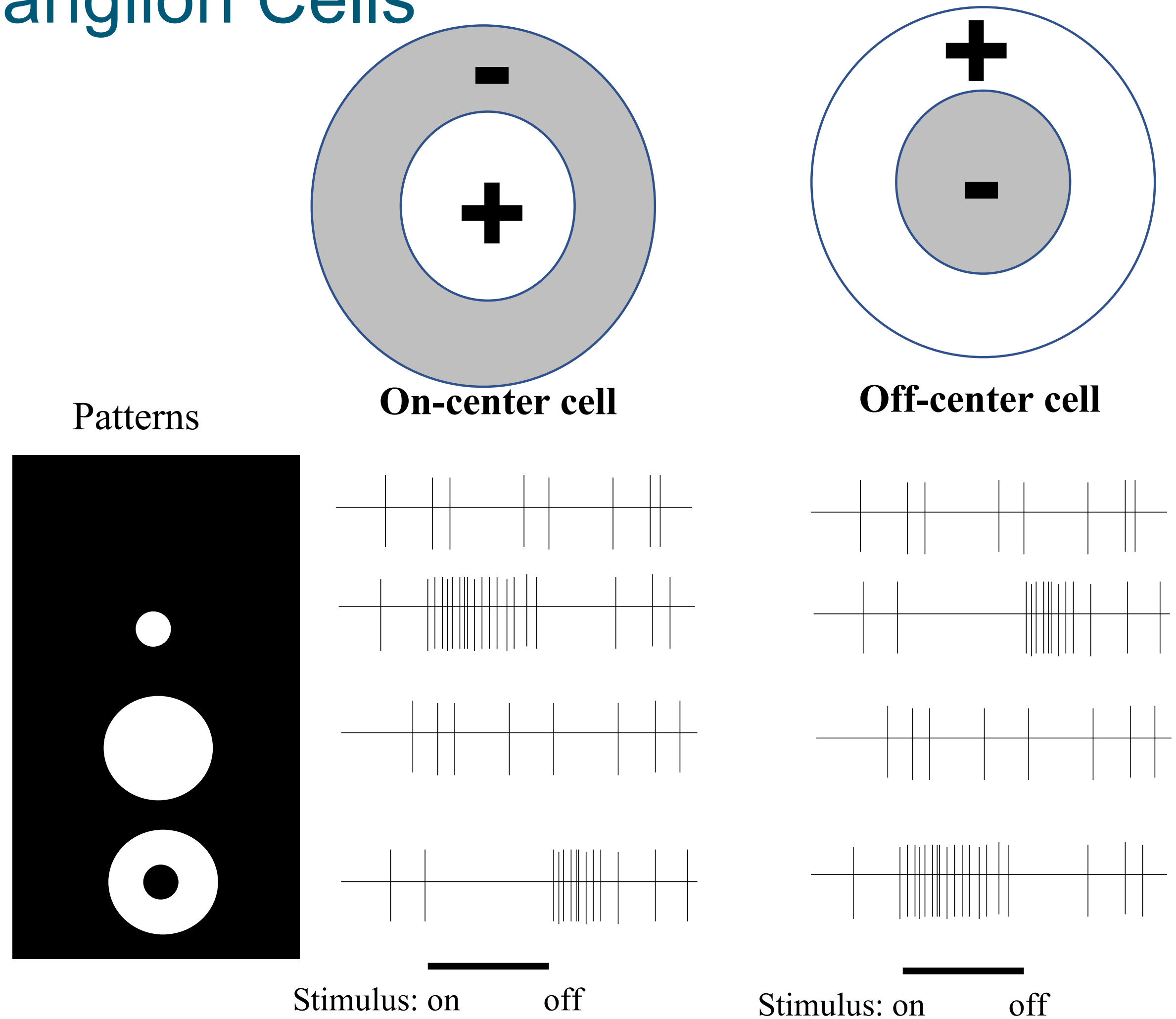


0	1	0
1	-4	1
0	1	0

Ref: Gonzalez and Woods, “Digital Image Processing”

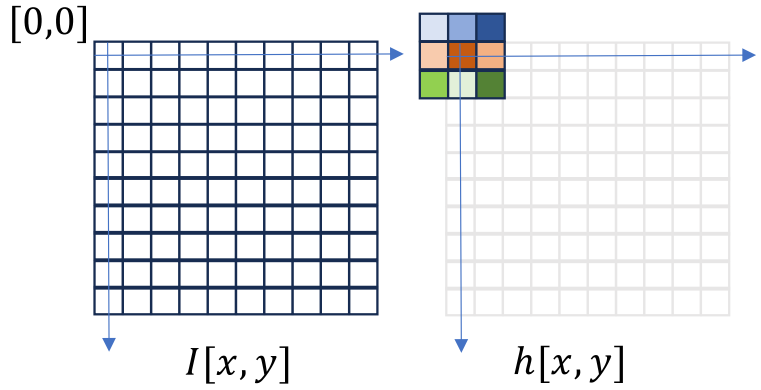
Receptive Fields of Retinal Ganglion Cells



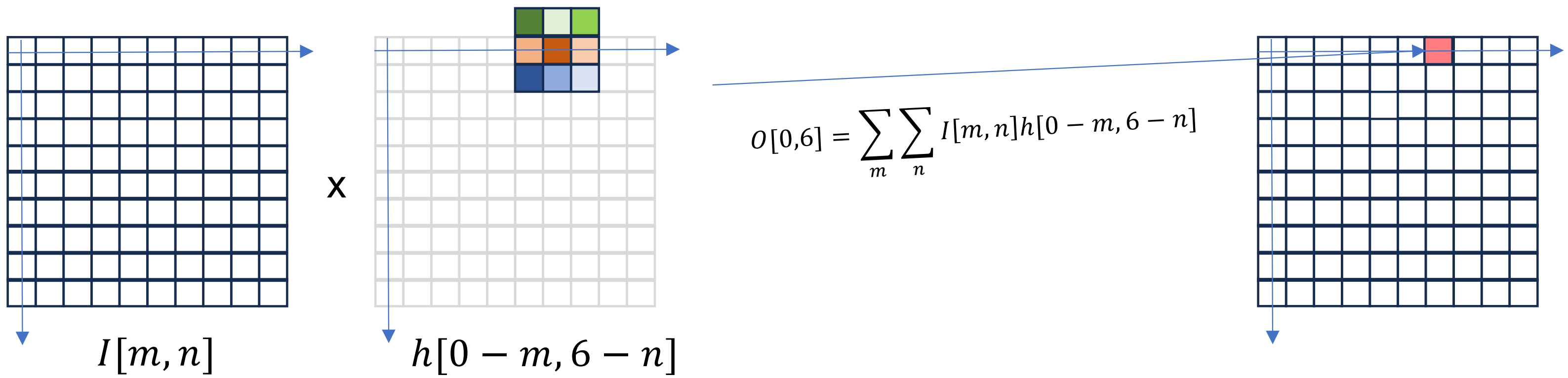


Padding

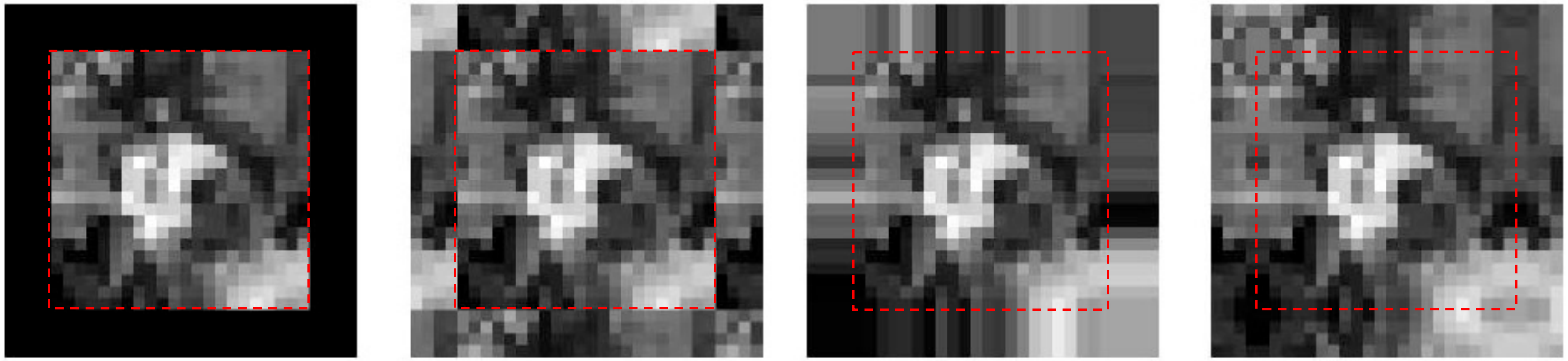
$$O[x,y] = \sum_m \sum_n I[m,n]h[x-m,y-n]$$



Example:



Padding



Zero **Wrap** **Clamp** **Mirror**

Ref: Rechard Szeliski, “Computer Vision: Algorithms and Applications”.

Separable Filtering (1/3)

$W[x, y]$ is called a separable kernel if it can be represented as $W[x, y] = W_x[x]\, W_y[y]$.

$$O[p, q] = \sum_{i=-M}^{M} \sum_{j=-N}^{N} W[i, j] \cdot I[p - i, q - j] = \sum_{i=-M}^{M} \sum_{j=-N}^{N} W_x[i] W_y[j] I[p - i, q - j]$$

$$= \sum_{i=-M}^{M} W_x[i] \left[\sum_{j=-N}^{N} W_y[j] I[p - i, q - j] \right] = \sum_{i=-M}^{M} W_x[i] \hat{I}[p - i, q]$$

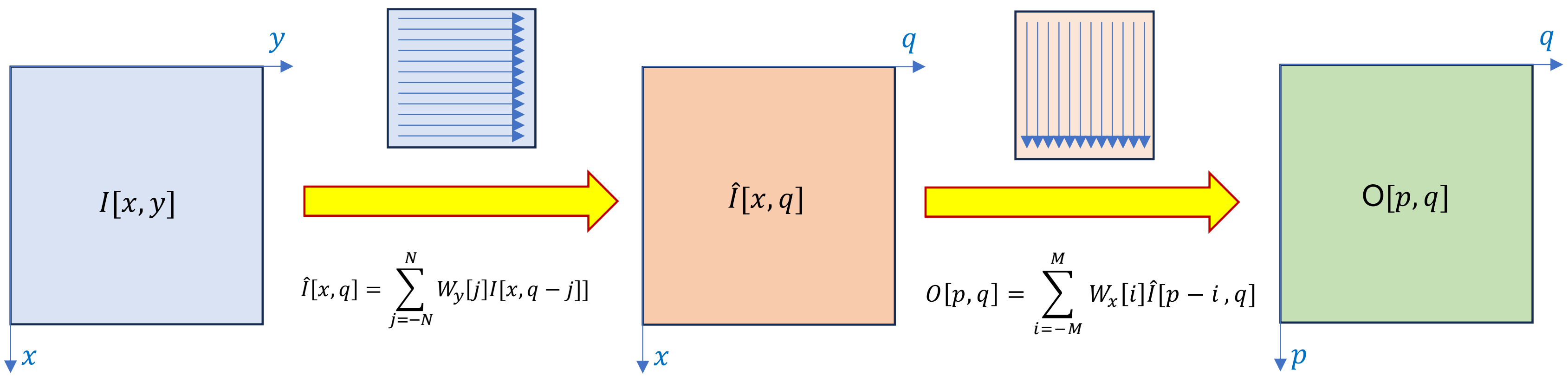
where $\hat{I}[x, q] \equiv \sum_{j=-N}^{N} W_y[j] I[x, q - j]$

Separable Filtering (2/3)

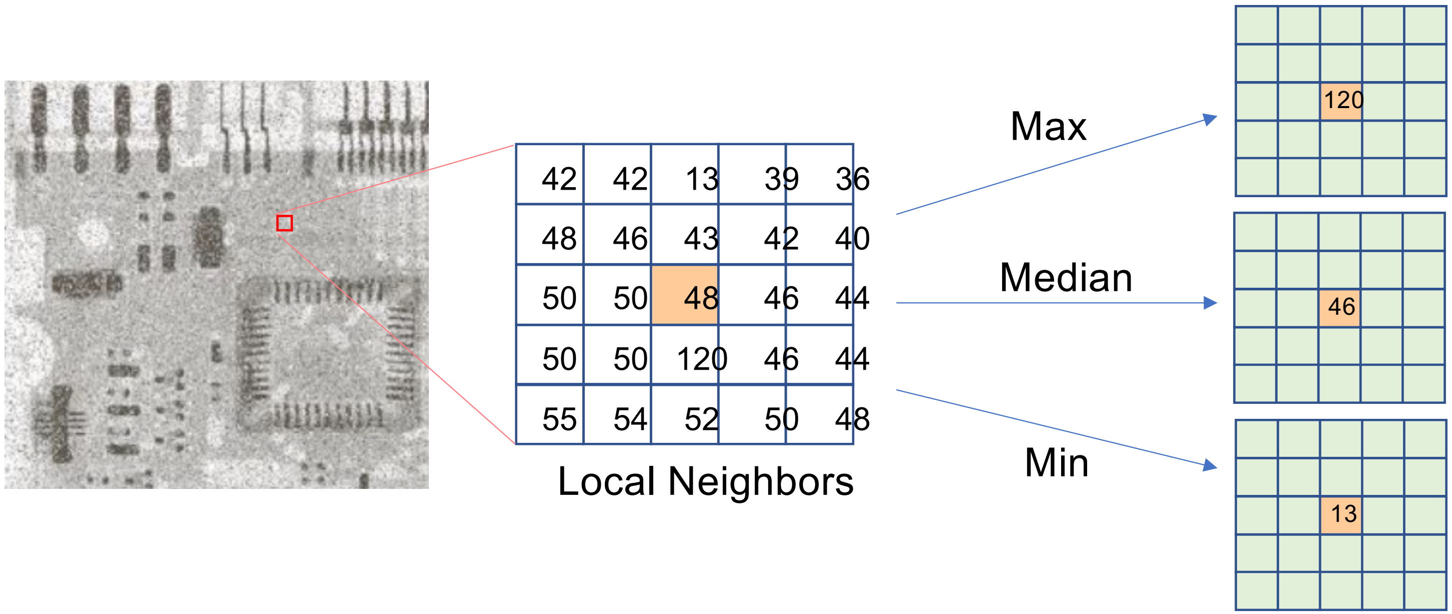
$$\frac{1}{16}\begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix} \qquad \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

$$\frac{1}{16}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

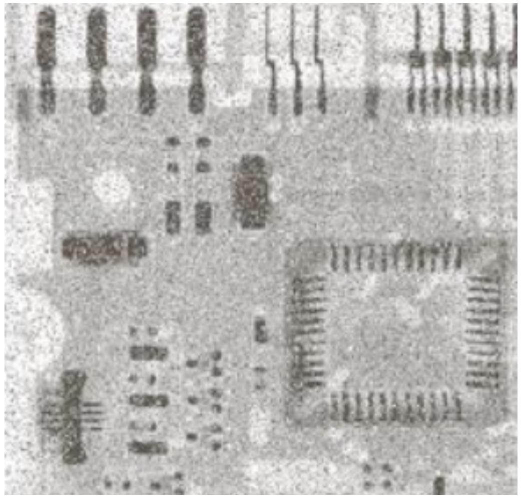
Separable Filtering (3/3)

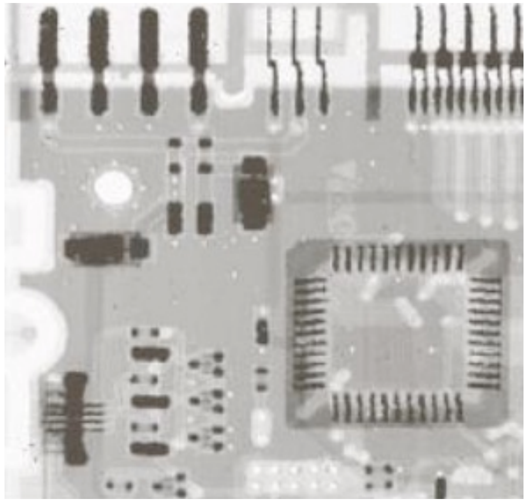


Non-linear Filter



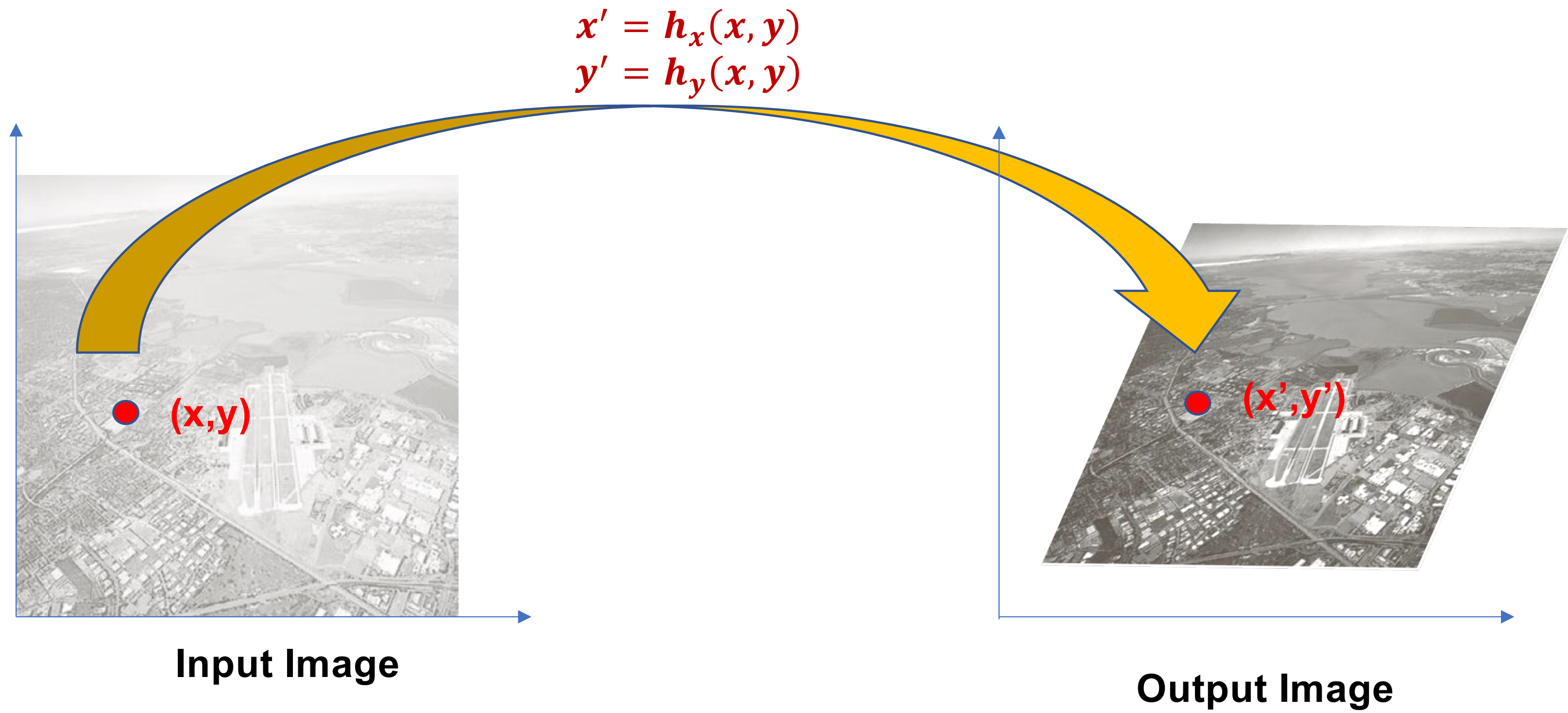
Median Filter





Ref: Gonzalez and Woods, “Digital Image Processing”

Geometric Transformation


$x' = h_x(x, y)$
$y' = h_y(x, y)$
(x,y)
(x',y')
Input Image
Output Image

Forward Warping

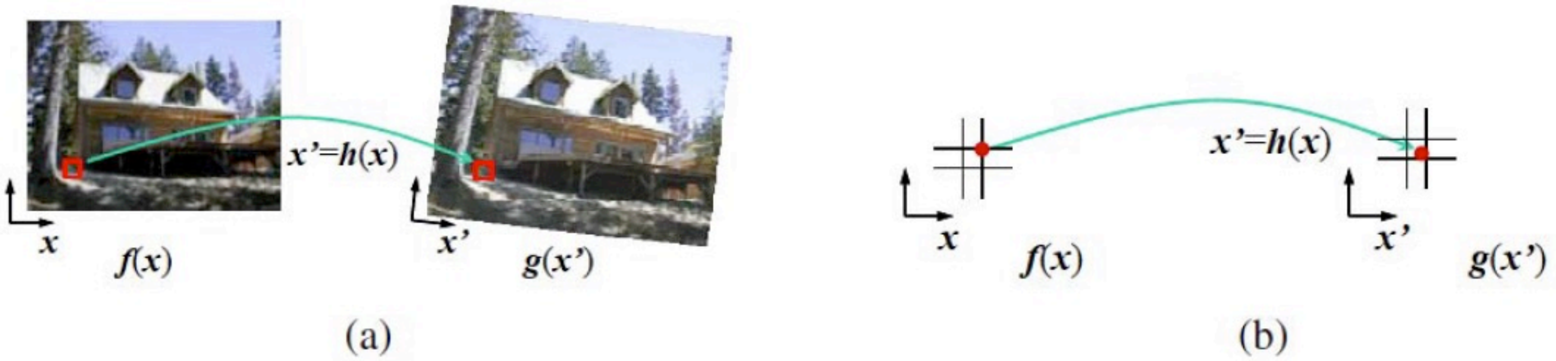


Figure 3.46 Forward warping algorithm: (a) a pixel $f(\boldsymbol{x})$ is copied to its corresponding location $\boldsymbol{x}' = \boldsymbol{h}(\boldsymbol{x})$ in image $g(\boldsymbol{x}')$; (b) detail of the source and destination pixel locations.

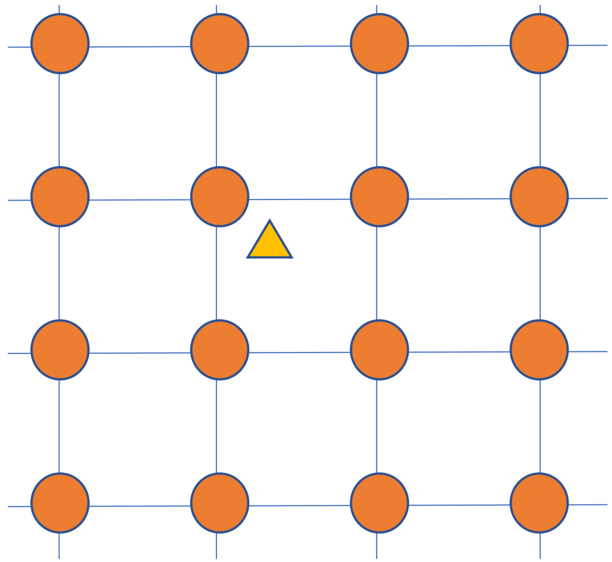
Ref: Rechard Szeliski, “Computer Vision: Algorithms and Applications”.

Inverse Warping

Figure 3.47 Inverse warping algorithm: (a) a pixel $g(\boldsymbol{x}')$ is sampled from its corresponding location $\boldsymbol{x} = \hat{\boldsymbol{h}}(\boldsymbol{x}')$ in image $f(\boldsymbol{x})$; (b) detail of the source and destination pixel locations.

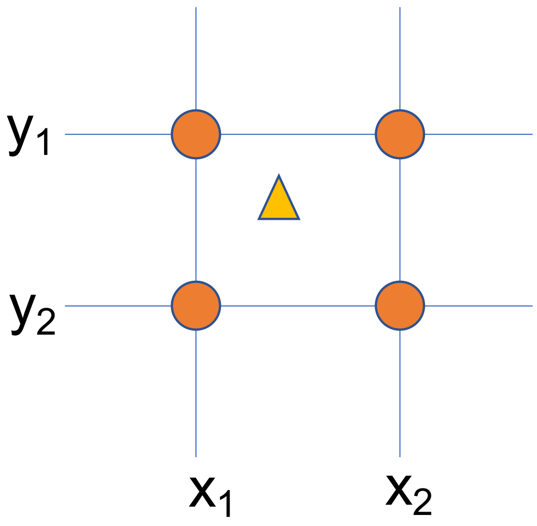
Ref: Rechard Szeliski, "Computer Vision: Algorithms and Applications".

Interpolation (1/2)



- **Nearest-Neighbor Interpolation**
- **Bilinear Interpolation**
- **Bicubic Interpolation**

Interpolation (2/2)



$$I(x, y) = a_0 + a_1 x + a_2 y + a_3 xy$$

where $a_0, a_1, a_2,$ and a_3 can be found by solving

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1 y_1 \\ 1 & x_1 & y_2 & x_1 y_2 \\ 1 & x_2 & y_1 & x_2 y_1 \\ 1 & x_2 & y_2 & x_2 y_2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} I(x_1, y_1) \\ I(x_1, y_2) \\ I(x_2, y_1) \\ I(x_2, y_2) \end{bmatrix}$$

Remark: Bicubic Interpolation $I(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$