Maneesh Wijewardhana

DIP (Digital Image Processing)

Imaging in the Electromagnetic Spectrum

- image is a function
- Most images = electromagnetic spectrum
- thought of a string of massless photons
- lambda * mu = c (speed of light)
- lambda (wavelength) is a cycle
- mu = frequency = # of cycles / sec = hertz
- energy is related to freq of wave (more wave(short wavelength) = higher energy and vice versa)
- E = h*mu (h is planks constant) unit for energy is electron-volt
- spectral color = singe wavelength of light in the visible spectrum
- Most images are based in radiation in the electromagnetic spectrum (not all)
- Gamma rays
  - medical application
- x-rays
  - medical, astronomy, electronics
  - if X-ray is received by receptor, becomes black
  - cannot go through bone well so becomes lighter
- ultraviolet
  - astronomy, medical, biological
- visible spectrum
  - astronomy, satellites, law enforcement, industries
- infrared
  - satellites
- radio-band
  - medical(MRI's), astronomy
- monochromatic (grayscale)
- panchromatic field (type of black and white photography) ← still grayscale but full visible spectrum
- 2 kinds of receptors in the eye: rodes(b&w, low light), cones(high light level, photopic vision, colour)
- 3 different types of cones
  - dedicated to red spectral colour
  - dedicated to blue spectral colour
  - dedicated to green spectral colour
- blue (440-490nm)
- green (520-570nm)
- red (630-740nm)
- A greylevel image is a **function**
- A colour image is a **triple** (order matters)
- multi-spectral image (**tuple**)
- false colour image is what the human eye cant see which can be done by altering what red blue green is in the wavelength range
  - this can be useful to make things appear that the human eye couldn't see (some ways are useful and some ways are not)

Other Imaging Modalites

- Some images are not based on the radiation from the em spectrum
  - Ultrasound
    - millions of pulses are sent and echos are received
    - build image from that
  - Synthetic images
    - Computer generated images
    - criminal forensics
    - video games

Digital Image Processing Fundamentals

Main steps in DIP

- Steps:
  - ◇ **Image Preprocessing**, **Low Level** (Acquisition and improvement of pictorial information for human interpretation and analysis. Output is image)

1. Image Acquisition
2. Image Restoration (correct the images based on the lengths you are using. An objective change of the image based on mathematical models)
3. Image Enhancement (subjective change of the image)

  - ◇ **High Level** (Autonomous machine perception. Outputs are image attributes)

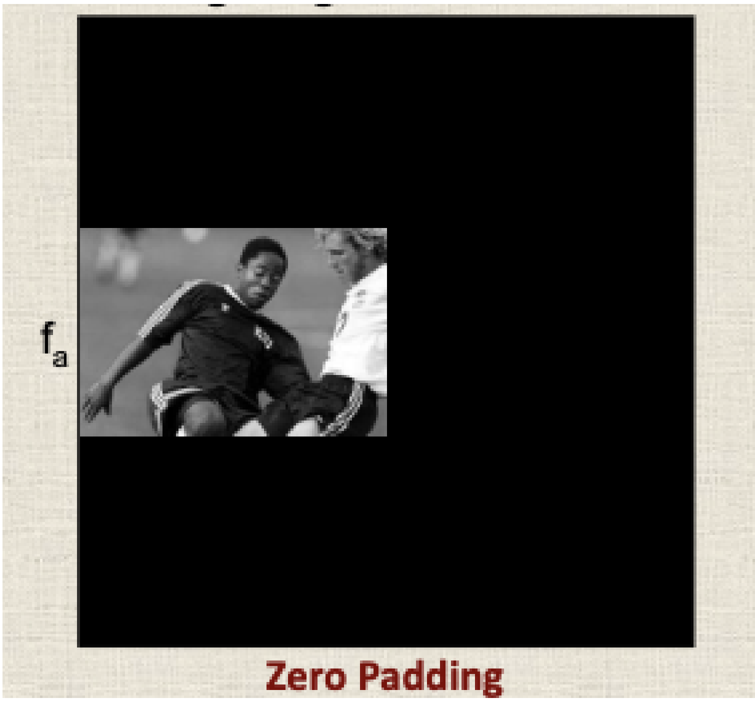
4. Segmentation (Break into pieces where each piece means something. Result of segmentation is a partition)
5. Representation & Description (Extract numerical features and interpret i.e. label the parts from segmentation)
6. Object Recognition (Use features to recognize the image)

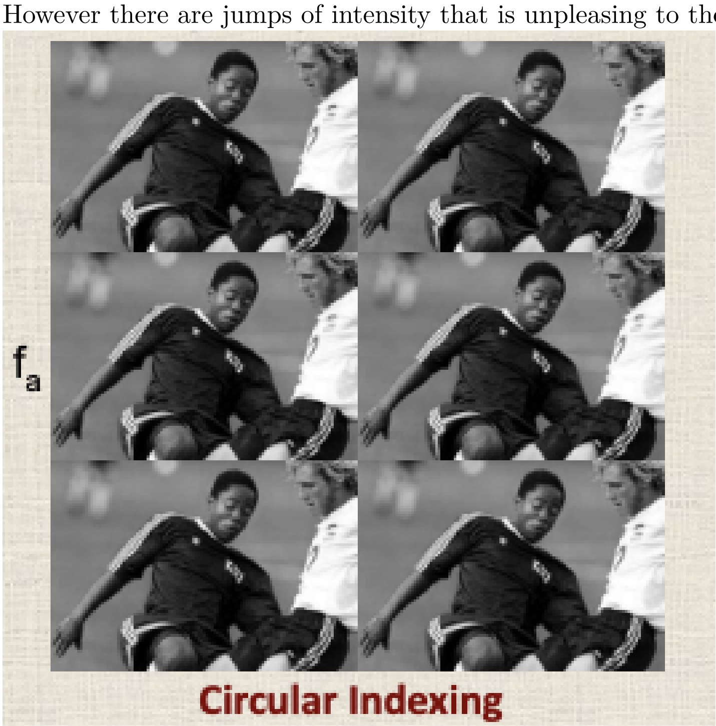
Details

- 4. Partitioning a set
  - ◇ Image is a set of pixels
  - ◇ non-empty set S
    - ∗ $\{S_1, S_2, ..., S_n\} n \in \mathbb{Z}^+$ is a partition (n-partition) $\iff S_1 \cup S_2 \cup \ldots S_n = S$
    - ∗ $\wedge \forall i \in 1..n, S_i \neq \varnothing$
    - ∗ $\wedge \forall (i, j) \in (1..n)^2, i \neq j \to S_i \cap S_j = \varnothing$
- 5. Let's define an attribute called compactness with variables Area and Perimeter
  - ◇ [0, 1] where 0 is not compact at all and 1 is perfectly compact (circle)
  - ◇ $C = 4\pi A/P^2$ and need to minimize P for a given A (P^2 bc we can remove units and 4π to keep in [0, 1])

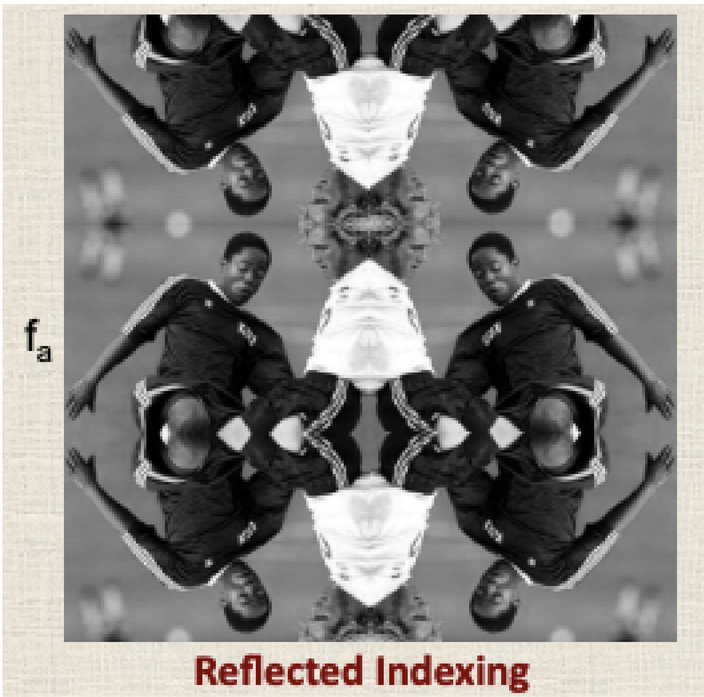
Image Definition and Representation

- **Analog Image**
  - ◇ $(x, y) \in \mathbb{R}^2 [0, +\infty[$ (total function which means infinite image, every $\mathbb{R}^2$ is defined by an intensity level)
  - ◇ **Domain** of definition is the set of elements that have an image
  - ◇ function is **total** $\iff$ all domain elements have a defined **co-domain**
  - ◇ **Range** is the set of elements in the co-domain with a **pre-image**
  - ◇ Let M and N be two positive real numbers
    - ∗ assume the domain of definition of f_a is:
    - ∗ $\{(x, y) \in \mathbb{R}^2 \mid 0 \leq x \leq M \wedge 0 \leq y \leq N\}$
  - ◇ $f_a \mid \mathbb{R}^3 \to \mathbb{R}$ (3D analog image)
  - ◇ $f_a \mid \mathbb{R}^2 \to \mathbb{R}^3$ (colour 2D analog image)
  - ◇ $f_a \mid \mathbb{R}^2 \to C$ (set of complex #'s, ex. spacial frequency in radio image)
  - ◇ $(x, y) \in \mathbb{R}^2 [0, +\infty[$
    - ∗ is the typical case and $f_a(x, y)$ is the intensity of f_a at (x, y)
  - ◇ Zero padding is the way to make a total function

* 

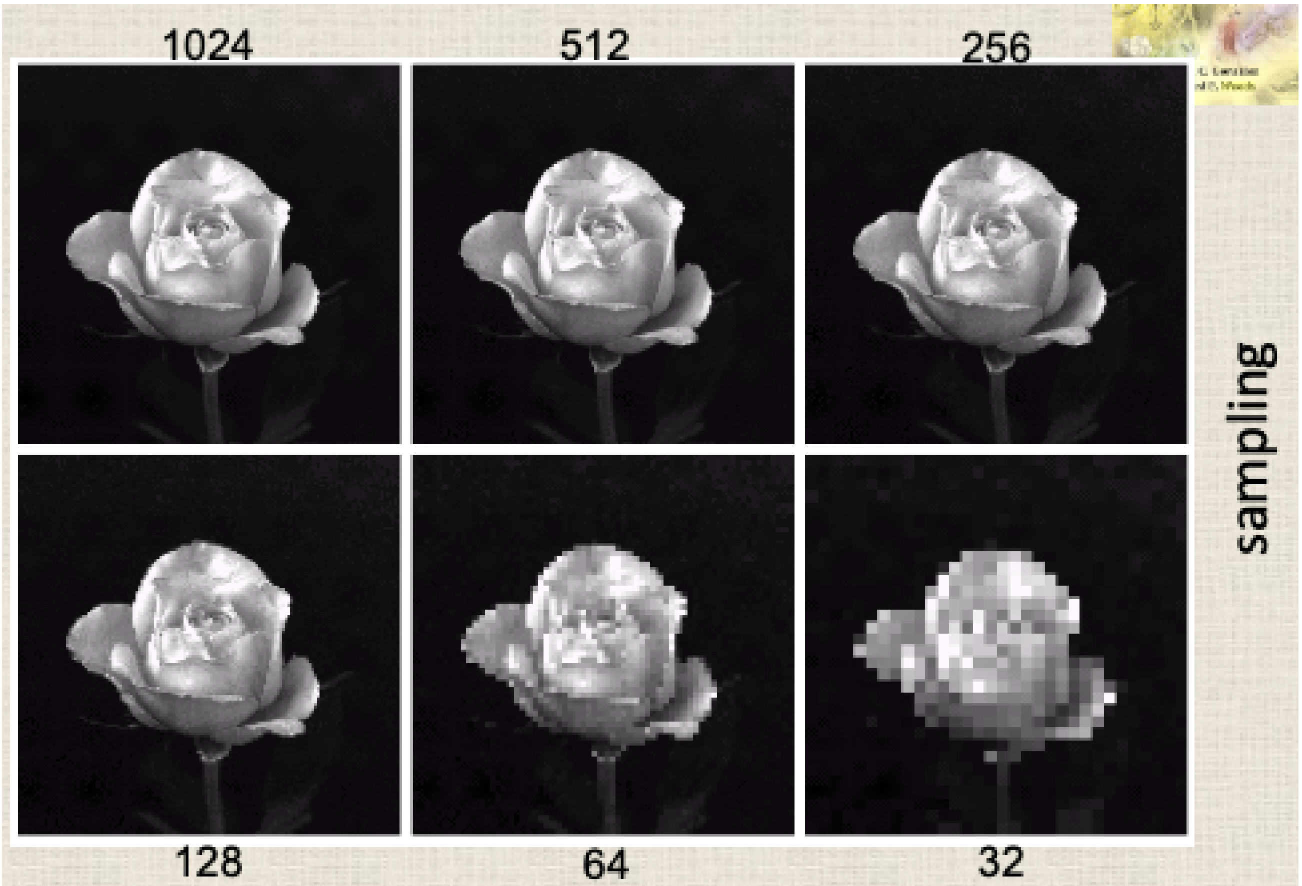
◇ Circular Indexing is the way to tile the image (this is a periodic function)
  * $\forall (x, y) \in \mathbb{R}^2, \forall (i, j) \in \mathbb{Z}^2, f_a(x + iM, y + jN) = f_a(x, y)$
  * This tells me intensity of anywhere in the plane
  * However there are jumps of intensity that is unpleasing to the eye
  * 

◇ Reflected Indexing (reflect images means no artificial jump in intensity)

- 

- **Digital Image**
  - $f \mid \mathbb{Z}^2 \to 0..+\infty$ (grayscale)
    - typical case
  - $f \mid \mathbb{Z}^3 \to \mathbb{Z}$ (3D)
  - $f \mid \mathbb{Z}^2 \to \mathbb{Z}^3$ (2D colour image)
  - $f \mid \mathbb{Z}^2 \to 0..\infty$ (grayscale)
    - $((x, y), f(x, y))$ is a pixel picture element
    - first term is location, second term is gray level (is a graph of f)
  - consider $f \mid \mathbb{Z}^2 \to 0..+\infty$
  - assume range included in 0..L-1 (lowest grayscale is 0, highest is L)
  - typically L = 2^l **l-bit grayscale image**
  - Assume domain of definition is 0..M-1 x 0..N-1
  - number of bits needed for 1 pixel is $l * M * N$
  - $f' \mid \mathbb{Z}^2 \to 0..+\infty$ (**zero padding**)
    - domain of def is $\mathbb{Z}^2$
    - f' is defined as the following
    1. $\forall (x, y) \in 0..M-1 \times 0..N-1, f'(x, y) = f(x, y)$
    2. $\forall (x, y) \in \mathbb{Z}^2, x < 0 \vee x > M \vee y < 0 \vee y > N \to f'(x, y) = 0$
  - $f' \mid \mathbb{Z}^2 \to 0..+\infty$ (**circular indexing**)
    1. $\forall (x, y) \in 0..M-1 \times 0..N-1, f'(x, y) = f(x, y)$
    2. $\forall (x, y) \in 0..M-1 \times 0..N-1, \forall (i, j) \in f'(x + iM, y + yN) = f(x, y)$

- **Digitization**
  - $f_a \mid \mathbb{R}^2 \to [0; +\infty[$
  - $f \mid \mathbb{Z}^2 \to 0..+\infty$
    - $f_a \to f = \textit{digitization}$
    - $\mathbb{R}^2 \to \mathbb{Z}^2 = \textit{sampling}$
    - $[0; +\infty[\to 0..+\infty = \textit{quantinization}$
  - If the size of the image increases, then the checkerboarding effect decreases and vice versa
  - drawback to increasing size is that more space is needed $\therefore$ processing takes longer
  - more quantization = false contouring and less(2 graylevels) = b&w

◇ 

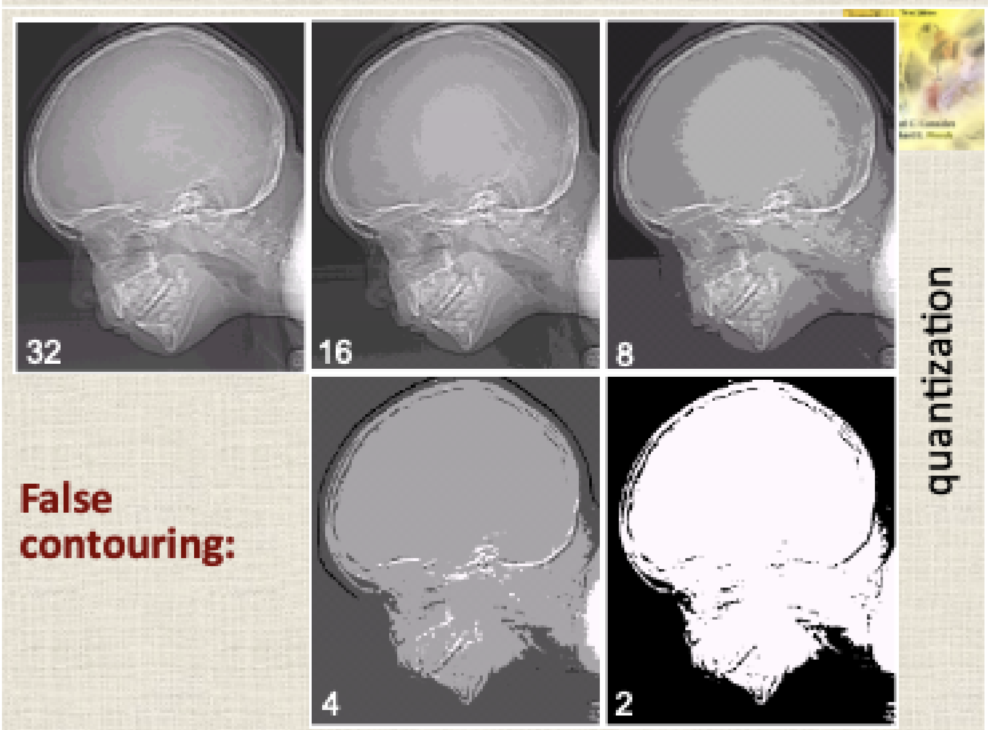
◇ 

Image operations and Transformations

- **Single-Pixel Operations**
  - ◇ $f \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
  - ◇ $g \mid \mathbb{Z}^2 \to \mathbb{R}$ (output image)
  - ◇ $f(x, y) \to T_i \to g(x, y)$ which is:
    - ∗ $g(x, y) = T_i(f(x, y))$
    - ∗ T_i is a function from 0..L-1 to 0..L-1 (intensity transformation function)

- ◇ Example:
  - ∗ $g = af + b$ from $\mathbb{Z}^2 \to \mathbb{R}$
    - · $(x, y) \mapsto (af + b)(x, y) = af(x, y) + b$
    - · $a \in \mathbb{R}$
    - · $b \in \mathbb{R}$
    - · $T_i \mid \mathbb{R} \to \mathbb{R}$
    - · $i \to ai + b$ (linear function)
- ◇ $\neg = 0, 1 \to 0, 1$ (propositional logic)
  - ∗ $x \mapsto 1 - x$
- ◇ $\neg = [0, 1] \to [0, 1]$ (fuzzy logic)
  - ∗ $x \mapsto 1 - x$
  - ∗ $x \mapsto 1 - x/(1 + \lambda x)$ (sugeno)
    - · $\lambda \in]-1, +\infty[$
- ◇ $\exists (x, y) \in \mathbb{Z}^2, g(x, y) \notin 0..255$ (how to make viewable version h, of g)
  - ∗ $h(x, y) =?$
  - ∗ 1. $\forall (x, y) \in \mathbb{Z}^2, h(x, y) = nint(min(255, max(0, g(x, y))))$
  - ∗ 2. $m = min_(x, y) \in \mathbb{Z}^2 g(x, y) \wedge M = max_(x, y) \in \mathbb{Z}^2 g(x, y)$
    - · $\therefore \forall (x, y) \in \mathbb{Z}^2, h(x, y) = nint(255(g(x, y) - m)/(M - m))$

Neighborhood Operations

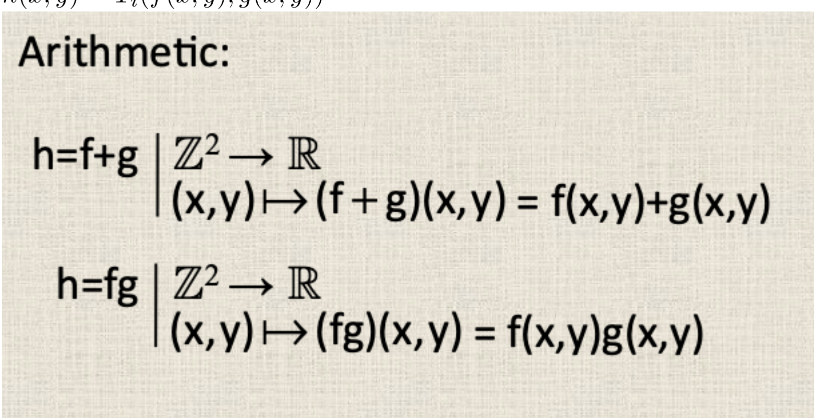
- **Euclidean** d_E (distance on $\mathbb{R}^2$)
  - ◇ $\mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$
  - ◇ $((x_1, y_1), (x_2, y_2)) \mapsto \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}$
  - ◇ **NOTE** $\mathbb{R}^2 \times \mathbb{R}^2 \neq \mathbb{R}^4$ because cartesian product is not associative
- What is the distance on a set S?
  - ◇ $d : S^2 \to \mathbb{R}$
  - ◇ 1. $\forall (x, y) \in S^2, x = y \iff d(x, y) = 0$ (definite)
  - ◇ 2. $\forall (x, y) \in S^2, d(x, y) = d(y, x)$ (symmetric)
  - ◇ 3. $\forall (x, y, z) \in S^3, d(x, y) \leq d(x, y) + d(y, z)$ (Triangle Inequality)
- Proposition
  - ◇ $\forall (x, y) \in S^2, d(x, y) \geq 0$
- Proof
  - ◇ Let x and y be two elements of S
  - ◇ Assume $d(x, y) = -1$
  - ◇ According to the Triangle Inequality $d(x, x) \leq d(x, y) + d(y, x)$
  - ◇ $\therefore 0 \leq -2$
  - ◇ contradiction $\therefore$ assumption is wrong
- **City-block** d_4 (distance on $\mathbb{Z}^2$) (typical in DIP)
  - ◇ $d_4 : \mathbb{Z}^2 \times \mathbb{Z}^2 \to \mathbb{R}$
  - ◇ $((x_1, y_1), (x_2, y_2)) \mapsto |x_2 - x_1| + |y_2 - y_1|$
- $(x_0, y_0) \in \mathbb{Z}^2$ (what is the neighborhood of (x_0, y_0))?
  - ◇ The set of neighbors of (x_0, y_0) is $\{\ (x, y) \in \mathbb{Z}^2 \mid d_4((x, y), (x_0, y_0)) = 1\ \}$
    - ∗ $p \in \mathbb{Z}^2$
    - ∗ $\{\ q \in \mathbb{Z}^2 \mid d_4(q, p) = 1\ \}$
- **chessboard**
  - ◇ $d_8 : \mathbb{Z}^2 \times \mathbb{Z}^2 \to \mathbb{R}$
  - ◇ $((x_1, y_1), (x_2, y_2)) \mapsto max(|x_2 - x_1|, |y_2 - y_1|)$
- **General formula for all d**
  - ◇ $d_n : ((x_1, y_1), (x_2, y_2)) \mapsto \sqrt[n]{|x_2 - x_1|^n + |y_2 - y_1|^n}$
- **4-neighbors** of p means city block and **8-neighbors** of p means chessboard
- The bigger the neighborhood, the more blurry it will be

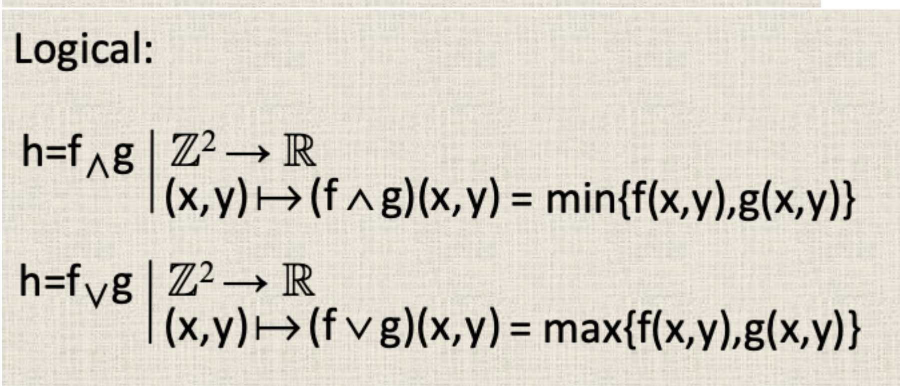
Geometric Spatial Transformations (uses inverse T)

- $f \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- $g \mid \mathbb{Z}^2 \to \mathbb{R}$ (output image)
- $g(u, v)$ depends on the gray levels from some neighborhood of (x, y) in f
- f is an image from $M - 1 \to N - 1$ and g is an image from $P - 1 \to Q - 1$
- $T_s : \mathbb{R}^2 \to \mathbb{R}^2$
- $(x, y) \mapsto T_s(x, y) = (u, v)$

- $u = x(P - 1/M - 1) \wedge v = y(Q - 1/N - 1)$
- T_s^- is $x = u(M - 1/P - 1) \wedge y = v(N - 1/Q - 1)$
- Note:
  - ◇ $f : \mathbb{Z}^2 \to \mathbb{R}$
  - ◇ $\overline{f} : \mathbb{R}^2 \to \mathbb{R}$
  - ◇ $(x, y) \mapsto \overline{f}(x, y)$
  - ◇ When $(x, y) \in \mathbb{Z}^2, \overline{f}(x, y) = f(x, y)$
- **Bilinear interpolation** (do it for x and y)
  - ◇ $\overline{f}(x, y_0) = (1 - s) * f(x_0, y_0) + s * f(x_0 + 1, y_0)$
  - ◇ $\overline{f}(x, y_0 + 1) = (1 - s) * f(x_0, y_0 + 1) + s * f(x_0 + 1, y_0 + 1)$
  - ◇ $\overline{f}(x, y) = (1 - s) * \overline{f}(x, y_0) + t * \overline{f}(x, y_0 + 1)$
  - ◇ $s = x - x_0 \wedge t = y - y_0$

Image Transforms (cartesian coordinates to polar coordinates)

- **Binary single-pixel operations**
- $f \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- $g \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- $h \mid \mathbb{Z}^2 \to \mathbb{R}$ (output image)
- $h(x, y) = T_i(f(x, y), g(x, y))$
- 

- 

  - ◇ $\wedge$: {0,1} × {0,1} → {0, 1} (classical)
  - ◇ $\wedge : [0, 1] \times [0, 1] \to [0, 1]$ (fuzzy conjunction)
    - ∗ we need this to be commutative, associative, distributive, have a neutral element($\wedge : 1, \vee : 0$), and compliment laws ($\neg x \wedge x = 0, \neg x \vee x = 1$)
    - ∗ also need monotonicity and continuity
  - ◇ $(x, y) \mapsto xy$ (algebreic, violates compliment laws)
  - ◇ $(x, y) \mapsto min(x, y)$ (standard conjunction, violates compliment laws but keeps others)
  - ◇ $(x, y) \mapsto max(x, y)$ (standard disjunction, violates compliment laws but keeps others)
  - ◇ $(x, y) \mapsto x + y - xy$

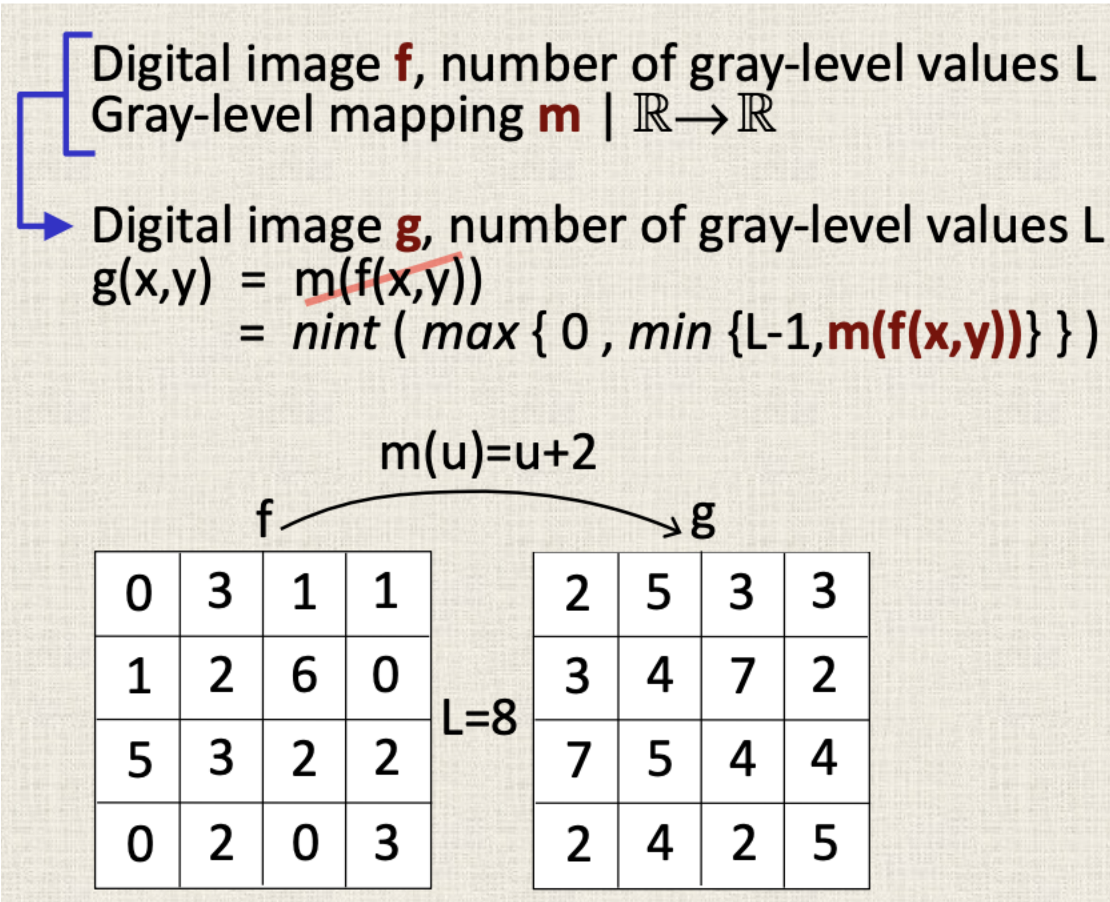
Image Registration (2 images, different angles)

- $f \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- $f_2 \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- What is T_s? (the spatial transformation)

Single-Pixel Operations (in depth)

- $f \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- $g \mid \mathbb{Z}^2 \to \mathbb{R}$ (output image)
- $f(x, y) \to T_i \to g(x, y)$
- $g(x, y) = T_i(f(x, y))$

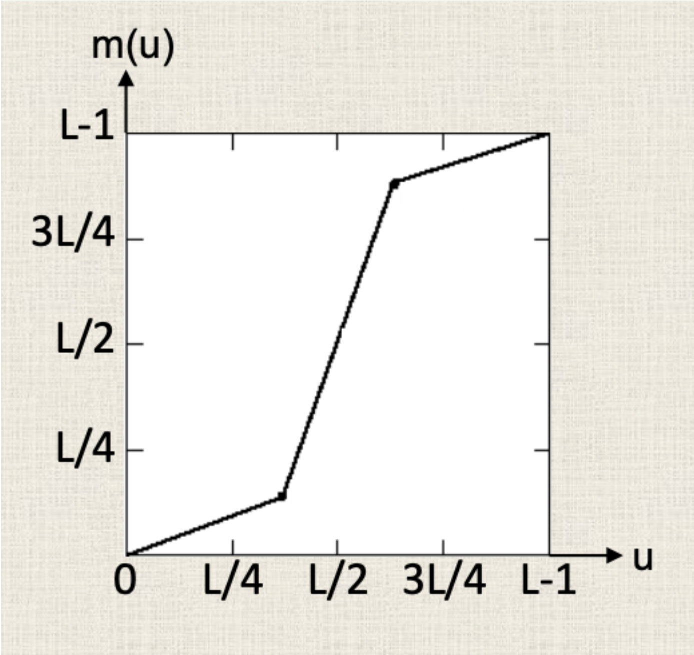
Gray-Level Mappings

- 

Linear Mappings

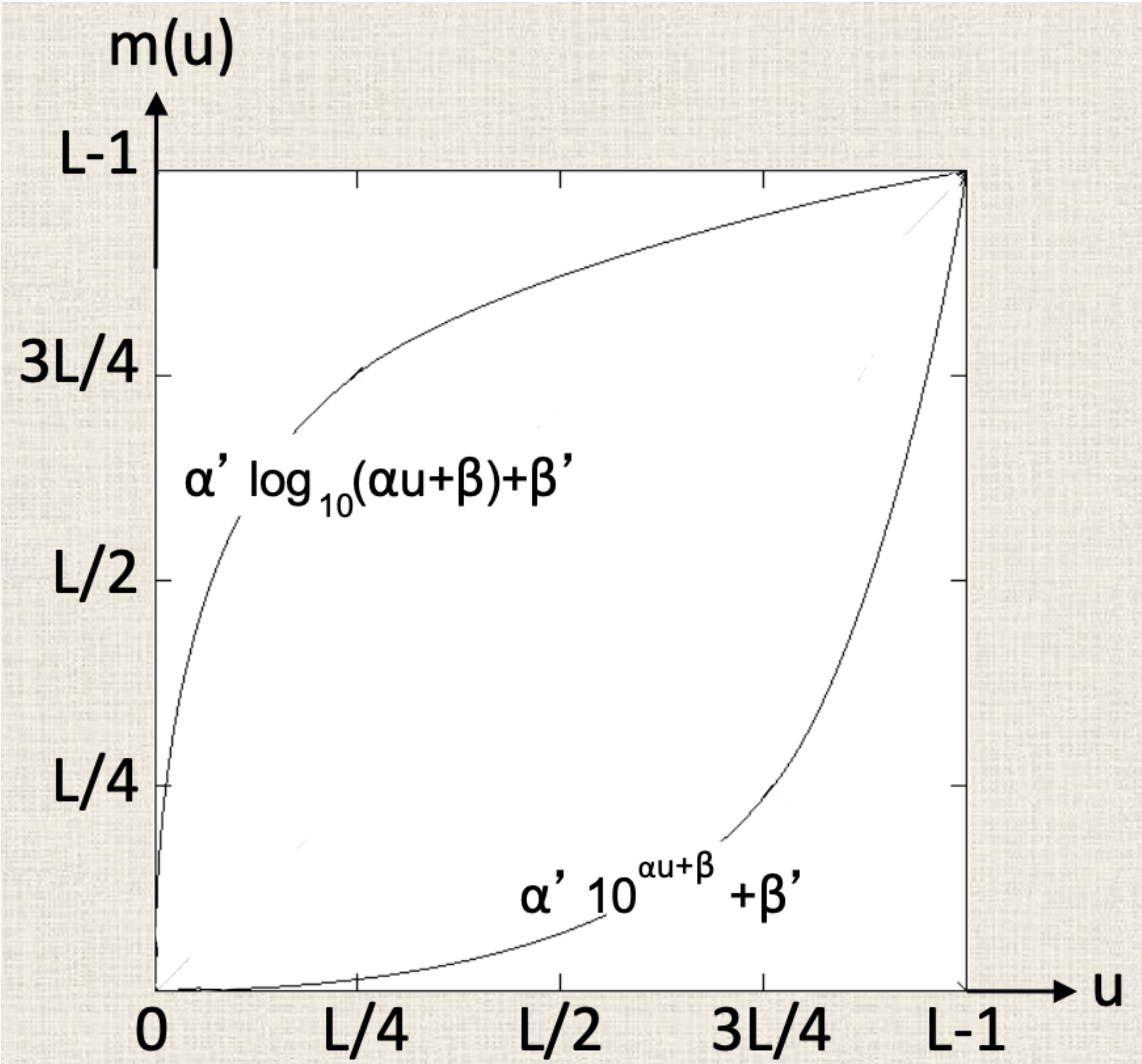
- $m(u) = au + b$
  - ◇ b is the **bias** (impacts **brightness**)
  - ◇ a is the **gain** (impacts **contrast**)
- Example:
  - ◇ if $f(x, y) \in 0..127 \to g(x, y) = 0$
  - ◇ if $f(x, y) \in 128..255 \to g(x, y) = 255$
  - ◇ What is a and b?
  - ◇ $m(u) = 255u - (127 * 255)$
  - ◇ meaning $(a >> 1 \wedge b << 0)$

Piecewise Linear Mappings (can be continuous or discontinuous)

- 

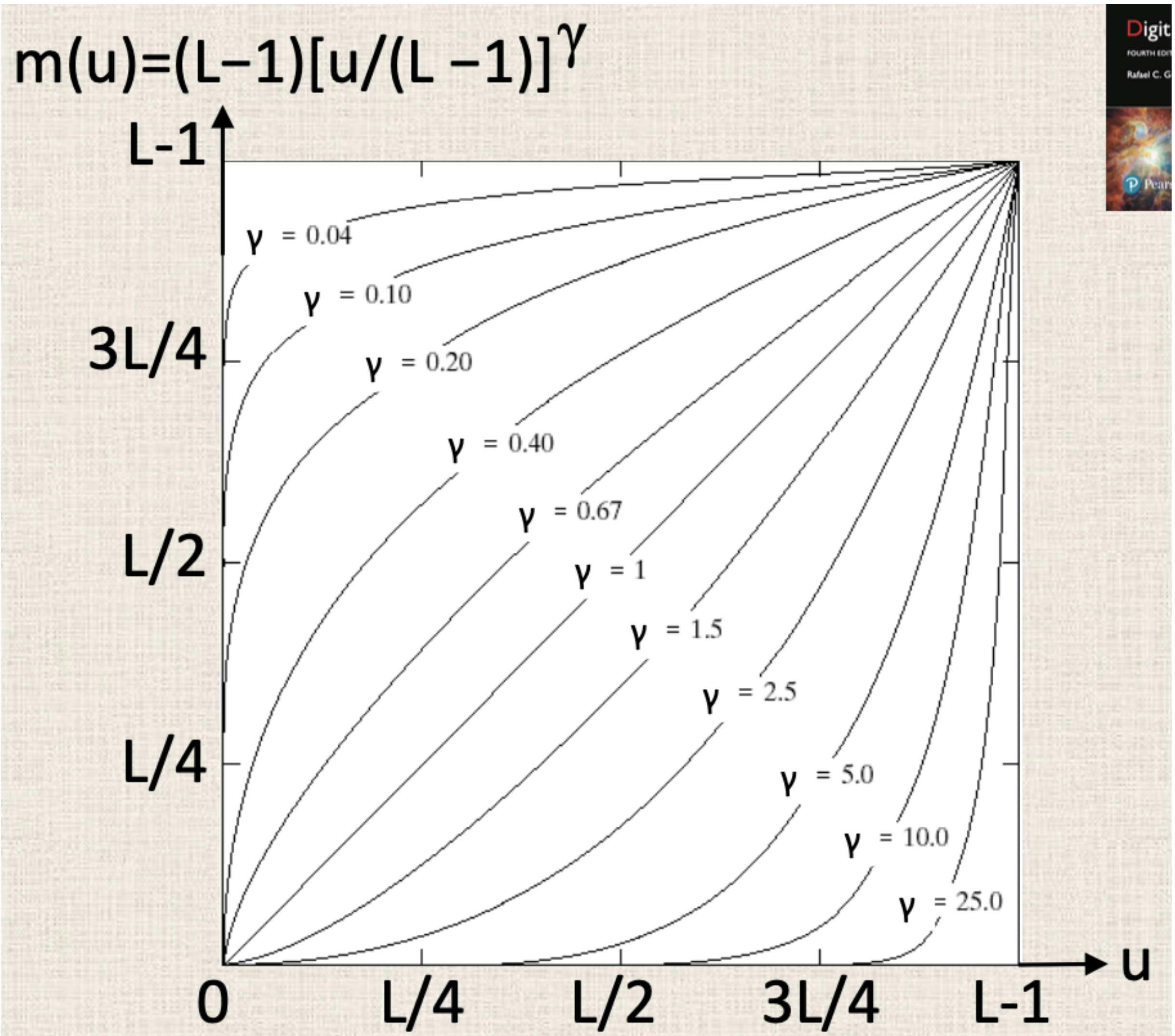
- decreases contrast in the darker and lighter areas
- increases contrast in the mid-grey level

Logarithmic and Exponential Mappings

- 

- $m(u) = (L-1)\log(1 + (9/L-1)u)$
- log
  - decreasing contrast for brighter region and increasing for darker
- exponential
  - decreasing contrast for darker region and increasing for lighter

Power-law Mappings

- 

- **Gamma Correction**
- $m_\gamma(u) = (L-1)[u/(L-1)]^\gamma$
- $m_{1/\gamma}(m_\gamma(u)) =?$
  - $m_{1/\gamma}((L-1)(u/L-1)^\gamma$
  - $(L-1)[(L-1)(u/L-1)^\gamma)/L-1]^{1/\gamma}$
  - $= u$
  - $\therefore m_{1/\gamma} \; o \; m_\gamma = I = m_\gamma \; o \; m_{1/\gamma}$ (bijection)
- $m_\gamma : [0, L] \to [0, L]$
  - $u \mapsto (L-1)(u/L-1)^\gamma$
  - $(m_\gamma^{-1}) = m_{1/\gamma}$

Final Remarks

- Gray-level mappings useful for image enhancement (brightness and contrast adjustment)
- There is often loss of information (even if there is perceptual improvement)
- **Subjective quality can increase, but objective quality can decrease**

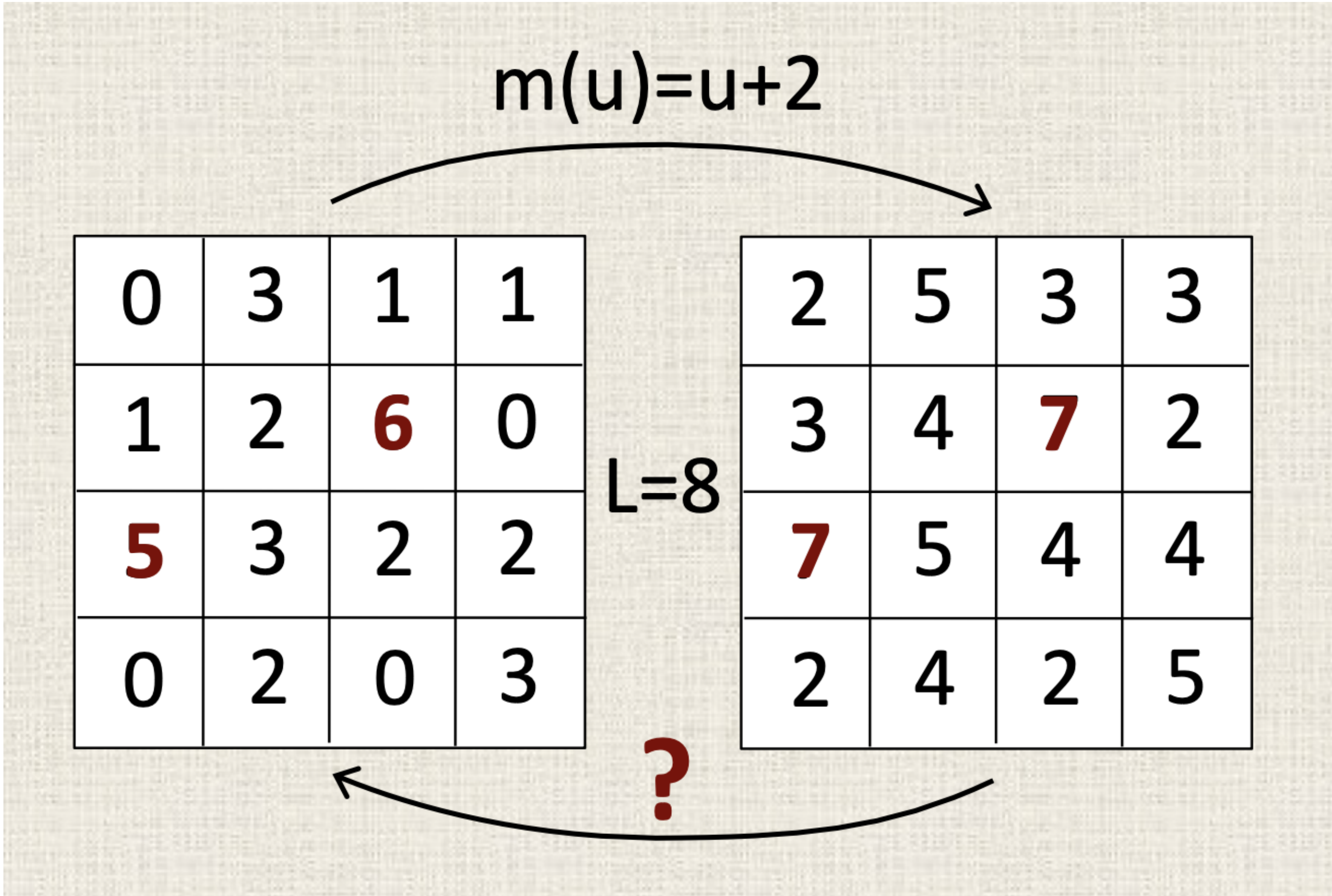
- 

Image Histograms

- Consider image $f : 0..M-1 \times 0..N-1 \to 0..L-1$
- $H_f : 0..L-1 \to 0..MN$ (amount of pixels in the image with gray level u)
  - $u \mapsto H_f(u)$
  - $H_f(u) = \mid \{(x,y) \in 0..M-1 \times 0..N-1 \mid f(x,y) = u\} \mid$
- $H_f^n : 0..L-1 \to [0,1]$ (normalized)
  - $u \mapsto H_f^n(u)$
  - $H_f^n(u) = H_f(u)/MN$
- $H_f^c : 0..L-1 \to 0..MN$ (cumulative)
  - $u \mapsto H_f^c(u)$
  - $H_f^c(u) =\mid \{(x,y) \in 0..M-1 \times 0..N-1 \mid f(x,y) \leq u\} \mid$
  - $H_f^c(L-1) = MN$
  - $H_f^c(0) = H_f(0)$
  - $H_f^c(1) = H_f(0) + H_f(1)$
  - **better definition**
    - H_f^c is the function from $0..L-1 \to 0..MN$ defined as follows:
    - $\forall u \in 0..L-1, H_f^c(u) = \sum_{i=0}^{u} H_f(i)$
- $H_f^{cn} : 0..L-1 \to 0..MN$ (cumulative normalized)
  - $u \mapsto H_f^{cn}(u)$
  - H_f^{cn} is the function from $0..L-1 \to [0,1]$ defined as follows:
  - $\forall u \in 0..L-1, H_f^{cn}(u) = \sum_{i=0}^{u} H_f^n(i)$
    - **or** $= \sum_{i=0}^{u} H_f(i)/MN$
    - **or** $= H_f^c(u)/MN$

Neighborhood Operations (in depth)

- $f \mid \mathbb{Z}^2 \to \mathbb{R}$ (input image)
- $g \mid \mathbb{Z}^2 \to \mathbb{R}$ (output image)

Convolution

- **Principle**
  - ◇ $g(x, y)$ is a **weighted sum** of gray levels from some neighborhood of $(x, y) \in f$
  - ◇ Weighted sum can depend on if we use zero-padding or some other type of indexing as the values can change accordingly
  - ◇ Remember that the middle of the mask is what goes on each gray level and multiplying them gets the value at that location
  - ◇ $h : m \times n$ (m and n are odd in order to center the kernel on the pixel of interest)
  - ◇ $f : M \times N$ (assume zero-padding)
  - ◇ g?
    - ∗ For any $(x, y) \in \mathbb{Z}^2, g(x, y) = \sum_{i=-m-1/2}^{m-1/2} \sum_{j=-n-1/2}^{n-1/2} h(i, j)f(x - i, y - j)$
    - ∗ This output image $g(x, y)$ is $h * f$
    - ∗ This double sum is $(h * f)(x, y)$
      - · $h * f$ is the **convolution** of h with f
- Show how convolution is commutative $(h * f) = (f * h)$
  - ◇ $(h * f)(x, y) = \sum_{i=-m-1/2}^{m-1/2} \sum_{j=-n-1/2}^{n-1/2} h(i, j)f(x - i, y - j)$
  - ◇ *Let* $I = x - i \wedge J = y - j$
  - ◇ $(h * f)(x, y) = \sum_{I=-m-1/2}^{m-1/2} \sum_{J=-n-1/2}^{n-1/2} h(I, J)f(x - I, y - J)$
  - ◇ $(h * f)(x, y) = \sum_{I=-m-1/2}^{m-1/2} \sum_{J=-n-1/2}^{n-1/2} f(I, J)h(x - I, y - J)$
  - ◇ $= (f * h)(x, y)$ which means $(h * f) = (f * h)$
- Show how convolution is linear (a is some constant)
  - ◇ $f * g$
  - ◇ 1. $(af) * g = a(f * g)$
  - ◇ 2. $(f_1 + f_2) * g = f_1 * g + f_2 * g$
  - ◇ **or**
  - ◇ 1. $f * (ag) = a(f * g)$
  - ◇ 2. $f * (g_1 + g_2) = f * g_1 + f * g_2$
  - ◇ $(h * f)(x, y) = \sum_{i=-m-1/2}^{m-1/2} \sum_{j=-n-1/2}^{n-1/2} h(i, j)f(x - i, y - j)$
    - ∗ $((ah) * f)(x, y) = \sum_{i=-m-1/2}^{m-1/2} \sum_{j=-n-1/2}^{n-1/2} ah(i, j)f(x - i, y - j)$ (**can factorize**)
    - ∗ $= a \sum_{i=-m-1/2}^{m-1/2} \sum_{j=-n-1/2}^{n-1/2} h(i, j)f(x - i, y - j)$
    - ∗ $= a(h * f)(x, y)$
    - ∗ $(ah) * f = a(h * f)$ ■
- **Handling of the Borders**
  - ◇ $M - (m - 1)$
  - ◇ $N - (n - 1)$

Filtering

- **principle**
  - ◇ $f(t) = \sum_{u=0}^{+\infty} c_u cos(\omega ut + \varphi_u)$
  - ◇ $u = 0$ $\quad c_0 cos(\varphi_0)$
  - ◇ $\varphi_0 = 0$ $\quad c_0$
  - ◇ $u = 1$ $\quad 2\pi/t$ in radians/s
  - ◇ For better approximations, divide the period by 2, 3, etc. . .
  - ◇ Decreasing amplitude of low freq = high-pass filter
  - ◇ Decreasing amplitude of high freq = low-pass filter
  - ◇ To blur the image = low-pass filter. brightness is x-axis (vertical axis)
  - ◇ To sharpen the image = high-pass filter
- **Linear Low-Pass Filtering** (Convolution kernel with positive coeff)

- ◇ **mean filtering** (uniform)
  - ∗ 3x3 mean mask [[1 1 1] [1 1 1] [1 1 1]] (seperable)
- ◇ For a smoother kernel, use **Gaussian filtering** (non-uniform)
  - ∗ 5x5 Gaussian mask (seperable with real values but not the rounded integer example)
  - ∗

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	4
2	4	5	4	2

  - ∗ probability distribution (double integral = 1)

- **Linear High-Pass Filtering** (Convolution kernel with mixture of positive and negative coeff)
  - ◇ **Laplacian filtering** (omnidirectional) (og image disappears but edges are defined)
  - ◇

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

3x3 Laplacian masks

  - ◇ To sharpen, we do Laplacian but also add the image
    - ∗ **unsharp masking** = subtracting from image output from low pass filter
    - ∗ **high-boost filtering** = adding to image output from high pass filter
    - ∗

0	−1	0
−1	$A + 4$	−1
0	−1	0

−1	−1	−1
−1	$A + 8$	−1
−1	−1	−1

    - ∗ Bigger “A” value means less sharpening as the -1’s have less significance

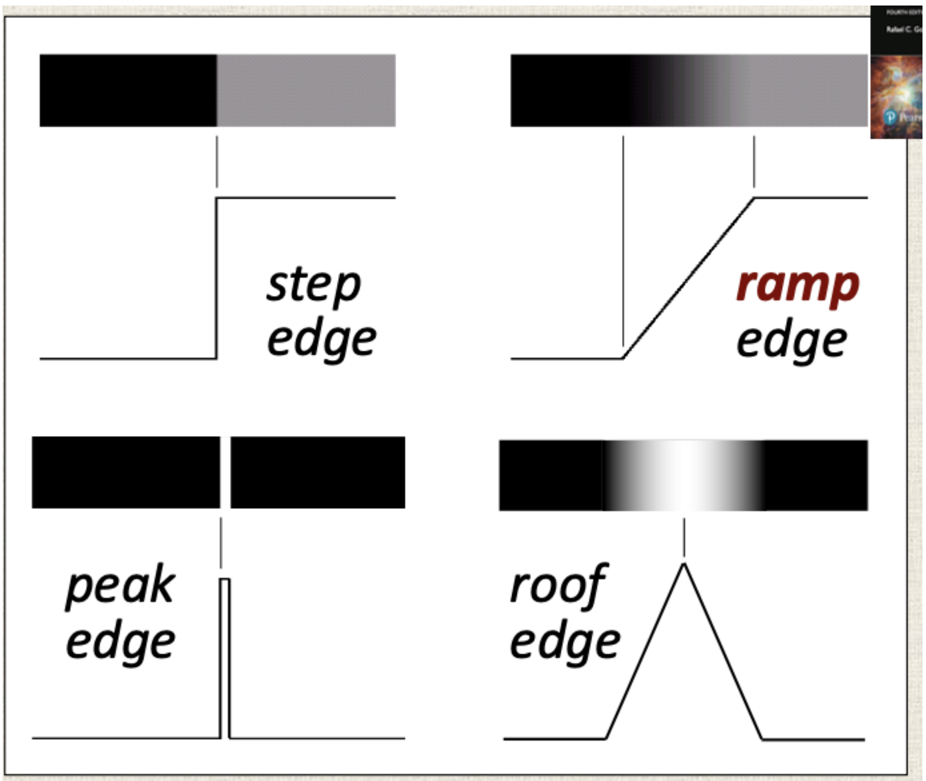
- **Order Statistic (Non-Linear) Filtering**
  - ◇ If you take the min in sorted list(min filter), you are removing white(salt) noise
    - ∗ The darker regions get bigger as well
    - ∗ SN: probability that pixel is 255 (we will define that probability)
  - ◇ If max(max filter), you are removing black(pepper) noise
  - ◇ If median(median filter), you are removing salt and pepper noise
    - ∗ to avoid cutting corners off of rectangle, use ‘+’ shape mask. eg. [[0 1 0][1 1 1][0 1 0]]

- **α-Trimmed Mean (Non-Linear) Filtering**

- ◇ discard first α values and last α values
  - ◇ for remaining values, compute average which will be the value attached to the output pixel
  - ◇ Choosing $\alpha = 4$ considering chessboard neighborhood we get a median filter
  - ◇ $\sum_{i=\alpha+1}^{mn-\alpha} u_i/MN - 2\alpha$

- **Adaptive (Non-Linear) Filtering**
  - ◇ we calculate mean gray level: $m(p)$ and variance: $V(p)$
  - ◇ how to calculate variance?
    - ∗ $\sum_i (f(q_i) - m(p))^2/9$
  - ◇ $g(p) = f(p) - [f(p) - m(p)]V_n/V(p)$
    - ∗ V_n is the estimate of the noise variance
      - · we look at the homogenous regions of the image since noise will not make them uniform (ideally, its is 0 which means no noise)

Edge Detection

- 

- **principle**
  - ◇ Step 1 = **noise reduction**
    - ∗ 1x3 filter to remove noise [1 1 1] without smoothing away the meaningful edge
  - ◇ Step 2 = **edge enhancement**
    - ∗ 3x1 filter like [-1 0 1]
  - ◇ Step 3 = **edge localization**
    - ∗ with 2 filters convolved, we get a 3x3 separable kernel
    - ∗ decide which output values are meaningful edges
- In ramp gradient, we need to calculate the intensity profile and figure out if there is a maximum/inflection point using first derivative
- When edge is horizontal, $v' = 0$ and $u' \neq 0$
- When edge is vertical, $u' = 0$ and $v' \neq 0$
- We care about both, so we create a **gradient vector**
- **steps to detect edges in DIP**
  - ◇ Convolve with sobel kernel or prewitt kernel wrt to x and y (do convolution with first, then convolve with second)
  - ◇ from the two partial derivatives images, add the abs value of the
  - ◇ using the magnitudes, use a threshold to assign 255 value

Image Laplacian

- **principle**
  - Same as regular but we take the zero point(zero-crossing) in second derivative
  - Thinner edges than regular, results in spaghetti effect
- **Laplacian of Gaussian**
  - Marr-Hildreth is what the final kernel is called

Thresholding and Labeling (High level step)

- It helps to calculate the histogram so we know what threshold to choose
- To calculate threshold automatically:
  - $u_{new} \leftarrow average f(x, y)$ (can find average naively by scanning image and adding up all pixels and dividing by number of pixels) **OR**
  - average f(x,y) = $\sum_{u}^{255} u\, H_f(u) / \sum_{u=0}^{255} H_f(u)$

Connected Components

- **Adjacency**
  - Let $S_1 \wedge S_2$ be two subsets of $\mathbb{Z}^2$
  - We say that S_1 is adjacent to S_2 iff:
  - $min\; d(p_1, p_2) = 1$
    - $p_1 \in S_1 \wedge p_2 \in S_2$
  - Tangency into the discrete space is adjacency
- **Paths** (a tuple)
  - Empty tuple is not a path, need at least 1
  - $\Pi = (p_0, p_1, ...p_n)$
  - Π is a 4 or 8 path in S iff:
    - $\forall p_i \in \Pi, p_i \wedge p_{i+1}$ are adjacent
  - p is 8 connected to q in S iff:
    - there exists a path from p to q (if 8 connected we need an 8 path, same for 4)
  - Adjacency is not a equivalence relation
    - 8-connected and 4-connected is however
    - equivalence classes are the connected components of S (they define a partition of S)
    - In Example:
      - 1 8-connected component
      - 3 4-connected component
    - S is connected iff:
      - $\forall (p, q) \in S^2, \exists\, \Pi \ni_S \;|\; \Pi_{first} = p \wedge \Pi_{last} = q$
- **Labeling**
  - Scan image from left to right (top to bottom) and while scanning, build a relation (we will worry about transitive later)
  - If two labels overlap then choose either and adjust relation matrix to make them symmetric
  - **Warshall Algorithm** $\mathbb{R}^+$ to find Transitive closure (flip as few 0's to make it transitive)

    *
```
ℜ+ = ℜ
% ℜ+ is now [ r_ij ]
for k=1 to n
    for i=1 to n
        for j=1 to n
            r_ij=max{r_ij,min{r_ik,r_kj}}
            % if r_ik and r_kj then r_ij=1
            % else keep r_ij as is
```
  - ◇ Identify equivalence classes
  - ◇ Scan image again and replace labels according to equivalence classes

Towards Object Recognition

Representation of Image regions

- **Polygonal Approximations**
  - ◇ Consider two farthest points
  - ◇ Look at two halves of the shape determined by the major axis (2 farthest point) (also the diameter of the shape)
  - ◇ Calculate max distance when walking on the boundary
  - ◇ Do max distance again and split again (stop splitting when max distance below some threshold)
  - ◇ Take proportion of diameter as an example of a threshold
- **Signatures**
  - ◇ First calculate centroid of shape
    - ∗ Get average x coordinates of all pixels that define the region (do same for y)
  - ◇ consider left to right axis
  - ◇ Walk around perimeter and calculate the distance for every theta value (for a circle, it will all be the same)
  - ◇ Result is a 1D function from a 2D shape which is the **Signature** of the image
  - ◇ If you scale, the signature will not change, if you rotate, it will be shifted
  - ◇ If you want to make it rotation invariant, find a shape that will always get the same signature
    - ∗ We would have to normalize the signature
      - · Choose the max, and shift (can choose min too)
  - ◇ If you divide all the distance by the average, we made the signature invariant to scaling

Boundary and Regional Descriptors

- **Geometric Descriptors**
  - ◇ Boundary Descriptors are external, Regional is internal
  - ◇ Eccentricity is invariant to translations, rotation, and scaling (we are still using the same diameter)
- **Shape Numbers**
  - ◇ By calculating the first difference of the chain codes, rotations do not change it but changing the starting point **does** change it
  - ◇ By calculating the shape number, it is now rotation and starting point invarient
    - ∗ It is calculated by shifting the first difference until it is the smallest possible number if you consider the entire difference as a decimal number
  - ◇ Higher order number is not necessarily better since in digital images, you will capture a lot of noise but it should not be too small because then we cannot capture the essence of the shape

Texture

- **Descriptors**

- ◇ $max\ p_{ij}$
- ◇ $energy\ \sum p_{ij}^2$
  - ∗ Max value is when the image is constant grey level
  - ∗ Min value is when all p_{ij} is small
- ◇ $entropy\ -\sum p_{ij} log_2 p_{ij}$ (measures randomness also known as disorder)
  - ∗ Max value is when all values are equal
    - · $2 * log_2 L$
    - · If we have 256 grey levels, max will be 16
    - · For 8 bit grey level images, it ranges from 0-16
  - ∗ Min value is when image is constant grey level
    - · When image is uniform, there is no disorder