Intensity Transformation and Spatial Filtering

Spatial domain

- Image plane
- Image processing methods based on direct manipulation of pixels
- Two principal image processing technique classifications
  1. Intensity transformation methods
  2. Spatial filtering methods

Background

- Spatial domain
  - Aggregate pixels composing an image
  - Computationally more efficient and require less processing resources for implementation
- Spatial domain processes denoted by the expression

$$g(x, y) = T[f(x, y)]$$

  - $f(x, y)$ is input image
  - $g(x, y)$ is output image
  - T is an operator on f, defined over some neighborhood of (x, y)
  - T may also operate on a set of images (adding two images)
- Neighborhood of a point (x, y)
  - Square or rectangular subimage area centered at (x, y) (Figure 3.1)
    * Typically, the neighborhood is much smaller than the image
  - Center moves over each pixel in the image
  - T is applied at each point to get g at that location
    * Compute the average intensity of the neighborhood
  - Also possible to have neighborhood approximations in the form of a circle
  - The above application is also called spatial filtering
    * Neighborhood may be extracted by a *spatial mask*, or *kernel*, or *template*, or *window*
  - Handling pixels at image border
    * Part of the neighborhood is outside the image frame
    * Outside pixels can be ignored; replaced by a uniform gray scale; replaced by 0; or inner pixels can be *reflected* outside
- Single pixel neighborhood, or point processing techniques
  - Simplest form of T
  - Smallest possible neighborhood of size 1×1
    * g depends only on the value of f at a single point (r, c)
  - Gray-level (or intensity) transformation function of the form

$$s = T(r)$$

    - r and s denote the gray level of $f(x, y)$ and $g(x, y)$
  - Contrast stretching and thresholding (Figure 3.2)
- Larger pixel neighborhoods (mask processing or filtering)
  - Use a neighborhood around (x, y) to determine the value of $g(x, y)$
  - Neighborhood defined by masks, filters, kernels, templates, or windows (all refer to the same thing)
  - A kernel is a small 2D array whose coefficients determine the nature of the process

Basic gray level transformations

- Pixel values denoted by r and s before and after transformation
- Transform denoted by $s = T(r)$
- Mapping performed by lookup tables (LUTs)
- Figure 3.3 – Basic transformations
- Image negatives
  - Equivalent of a photographic negative
  - Input gray level range: $r = [0, L - 1]$
  - Transformation is given by $s = (L - 1) - r$
  - Suited for enhancing white or gray detail in dark areas of image, specially when dark areas are dominant in size (Figure 3.4)
- Flipping and flopping
  - Simplest geometric transformation
  - Reflect the image about a horizontal axis (flip) or a vertical axis (flop)
  - Figure 3.1 and 3.2 (Birchfield)
  - Assume an image of size w columns and h rows
  - Flip

$$x' = x; y' = h - 1 - y$$

    - Flip, forward mapping

$$g(h - 1 - r, c) = f(r, c)$$

    - Flip, reverse mapping

$$g(r, c) = f(h - 1 - r, c)$$

  - Flop

$$x' = w - 1 - x; y' = y$$

    - Flop, forward mapping

$$g(r, w - 1 - c) = f(r, c)$$

    - Flop, reverse mapping

$$g(r, c) = f(r, w - 1 - c)$$

  - Flip-flop is a flip operation followed by a flop
- Log transformations
  - General form given by: $s = c \log(1 + r)$ (Figure 3-3)
    - c is a constant

        * $r \geq 0$
    - Maps a narrow range of gray level values in input image to a wider range of output levels, or the other way round with inverse log transform
    - Log function compresses the dynamic range of images with large variation in pixel values
    - Easiest way to generate log transforms is by using a lookup table, and scaling the input to the range [0,1]
    - Example given by Fourier spectrum (Figure 3-5)

- Power-law or gamma transformations
    - General form given by: $s = cr^\gamma$ (Figure 3-6)
        * c and γ are positive constants
    - General form may also be written as $s = c(r + \epsilon)^\gamma$ to account for an offset
        * Offset is useful for some measurable output when the input is zero
        * Typically, an issue of display calibration and hence, normally ignored
    - Fractional value of γ map a narrow range of dark input values into a wider range of output values; opposite is true for higher values of input levels
    - The transformation reduces to identity values when $c = \gamma = 1$
    - Gamma-correction
        * Different devices respond differently to pixel values according to power law
        * Typical values of γ for CRTs is between 1.8 and 2.5
        * Important if displaying the image accurately on a computer screen is of concern
            · Images not properly corrected can look bleached out or too dark
    - General-purpose contrast manipulation
        * Playing with the gamma to enhance detail in a desired region (Figure 3.9)

- Piecewise-linear transformation functions
    - Form of function can be arbitrarily complex
    - Specification requires considerable user input
    - Contrast stretching
        * Image contrast could be low due to poor illumination, lack of dynamic range in the sensor, or wrong setting of lens aperture during image acquisition
        * Increase the dynamic range of gray levels in the image being processed to the full intensity range of recording medium or display device
        * Figure 3.10
        * Thresholding function
    - Gray-level slicing
        * Figure 3-11
        * Two approaches
            1. Display in one value (white) all the values in the range of interest, and change to black everything else
            2. Brighten or darken desired range of intensities leaving all other intensity levels unchanged
            3. Figure 3-12
    - Bit-plane slicing
        * Figure 3-13
        * Most changes can be captured in significant bits
        * You may not be able to perceive minute changes reflected in low bits
        * Figure 3-14

        · See how the border is black in some cases and white in others (pixel value 194, or 1100 0010)
      * Decomposition useful for image compression
      * Combine some of the bit planes
      * Figure 3-15

Histogram processing

- Discrete function $h(r_k) = n_k$
  - r_k is the kth gray level
  - n_k is the number of pixels in the image at gray level r_k
- Normalized histogram
  - Normalize a histogram by dividing each value by total number of pixels in the image MN
  - Given by: $p(r_k) = n_k/MN$, for $k = 0, 1, \dots, L-1$
  - $p(r_k)$ gives the probability of occurrence of gray level r_k
  - Sum of all components of a normalized histogram is 1
  - In your project, the histogram to be created is given by:

$$p(k) = \frac{n_k}{\max(n_0, n_1, \dots, n_{L-1})}$$

    for $k = 0, 1, \dots, L-1$
- Basis for many spatial processing techniques
  - Simple to calculate in software and economic hardware implementations
  - Can be used for image enhancement and to look at image statistics
  - Figure 3-16
- Histogram equalization
  - Method to increase the global contrast in images by better distributing the intensities in the histogram
    * Areas of lower local contrast gain higher contrast
    * Achieved by spreading out most frequent intensity values
  - Consider a continuous function, with r being the gray levels of the image to be enhanced
  - Let r be normalized to the interval $[0, 1]$ such that $r = 0$ represents black and $r = 1$ represents white
  - Consider the transformation of the form

$$s = T(r) \qquad\qquad 0 \le r \le 1$$

    to produce a level s for every pixel value r in the original image
  - Let $T(r)$ satisfy the following conditions
    1. $T(r)$ is single-valued and monotonically increasing in the interval $0 \le r \le 1$
    2. $0 \le T(r) \le 1$ for $0 \le r \le 1$
    3. Figure 3-17
       * Monotonic vs strictly monotonic
       * Strictly monotonic allows for the existence of inverse
  - Single-valued requirement guarantees the existence of inverse transform

- Monotonicity preserves the increasing order of black to white in output image, preventing artifacts generated by reversals of intensity
- Second condition guarantees that gray levels are preserved in output image
- Inverse transform from s back to r is denoted

$$r = T^{-1}(s) \qquad\qquad 0 \leq s \leq 1$$

  * T^{-1} may fail to be single valued unless we require $T(r)$ to be *strictly* monotonically increasing in the interval [0,1]
- Gray values in an image may be viewed as random variables in the interval $[0, 1]$
- Probability density function
  * Used to describe a random variable
  * Let $p_r(r)$ and $p_s(s)$ denote the probability density functions of random variables r and s, respectively
    · p_r and p_s are different functions
  * If $p_r(r)$ and $T(r)$ are known, and $T(r)$ is continuous and differentiable over the range of values of interest, the PDF of the transformed variable s can be obtained using the simple formula

$$p_s(s) = p_r(r) \left| \frac{dr}{ds} \right|$$

  * A transformation function of importance in image processing is:

$$s = T(r) = \int_0^r p_r(w)dw$$

  where w is a dummy variable of integration
    · The integral gives the cumulative distribution function (CDF) of random variable r
    · Since PDFs are positive, the area under the function cannot decrease as r increases
    · At $r = 1$, the integral evalutes to 1 (the area under a PDF curve is always 1)
  * Finding $p_s(s)$ corresponding to the transformation
    · Leibniz's rule: Derivative of a definite integral with respect to its upper limit is the integrand evaluated at the limit

$$\begin{aligned}\frac{ds}{dr} &= \frac{dT(r)}{dr} \\ &= \frac{d}{dr}\left[\int_0^r p_r(w)dw\right] \\ &= p_r(r)\end{aligned}$$

    · Substituting the result in the equation for the PDF of transformed variable, we get

$$\begin{aligned}p_s(s) &= p_r(r)\left|\frac{dr}{ds}\right| \\ &= p_r(r)\left|\frac{1}{p_r(r)}\right| \\ &= 1 \qquad 0 \leq s \leq 1\end{aligned}$$

    · The form of $p_s(s)$ is a uniform PDF, or performing the intensity transformation yields a random variable s, characterized by a uniform PDF
    · $T(r)$ depends on $p_r(r)$ but the resulting $p_s(s)$ always is uniform
    · Figure 3.18

- For discrete values for an image of size $n = M \times N$ pixels, we have

$$p_r(r_k) = \frac{n_k}{n} \qquad\qquad k = 0, 1, 2, \ldots, L-1$$

- The discrete transformation function is given by

$$\begin{aligned} s_k &= T(r_k) = \sum_{j=0}^{k} p_r(r_j) \\ &= \sum_{j=0}^{k} \frac{n_j}{n} \qquad\qquad k = 0, 1, 2, \ldots, L-1 \end{aligned}$$

- Processed image obtained by mapping each pixel with level r_k into a corresponding pixel with level s_k
  * Histogram equalization or histogram linearization
- Transform spreads the histogram of the input image so that the levels of histogram-equalized image span a fuller range of gray scales
- Discrete form does not guarantee uniform distribution of probability density function
- Advantages
  * More dynamic range of gray level distribution
  * Fully automatic algorithm; no parameters need be specified
  * Simple to compute
- Example: Consider a 3-bit image (8 intensity levels) of size 64×64 pixelswith the intensity distribution as follows:

r_k	n_k	$p_r(r_k) = n_k/n$
0	790	0.19
1	1023	0.25
2	850	0.21
3	656	0.16
4	329	0.08
5	245	0.06
6	122	0.03
7	81	0.02

  * Histogram in Figure 3.19a
  * Values of histogram transformation are obtained by (last column shows the value rounded to nearest integer)

$s_0 = T(r_0)$	$7\sum_{j=0}^{0} p_r(r_j) = 7p_r(r_0)$	1.33	1
$s_1 = T(r_1)$	$7\sum_{j=0}^{1} p_r(r_j) = 7p_r(r_0) + 7p_r(r_1)$	3.08	3
$s_2 = T(r_2)$		4.55	5
$s_3 = T(r_3)$		5.67	6
$s_4 = T(r_4)$		6.23	6
$s_5 = T(r_5)$		6.65	7
$s_6 = T(r_6)$		6.86	7
$s_7 = T(r_7)$		7.00	7

  * Transformation function and equalized histogram in Figure 3.19 b and c
  * Some intensity levels are mapped to a single intensity level, hence fewer intensity levels in the equalized histogram
  * Intensity levels of the equalized histogram span a wider range of intensity scale, leading to contrast enhancement
  * Perfectly flat histograms
    · Not possible because of two reasons
    1. Histograms are approximations to PDFs

2. We cannot create new intensity levels when we perform histogram equalization because of discrete values
   · Spreading the histogram of input image results in a wider span of intensities for equalized image

- Inverse transform from s to r is denoted by

$$r_k = T^{-1}(s_k) \qquad\qquad k = 0, 1, 2, \ldots, L-1$$

  * Satisfies the two conditions outlined earlier only if none of the levels $r_k, k = 0, 1, 2, \ldots, L-1$ are missing from input image (none of the components of the equalized histogram are zero)
  * Plays a central role in histogram matching scheme

- Figure 3.20
  * Significant improvement in low contrast images (first three)
  * Transformation functions (Figure 3.21)
    · Transformation 4 as nearly linear shape, indicating that the inputs were mapped to nearly equal outputs
  * Transformation function changes contrast by redistributing pixel gray levels but does not alter the number (content given by probability) of matched gray levels

• Histogram matching/Histogram specification
  - Histogram equalization gives an image with uniformly distributed gray levels automatically
  - Histogram matching allows us to specify the shape of the histogram of new image
    * Useful when you have to create two images with same contrast and brightness
  - Development of method
    * Consider continuous gray levels r and z (continuous random variables)
      · r is the intensity level of input image
      · z is the intensity level of processed image
    * Let $p_r(r)$ and $p_z(z)$ denote their continuous probability density functions
    * $p_r(r)$ is *estimated* from input image
    * $p_z(z)$ is the *specified* probability density function desired in output image
    * Let s be a random variable with the property

$$s = T(r) = \int_0^r p_r(w)dw$$

      This expression is a continuous version of histogram equalization
    * Define a random variable z with the property

$$G(z) = \int_0^z p_z(t)dt = s$$

      using t as a dummy variable of integration
    * It follows that $G(z) = T(r)$ and z must satisfy the following condition:

$$z = G^{-1}[T(r)] = G^{-1}(s)$$

      · $T(r)$ can be obtained by estimating $p_r(r)$ from the input image using $T(r) = \int_0^r p_r(w)dw$
      · $G(z)$ can be computed from a given $p_z(z)$ using $G(z) = \int_0^z p_z(t)dt$
    * An image with specified probability density function can be obtained from a given image using the procedure:
      1. Compute $p_r(r)$ from the input image and obtain the values of $T(r)$ or s
      2. Compute $G(z)$ using the specified PDF in $G(z) = \int_0^z p_z(t)dt$
      3. Obtain the inverse transformation $z = G^{-1}(s)$ by mapping from s to z

4. Get the output image by first equalizing the input image; pixel values in this image are s values. For each pixel in the equalized image, perform the inverse mapping $G^{-1}(s)$ to get the corresponding pixel in output image.

– Example: Let an image have the intensity PDF given by

$$p_r(r) = \begin{cases} \frac{2r}{(L-1)^2} & 0 \leq r \leq (L-1) \\ 0 & \text{otherwise} \end{cases}$$

Find the transformation function that will produce an image whose intensity PDF is

$$p_z(z) = \begin{cases} \frac{3z^2}{(L-1)^3} & 0 \leq z \leq (L-1) \\ 0 & \text{otherwise} \end{cases}$$

  ∗ Find the histogram equalization transformation for the interval $[0, L-1]$

$$\begin{aligned} s &= T(r) \\ &= (L-1)\int_0^r p_r(w)\,dw \\ &= \frac{2}{L-1}\int_0^r w\,dw \\ &= \frac{r^2}{L-1} \end{aligned}$$

  This gives a uniform PDF due to histogram equalization transform

  ∗ Since we are interested in an image with a specified histogram, we compute

$$\begin{aligned} G(z) &= (L-1)\int_0^z p_z(w)\,dw \\ &= \frac{3}{(L-1)^2}\int_0^z w^2\,dw \\ &= \frac{z^3}{(L-1)^2} \end{aligned}$$

  over the interval $[0, L-1]$

  ∗ Finally, require that $G(z) = s$ leading to $z^3/(L-1)^2 = s$ and

$$z = \left[(L-1)^2 s\right]^{1/3}$$

  ∗ Multiply every histogram-equalized pixel by $(L-1)^2$ and compute its cube root giving an image whose intensities z have the PDF $p_z(z) = 3z^2/(L-1)^2$ in the interval $[0, L-1]$

  ∗ Since $s = r^2/(L-1)$, we can generate z directly from r, the input intensity by

$$\begin{aligned} z &= \left[(L-1)^2 s\right]^{1/3} \\ &= \left[(L-1)^2 \frac{r^2}{(L-1)}\right]^{1/3} \\ &= \left[(L-1)r^2\right]^{1/3} \end{aligned}$$

– Discrete formulation of histogram equalization transformation

  ∗ Mapping from input levels in original image into corresponding s_k based on the histogram of original image is given by

$$s_k = T(r_k)$$

$$\begin{aligned} &= (L-1)\sum_{j=0}^{k} p_r(r_j) \\ &= (L-1)\sum_{j=0}^{k} \frac{n_j}{n} \qquad\qquad k = 0, 1, 2, \ldots, L-1 \\ &= \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j \end{aligned}$$

  * Given a specific value of s_k, the discrete formulation for $G(z)$ is based on computing the transformation function

    $$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_i)$$

    for a value of q, so that

    $$G(z_q) = s_k$$

    where $p_z(z_i)$ is the ith value of the specified histogram
  * Find the desired value z_q by the inverse transformation

    $$z_q = G^{-1}(s_k)$$

  * This gives a value of z for each value of s, performing a mapping from s to z

- Summary of histogram specification process
  1. Compute the histogram $p_r(r)$ of given image and use it to find histogram equalization transform

     $$s_k = \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j$$

     Round the resulting values s_k to the integer range $[0, L-1]$
  2. Compute all values of transformation function G using

     $$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_i)$$

     for $q = 0, 1, 2, \ldots, L-1$ where $p_z(z_i)$ are the values of the specified histogram; Round the values of G to integers in the range $[0, L-1]$ into an LUT
  3. For every value of $s_k \in [0, L-1]$, use the stored value of G from step 2to find the corresponding value of z_q so that $G(z_q)$ is closest to s_k and store these mappings from s to z
  4. Histogram equalize the input image; map every histogram-equalized value andf map every qualized pixel s_k of this image to corresponding value z_q in the histogram-specified image using the mappings from step 3

- Example: Using the 64×64 pixel image from histogram equalization example
  * Histogram in Figure 3.22(a)
  * Specified histogram is in column 2 below

z_q	Specified $p_z(z_q)$	Actual $p_z(z_k)$
$z_0 = 0$	0.00	0.00
$z_1 = 1$	0.00	0.00
$z_2 = 2$	0.00	0.00
$z_3 = 3$	0.15	0.19
$z_4 = 4$	0.20	0.25
$z_5 = 5$	0.30	0.21
$z_6 = 6$	0.20	0.24
$z_7 = 7$	0.15	0.11

  * Obtain the scaled histogram equalized values from last example as

$$s_k = \{1, 3, 5, 6, 6, 7, 7, 7\}$$

  * Compute all the values of the transformation function G

$G(z_0)$	$7\sum_{j=0}^{0} p_z(z_j)$	0.00
$G(z_1)$	$7\sum_{j=0}^{1} p_z(z_j) = 7[p(z_0) + p(z_1)]$	0.00
$G(z_2)$		0.00
$G(z_3)$		1.05
$G(z_4)$		2.45
$G(z_5)$		4.55
$G(z_6)$		5.95
$G(z_7)$		7.00

  * The fractional values are converted to integers in our valid range $[0, 7]$ giving

$$G(z_i) = \{0, 0, 0, 1, 2, 5, 6, 7\}$$

  * The transformation function is sketched in Figure 3.22(c)
    · G is not strictly monotonic and has to be handled
    · Find the smallest value of z_q such that $G(z_q)$ is closest to s_k
    · In our example, $s_0 = 1$ and $G(z_3) = 1$; that is a perfect match, giving us $s_0 \rightarrow G(z_3)$
    · Every pixel with value 1 in histogram equalized image will map to value 3 in the histogram-specified image
    · Final mapping

s_k	$\rightarrow$	z_q
1	$\rightarrow$	3
3	$\rightarrow$	4
5	$\rightarrow$	5
6	$\rightarrow$	6
7	$\rightarrow$	7

    · In the final step, use the above mappings to map every pixel in the histogram equalized image into a corresponding pixel in the histogram-specified image
    · Listed in the third column of table on top as third column ($p_z(z_k)$) (Histogram in Figure 3.22(d))
    · Since $s = 1 \Rightarrow z = 3$ and there are 790 pixels in the histogram-equalized image at intensity 1, $p_z(z_3) = 790/4096 = 0.19$

- Example
  * Figure 3.23 – Mars moon Phobos and its histogram
  * Image dominated by large dark areas
  * Figure 3.24 – Histogram equalized
  * Image histogram quickly rises from 0 to 190, resulting in almost all the pixels concentrated towards the upper end of the dynamic range, giving a light, washed out appearance
  * Fixed by manual specification of the histogram
  * Sample the function into 256 equally spaced discrete values
    · Resulting transformation function $G(z)$ in Figure 3.25
    · Smoother transition of levels in the dark regions of gray scale
- Comments
  * Manual specification of histogram is by trial-and-error
  * Practical use is to adjust the contrast by using the histogram of a different image

• Local enhancement

- Global methods (histogram) modify pixels by transformation functions based on entire image

- Local methods work with neighborhood of each pixel to find a transformation
- A simple approach would be to define small rectangles on the image and process the pixels in selected rectangle using the techniques already seen
  - May have overlapping or non-overlapping rectangles
  - You could also define one rectangle and move its center from pixel to pixel
  - Compute the neighborhood and obtain a histogram equalization or histogram specification transformation
- Example – Blurring to reduce noise content
  - Figure 3-26a – slightly noisy
  - Figure 3.26b – Global histogram equalization
  - Figure 3.26c – Local histogram with a 3×3 neighborhood
  - Local vs global histogram equalization
    - Intensity values of objects too close to the intensity of large squares, and their sizes too small to influence global histogram equalization significantly enough to show the detail

- Use of histogram statistics for image enhancement
  - Let r be the discrete random variable representing intensity values in the range $[0, L-1]$, and $p(r_i)$ the normalized histogram component corresponding to r_i
  - Mean: Average gray scale intensity

$$m = \sum_{i=0}^{L-1} r_i p(r_i)$$

    $p(r_i)$ is the probability of occurrence of gray level r_i
  - The nth moment of r about its mean is defined by

$$\mu_n(r) = \sum_{i=0}^{L-1} (r_i - m)^n p(r_i)$$

    Obviously, $\mu_0 = 1$ and $\mu_1 = 0$
  - Variance (second moment): Average contrast

$$\sigma^2(r) = \mu_2(r) = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i)$$

  - Standard deviation is defined simply as $\sigma = \sqrt{\mu_2}$
  - Mean and variance can also be measured directly from sample values (sample mean and sample variance) as

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)$$

$$\sigma^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - m]^2$$

  - Example: 2-bit ($L = 4$) image of size 5×5

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

    - Image histogram is given by: $(6, 7, 7, 5)$, or after normalization, $p(r_i) = (0.24, 0.28, 0.28, 0.20)$

- The average value of intensities in the image is

$$\begin{aligned} m &= \sum_{i=0}^{3} r_i p(r_i) \\ &= (0)(0.24) + (1)(0.28) + (2)(0.28) + (3)(0.20) \\ &= 1.44 \end{aligned}$$

- Sample mean is computed by

$$\begin{aligned} m &= \frac{1}{25} \sum_{x=0}^{4} \sum_{y=0}^{4} f(x, y) \\ &= 1.44 \end{aligned}$$

– Enhancement can be based on measuring global mean and variance over entire image and adjusting those values to change intensity and contrast

– These quantities can also be used for local enhancement by changing local mean and variance in predefined regions about each pixel
  - Consider pixel at location (x, y)
  - Let S_{xy} be a subimage/neighborhood of specified size, centered at (x, y)
  - Mean value of the neighborhood is

$$m_{S_{xy}} = \sum_{i=0}^{L-1} r_i p_{S_{xy}}(r_i)$$

    · $p_{S_{xy}}$ is the histogram of region S_{xy}
    · Many of the p_i's will be zero, depending on the size of S_{xy}
  - The gray level variance is given by

$$\sigma^2_{S_{xy}} = \sum_{i=0}^{L-1} [r_i - m_{S_{xy}}]^2 p_{S_{xy}}(r_i)$$

  - Local mean and local variance are a measure of average intensity and average contrast, respectively in the neighborhood

– Figure 3-27
  - Scanning electron microscope image of a tungsten filament wrapped around a support
  - The secondary filament on the right side of the image
  - Enhance using local enhancement techniques

– Enhancement of dark areas only
  - Compare local average gray level $m_{S_{xy}}$ in a neighborhood around point (x, y) to the global mean m_G
  - A pixel at point (x, y) is considered a candidate for enhancement if $m_{S_{xy}} \leq k_0 m_G$, $0 < k_0 < 1.0$
  - Low contrast areas are also candidates for enhancement, if $\sigma_{S_{xy}} \leq k_2 \sigma_G$ where $k_2 > 0$ and σ_G is global standard deviation
    · $k_2 > 1.0$ to enhance light areas
    · $k_2 < 1.0$ to enhance dark areas
  - We need a lower bound on contrast so as not to enhance uniform intensity areas with standard deviation 0
    · This can be achieved by requiring $k_1 \sigma_G \leq \sigma_{S_{xy}}$, with $k_1 < k_2$
  - A pixel meeting all the requirements can be enhanced by multiplying it by a specific constant E to increase/decrease its gray level
  - Let $f(x, y)$ and $g(x, y)$ be the input and enhanced values of the pixel at location (x, y)

$$g(x, y) = \begin{cases} E \cdot f(x, y) & \text{if } m_{S_{xy}} \leq k_0 m_G \text{ \&\& } k_1 \sigma_G \leq \sigma_{S_{xy}} \leq k_2 \sigma_G \\ f(x, y) & \text{otherwise} \end{cases}$$

        * Values chosen: $E = 4.0$, $k_0 = 0.4$, $k_1 = 0.02$, $k_2 = 0.4$
        * Can a multiple of 4 for pixels cause overflow?
        * A small area under S_{xy} preserves detail and reduces computational burden

Basics of spatial filtering

- Filtering
    - Roots in the use of Fourier transform for signal processing in frequency-domain
    - Accepting or rejecting certain frequency components
    - Low-pass filter, high-pass filter, band-pass filter
    - Filtering effects of frequency-based filters can be achieved by using spatial filters
    - Filter, mask, kernel, or window
        * Values in filter subimage are referred to as coefficients
    - Spatial filters have a 1:1 correspondence to frequency filters but can also do nonlinear filtering in addition (not possible in frequency based filters)
- Spatial filtering
    - Spatial filter consists of
        1. A neighborhood, typically a small square or rectangle
        2. A predefined operation performed on the image pixels in the neighborhood
    - Creates a new pixel at the coordinates of the neighborhood center as the result of filtering operation
        * Generally, the result is written into a new image as the pixels in the neighborhood may still be needed for the filtering of other pixels
    - Convolution – Moving the filter mask over pixels
    - Linear filtering
        * Product of filter coefficients with the corresponding pixels
        * $R = \sum w_{ij} f(x+i, y+j)$
        * Fig 3-28
        * Response $g(x, y)$ of the filter at point (x, y)
            · Given by the sum of products of filter coefficients and corresponding image pixels
            · For a 3×3 filter, we have

$$g(x,y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} w(i,j) f(x+i, y+j)$$

            · The center coefficient of the filter $(0, 0)$ aligns with the pixel at location (x, y)
        * Odd dimensions of the filter
            · For a mask of size $m \times n$, assume that $m = 2a + 1$ and $n = 2b + 1$, where $a > 0$ and $b > 0$
            · For a filter of size $m \times n$

$$g(x,y) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} w(i,j) f(x+i, y+j)$$

            varying x and y over the entire image
- Spatial correlation and convolution
    - Correlation
        * Process of moving the filter mask over the image and computing the sum of products at each location

- Convolution
  - Same as correlation except that the filter is first rotated by $180°$
- Figure 3.29
  - 1D function f and filter w
- Correlation is a function of displacement of the filter
  - First value of correlation corresponds to zero displacement of the filter
  - Second value corresponds to one unit displacement
- Correlating a filter w with a function that contains all 0s and a single 1 (unit impulse function or discrete unit impulse) yields a copy of w but rotated by $180°$
- Convolving a function with a unit impulse yields a copy of the function at the location of the impulse
  - Prerotating the function by $180°$ and performing the sliding sum of products operations yields the desired result
  - If the filter mask is symmetric, correlation and convolution yield the same result
- Issue of border crossing by the kernel
  - Limiting the image to the kernel overlap
  - Limiting the kernel to the image overlap
  - Padding image by replication or reflection
- Extending the concept to images (2D)
  - Figure 3.30
  - If f contains a region identical to w, value of correlation is maximum when w is centered on that region of f
    - This property can be used to find matches between images
  - Correlation is given by

$$w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

  - Convolution is given by

$$w(x,y) \bigstar f(x,y) \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t)$$

- Vector representation of linear filtering
  - Characteristic response R of a mask for correlation is given by

$$\begin{aligned} R &= w_1 z_1 + w_2 z_2 + \cdots + w_{mn} z_{mn} \\ &= \sum_{k=1}^{mn} w_k z_k \\ &= \mathbf{w}^T \mathbf{z} \end{aligned}$$

    where ws are coefficients of an $m \times n$ filter and zs are corresponding image intensities
  - The same equation can be used for convolution by rotating the mask by $180°$
  - A general 3×3 mask is labeled as

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

  - Again, R is given by $\mathbf{w}^T \mathbf{z}$ as above, where $\mathbf{w}$ and $\mathbf{z}$ are 9-dimensional vectors formed from the coefficients of the mask and image intensities, respectively
- Generating spatial filter masks

- Generating an $m \times n$ linear spatial filter mask requires the specification of mn mask coefficients
- The coefficients are selected based on the purpose of the filter
  * All a filter does is to compute sum of products
- Replacing the pixels by the average intensity of a 3×3 neighborhood centered at the pixels
  * Average value at location (x, y) is the sum of nine intensity values in the 3×3 neighborhood centered on (x, y) divided by 9
  * With $z_i,\ i = 1, 2, \ldots, 9$, the average is
$$R = \frac{1}{9} \sum_{i=1}^{9} z_i$$
  * This is the same as
$$R = \sum_{i=1}^{9} w_i z_i$$
    with $w_i = 1/9$
- Generating a spatial mask based on a continuous function of two variables
  * Gaussian function of two variables has the basic form
$$h(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$
    where σ is the standard deviation and the coordinates x and y are integers
  * The mask for a 3×3 neighborhood can be generated as $w_1 = h(-1, -1)$, $w_2 = h(-1, 0)$, …, $w_9 = h(1, 1)$
  * 2D Gaussian has a bell shape with standard deviation controlling the tightness of the bell
- Generating a nonlinear filter requires the size of the neighborhood and the operations to be performed on the image pixels contained in the neighborhood
  * Max operation

Smoothing spatial filters

- Used for blurring and noise reduction
  - Noise reduction can be achieved by blurring with a linear filter or by a nonlinear filter
- Smoothing linear filters
  - Averaging filter or lowpass filter
    * Average of pixels in a neighborhood
    * Reduction in sharp transitions in gray levels
      · Random noise is characterized by sharp transitions in intensity levels
      · Averaging reduces the sharp transitions
      · A side effect is the blurring of edges which are also characterized by sharp transitions in intensity levels
    * The filter can also be used to reduce false contouring
  - Standard average of pixels in a 3×3 neighborhood
$$\frac{1}{9} \times \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$
    * Same as $g(x, y) = \frac{1}{9} \sum_{c=-1}^{1} \sum_{r=-1}^{1} f(x + c, y + r)$
    * Also called box filter (all coefficients of filter are equal)
    * The entire image can be divided by 9 at the end of filtering

- Weighted average of pixels in a 3×3 neighborhood

$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

  * Pixels are multiplied by different coefficients, giving more importance to some pixels at the expense of others
  * Reduces blurring in the smoothing process
  * Optimization by shifting right by four bits instead of dividing by 16
  * General implementation to filter an image with a weighted averaging filter of size $m \times n$ is given by

$$g(x,y) = \frac{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)}{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)}$$

- Example: Figure 3.33
  * Results with square averaging filters of size $m = 3, 5, 9, 15, 35$ pixels
  * Pronounced black border with larger filters
    · Result of padding the border region with 0s
- Example: Figure 3.34
  * Blur an image to detect objects of interest
  * Image from Hubble telescope, and application of 15×15 averaging mask

• Order-statistics filters
  - Nonlinear spatial filters
  - Response based on ordering or ranking the pixels under the kernel
  - Median filter
    * Good for filtering impulse noise (or salt-and-pepper noise), with less blurring
    * Force points with distinct gray values to be more like their neighbors
    * Figure 3-35
    * Isolated clusters with area less than $n^2/2$ are eliminated by $n \times n$ median filter
  - Max filter
  - Min filter

Sharpening spatial filters

• Highlight fine detail in an image or enhance detail that has been blurred
  - Averaging is same as integration
  - Achieve sharpening by differentiation
    * Strength of response of a derivative operator is proportional to the degree of discontinuity of image at the point where the operator is applied
    * Differentiation enhances edges and discontinuities (including noise) and deemphasizes slow varying gray scale values

• Foundation
  - One dimensional derivatives to simplify the discussion
  - Behavior of derivatives at the beginning (step) and end (ramp) of discontinuities and along gray-level ramps
    * Discontinuities used to model noise points, lines, and edges

- Derivative of a digital function
  * Defined in terms of differences
  * Definition for a first derivative must be
    · Zero in flat segments (areas of constant gray level)
    · Nonzero at the onset of a gray-level step or ramp
    · Nonzero along ramps
  * Definition of a second derivative must be
    · Zero in flat areas
    · Nonzero at the onset and end of a gray level step or ramp
    · Zero along ramps of constant slope
  * Keep in mind the finite nature of digital functions
    · Finite values
    · Finite maximum change
    · Shortest distance for maximum change is adjacent pixels
  * Basic definition of first-order derivative of a one-dimensional function given by the difference

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

  * Second-order derivative is defined as the difference

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

    · The two definitions satisfy the conditions laid out above
    · Fig 3-36
    · Scan line contains an intensity ramp, three sections of constant intensity, and an intensity step
    · First-order derivatives are indicated by dots while the second-order derivatives are indicated by squares
    · Circles indicate the onset or end of intensity transitions
    · First-order derivative produces thick edges while second-order derivative produces fine edges
    · For isolated point, second-order derivative gives a much stronger response than first-order derivative
    · Second-order derivative can enhance fine detail much more than the first-order derivative
  * Conclusions from above
    · First-order derivatives produce thicker edges in an image
    · Second-order derivatives have stronger response to fine detail
    · First-order derivatives have stronger response to gray-level step
    · Second-order derivatives produce a double response at step changes in gray level
  * In most applications, second derivative is better suited than first derivative because of enhancement of fine detail
  * Zero-crossing property
    · Sign of the second derivative changes at the onset and end of a step or ramp
    · In a step transition, a line joining the two values crosses the horizontal axis midway between the two extremes
    · Useful to detect edges

- Use of second derivative for enhancement – the Laplacian
  - Define a discrete formulation of second order derivative and construct a filter mask based on that formulation
  - Preference for isotropic filters
    * Response is independent of direction of discontinuities in image
    * Rotationally invariant filters

- Rotating the image and then applying the filter gives the same result as applying the filter first and then rotating the image

- Development of method
  - Laplacian – simplest derivative operator
  - Defined as $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$
    - Since derivative of any order is a linear operation, Laplacian is a linear operator
  - Discrete Laplacian operator
    - Must satisfy the properties of second derivative
    - Partial-order derivative in x direction

$$\frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

    - Partial-order derivative in y direction

$$\frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

    - Discrete Laplacian in two dimensions is given by taking the sum of partials

$$\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

    - The mask is given by

0	1	0
1	-4	1
0	1	0

    - The mask gives isotropic result in increments of $90°$
  - Diagonal directions can be added to Laplacian by adding two more terms to above equation resulting in the mask as

1	1	1
1	-8	1
1	1	1

    - The mask gives isotropic result in increments of $45°$
  - Properties of Laplacian – As derivative operator
    - Highlights gray level discontinuities in an image
    - Deemphasizes slowly varying gray level changes
    - Produces images with grayish discontinuities superimposed on a dark featureless background
    - Background features can be recovered by adding original and Laplacian images, if the center is a positive coefficient, or subtracting the Laplacian image from original if center is negative coefficient

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if center is negative} \\ f(x,y) + \nabla^2 f(x,y) & \text{if center is positive} \end{cases}$$

    - Figure 3.38
  - Scaling Laplacian
    - Add to the image its minimum value to bring the new minimum to zero and then, scale the result to the full $[0, L-1]$ range
- Simplifications
  - Computation of Laplacian and subtraction can be done in a single pass of a single mask

$$\begin{aligned} g(x,y) &= f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] + 4f(x,y) \\ &= 5f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] \end{aligned}$$

        * The mask is defined as

0	-1	0
-1	5	-1
0	-1	0

- Unsharp masking and high-boost filtering
  - Unsharp masking
    * Used in publishing industry
    * Subtract a blurred version of the picture from itself
    * Obtain the mask as:
$$g_{\text{mask}}(x,y) = f(x,y) - \bar{f}(x,y)$$
    * Add a weighted portion of the mask back to original image:
$$g(x,y) = f(x,y) + k \times g_{\text{mask}}(x,y)$$
      $k \geq 0$ is a weight for generality
      · $k = 1$ leads to unsharp masking as defined above
      · $k > 1$ leads to high-boost filtering
      · $k < 1$ de-emphasizes the contribution of unsharp mask
    * Figure 3-39
      · Unsharp mask is similar to the second-order derivative
      · The points of change of slope in intensity get emphasized
      · Possible for the final result to have negative intensities, especially if input image has zero values or k is chosen as large enough
      · Negative intensities can lead to objectionable results (dark halo around edges)
    * Figure 3-40
      · Blurred with a 5×5 Gaussian smoothing filter with $\sigma = 3$
      · Figure 3.40e shows result with $k = 4.5$, the largest possible value that will keep all intensities positive in the output
  - High-boost filtering
    * Generalization of unsharp masking
    * Defined as:
$$g_{hb}(x,y) = Af(x,y) - \bar{f}(x,y)$$
      where $A \geq 1$ and $\bar{f}$ is a blurred version of f
    * It can also be written as:
$$\begin{aligned} g_{hb}(x,y) &= (A-1)f(x,y) + f(x,y) - \bar{f}(x,y) \\ &= (A-1)f(x,y) + g_{\text{mask}}(x,y) \end{aligned}$$
  - Using Laplacian
$$g(x,y) = \begin{cases} Af(x,y) - \nabla^2 f(x,y) & \text{if center is negative} \\ Af(x,y) + \nabla^2 f(x,y) & \text{if center is positive} \end{cases}$$
  - High-boost filtering is standard Laplacian sharpening when $A = 1$
  - If A is large, high-boost image is approximately equal to the original image multiplied by a constant
- Use of first derivative for enhancement – the gradient
  - First derivative implemented using the magnitude of gradient

- Gradient of f at (x, y) is defined as the column vector

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

  * Important geometric property of the vector: it points in the direction of the greatest rate of change of f at (x, y)

- Magnitude of vector ∇f, denoted by $M(x, y)$, is given by

$$\begin{aligned} M(x,y) &= \text{mag}(\nabla f) \\ &= \sqrt{g_x^2 + g_y^2} \\ &= \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2} \end{aligned}$$

  * $M(x, y)$ is an image of the same size as the original
  * Commonly referred to as the *gradient image*

- Partial derivatives are not rotationally invariant but magnitude of gradient is
- The gradient operator is computationally expensive, and is therefore, approximated as

$$\nabla f \approx |G_x| + |G_y|$$

  * Simpler to compute and preserves relative changes in gray levels
  * Does not preserve isotropic feature property

- Digital approximations to compute appropriate filter masks
  * Use the following notation to denote intensities in a 3×3 region

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

  * Simplest approximations

$$\begin{aligned} g_x &= f(x, y+1) - f(x, y) \\ g_y &= f(x+1, y) - f(x, y) \end{aligned}$$

  * Robert's definition, based on cross differences

$$\begin{aligned} g_x &= f(x+1, y+1) - f(x, y) \\ g_y &= f(x, y+1) - f(x+1, y) \end{aligned}$$

    · Compute the gradient as

$$M(x, y) = \sqrt{[f(x+1, y+1) - f(x, y)]^2 + [f(x, y+1) - f(x+1, y)]^2}$$

    · Using absolute values, the gradient is given by

$$M(x, y) \approx |f(x+1, y+1) - f(x, y)| + |f(x, y+1) - f(x+1, y)|$$

  * Implemented with the mask (Roberts cross-gradient operator)

-1	0
0	1

0	-1
1	0

  * Even-sized masks are different to implement due to lack of center of symmetry

* An approximation using absolute values at point $f(x,y)$ using a 3×3 mask is given by Sobel operators

$$\begin{aligned} M(x,y) \approx\ & |(f(x-1,y+1)+2f(x,y+1)+f(x+1,y+1)) - \\ & (f(x-1,y-1)+2f(x,y-1)+f(x+1,y-1))| + \\ & |(f(x+1,y-1)+2f(x+1,y)+f(x+1,y+1)) - \\ & (f(x-1,y-1)+2f(x-1,y)+f(x-1,y+1))| \end{aligned}$$

or, pictorially as

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

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

  · Difference in first and third row in the left mask gives partial derivative in the vertical direction
  · Difference in first and third column in the right mask gives partial derivative in the horizontal direction
* Mask gives gradient in x and y directions, and coefficients sum to zero indicating no change in constant gray level areas
* Used for edge detection (Fig 3-42)

Combining spatial enhancement methods

- Use a combination of the filters to enhance the image depending on the application
- Figure 3.43
- Reading assignment