Basic image processing operations

COS 791

Chapter 3

Overview

- This chapter concerns basic operations that can improve the appearance and quality of images
  - Image brightness histogram
  - Point operations
    - Histogram processing
    - Thresholding
  - Group operations

Image formation

- A computer image is a matrix (2D array) of pixels
- The value of each pixel is proportional to the brightness of the corresponding point in the image
- We shall describe an image as an N x N m-pixels where N is the number of points and m controls the number of brightness levels

Histograms

- Intensity histogram shows how individual brightness levels are occupied in an image
- Image contrast is measured by the range of brightness
- A histogram plots the number of pixels with a particular brightness level

Histograms

- For 8-bit pixels, the brightness ranges from 0 (black) to 255 (white)
- The shape and location of the values in a histogram can tell us a lot about the features of an image
- Smaller ranges shows a low contrast in the image while a larger range of intensities indicates a higher contrast

Histograms

- Histograms can show how much of the range of intensity values have been used in the image and if the range can be increased
- A histogram can also reveal if there is much noise in the histogram, if the ideal histogram is known

Histograms

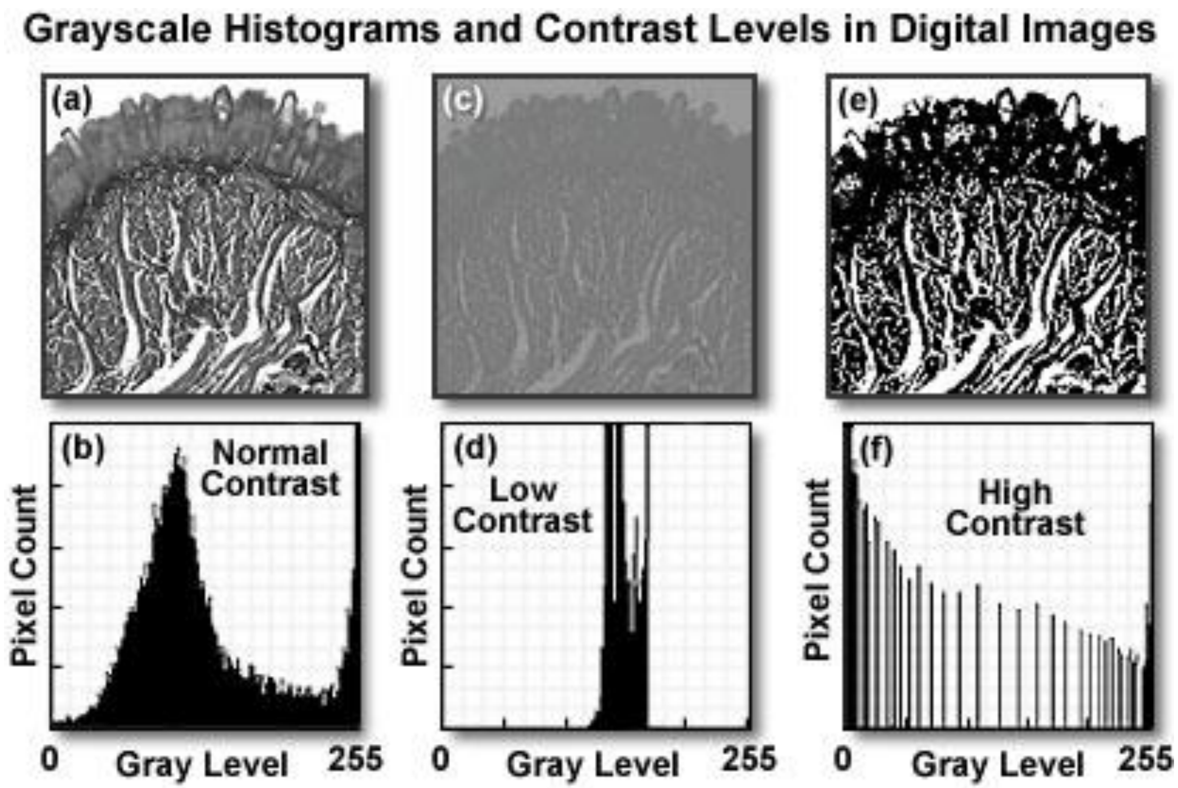
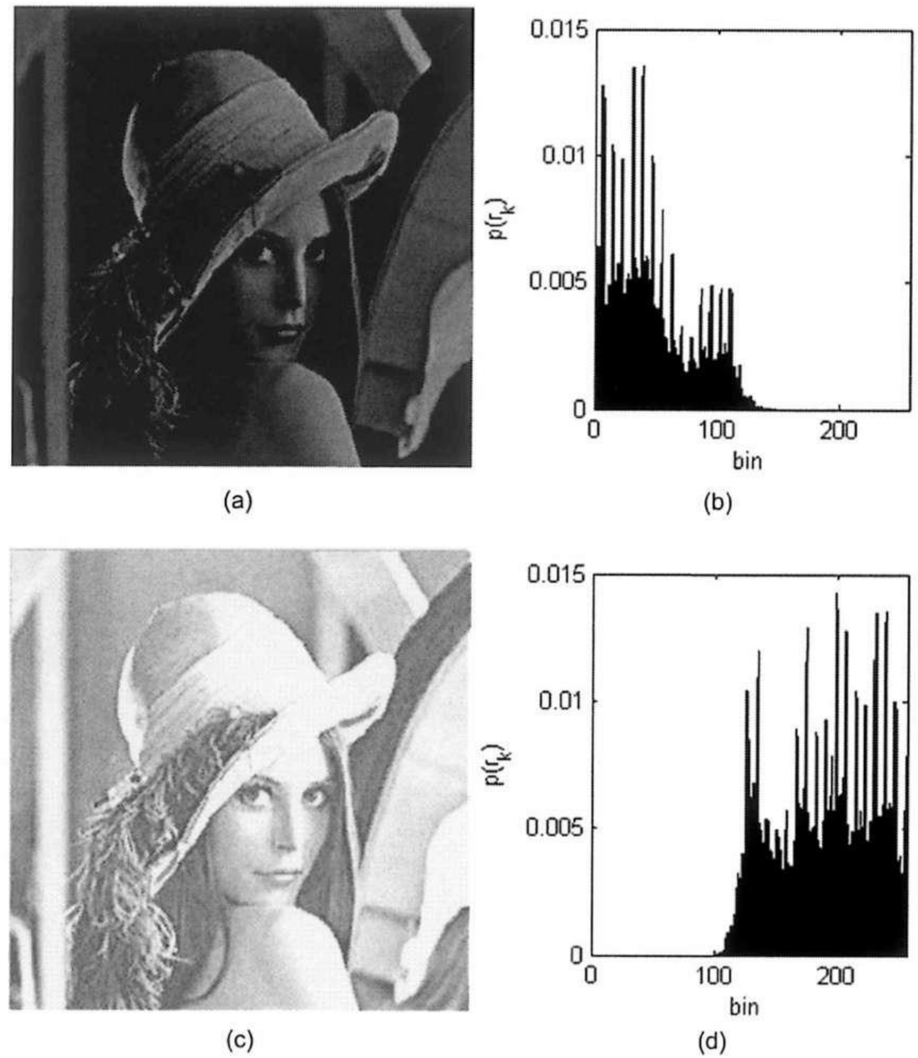


Figure 7

Histograms



Images from http://what-when-how.com/embedded-image-processing-on-the-tms320c6000-dsp/spatial-transform-functions-and-the-image-histogram-image-processing/

Point operators

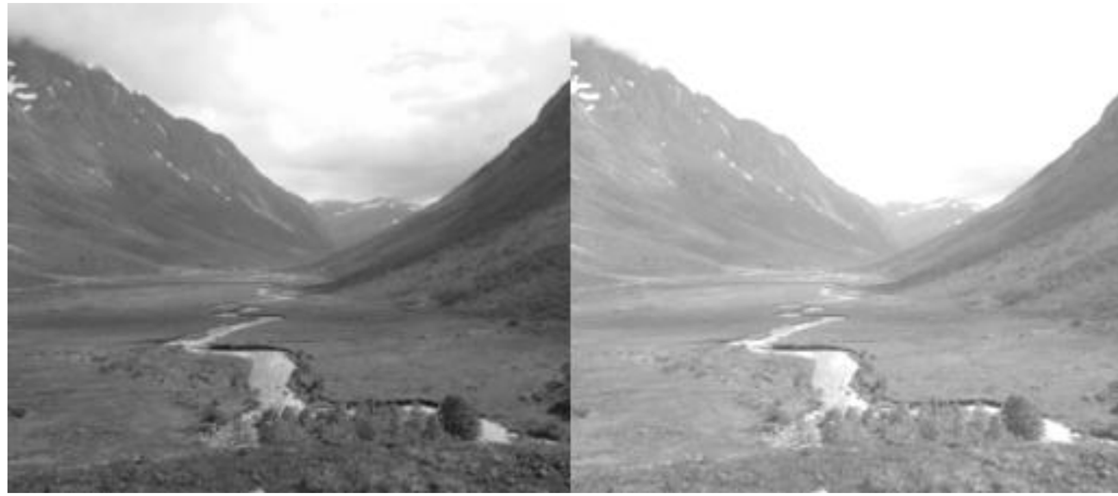
- Most basic image processing operations
- Each pixel value is replaced with a new value obtained from the old one
- Images often do not use all available gray levels
- We can process the images to use more of the intensity levels and thus produce a better appearance

Point operators - Brightness

- When changing the brightness of an image, a constant is added or subtracted from all sample values.
- This effectively shifts the contents of the histogram left (subtraction) or right (addition)

Point operators - Brightness

Increase in brightness (addition):



Decrease in brightness (subtraction):

Images from http://pippin.gimp.org/image_processing/chap_point.html

Point operators - Contrast

- To increase the brightness of an image by stretching the contrast, each pixel can be multiplied by a constant value, say 2, to double the range
- Visualized in the histogram it is equivalent to expanding or compressing the histogram

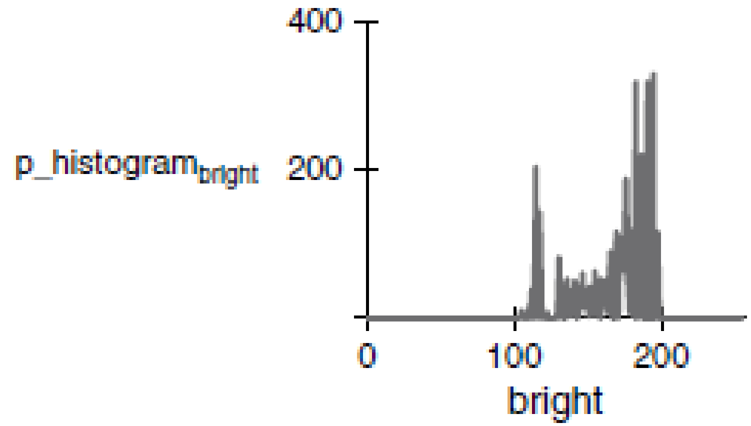
Point operators - Contrast

- If the overall brightness is controlled by level l and the range is controlled by a gain, k, the brightness of the point in a new image N can be related to the brightness of the old image O by

$$N_{x,y} = k \times O_{x,y} + l$$

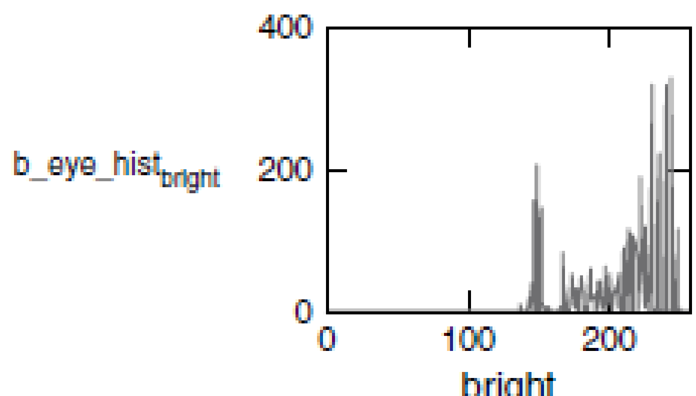
Where $k = 1.2$ and $l = 10$

(a) Image of eye



(b) Histogram of eye image

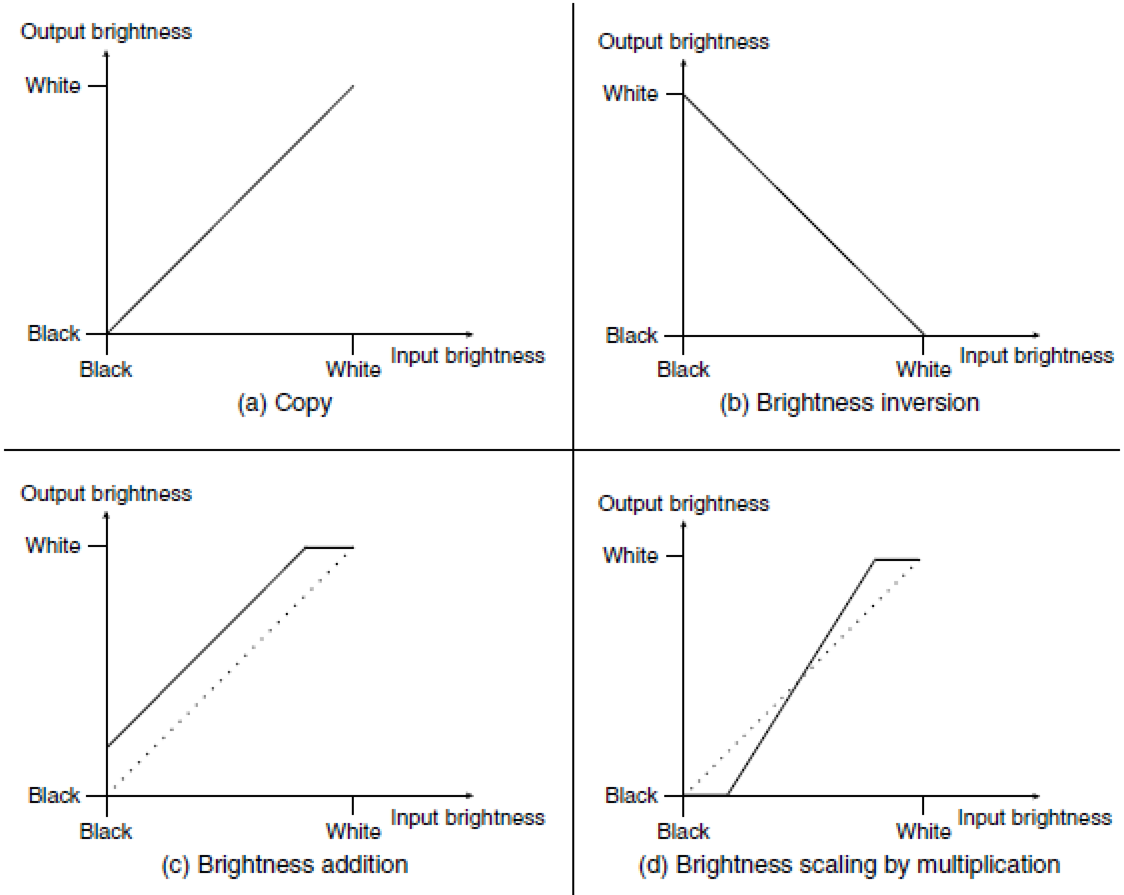
(a) Image of brighter eye



(b) Histogram of brighter eye

Point operators

- The stretching can be displayed as a mapping between the input and output ranges, according to a specified relationship.



Point operators

- In these mappings, if the mapping produces values that are outside the range of expected values (a negative value for example), then a clipping process can be used to set the output values to the chosen level
- For example, if an output point of more than 255 (white) is produced then that output point can be set to white

Histogram normalization

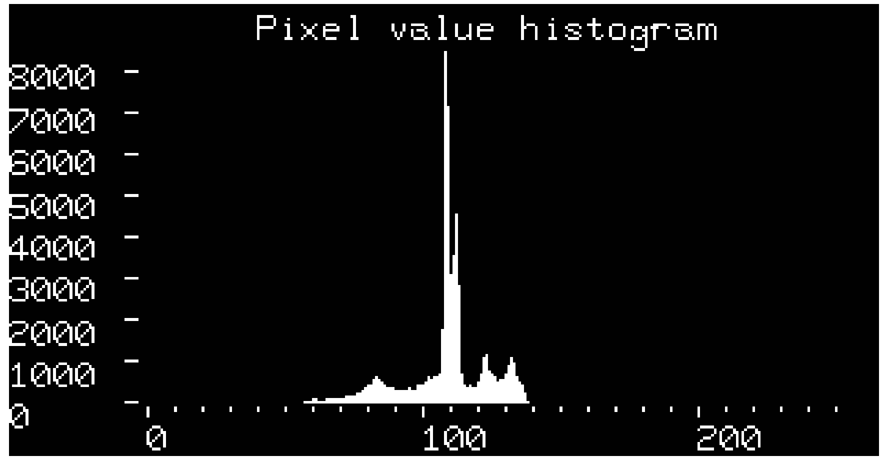
- Also known as contrast stretching
- Attempts to improve the contrast in an image by stretching the range of intensity values it contains to cover all 256 possible levels

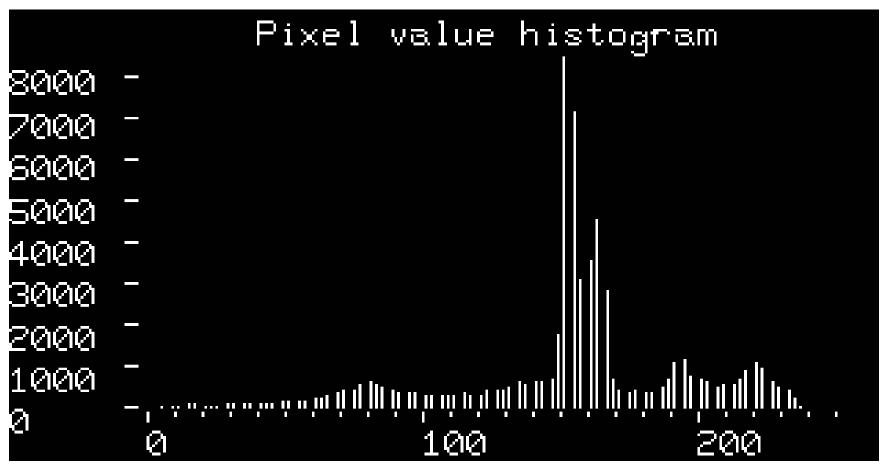
Histogram normalization

- Have to specify the upper and lower pixel values over which the image is to be normalized – normally 0 and 255 for 8-bit images
  - Called N_{min} and N_{max} respectively
- The image is scanned to find the lowest and highest pixel values currently present in the image
  - These are O_{min} and O_{max}
- Each new pixel is then calculated as:

$$N_{x,y} = \left(O_{x,y} - O_{min}\right)\left(\frac{N_{max} - N_{min}}{O_{max} - O_{min}}\right) + N_{min}$$

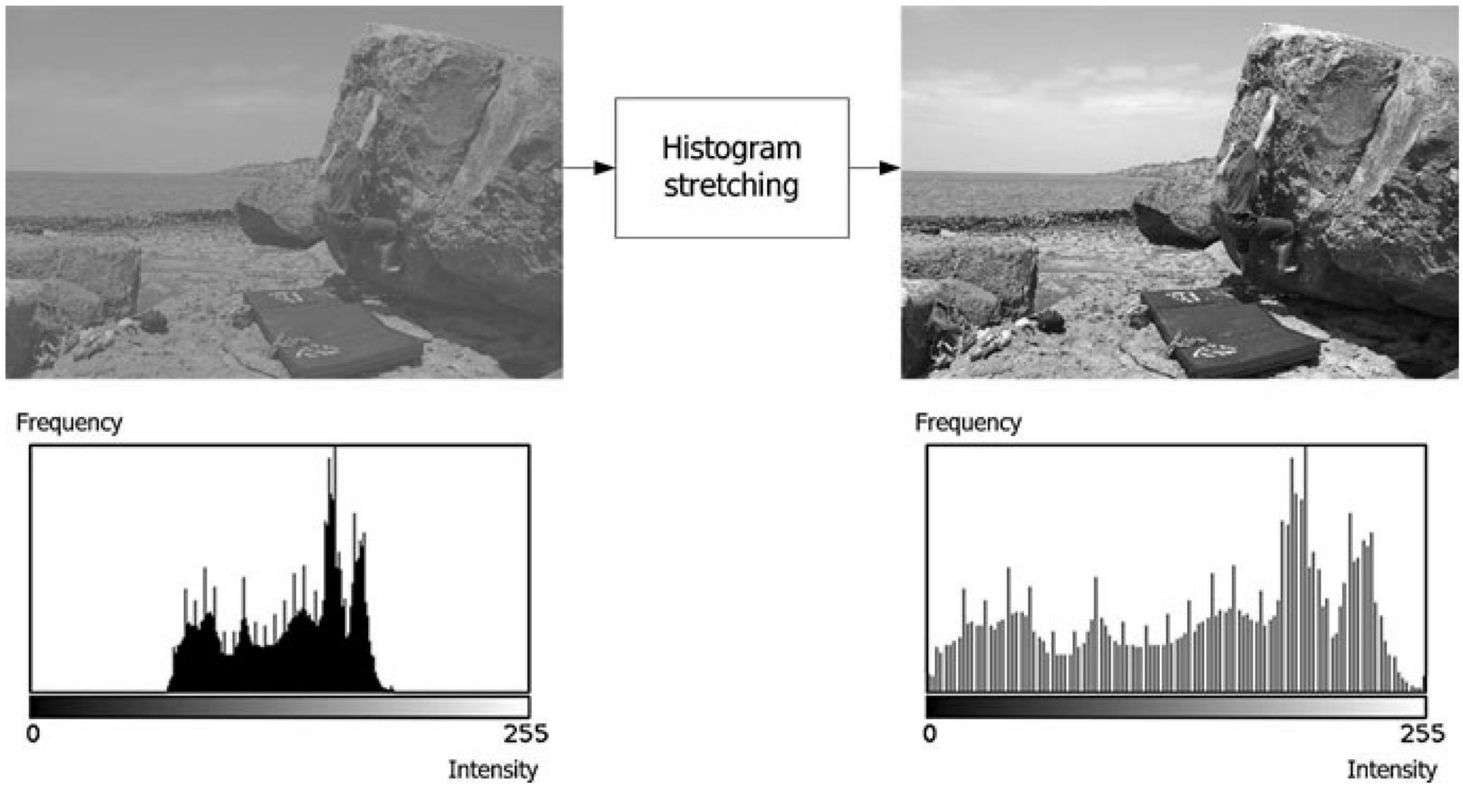
Histogram normalization





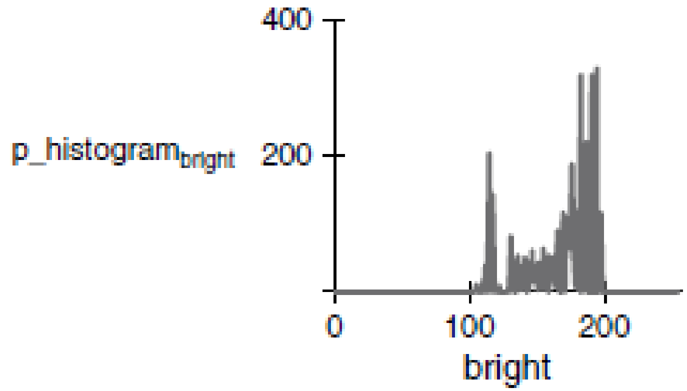
Images from http://homepages.inf.ed.ac.uk/rbf/HIPR2/stretch.htm

Histogram normalization

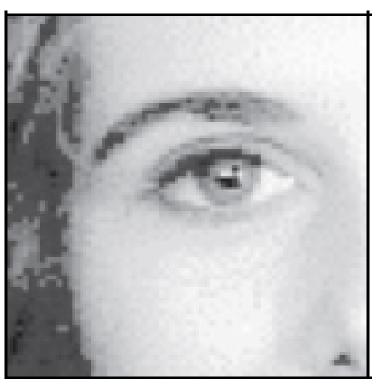


Histogram normalization

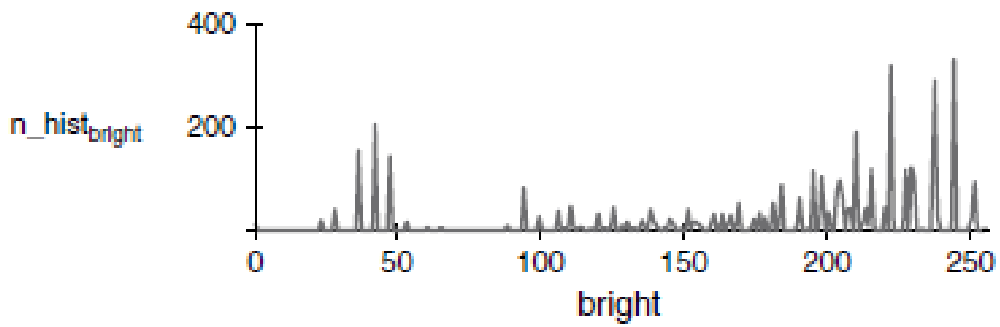
(a) Image of eye



(b) Histogram of eye image



(a) Intensity normalized eye



(b) Histogram of intensity normalized eye

Histogram equalization

- Histogram equalization is a nonlinear process aimed to highlight image brightness similar to human visual analysis
- Histogram equalization aims to produce a flatter histogram where all levels are equiprobable

Histogram equalization

- For a range of M levels, the number of points per level is denoted as $O(l)$ and $N(l)$
- For square images, there are K^2 points in the input and output images, so the sum of points per level should be equal:

$$\sum_{i=0}^{M} O(l) = \sum_{i=0}^{M} N(l)$$

Histogram equalization

- Since we are aiming for a uniformly flat histogram, this should be the same for an arbitrarily chosen level p
- The cumulative histogram up to level p should be transformed to cover up to level q in the new histogram:

$$\sum_{i=0}^{p} O(l) = \sum_{i=0}^{q} N(l)$$

Histogram equalization

- Since the output histogram is uniformly flat, the number of points per level in the output picture is the ratio of the number of points in the image to the range of levels:

$$N(l) = \frac{K^2}{N_{max} - N_{min}}$$

- So the cumulative histogram of the new image is:

$$\sum_{i=0}^{q} N(l) = q \times \frac{K^2}{N_{max} - N_{min}}$$

Histogram equalization

- By a previous equation, this is equal to the cumulative histogram of the input image

$$q \times \frac{K^2}{N_{max} - N_{min}} = \sum_{i=0}^{p} O(l)$$

- This gives us a mapping for the output pixels at level q from the input pixels at level p as:

$$q = \frac{N_{max} - N_{min}}{K^2} \times \sum_{i=0}^{p} O(l)$$

Histogram equalization

- The equalizing function E of the level q for the image O is thus:

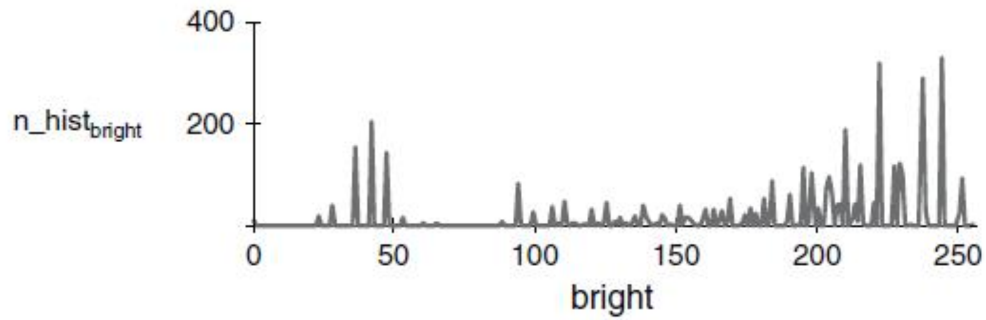
$$E(q,O) = \frac{N_{max} - N_{min}}{K^2} \times \sum_{i=0}^{p} O(l)$$

- And the output image is:

$$N_{x,y} = E(O_{x,y}, O)$$

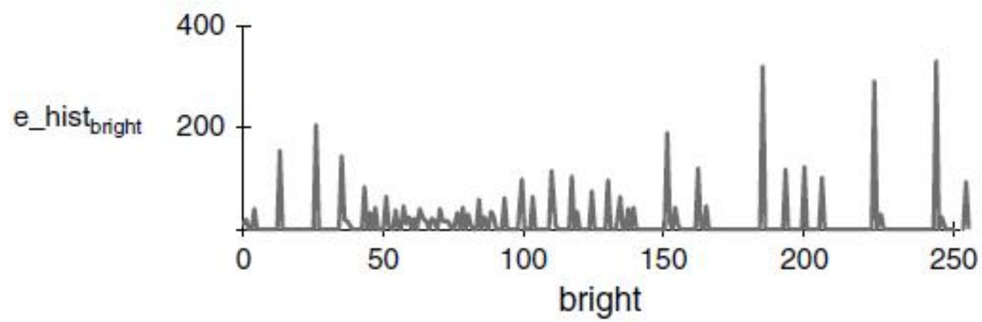
Histogram normalization vs equalization

(a) Intensity normalized eye



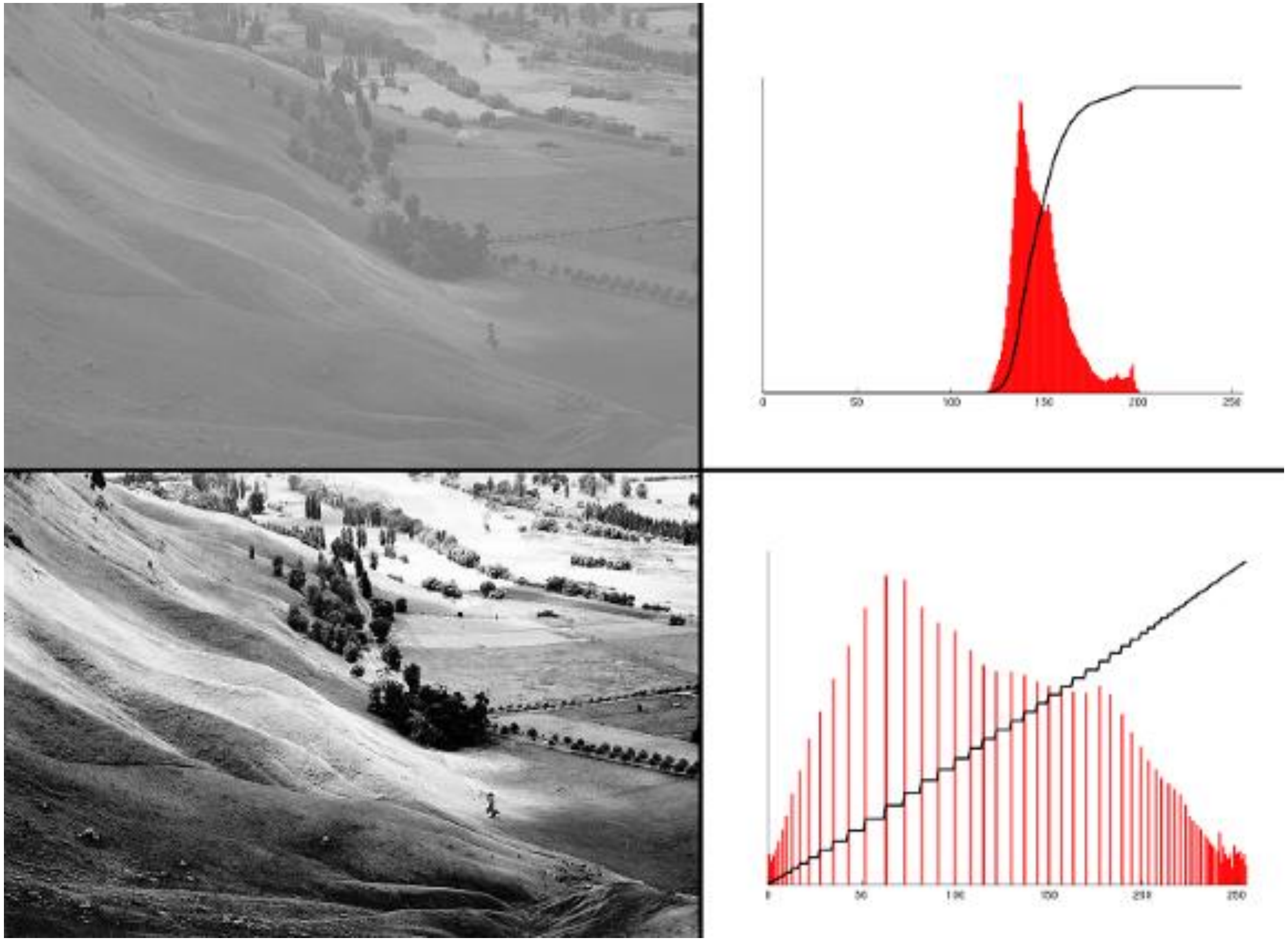
(b) Histogram of intensity normalized eye

(c) Histogram of equalized eye



(d) Histogram of histogram equalized eye

Histogram equalization



Images from http://siret.ms.mff.cuni.cz/facereco/method/

Histogram equalization

- Unfortunately noise in the image acquisition process can affect the shape of the original histogram and also the equalized version – you could end up enhancing the noise
- Histogram equalization is a nonlinear process and thus irreversible – histogram normalization on the other hand is linear and fully reversible

Thresholding

- Point operator that changes an image from gray level to binary
- Effectively separates the background from the foreground
- There are two main forms:
  - Uniform thresholding
  - Adaptive thresholding

Uniform thresholding

- Pixels above a specified value (the intensity threshold) are set the white, those below the threshold are set to black – or vice versa

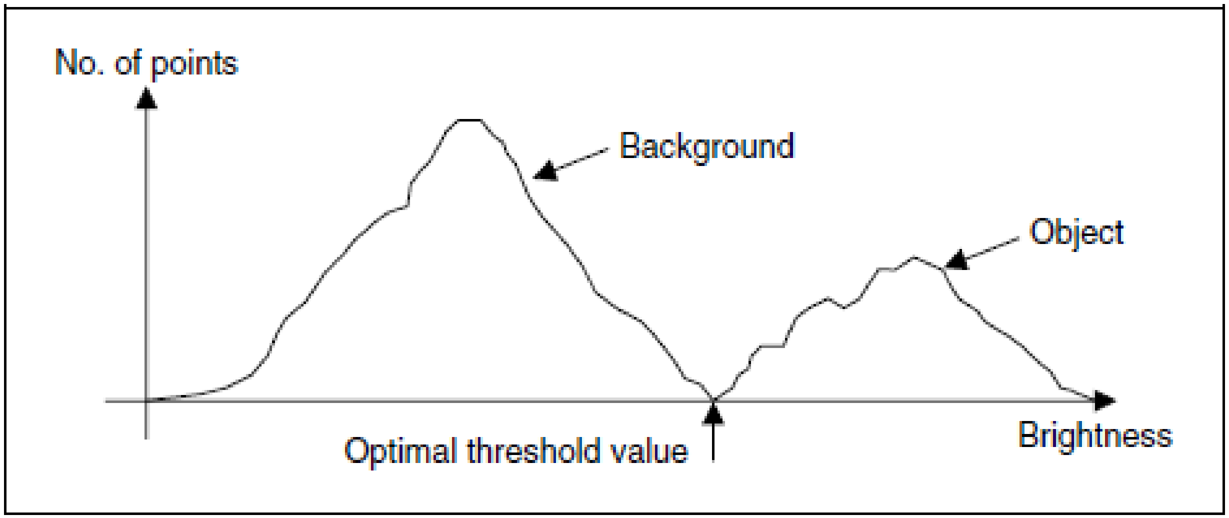
Uniform thresholding

- Not all images can be neatly segmented into foreground and background using simple thresholding.
- Whether or not an image can be correctly segmented this way can be determined by looking at an intensity histogram of the image.
- These more advanced techniques are called optimal thresholding

Optimal thresholding

- Images that are well suited to thresholding is when the intensity of pixels within foreground objects is distinctly different from the intensity of pixels within the background.
- In this case, we expect to see a distinct peak in the histogram that separates the foreground from the background.
- If such a peak does not exist, then it is unlikely that simple thresholding will produce a good segmentation.

Optimal thresholding



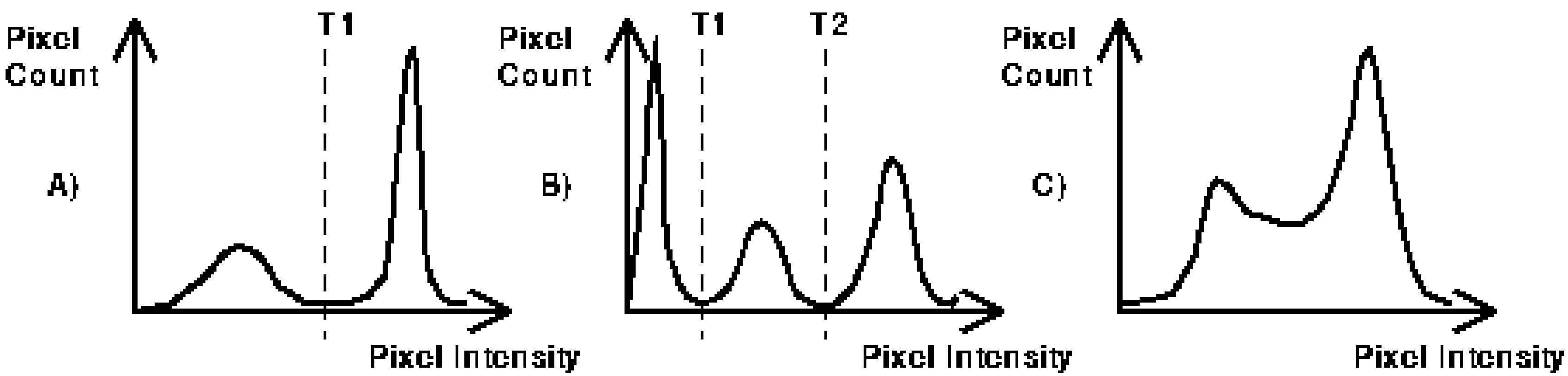


Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/threshld.htm

Optimal thresholding

- Otsu's method is one of the most popular techniques for optimal thresholding
- Otsu's technique maximizes the likelihood that the chosen threshold will split the image between an object and its background
- Otsu's technique is automatic as opposed to manual in some other thresholding techniques

Thresholding

Original image and histogram:

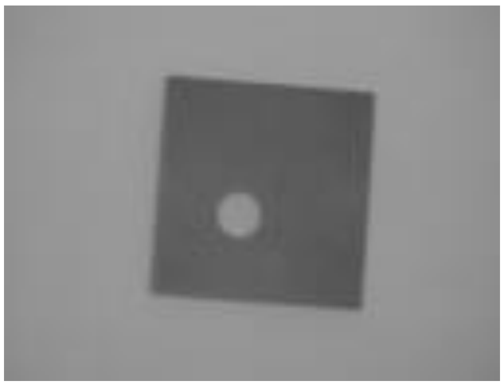
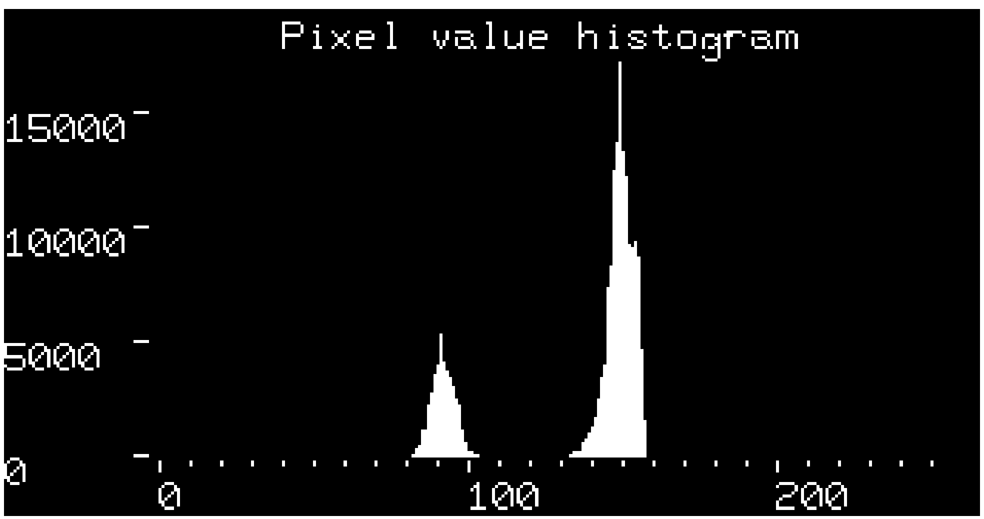


Image after thresholding:



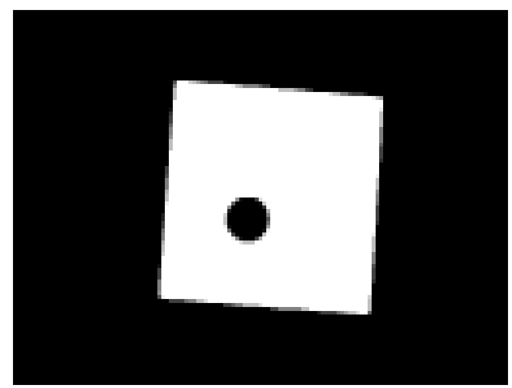
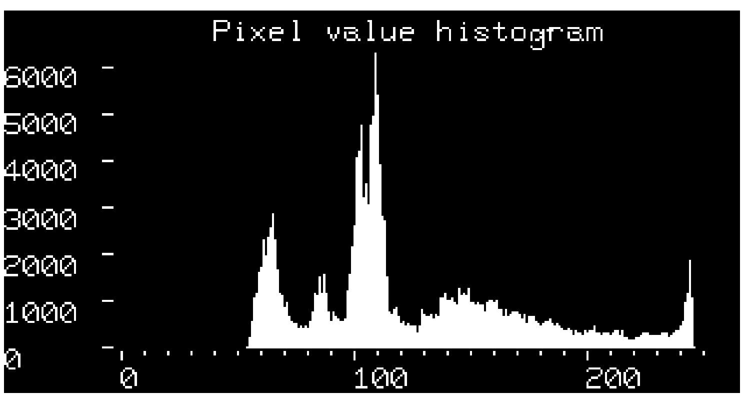


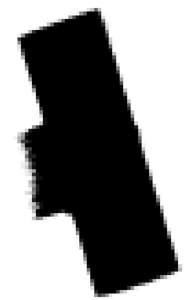
Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/threshld.htm

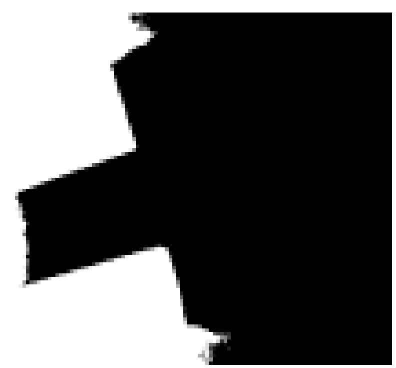
Thresholding

Original image and histogram:



Images after thresholding:





Group operations

- Group operations calculate new pixel values from a pixel’s neighbourhood
- The group operation is usually expressed in terms of template convolution
  - Template is a set of weighting coefficients
  - Template is usually square and its size is usually odd
  - For example 3 x 3 pixels

Group operations

- New pixel values are calculated by placing the template at the point of interest
- Pixel values are multiplied by the corresponding weighting coefficient and added to an overall sum
- The sum (usually) becomes the new value for the center pixel in the output image
- The template is then moved horizontally by one position until the end of the row

Group operations

w_0	w_1	w_2
w_3	w_4	w_5
w_6	w_7	w_8

- To calculate the value in new image N, at coordinates (x,y) the template operates on the original image O as follows:

$$N_{x,y} = (w_0 \times O_{x-1,y-1}) + (w_1 \times O_{x,y-1}) + (w_2 \times O_{x+1,y-1}) + \\ (w_3 \times O_{x-1,y}) + (w_4 \times O_{x,y}) + (w_5 \times O_{x+1,y}) + \\ (w_6 \times O_{x-1,y+1}) + (w_7 \times O_{x,y+1}) + (w_8 \times O_{x+1,y+1})$$

Group operations

- Note that we cannot ascribe values to the picture's borders since the template cannot extend beyond the image
- To calculate values for the borders, we have three options:
  - Set the border to black (or deliver a smaller image)
  - Assume that the image replicates to infinity along both dimensions and calculate new values by cyclic shift
  - Calculate the pixel value from a smaller area

Averaging operator

- For an averaging operator, the template weighting values are equal for all pixels in the template and the sum of the weighting values are 1
- In a template for a 3 x 3 averaging operator, each weight is thus 1/9

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Averaging operator

- Result of averaging with a 3 x 3 operator

Averaging operator

- The result of the averaging operator is that much of the detail disappears, revealing the broad image structure.
- The effect of the averaging operator is to reduce noise – advantage
- Disadvantage is that averaging causes blurring that reduces detail in an image

Averaging operator

- Larger template sizes can also be used
  - Common ones are 3 x 3, 5 x 5 and 7 x 7
  - Beyond this averaging imposes high computational costs
- Larger averaging operators smooth the image more and removes more detail

Averaging operators

(a) 5×5 (b) 7×7 (c) 9×9

Gaussian averaging operator

- Considered to be optimal for image smoothing
- Template values are set by Gaussian relationship
- The Gaussian function g at coordinate (x,y) is controlled by the variance σ^2 according to:

$$g(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

- This equation is used to calculate coefficients for the Gaussian template

Gaussian averaging operator

- The Gaussian function essentially removes the influence of points greater than 3σ in distance from the center of the template
- The Gaussian averaging operator retains more features than direct averaging and more noise is removed

Gaussian averaging operator

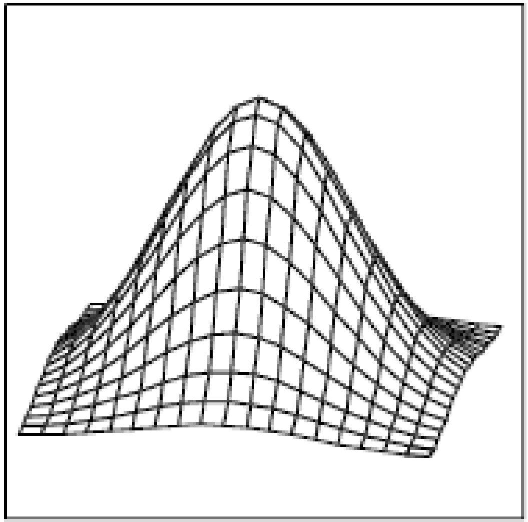
(a) 3×3

(b) 5×5

(c) 7×7

- Again we see that the effect of larger template size is the loss of detail

Gaussian averaging operator



Gaussian_template(19, 4)

- Surface plot of the Gaussian function shows a bell shape

- The central points are thus weighted to contribute more than the peripheral points
- The variance is chosen to ensure that template coefficients drop to near zero at the template's edge

Gaussian averaging operator

- Example of template size 5 x 5 with variance of one:

0.002	0.013	0.220	0.013	0.002
0.013	0.060	0.098	0.060	0.013
0.220	0.098	0.162	0.098	0.220
0.013	0.060	0.098	0.060	0.013
0.002	0.013	0.220	0.013	0.002

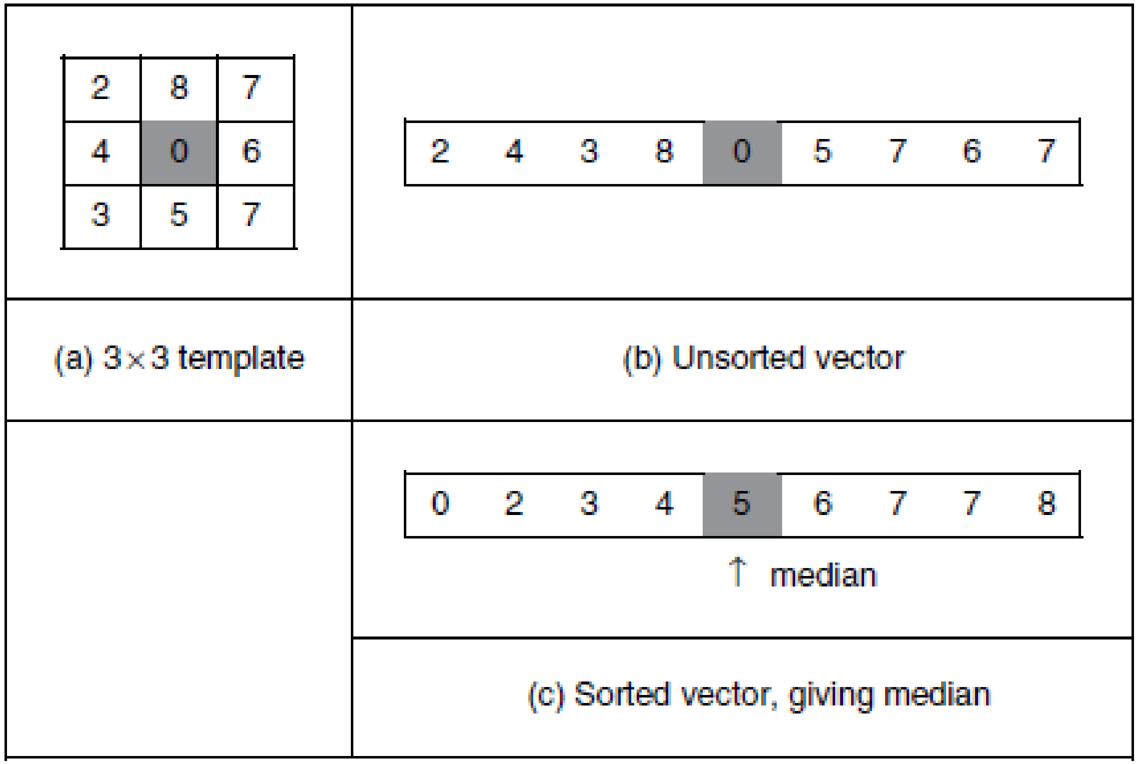
- It is possible to give the template assymetric properties by scaling the x and y coordinates

Gaussian averaging operator

Median filter

- A statistical operator
- The median is the centre of a ordered distribution
- The median is taken from a template centered on the point of interest
- Instead of a calculation, the value of the new pixel is the median of the template

Median filter



(a) 3×3 template

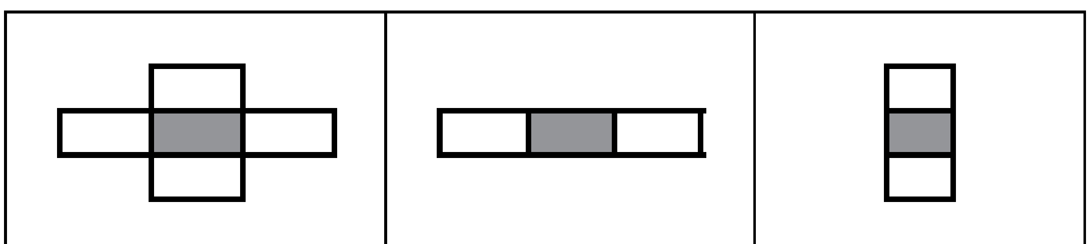
(b) Unsorted vector

(c) Sorted vector, giving median

- The value of the new pixel will thus be 5 (the middle component in the list)

Median filter

- When using a 3 x 3 template, the values first have to be sorted in a list to find the median
- Larger template sizes are also possible, but becomes computationally demanding and slow
- An alternative is to use template shapes other than a square

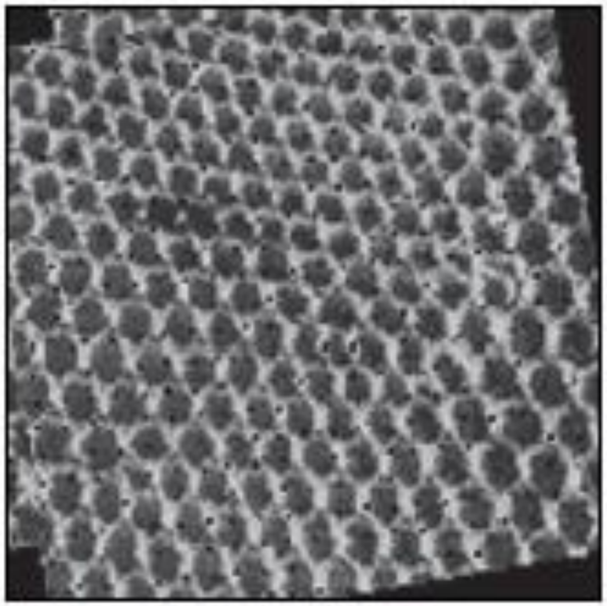


(a) Cross | (b) Horizontal line | (c) Vertical line

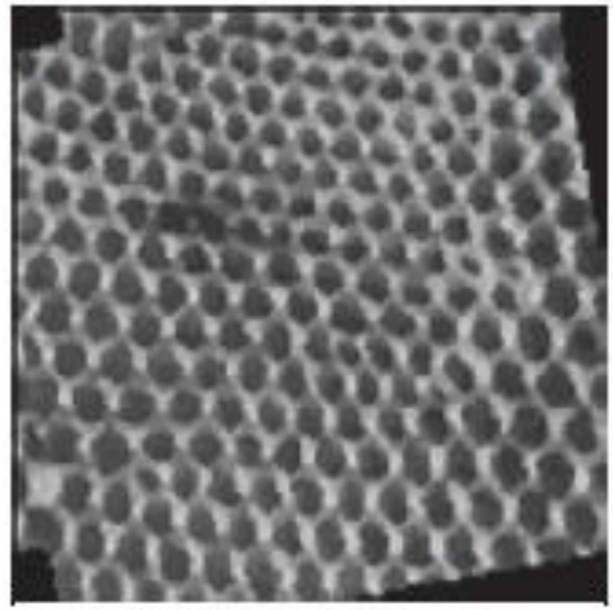
Median filter

- The median has a well-known ability to remove salt-and-pepper noise – isolated black and white points
- Since the salt-and-pepper noise will appear at either end of the list, they are removed by the median process
- Median filter is also good at retaining edges while suppressing noise contamination

Median filter



(a) Rotated fence



(b) Median filtered

Median filter

- Removes salt-and-pepper noise

Image from https://en.wikipedia.org/wiki/Median_filter

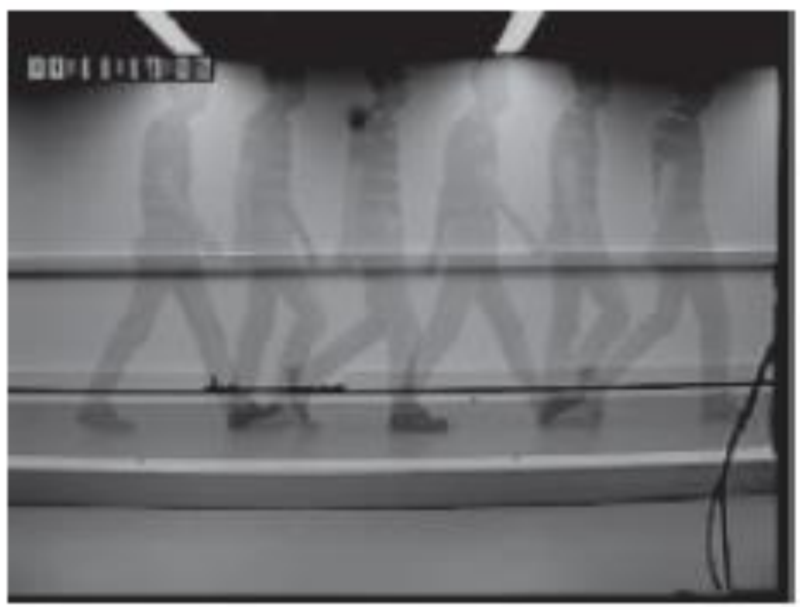
Operators implementation

- Finding the background of am image
  - We have a sequence of images of a walking subject
  - We want to be able to find the background

(a) Image 1 | (b) Image 2 | (c) Image 3 | (d) Image 4 | (e) Image 5 | (f) Image 6

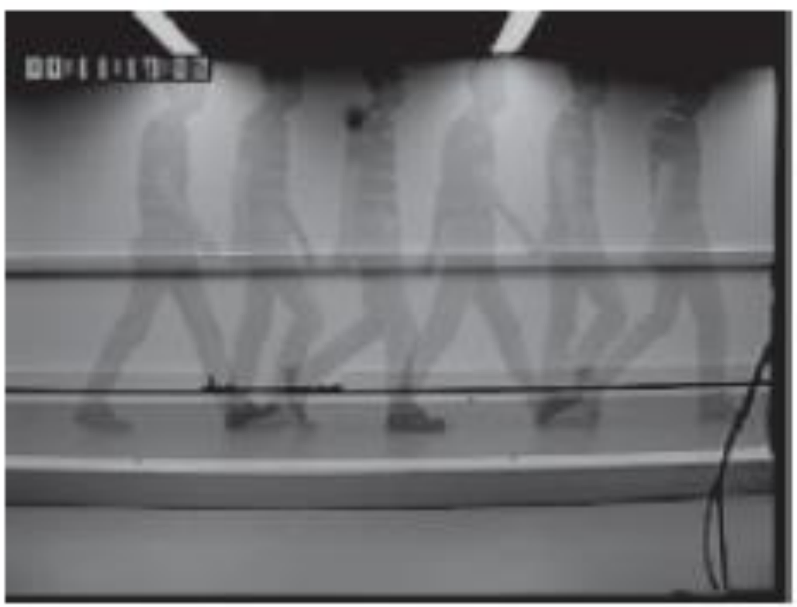
Operators implementation

- Average the images the images to find the background
  - Temporal average – image where each point is the average of the points in the same position in each of the images



Operators implementation

- To reduce the effect of the walking shadow we can include spatial averaging
  - This gives spatiotemporal averaging



Operators implementation

- As an alternative we can take the median of the points in the six images
  - Temporal median

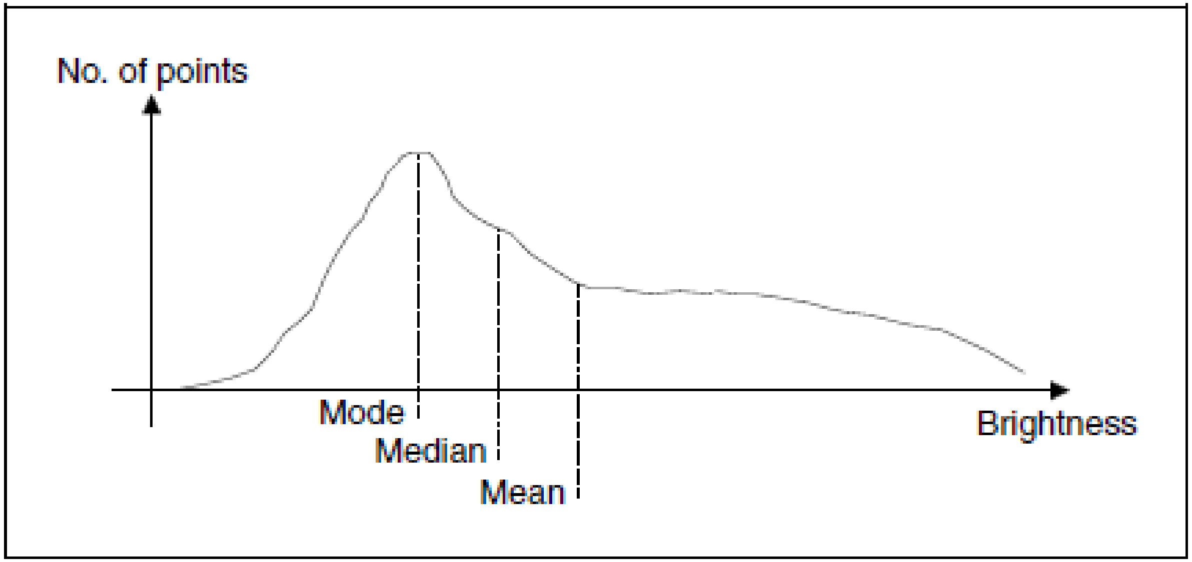
Mode filter

- The mode of a template is the most frequently occurring pixel value
- The center pixel can thus be replaced with the mode to remove noise
- It is however very difficult to determine the mode for a small population and we have to estimate the mode using a truncated median filter

Mode filter

- Truncated median filter is based on the principle that for many non-Gaussian distributions, the order of the mean, median and mode are the same for many images
- Accordingly, if we truncate the distribution, the median of the truncated distribution will approach the mode of the original distribution

Mode filter



- The operator first finds the mean and median of the current template
- The distribution of points is then truncated on the side of the mean so that the median now halves the distribution of the remaining points

Mode filter

- If the median is less than the mean then the point of truncation is

$$\text{upper} \quad = \text{median} + (\text{median} - \min(\text{distribution}))$$
$$= 2 \text{ x median} - \min(\text{distribution})$$

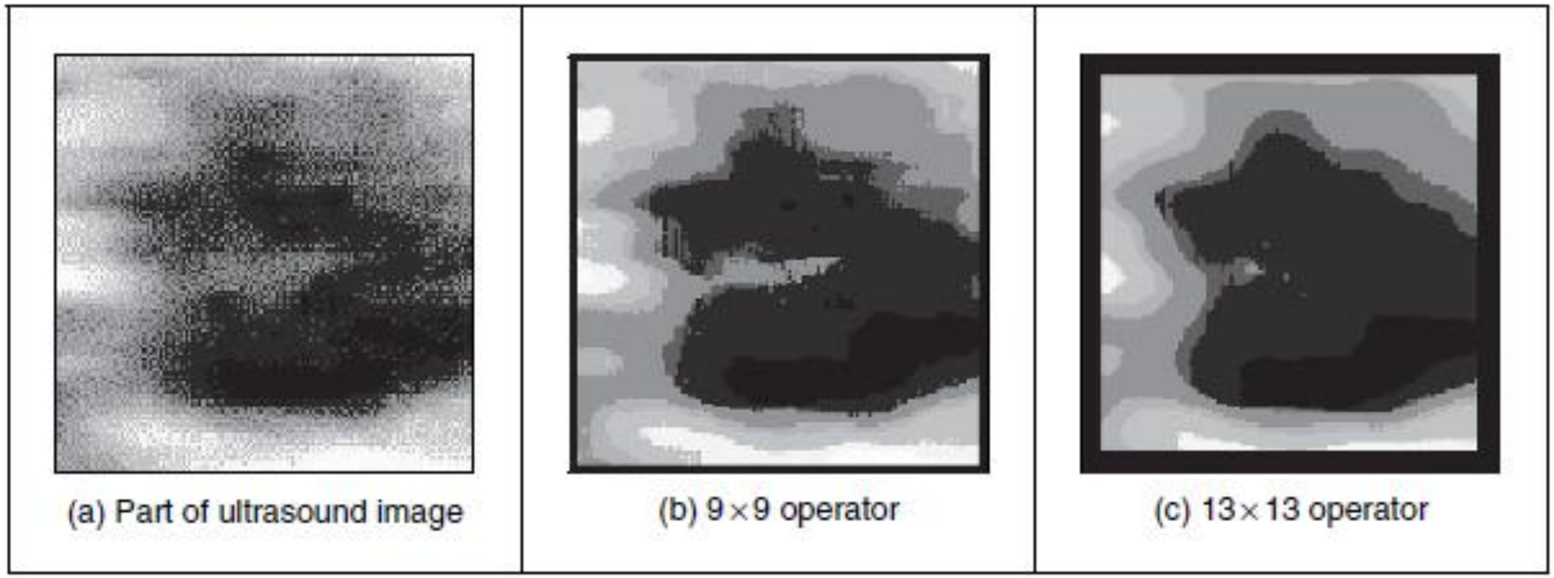
- If the median is greater than the mean then the point of truncation is

$$\text{lower} = 2 \text{ x median} - \max(\text{distribution})$$

Mode filter

- The median of the remaining distribution then approaches the mode
- The template size is usually large, say 7 x 7 or 9 x 9
- The effect is to remove noise while retaining feature boundaries

Mode filter



(a) Part of ultrasound image | (b) 9×9 operator | (c) 13×13 operator

Anisotropic diffusion

- The most advanced form of smoothing is achieved by preserving the boundaries

(a) Original image (b) Anisotropic diffusion (c) Gaussian smoothing

- The feature boundaries in the smoothed image are crisp and the skin is more matte

Anisotropic diffusion

- Anisotropic diffusion has its basis in heat flow and scale space
- Essential idea of scale space is that there is a multiscale representation of images – from low resolution (coarsely sampled) to high resolution (finely sampled)

Anisotropic diffusion

- A scale space set of images can be derived by convolving an original image with a Gaussian function:

$$\mathbf{P}_{x,y}(\sigma) = \mathbf{P}_{x,y}(0) * g(x, y, \sigma)$$

- Where $\mathbf{P}_{x,y}(0)$ is the original image, $g(x,y,\sigma)$ is the Gaussian template as per Gaussian averaging and $\mathbf{P}_{x,y}(\sigma)$ is the image at level σ
- Coarser levels correspond to larger values for σ

Anisotropic diffusion

- The family of images derived this way can be viewed as the solution of the heat equation
- The heat equation is the anisotropic diffusion equation:

$$\frac{\partial \mathbf{P}}{\partial t} = \nabla \cdot (c_{x,y}(t) \nabla \mathbf{P}_{x,y}(t))$$

- Where $\nabla\cdot$ is the divergence operator (measures how the density within a region changes) and diffusion coefficient $c_{x,y}$ controls the amount of local change

Anisotropic diffusion

- By approximating differentiation by differencing the rate of change of the image between step *t* and step *t+1* is

$$\frac{\partial \mathbf{P}}{\partial t} = \mathbf{P}(t+1) - \mathbf{P}(t)$$

- This implies we have an iterative solution
- Image P at time step *t+1* is denoted as $\mathbf{P}^{<t+1>}$

$$\mathbf{P}^{<t+1>} - \mathbf{P}^{<t>} = \nabla \cdot (c_{x,y}(t) \nabla \mathbf{P}_{x,y}^{<t>})$$

Anisotropic diffusion

- Using differences over the four compass directions:

$$\nabla_N\left(\mathbf{P}_{x,y}\right) = \mathbf{P}_{x,y-1} - \mathbf{P}_{x,y}$$
$$\nabla_S\left(\mathbf{P}_{x,y}\right) = \mathbf{P}_{x,y+1} - \mathbf{P}_{x,y}$$
$$\nabla_E\left(\mathbf{P}_{x,y}\right) = \mathbf{P}_{x-1,y} - \mathbf{P}_{x,y}$$
$$\nabla_W\left(\mathbf{P}_{x,y}\right) = \mathbf{P}_{x+1,y} - \mathbf{P}_{x,y}$$

- When we use this as an approximation to the previous equation we get:

$$\mathbf{P}^{<t+1>} - \mathbf{P}^{<t>} = \lambda(cN_{x,y}\nabla_{\mathrm{N}}(\mathbf{P}) + cS_{x,y}\nabla_{\mathrm{S}}(\mathbf{P}) + cE_{x,y}\nabla_{\mathrm{E}}(\mathbf{P}) + cW_{x,y}\nabla_{\mathrm{W}}(\mathbf{P}))$$

Anisotropic diffusion

- By rearrangement we obtain the anisotropic diffusion equation:

$$\mathbf{P}^{<t+1>} = \mathbf{P}^{<t>} + \lambda(c\mathrm{N}_{x,y}\nabla_N(\mathbf{P}) + c\mathrm{S}_{x,y}\nabla_S(\mathbf{P}) + c\mathrm{W}_{x,y}\nabla_W(\mathbf{P}) + c\mathrm{E}_{x,y}\nabla_E(\mathbf{P})), \mathbf{P} = \mathbf{P}_{x,y}^{<t>}$$

- This shows that the solution is iterative: images at one step (denoted by <t+1>) are computed from images at the previous step (denoted by <t>)

Anisotropic diffusion

- The first image is the original (noisy) image
- A new image is formed by adding a controlled amount of local change

Anisotropic diffusion

- Functions for the conduction coefficients ($c\mathrm{N}_{\mathrm{x,y}}$...) are chosen along the compass directions
- We need a function that tends to zero but increases with a larger difference (an edge or boundary) so that diffusion does not take place across the boundaries (edges)

Anisotropic diffusion

- The one function that can achieve this is:

$$g(x,k) = e^{-x^2/k^2}$$

- $cN_{x,y}$, $cS_{x,y}$, $cE_{x,y}$ and $cW_{x,y}$ is thus denoted as:

$$cN_{x,y} = g(\|\nabla_N(\mathbf{P})\|)$$
$$cS_{x,y} = g(\|\nabla_S(\mathbf{P})\|)$$

etc

Anisotropic diffusion

- This function has the desired properties since when the values of the differences ∇ are large, the function g is very small and vice versa
- k controls the rate at which the conduction coefficient decreases with increasing difference magnitude
- The aim is near unity for small differences and near zero for large differences

Anisotropic diffusion

- To recap;
  - We want to filter an image by retaining boundary points
  - These are retained according to the value of k
  - This function is operated in the four compass directions – to weigh the brightness differences in each direction
  - These contribute to an iterative equation that calculates a new value for an image point by considering the contribution from its fours neighbours

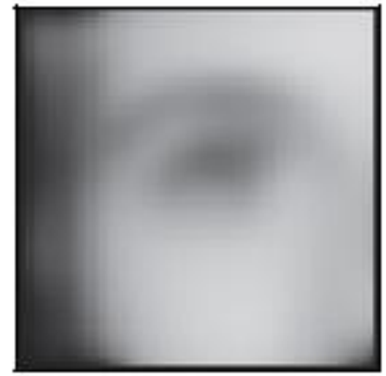
Anisotropic diffusion

- We now need to choose values for k and λ
- We also need to choose the number of iterations
- k is the conduction coefficient and low values preserve edges while high values allow diffusion (smoothing)
- λ controls the amount of smoothing

Anisotropic diffusion

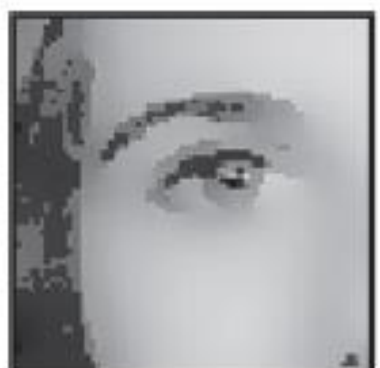
(a) $k = 100$ and $\lambda = 0.05$

(b) $k = 100$ and $\lambda = 0.15$

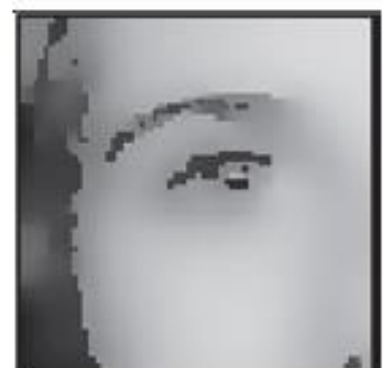


(c) $k = 100$ and $\lambda = 0.25$

(d) $k = 5$ and $\lambda = 0.25$



(e) $k = 15$ and $\lambda = 0.25$



(f) $k = 25$ and $\lambda = 0.25$

Mathematical morphology

- Mathematical morphology analyzes images by using operators developed using set theory
- Was originally developed for binary images and was extended to also include gray-level
- Images are processed according to shape
- Morphological operators define local transformations that change pixel values that are represented as sets

Mathematical morphology

- Pixels are changed according to hit or miss transformation
  - An object represented by a set X, is examined through a structural element B

Mathematical morphology

$$X \otimes B = \{x | B_x^1 \subset X \cap B_x^2 \subset X^c$$

x – one element of X (a pixel in an image)

X^c – complement of X (set of pixels that are not in the set X)

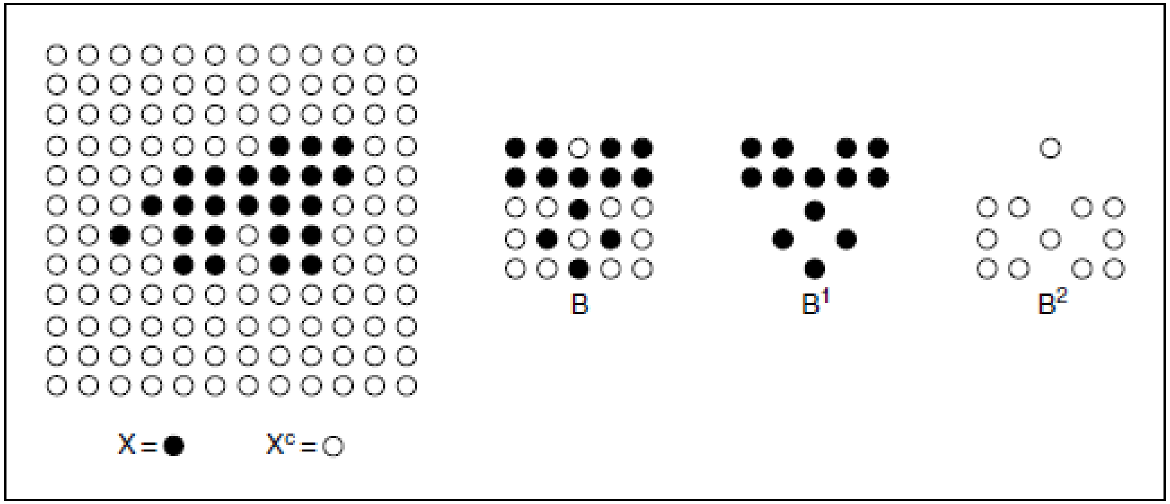
Structuring element B is represented by two parts: B^1 and B^2

Operation of B^1 on X is a “hit”

Operation of B^2 on X^c is a “miss”

Subindex x indicates that it is moved to position of element x

Mathematical morphology



- Illustrates binary image and structuring element
- Structural element is decomposed into two sets
- Each subset is used to analyze the set and its complement
- B^1 (black) and B^2 (white) are applied to X and X^c respectively

Mathematical morphology

- The structural element B is placed at each pixel
- It then performs a pixel by pixel comparison against the template B
- If the value of the image is the same as that of the structuring element then then image's pixel forms part of the resulting set $X \otimes B$

Morphological operators

- Simplest form of morphological operator is defined when either B^1 or B^2 is empty
- When B^1 is empty it is defined as erosion (reduction)

$$X \ominus B = \{x | B_x^1 \subset X\}$$

- When B^2 is empty it is defined as dilation (increase)

$$X \oplus B = \{x | B_x^2 \subset X^c\}$$

Morphological operators

(a) Original image

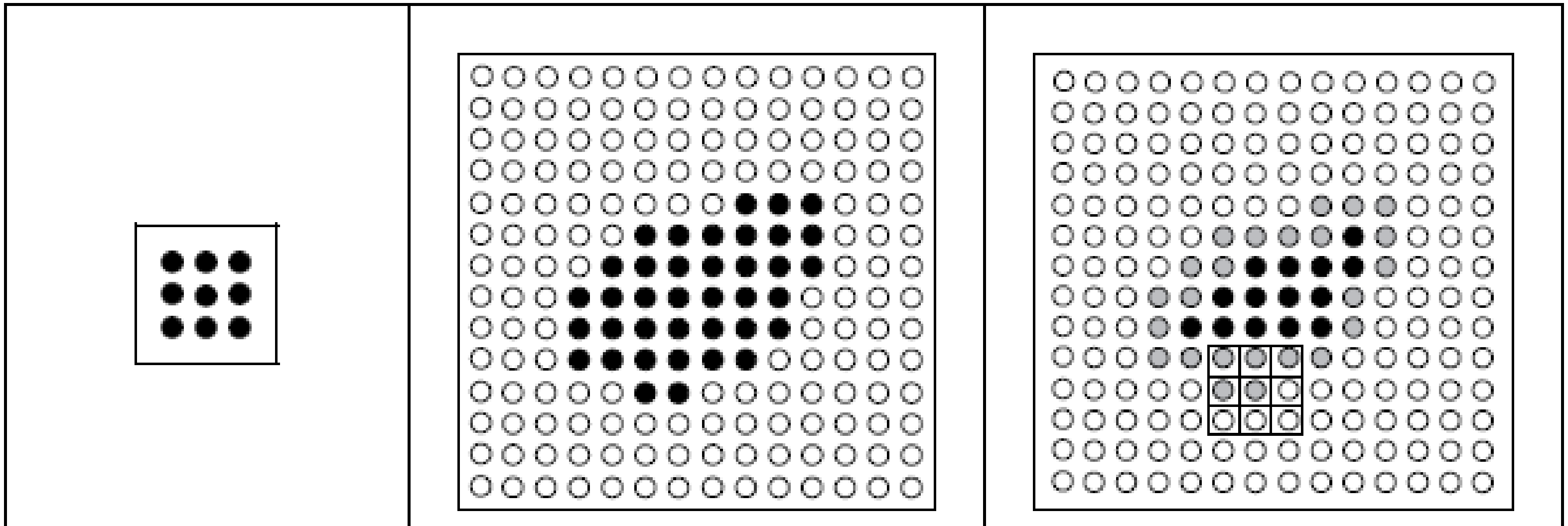
(b) Erosion

(c) Dilation

Erosion

- A pixel x belongs to the eroded set if each point of the element B^1 is on X
- This operator removes the pixels at the borders of objects and erodes or shrinks the set

Erosion



(a) Structural element (b) Image (c) Erosion

- The eroded set is formed only from black pixels
- Gray pixels were removed
- For a pixel x, that pixel is removed if all of the pixels in the structural element does not match X

Erosion

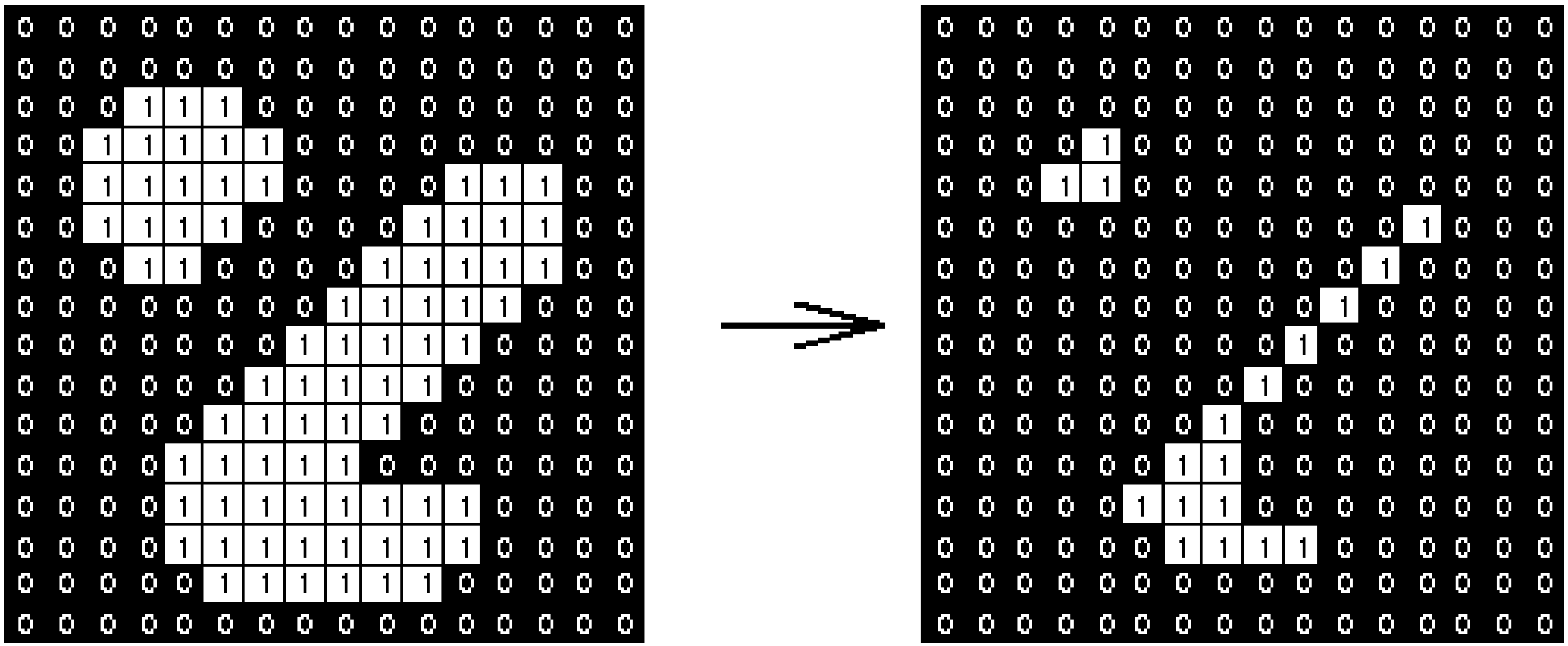


Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm

Erosion

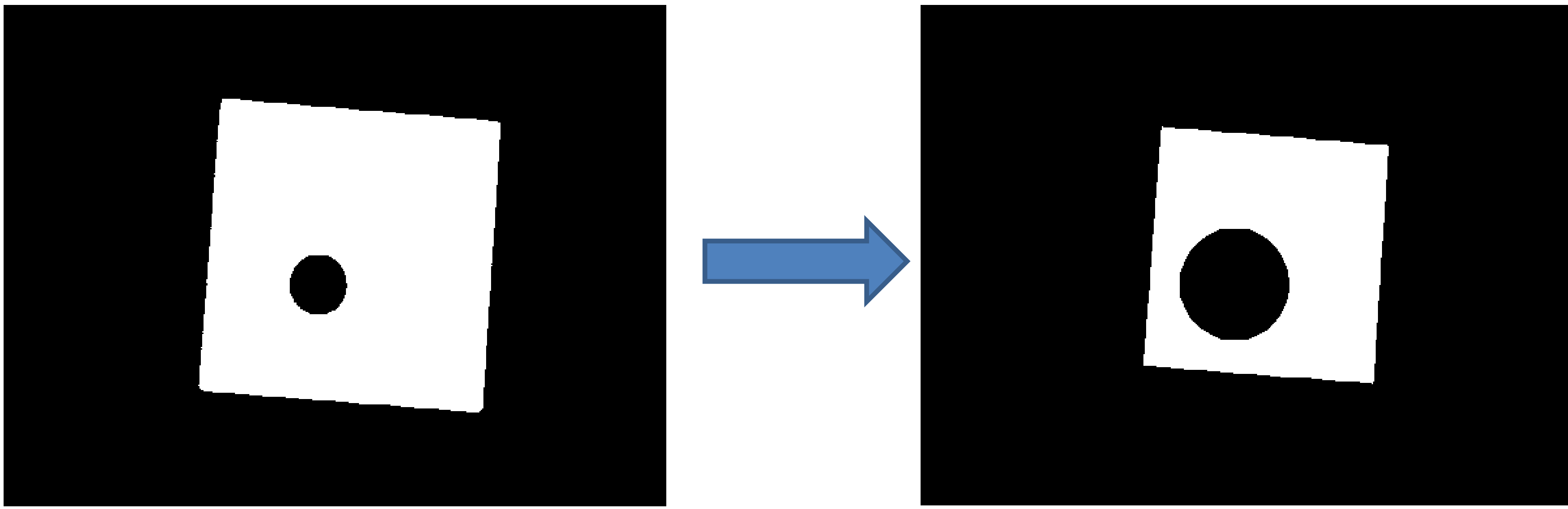
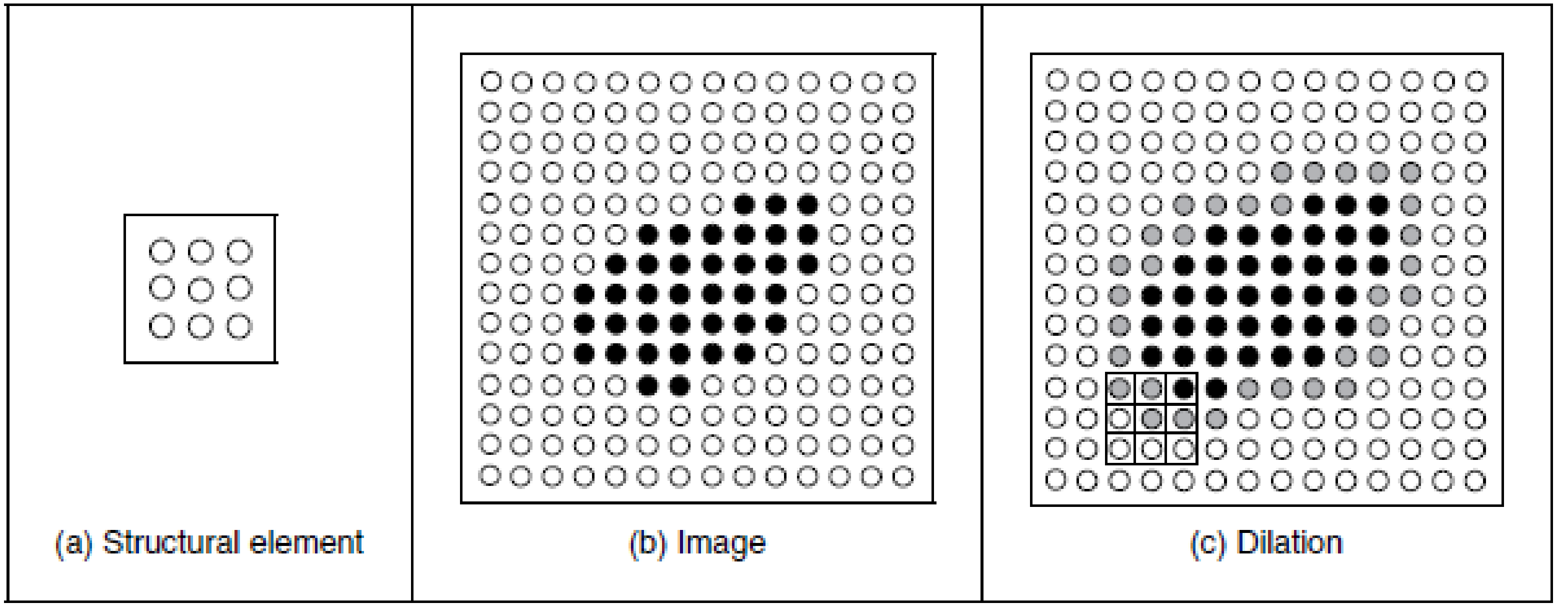


Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm

Dilation

- A point belongs to the dilated set when all the points in B^2 are in the complement
- This operator shrinks or erodes the complement which increases the set X

Dilation



- Gray pixels are added to the set
- Structural element is placed on each pixel in the complement (each white pixel)
- When it does not fully match the structural element then the pixel is removed from X^c (thus added to X)

Dilation

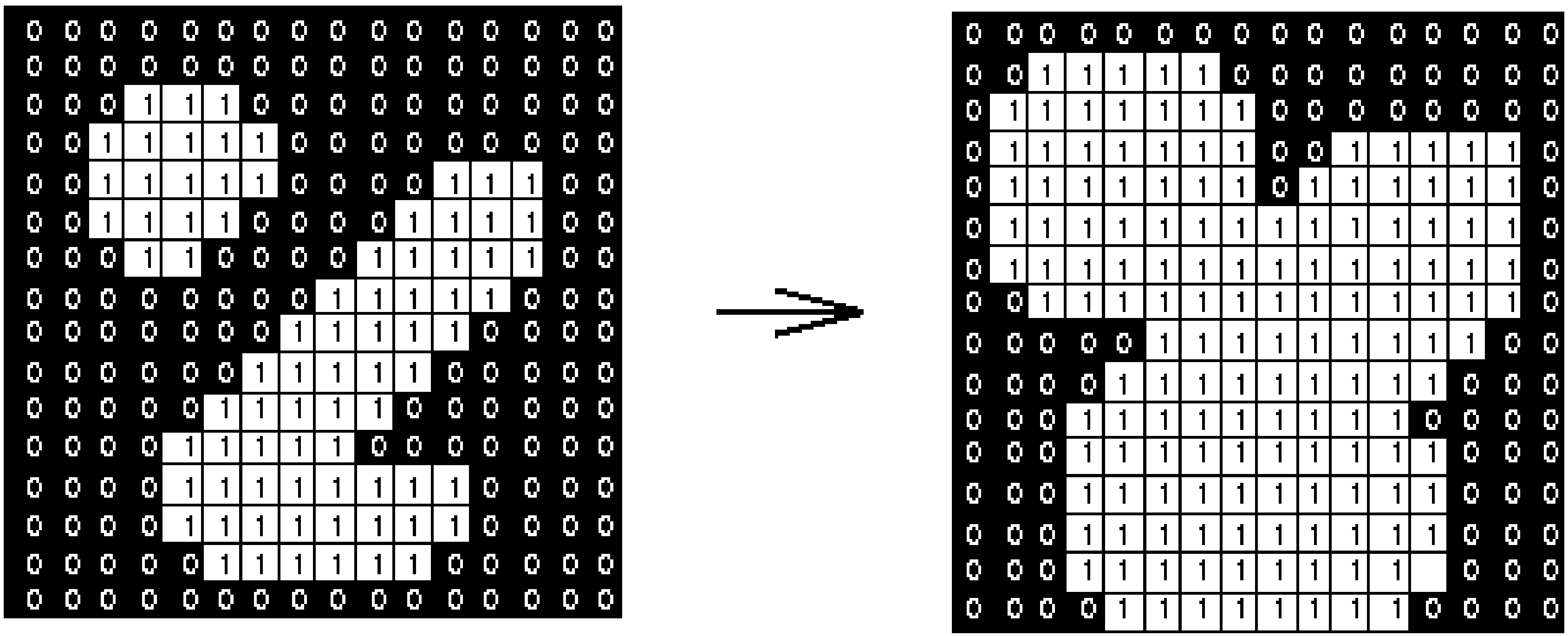


Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm

Dilation

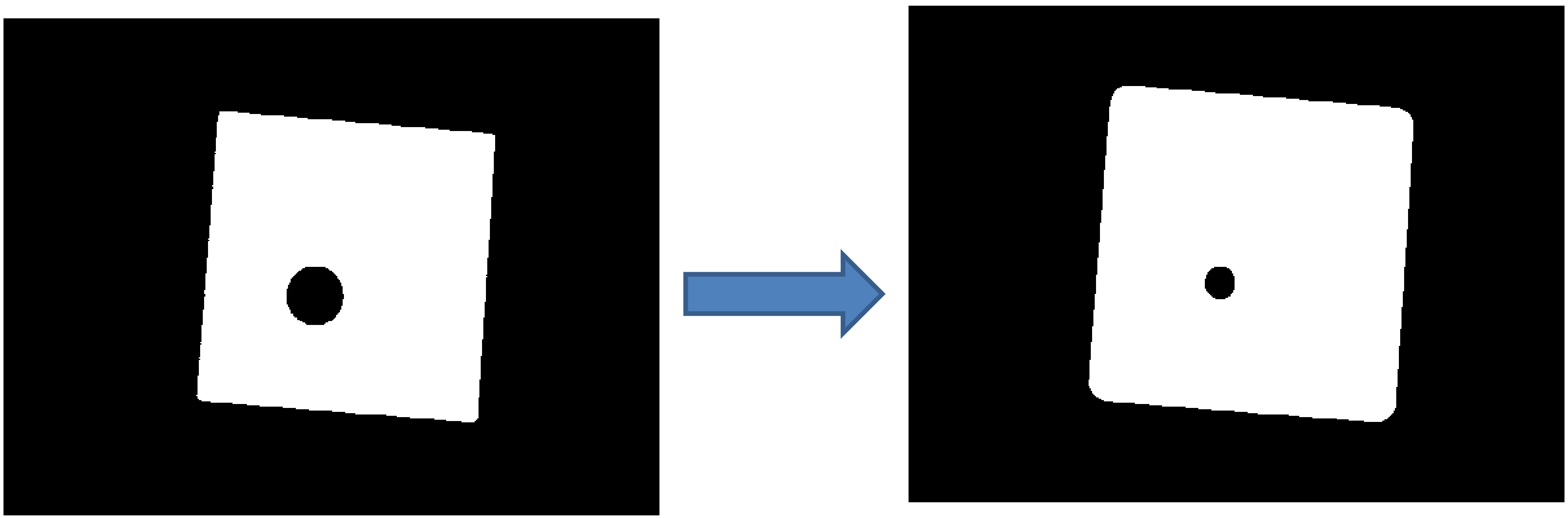


Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm

Morphological operators

- The structuring element does not require a specific shape
  - Square, circular, cross, triangle and diamond are often used with slightly different results
- The “strength” of the transformation is defined by the size of the structuring element
- In general applications use small structural elements (for speed)

Morphological operators

- A succession of transformations are often performed to achieve a desired result
- The opening operator is an erosion followed by a dilation

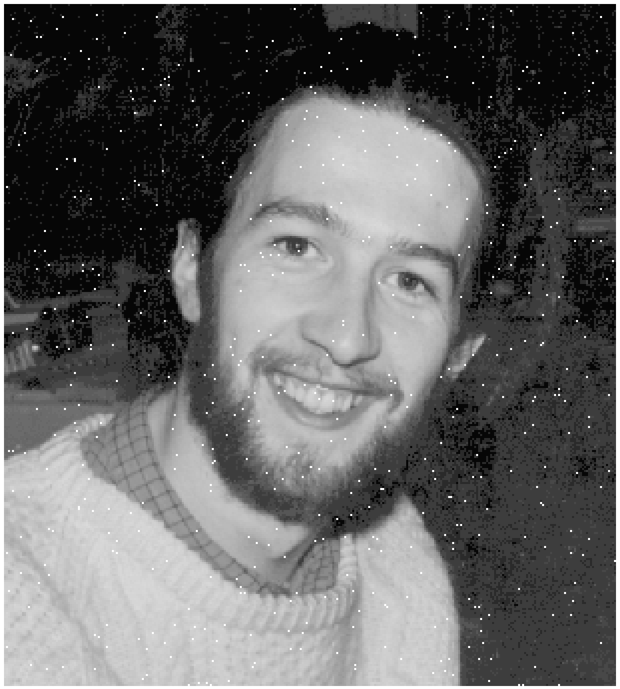
$$X \circ B = (X \ominus B) \oplus B$$

  - Opening is very good at removing salt noise
- The closing operator is a dilation followed by an erosion

$$X \bullet B = (X \oplus B) \ominus B$$

  - Closing is very good at removing pepper noise

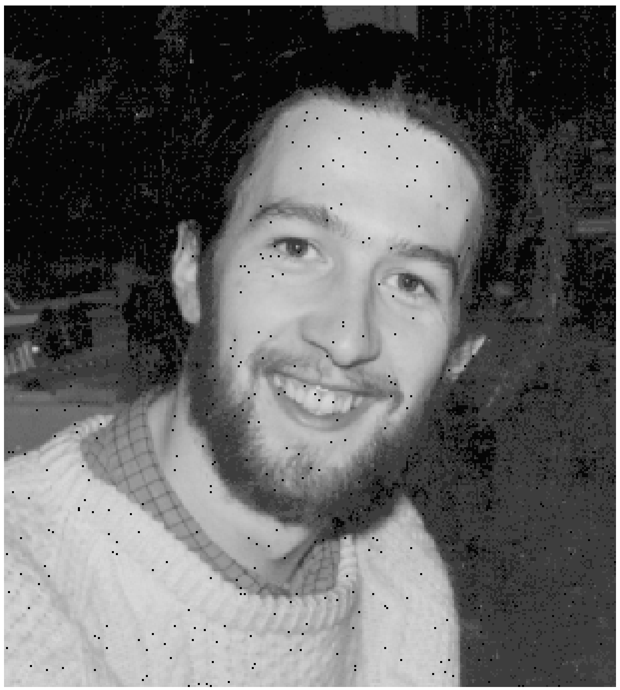
Opening



- 3x3 square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm

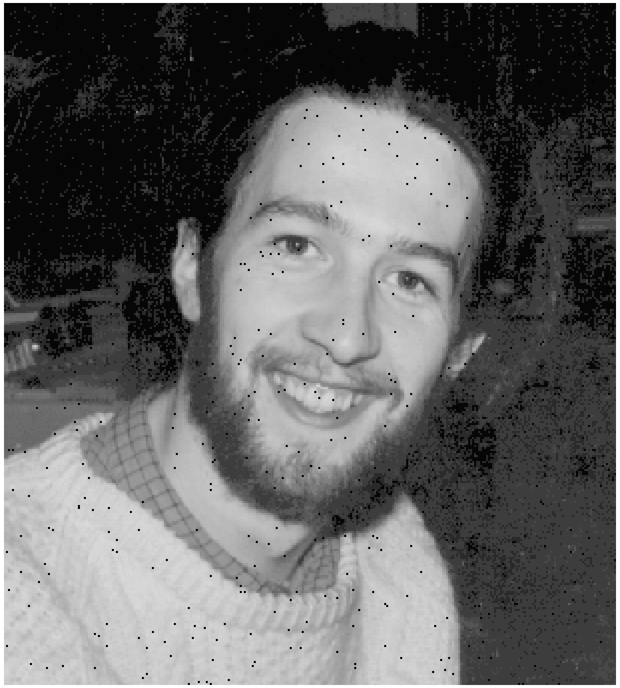
Opening



- 3x3 square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm

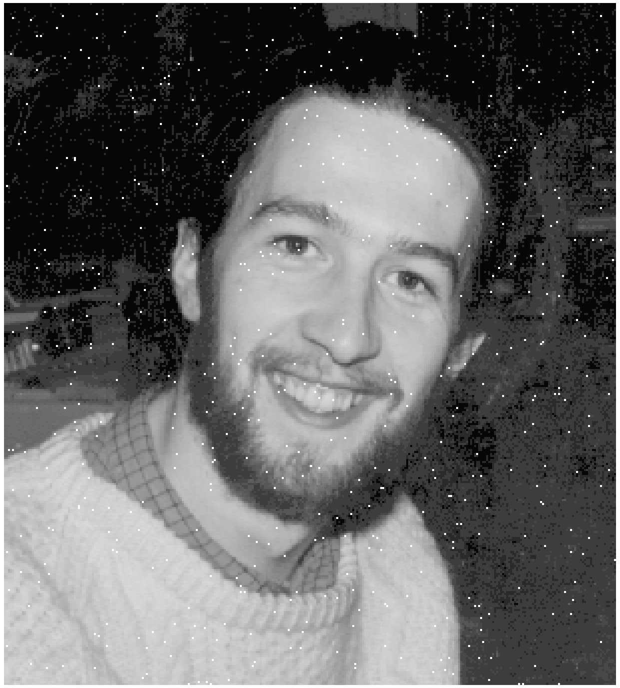
Closing



- 3x3 square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm

Closing



- 3x3 square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm

Morphological operators

- Opening and closing operators are generally used as filters to remove salt-and-pepper noise and to smooth surfaces
- Morphological operators are also used in edge detection
  - Edges can be detected by subtracting the original image and the one obtained by erosion or dilation

Gray-level morphology

- Gray-level morphology extends binary morphology to represent functions as sets
- There are two alternative representations:
  - Cross section
    - Gray-level images are defined as a stack of binary images
  - Umbra approach

Umbra approach

- Defines sets as the points contained below functions
- The umbra of function *f(x)* consists of all points below the function:

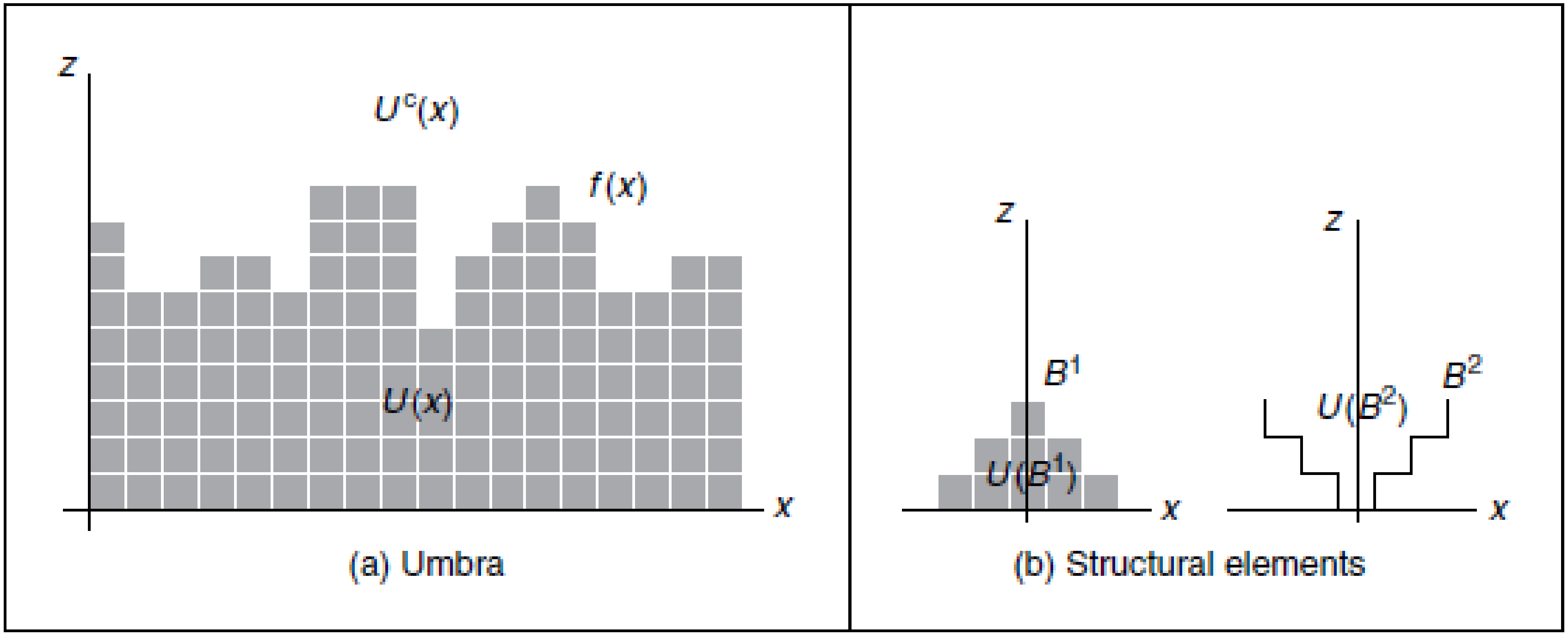
$$U(X) = \{(x, z) | z < f(x)\}$$

- Here *x* represents a pixel and *f(x)* its gray level
- The space *(x,z)* is formed by the combination of all pixels and their gray levels

Umbra approach

- For images, x is defined in 2D, thus all points of the form (x,z) define a 3D cube
- An umbra is a collection of points in this 3D space

Umbra approach



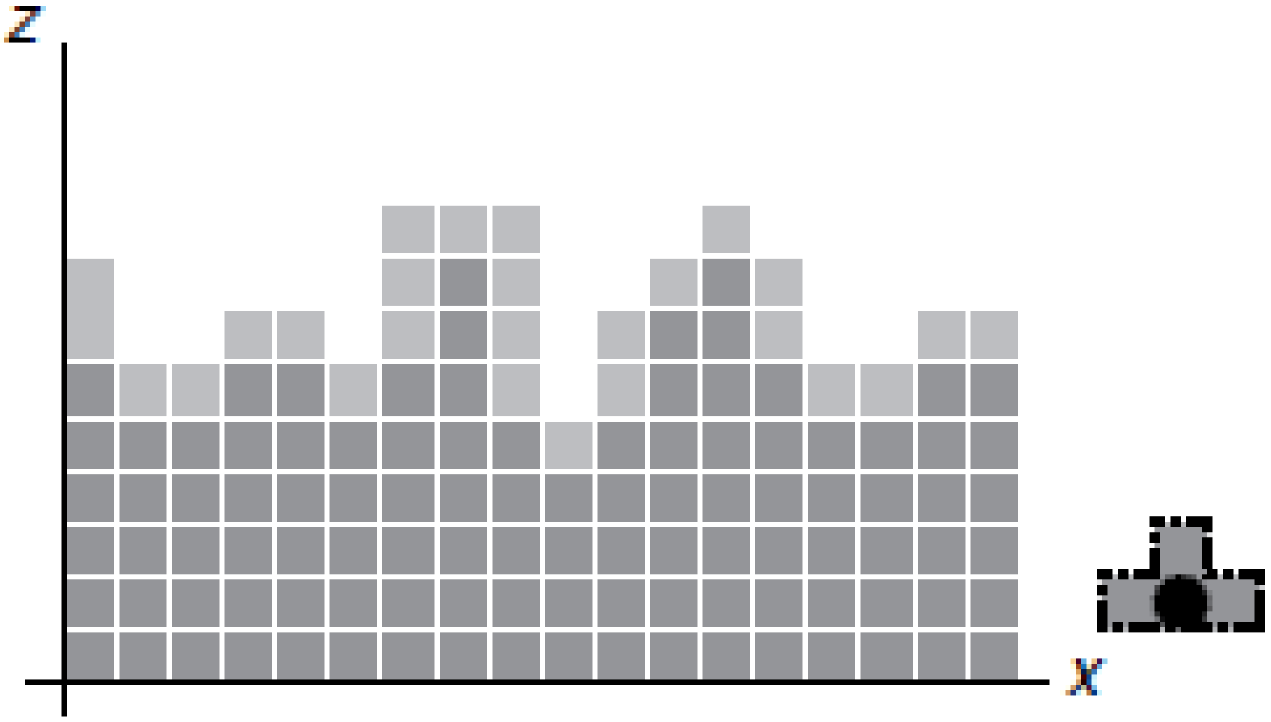
- *f(x)* is shown as 1D function for simplicity
- $U^c(X)$ is the complement of the umbra and is given by points on and above the curve

Umbra approach

- The hit or miss transformation of gray-level functions:
$$U(X \otimes B) = \{(x,z) | U(B^1_{x,z}) \subset U(X) \cap U(B^2_{x,z}) \subset U^c(X)\}$$
- The structural element is translated along the points *(x,z)*
- To know if a point *(x,z)* is in the transformed set, we move the structural element B^1 to the point and see if its umbra fully intersects $U(X)$

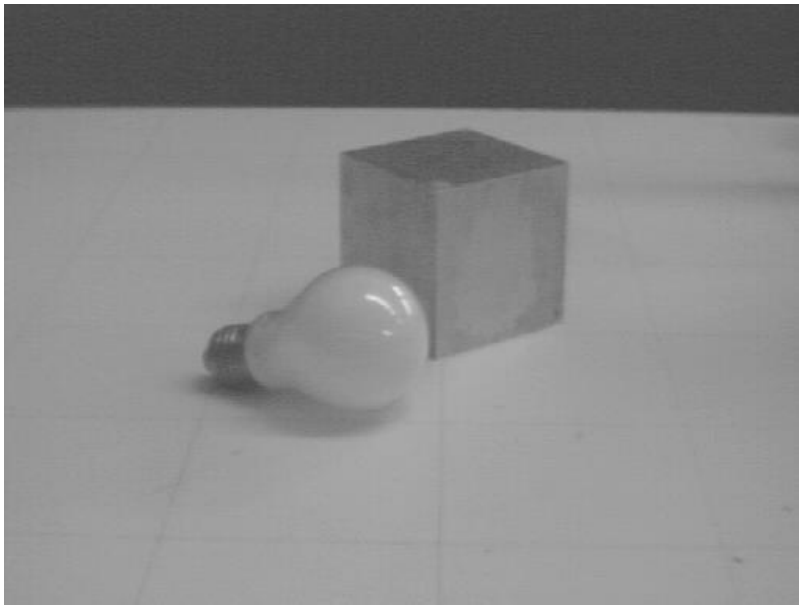
Gray level erosion

$$U(X \ominus B) = \{(x,z) | U(B^1_{x,z}) \subset U(X)\}$$

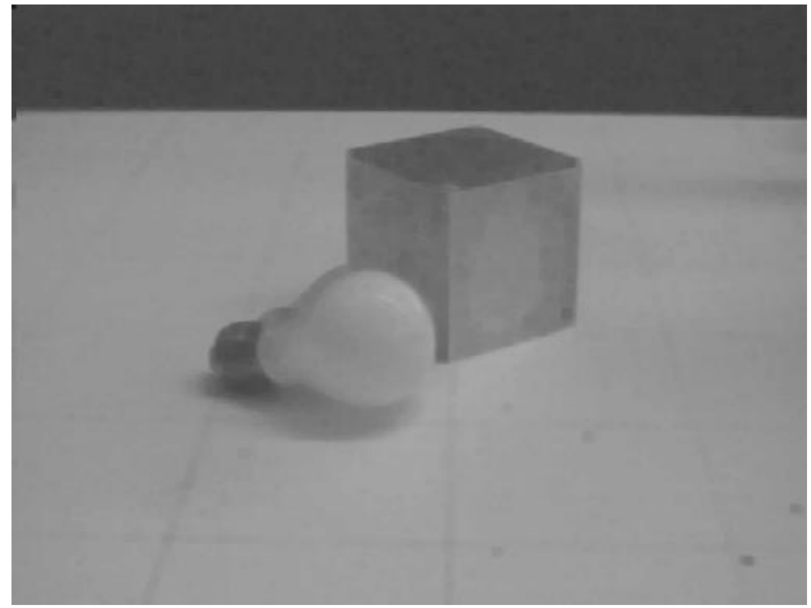


(a) Erosion

Gray level erosion



Original image

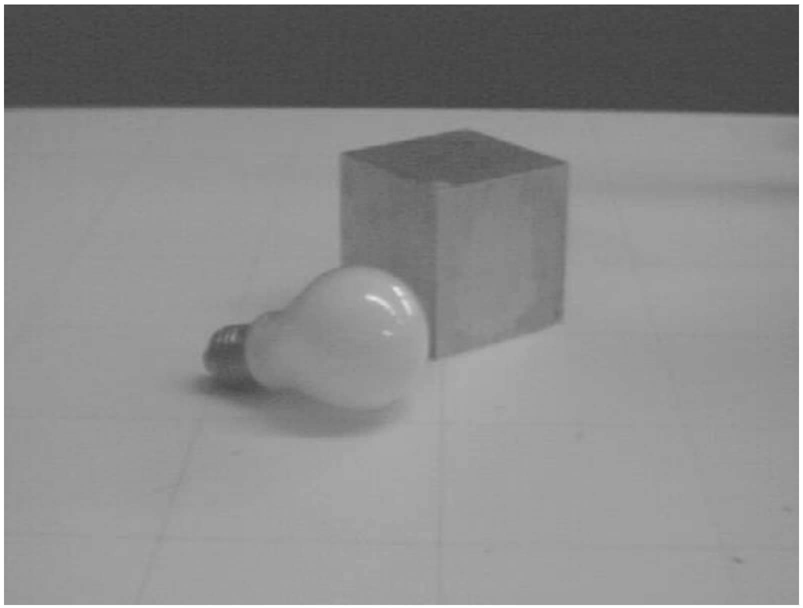


Erosion image

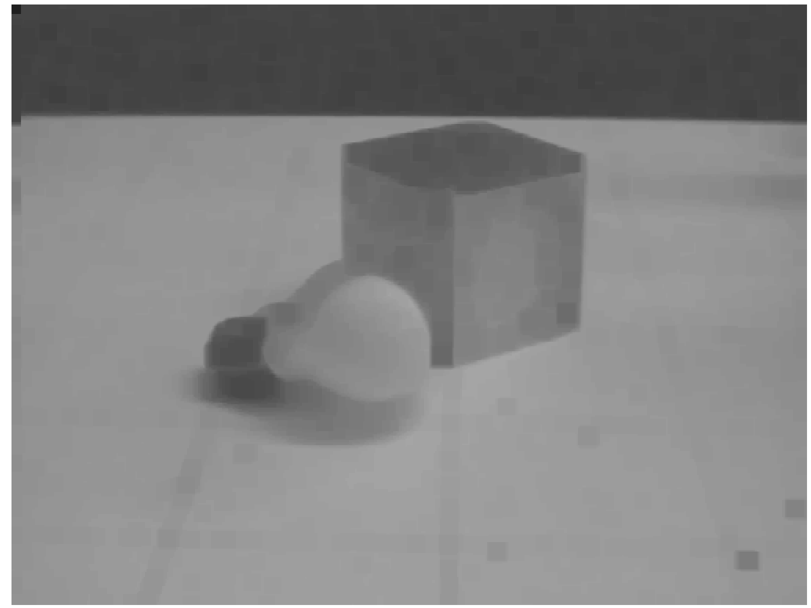
- After two erosion passes using a 3 x 3 flat square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm

Gray level erosion



Original image



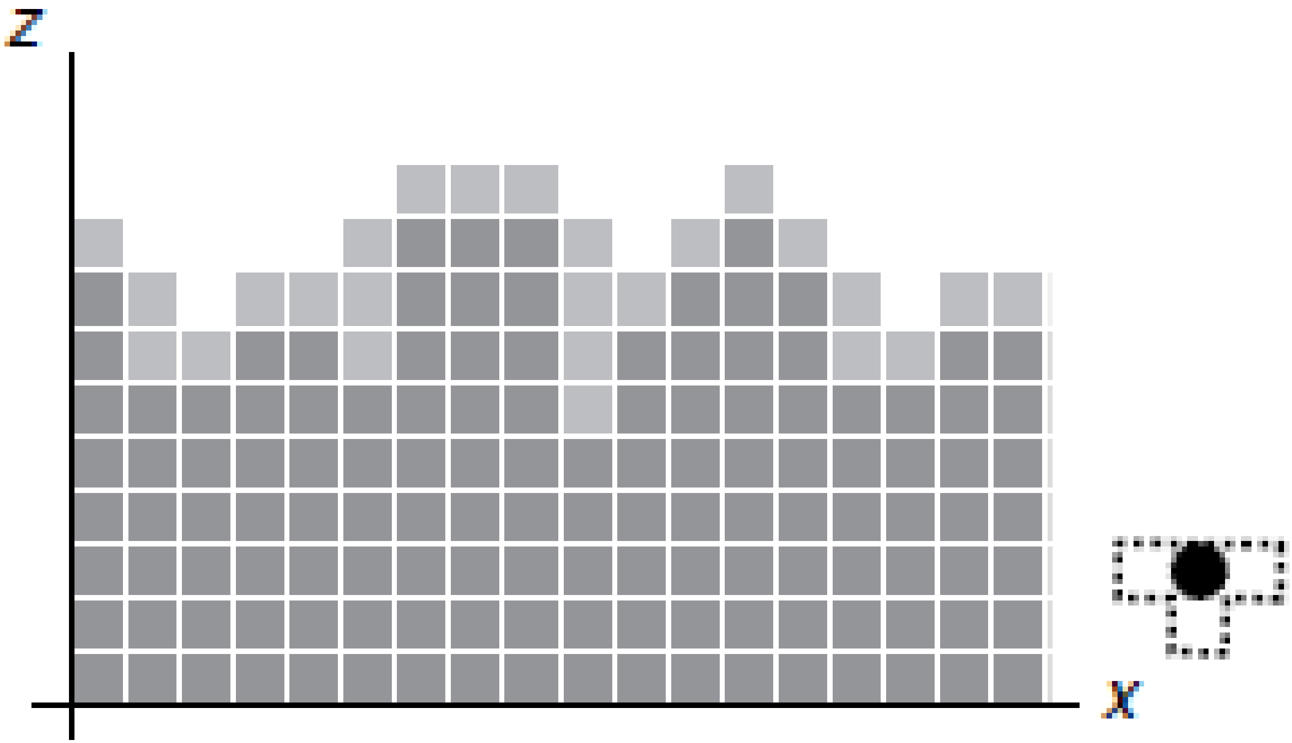
Erosion image

- After five erosion passes using a 3 x 3 flat square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm

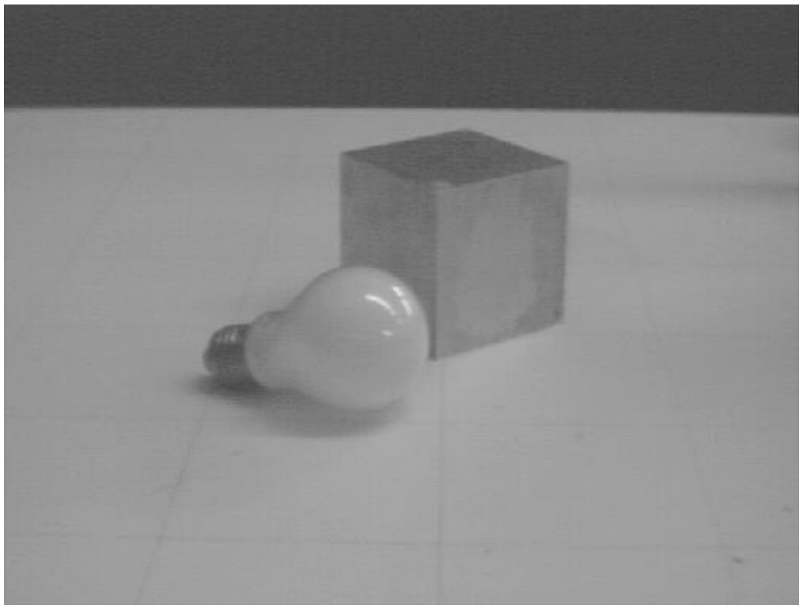
Gray level dilation

$$U(X \oplus B) = \{(x,z) | U(B^2_{x,z}) \subset U^c(X)\}$$

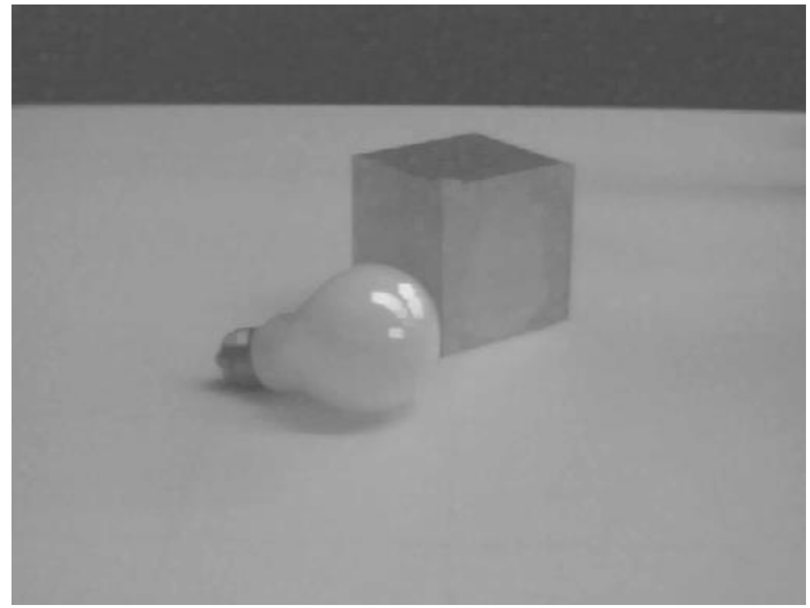


(b) Dilation

Gray level dilation



Original image

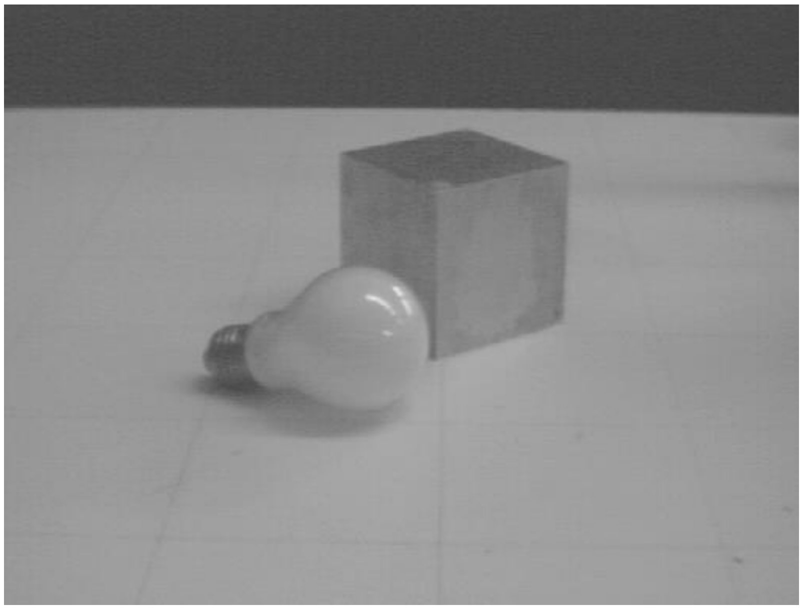


Dilation image

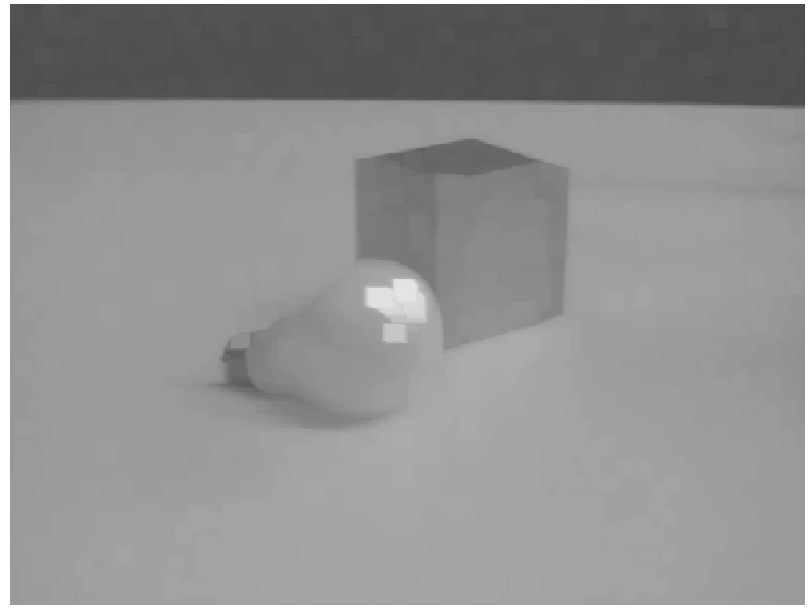
- After two erosion passes using a 3 x 3 flat square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm

Gray level dilation



Original image



Dilation image

- After five erosion passes using a 3 x 3 flat square structuring element

Image from http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm

Minkowski operators

- Erosion and dilation operators require the computation of intersections which is processing intensive
- Minkowski operators simplify these computations into mathematical operations

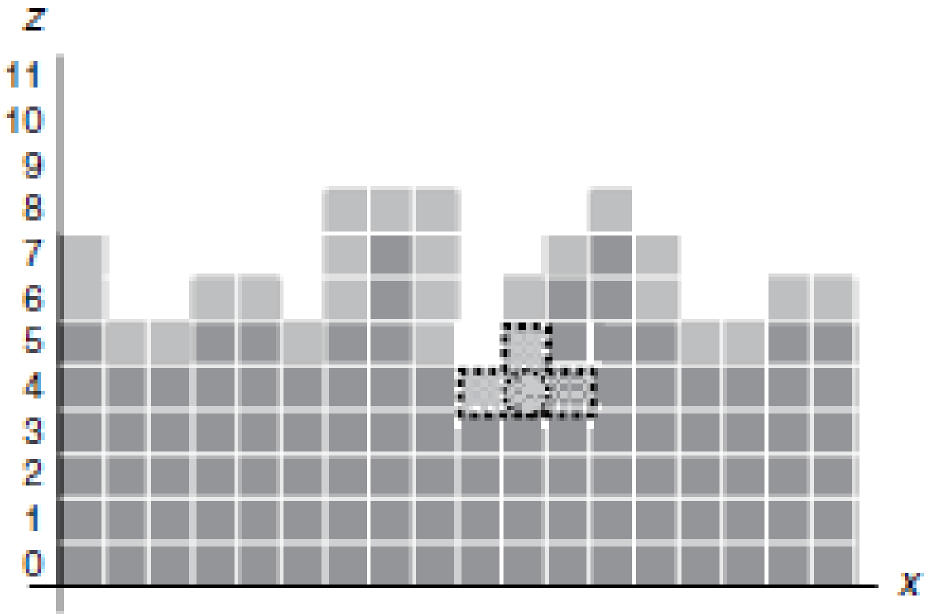
Minkowski operators - Erosion

- The value of a pixel can simple be calculated by comparing the gray level values of the structural element and corresponding image pixels
- The highest point that we can translate the structural element without intersecting the complement is given by the minimum value of the difference between the gray level of the image pixel and the corresponding pixel in the structural element

$$\ominus(x) = min_i\{f(x-i) - B(i)\}$$

- Where $B(i)$ denotes the value of the i-th pixel of the structural element

Minkowski operators - Erosion



(a) Erosion

- The structural element has three pixels with values 0, 1 and 0 respectively
- Subtractions:

  4 – 0 = 4

  6 – 1 = 5

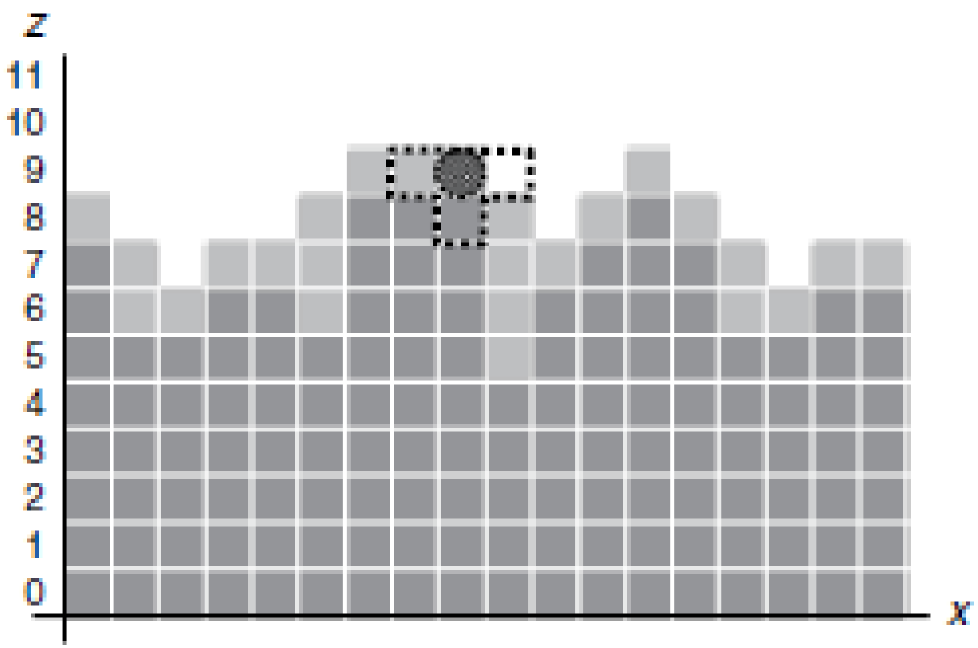
  7 – 0 = 7
- The minimum value is thus 4

Minkowski operators - Dilation

- The dilation can be obtained by comparing the gray-level value with the image and the structural element

$$\ominus (x) = max_i\{f(x-i) + B(i)\}$$

Minkowski operators - Dilation



(b) Dilation

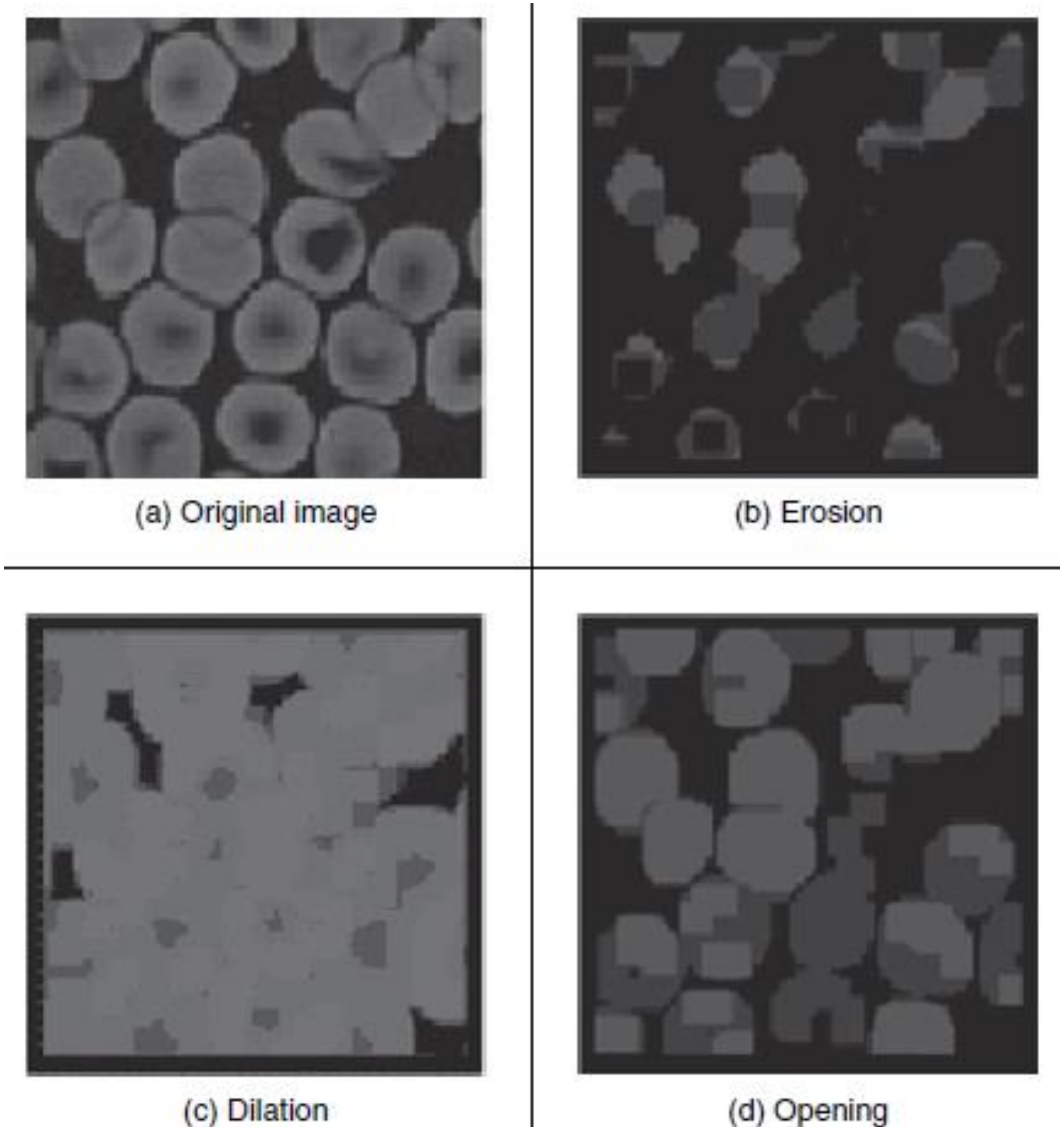
- Summation:

  8 + 0 = 8

  8 + 1 = 9

  4 + 0 = 4
- The maximum value is thus 9

Morphological operators



(a) Original image

(b) Erosion

(c) Dilation

(d) Opening

- Original image has 128x 128 pixels
- Structural element defined by 9 x 9 flat area
- Opening forms regular regions of similar size to original and removes small regions