Lecture 9
Morphological Image Processing

Prof. Lu FANG

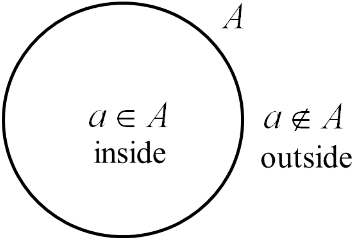
Tsinghua–Berkeley Shenzhen Institute

Morphological Image Processing

- **Morphology**: a branch of biology that deals with the form and structure of animals and plants.
- Morphological image processing is used to extract image components for representation and description of region shape, such as boundaries, skeletons, and the convex hull.
- We use the language in set theory to represent morphological operations.

Basic Concepts from Set Theory I

- Let A be a set in Z^2 ($2D$ integer space)
- If $a = (a_1, a_2)$ is an element of A, where a_1 and a_2 are coordinates of pixels in $2D$, then we write $a \in A$.
- If a is not an element of A, then we write $a \notin A$.



- If A is a set with no element, then we call it *null or empty set*,

$$A = \emptyset.$$

Basic Concepts from Set Theory II

- A set is specified by the contents of two braces: $\{\}$,
  - a. e.g. $C = \{w | w = -d, \text{for } d \in D\}$.
  - b. C represents a set of elements.
  - c. w represents the elements.
  - d. All elements inside C are formed by multiplying the element coordinates, d, by -1.
- A is a subset of B: every element of a set A is also an element of another set B. (every element of A is in B),

$$A \subseteq B.$$

- Union of two sets A and B: the set of all elements belonging to either A or B. (all elements are either in A or B)

$$C = A \cup B.$$

Basic Concepts from Set Theory III

- Intersection of two sets A and B: the set of all elements belonging to both A and B. (all elements are in A and B)

$$D = A \cap B.$$

- Two sets A and B are disjoint or mutually exclusive if their intersection is an empty set,

$$A \cap B = \emptyset.$$

- Complement of a set A: the set of elements not contained in A,

$$A^c = \{w | w \notin A\}.$$

- Difference of two sets A and B: the set of all elements belonging to A and not contained in B,

$$A - B = \{w | w \in A, w \notin B\} = A \cap B^c.$$

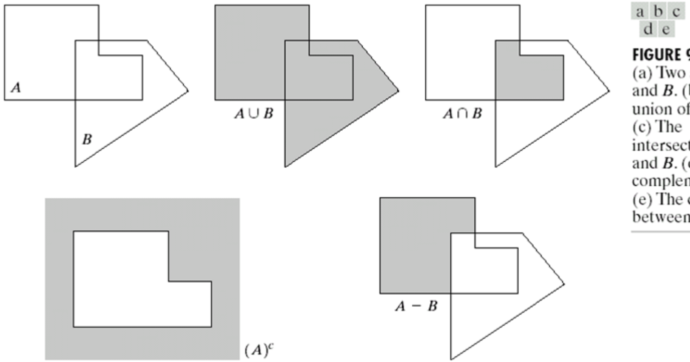


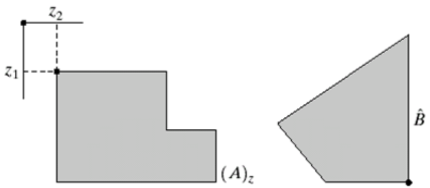
FIGURE 9.1 (a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

- Reflection of set B:

$$\hat{B} = \{w | w = -b, \text{for } b \in B\}.$$

- Translation of set A:

$$(A)_z = \{c | c = a + z, \text{for } a \in A\}.$$



a b

FIGURE 9.2 (a) Translation of A by z. (b) Reflection of B. The sets A and B are from Fig. 9.1.

Logic Operations Involving Binary Images

- All images are binary images, black indicates a binary 1 and white indicates a 0.
- Operations are limited to binary variables.
- Processing by logical functions is fast and simple.

TABLE 9.1
The three basic logical operations.

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also $\bar{p}$)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Examples

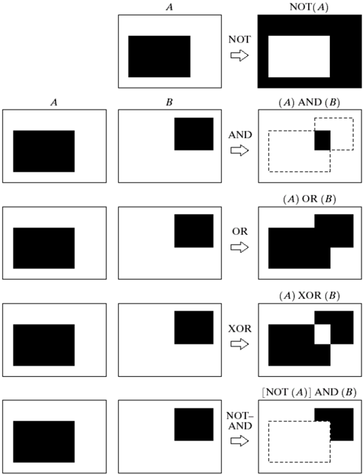


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

Black = 1
White = 0

Morphological Image Processing

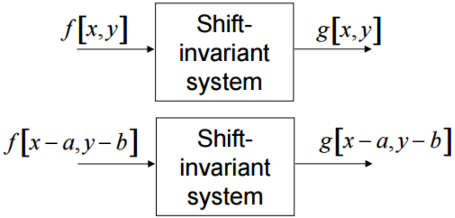
- Shift-invariant logical operations on binary images: "morphological" image processing.
  - Erosion and Dilation
  - Opening and Closing
  - Boundary Extraction
  - Region Filling
  - Skeletons
- Morphological image processing has been generalized to gray-level images via level sets.

Shift-invariance

- Assume that digital images $f[x,y]$ and $g[x,y]$ have infinite support

$$[x,y] \in \{\cdots,-2,-1,0,1,2,\cdots\} \times \{\cdots,-2,-1,0,1,2,\cdots\}$$

. . . then, for all integers a and b



- Shift-invariance does **<u>not</u>** imply linearity (or vice versa).

Structure Elements (SE)

Small sets or sub-images used to probe an image under study for properties of interest

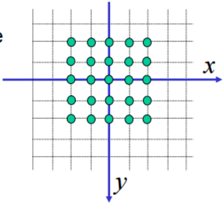
- Neighborhood "window" operator

$$W\left\{f\left[x,y\right]\right\}=\left\{f\left[x-x',y-y'\right]:\left[x',y'\right]\in\Pi_{xy}\right\}$$

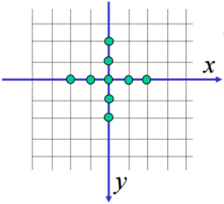
"structuring element"

- Example structuring elements Π_{xy}:

5x5 square



cross



Erosion

- Let A and B be sets in Z^2, the erosion of A by B is $A \ominus B = \{z | (B)_z \subseteq A\}$
- The set of all displacements, z, such that B, translated by z, is contained in A.

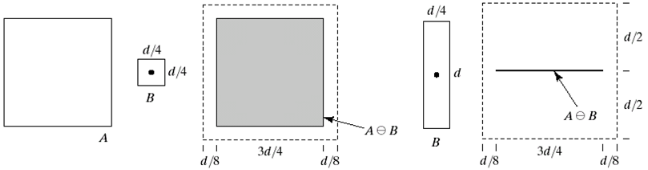
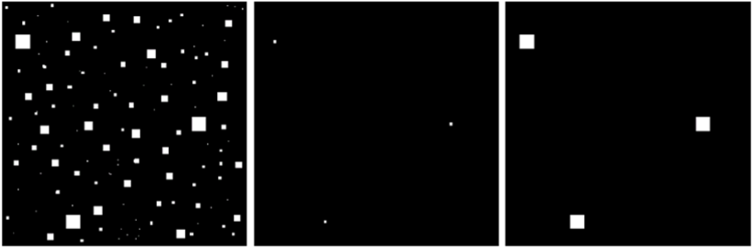


FIGURE 9.6 (a) Set A. (b) Square structuring element. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A using this element.

- Dashed lines show the original set for reference.
- Solid lines show the limit beyond which any further displacement of the origin of B by z would cause the erosion set not contained in A.
- z is in $A \ominus B$ when B is completely contained by the set A.

Erosion: application

- Eliminating irrelevant detail from a binary image.
- In the images, black indicates a binary 0 and white indicates a 1.
- All elements in the structuring element have the same binary values as the objects of interest.



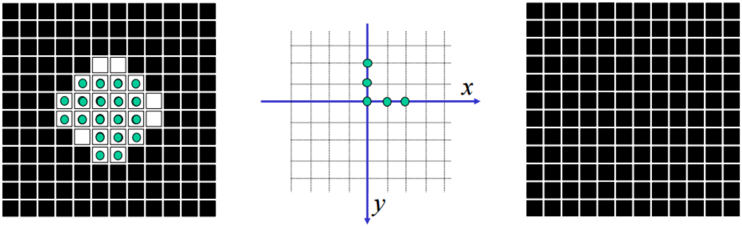
a b c

FIGURE 9.7 (a) Image of squares of size 1, 3, 5, 7, 9, and 15 pixels on the side. (b) Erosion of (a) with a square structuring element of 1's, 13 pixels on the side. (c) Dilation of (b) with the same structuring element.

Erosion: application

Binary erosion

$$g\left[x,y\right]=AND\left[W\left\{f\left[x,y\right]\right\}\right]:=erode\left(f,W\right)$$



$f[x,y]$ $\quad$ Π_{xy} $\quad$ $g[x,y]$

Dilation

- Let A and B be sets in Z^2.
- The dilation of A by B is defined as

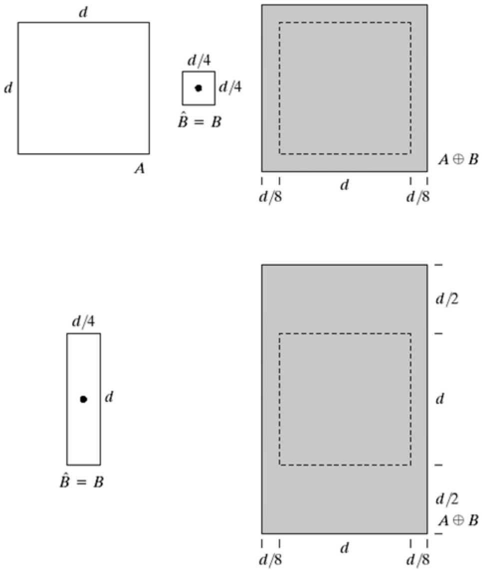
$$A \oplus B = \left\{ z | (\hat{B})_z \cap A \neq \emptyset \right\}.$$

- The set of all displacements, z, such that $\hat{B}$ and A overlap by at least one element.
- Similar to concept of convolution mask. Why?
- Another definition:

$$A \oplus B = \left\{ z | \left[(\hat{B})_z \cap A \right] \subseteq A \right\}.$$

a	b	c
d		e

FIGURE 9.4
(a) Set A.
(b) Square structuring element (dot is the center).
(c) Dilation of A by B, shown shaded.
(d) Elongated structuring element.
(e) Dilation of A using this element.



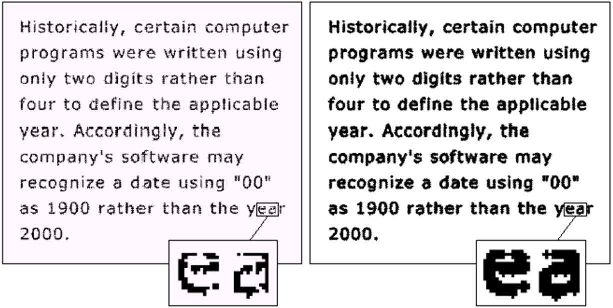
- Dashed lines (in Figs. c and e) show the original set for reference.
- Solid lines (in Figs c and e) show the limit beyond which any further displacements of the origin of $\hat{B}$ by z would cause the intersection of $\hat{B}$ and A to be empty.
- z is in $A \oplus B$ when A and $\hat{B}$ overlap by at least one element.
- Dilation and erosion are duals of each other.

$$(A \ominus B)^c = A^c \oplus \hat{B}.$$

- Dilation expands objects (represented by '1') in an image and erosion shrinks objects.

Dilation: application

Bridging gaps in broken characters:



a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

$$B = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

Relationship between dilation and erosion

Binary erosion with square structuring element

$$g[x,y] = AND\left[W\left\{f[x,y]\right\}\right] := erode(f,W)$$

Original (701x781)

erosion with
3x3 structuring element

erosion with
7x7 structuring element

- Shrinks the size of 1-valued objects
- Smoothes object boundaries
- Removes peninsulas, fingers, and small objects

Binary dilation with square structuring element

$$g[x,y] = OR\left[W\left\{f[x,y]\right\}\right] := dilate(f,W)$$

Original (701x781)

dilation with
3x3 structuring element

dilation with
7x7 structuring element

- Expands the size of 1-valued objects
- Smoothes object boundaries
- Closes holes and gap

Relationship between dilation and erosion

- Duality: erosion is dilation of the background

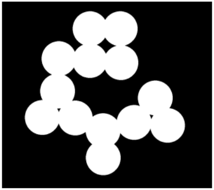
$$dilate(f,W) = NOT\left[erode\left(NOT[f],W\right)\right]$$

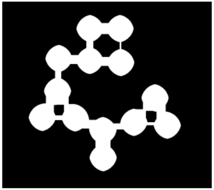
$$erode(f,W) = NOT\left[dilate\left(NOT[f],W\right)\right]$$

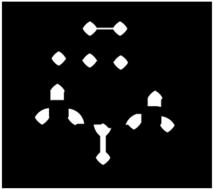
- But: erosion is <u>not</u> the inverse of dilation

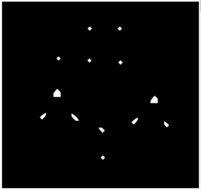
$$\begin{aligned} f[x,y] &\neq erode\left(dilate(f,W),W\right) \\ &\neq dilate\left(erode(f,W),W\right) \end{aligned}$$

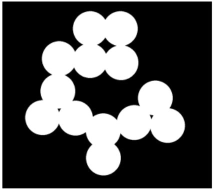
Example: blob separation/detection by erosion

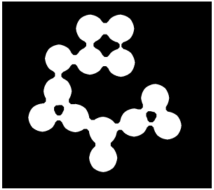

Original binary image
Circles (792x892)

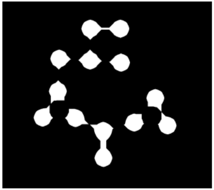

Erosion by 30x30
structuring element

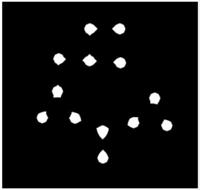

Erosion by 70x70
structuring element


Erosion by 96x96
structuring element

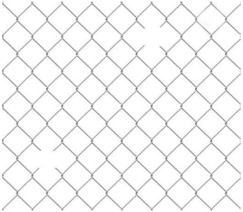

Original binary image
Circles (792x892)


Erosion by disk-shaped
structuring element
Diameter=15

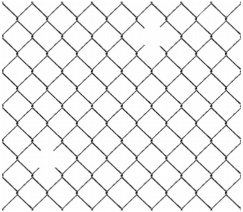

Erosion by disk-shaped
structuring element
Diameter=35


Erosion by disk-shaped
structuring element
Diameter=48

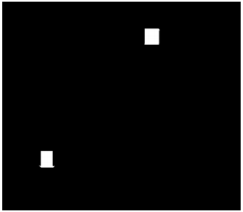
Example: chain link fence hole detection



Original grayscale image
Fence (1023 x 1173*)*



Fence thresholded
using Otsu's method



Erosion with 151x151
"cross" structuring element

Opening I

- Recall: Dilation expands an object and erosion shrinks it.
- In general, opening smoothes the object contour.

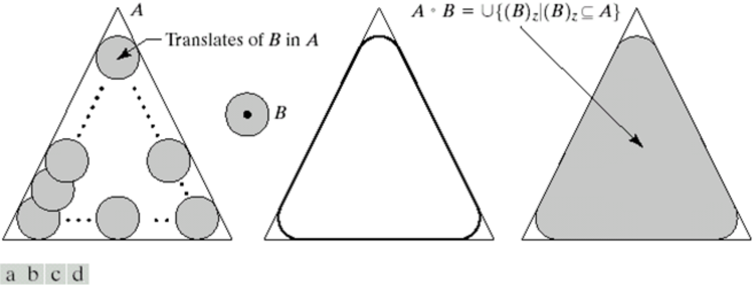


FIGURE 9.8 (a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded).

Opening II

- Definition

$$A \circ B = (A \ominus B) \oplus B.$$

- It represents erosion of A by B, followed by dilation of the result by B.
- It means that it takes the union of all translates of B that fit into A. (Imagine you roll the ball inside the object)
- Another definition

$$A \circ B = \cup \left\{ (B)_z | (B)_z \subseteq A \right\}.$$

- Subimage property: $A \circ B$ is a subset (subimage) of A.
- Convergence property

$$(A \circ B) \circ B = A \circ B.$$

Closing I

- In general, closing also smoothes the object contour.

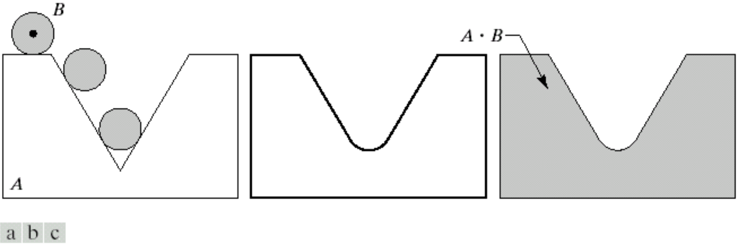


FIGURE 9.9 (a) Structuring element B "rolling" on the outer boundary of set A. (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).

- Definition

$$A \bullet B = (A \oplus B) \ominus B.$$

Closing II

- It represents dilation of A by B, followed by erosion of the result by B.
- Imagine that you roll the ball outside the object instead of rolling the ball inside the object.
- Opening and closing are dual

$$(A \bullet B)^c = A^c \circ \hat{B}.$$

- Subimage property: A is a subset (subimage) of $A \bullet B$.
- Convergence property

$$(A \bullet B) \bullet B = A \bullet B.$$

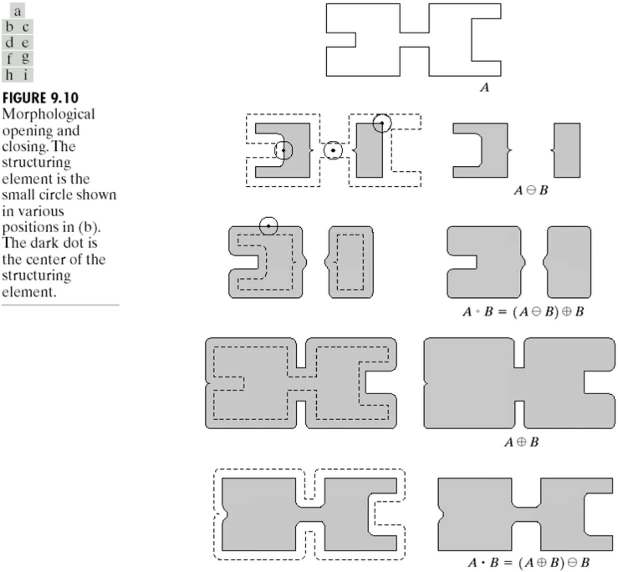
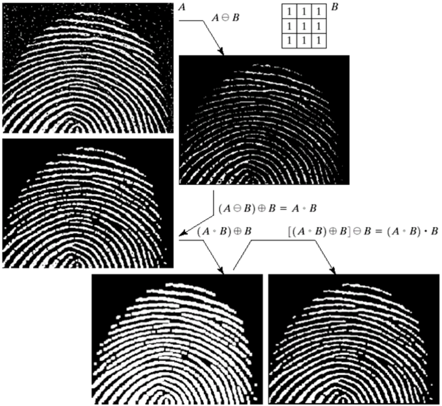


FIGURE 9.10 Morphological opening and closing. The structuring element is the small circle shown in various positions in (b). The dark dot is the center of the structuring element.



a b c d e f

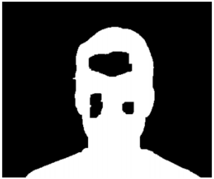
FIGURE 9.11
(a) Noisy image. (c) Eroded image. (d) Opening of A. (d) Dilation of the opening. (e) Closing of the opening. (Original image for this example courtesy of the National Institute of Standards and Technology.)

Duality of Opening and Closing

- Opening generally smoothes the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions.
- Closing tends to smooth sections of contours but it generates fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour.
- Opening and closing are duals of each other with respect to set complementation and reflection.
- Example: Small hole removal by closing


Original binary mask


Dilation


Closing 10x10


Difference to original mask

Boundary Extraction

- $\beta(A)$ denotes the boundary of a set A.
- The boundary is extracted by using

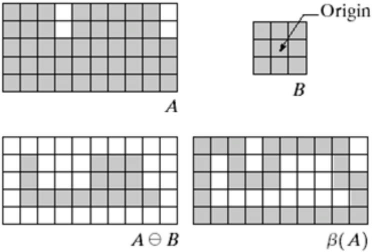
$$\beta(A) = A - (A \ominus B).$$

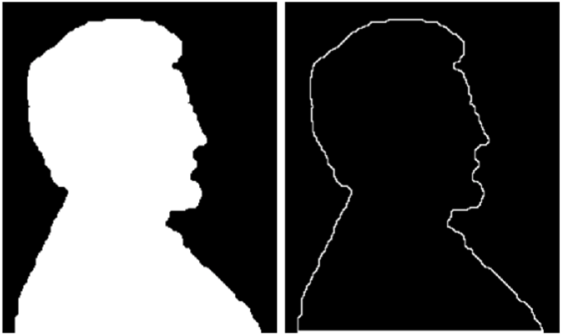
- The boundary is the difference between the object and the "eroded" object.

Shaded region = 1, white region = 0

a	b
c	d

FIGURE 9.13 (a) Set A. (b) Structuring element B. (c) A eroded by B. (d) Boundary, given by the set difference between A and its erosion.





a b

FIGURE 9.14
(a) A simple binary image, with 1's represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

Region Filling

- Beginning with a point p inside the boundary, the objective is to fill the entire region with 1s.
- The algorithm is defined as follows.

$$X_k = (X_{k-1} \oplus B) \cap A^c.$$

- k=1,2,3,...
- $X_0 = p$
- B is the symmetric structuring element.
- Intersection with A^c constrains the result to be inside the region of interest.
- The algorithm stops when $X_k = X_{k-1}$. This means the algorithm stops when there is no change in the region size.

Shaded region = 1, white region = 0

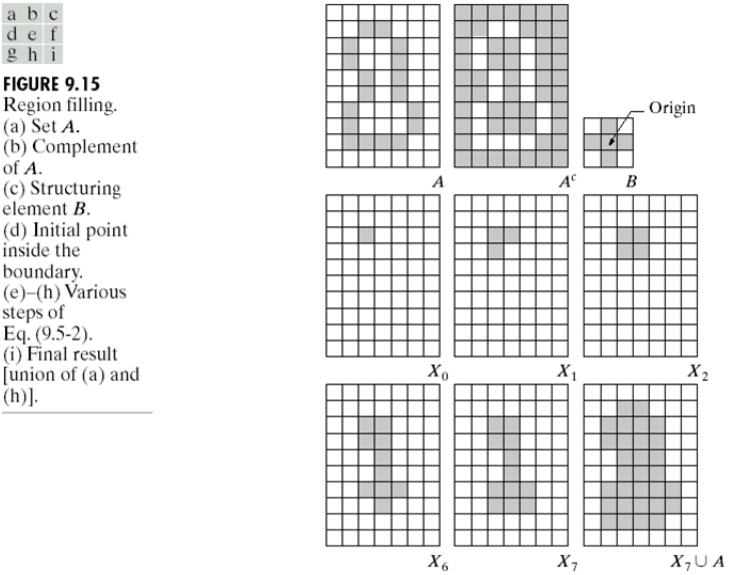


FIGURE 9.15 Region filling. (a) Set A. (b) Complement of A. (c) Structuring element B. (d) Initial point inside the boundary. (e)–(h) Various steps of Eq. (9.5-2). (i) Final result [union of (a) and (h)].

a b c

FIGURE 9.16 (a) Binary image (the white dot inside one of the regions is the starting point for the region-filling algorithm). (b) Result of filling that region (c) Result of filling all regions.

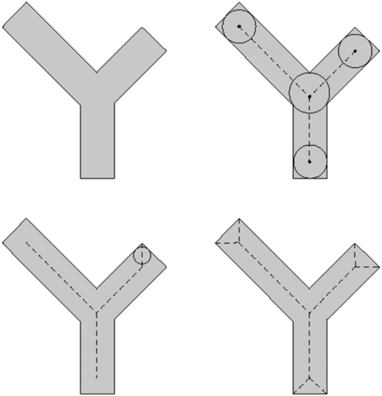
Skeletons I

- Definition of a skeleton, $S(A)$, of a set A is as follows
  - If
    - z is a point of $S(A)$ and
    - $(D)_z$ is the largest disk centered at z and contained in A,
  - then
    - one cannot find a larger disk (not necessarily centered at z) containing $(D)_z$ and included in A, and
    - the disk $(D)_z$ is called *maximum disk*.
    - the disk $(D)_z$ touches the boundary of A at two or more different places.

Skeletons II

a b
c d

FIGURE 9.23
(a) Set A.
(b) Various positions of maximum disks with centers on the skeleton of A.
(c) Another maximum disk on a different segment of the skeleton of A.
(d) Complete skeleton.



Skeletons III

- The algorithm is given by

$$S(A) = \bigcup_{k=0}^{K} S_k(A).$$

with

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B.$$

where B is a structuring element, and

$$(A \ominus kB) = (\ldots((A \ominus B) \ominus B) \ldots) \ominus B.$$

k successive erosions of A by B.

- K is the last iterative step before A erodes to an empty set.

$$K = \max\left\{k | (A \ominus kB) \neq \emptyset\right\}.$$

Reconstruction from skeletons

- A can be reconstructed by using the equation,

$$A = \bigcup_{k=0}^{K} \left(S_k(A) \oplus kB \right),$$

where

$$(S_k(A) \oplus kB) = (\ldots((S_k(A) \oplus B) \oplus B) \oplus \ldots) \oplus B,$$

k successive dilations of $S_k(A)$.

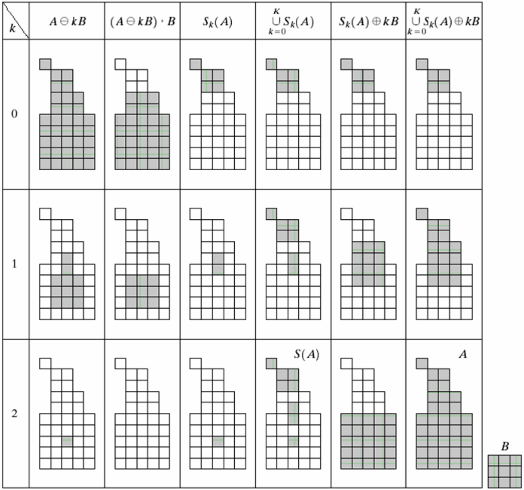


FIGURE 9.24 Implementation of Eqs. (9.5-11) through (9.5-15). The original set is at the top left, and its morphological skeleton is at the bottom of the fourth column. The reconstructed set is at the bottom of the sixth column.

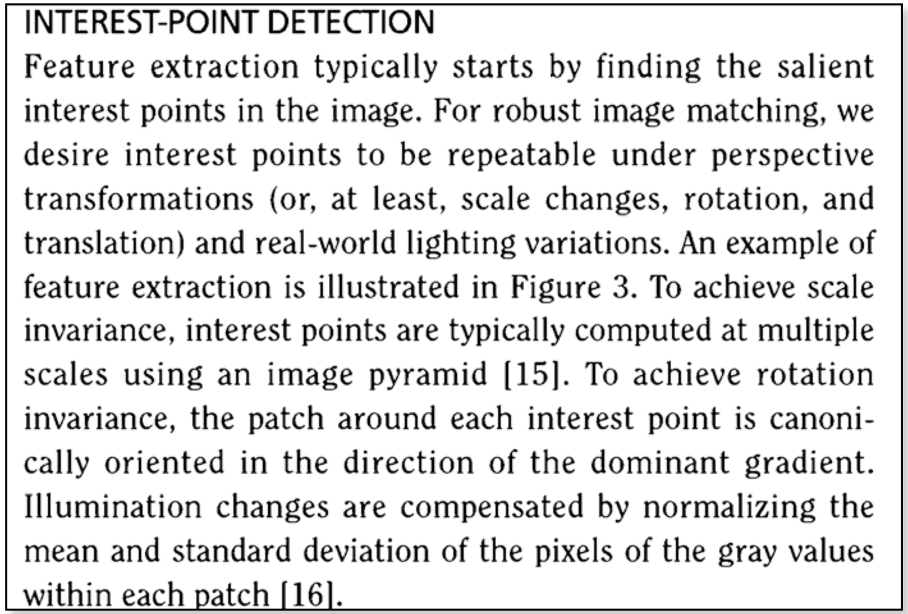
Supplementary Material
on Morphological Image Processing

From Bernd Girod

Recognition by erosion

Binary image f

1400

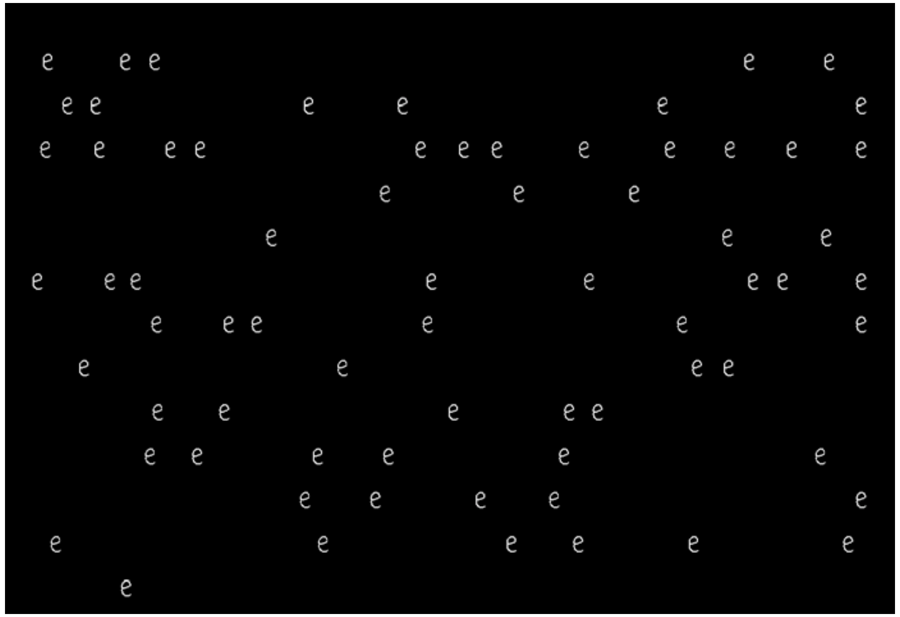


INTEREST-POINT DETECTION

Feature extraction typically starts by finding the salient interest points in the image. For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. An example of feature extraction is illustrated in Figure 3. To achieve scale invariance, interest points are typically computed at multiple scales using an image pyramid [15]. To achieve rotation invariance, the patch around each interest point is canonically oriented in the direction of the dominant gradient. Illumination changes are compensated by normalizing the mean and standard deviation of the pixels of the gray values within each patch [16].

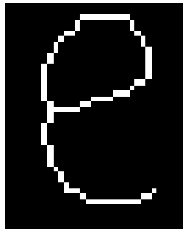
2000

$$open\left(NOT\left[f\right],W\right)=dilate\left(erode\left(NOT\left[f\right],W\right),W\right)$$



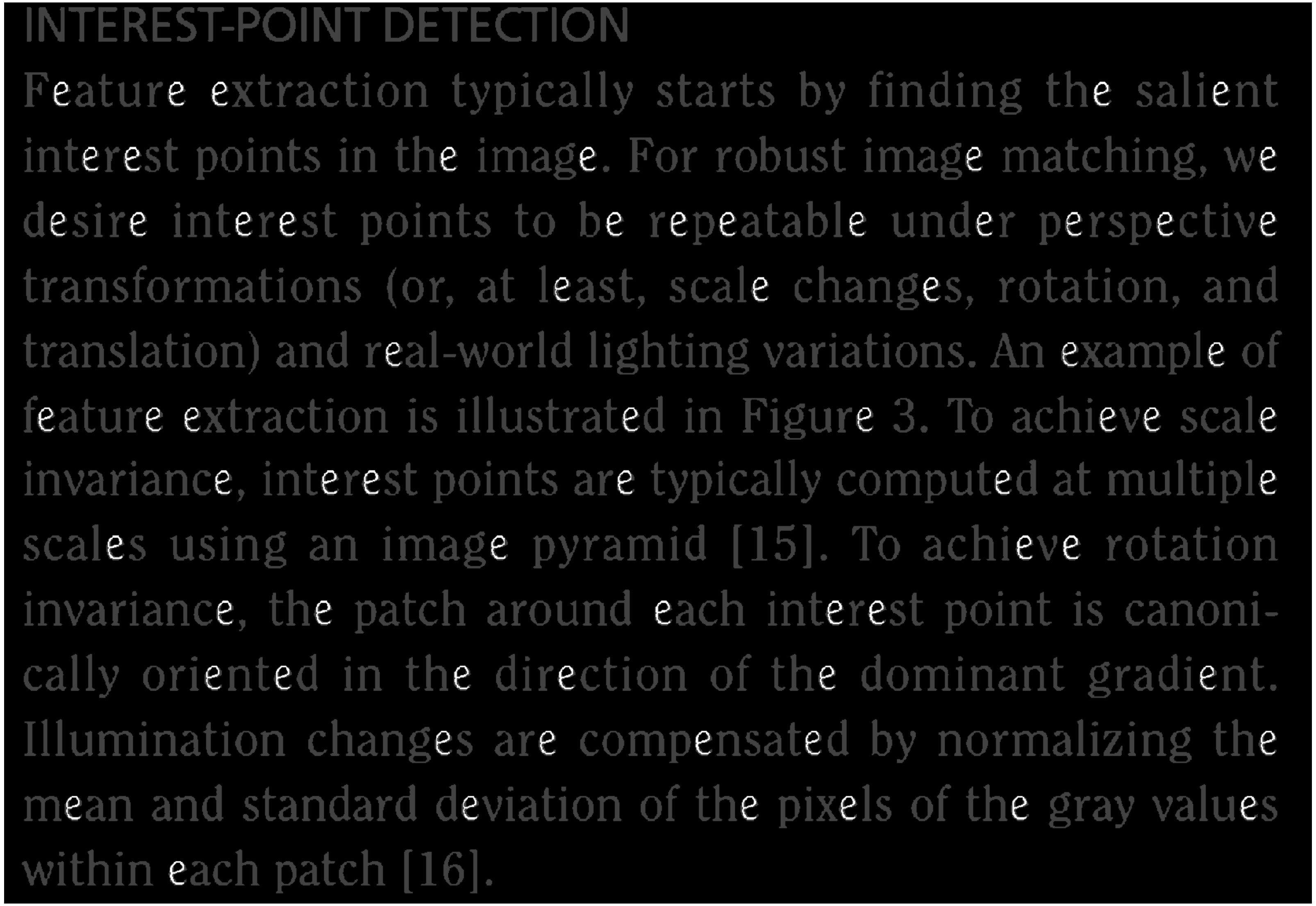
Structuring element W

44



34

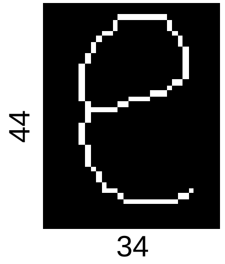
Recognition by erosion



INTEREST-POINT DETECTION

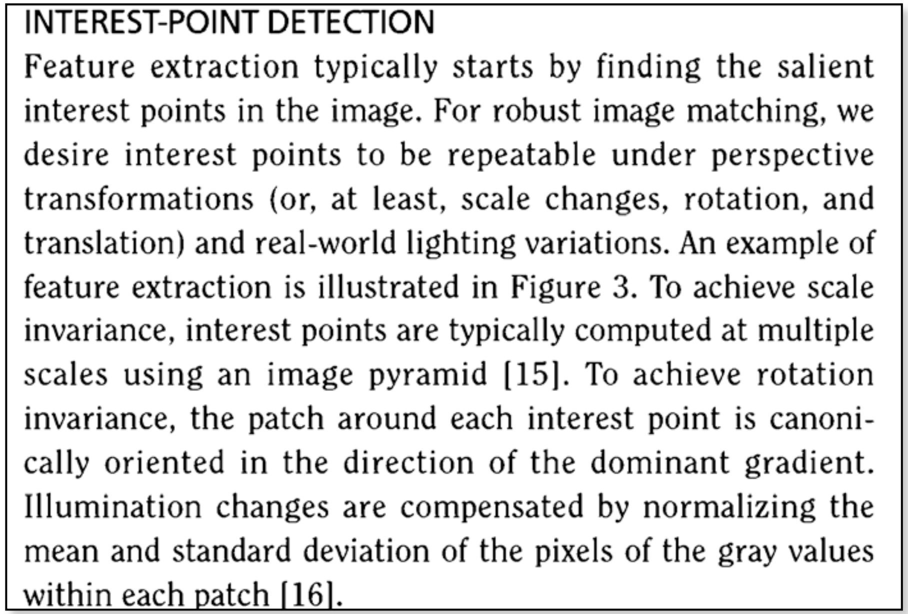
Feature extraction typically starts by finding the salient interest points in the image. For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. An example of feature extraction is illustrated in Figure 3. To achieve scale invariance, interest points are typically computed at multiple scales using an image pyramid [15]. To achieve rotation invariance, the patch around each interest point is canonically oriented in the direction of the dominant gradient. Illumination changes are compensated by normalizing the mean and standard deviation of the pixels of the gray values within each patch [16].

Structuring element W



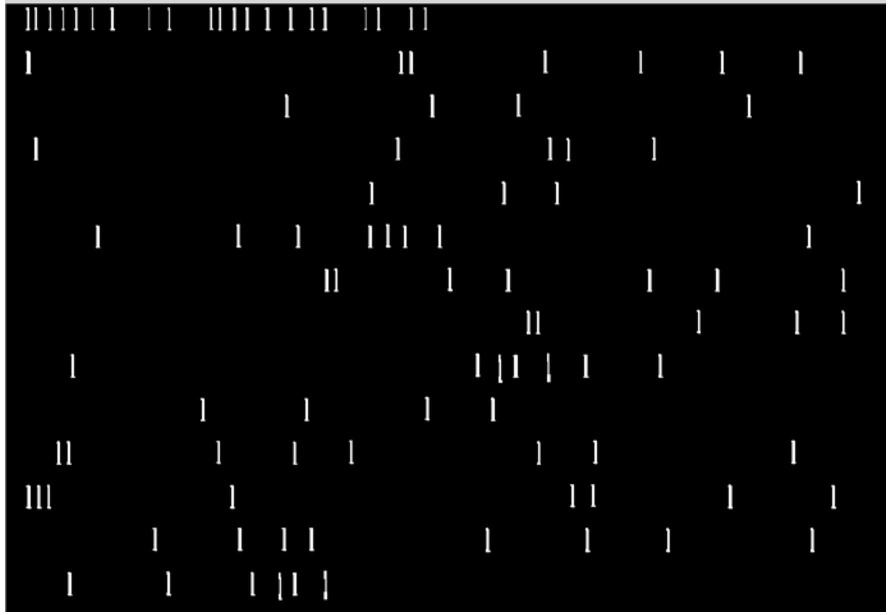
Recognition by erosion

Binary image f

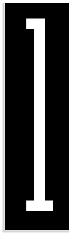

INTEREST-POINT DETECTION

Feature extraction typically starts by finding the salient interest points in the image. For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. An example of feature extraction is illustrated in Figure 3. To achieve scale invariance, interest points are typically computed at multiple scales using an image pyramid [15]. To achieve rotation invariance, the patch around each interest point is canonically oriented in the direction of the dominant gradient. Illumination changes are compensated by normalizing the mean and standard deviation of the pixels of the gray values within each patch [16].

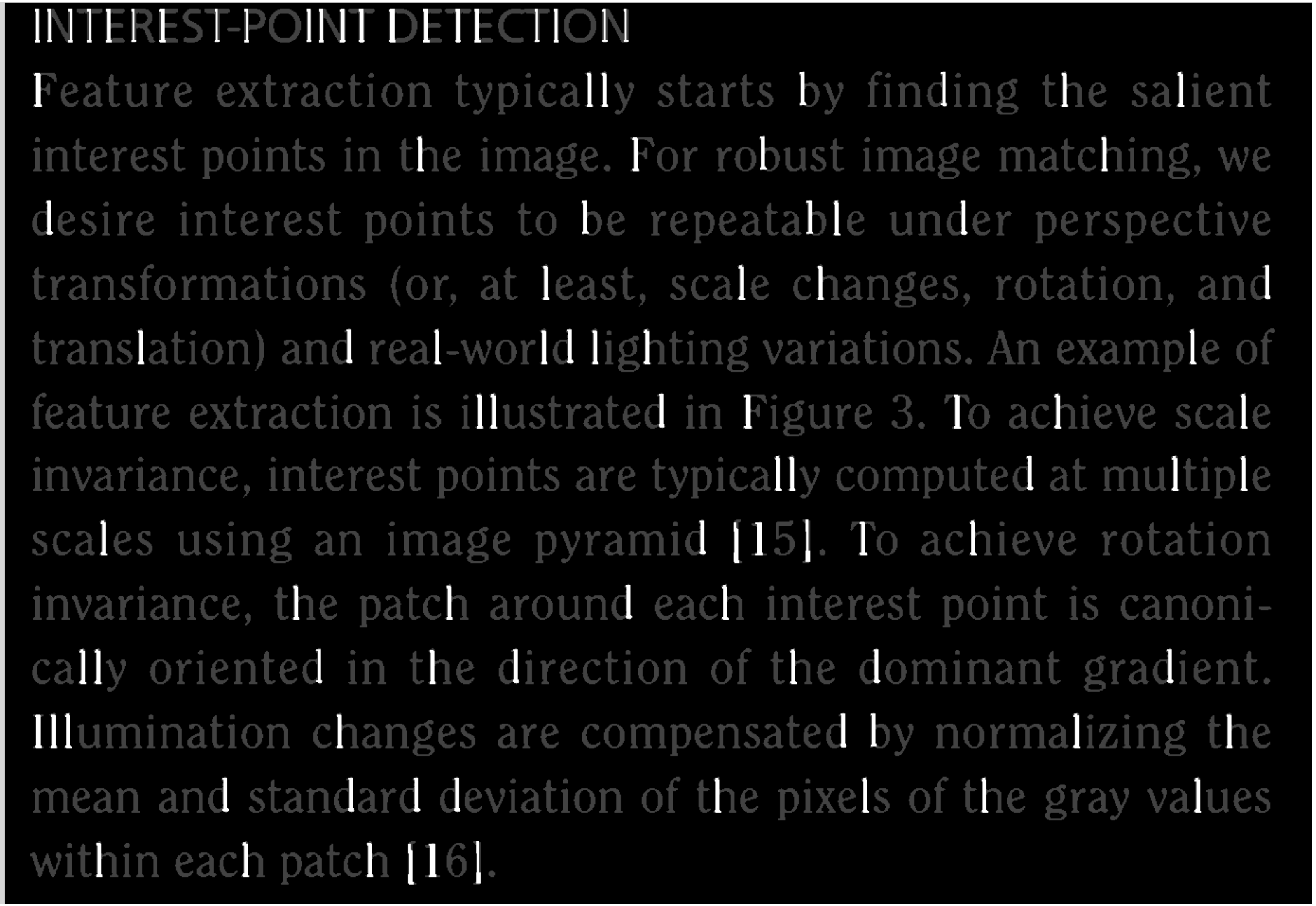
$$open\left(NOT\left[f\right],W\right) = dilate\left(erode\left(NOT\left[f\right],W\right),W\right)$$



Structuring element W



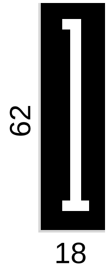
Recognition by erosion



INTEREST-POINT DETECTION

Feature extraction typically starts by finding the salient interest points in the image. For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. An example of feature extraction is illustrated in Figure 3. To achieve scale invariance, interest points are typically computed at multiple scales using an image pyramid [15]. To achieve rotation invariance, the patch around each interest point is canonically oriented in the direction of the dominant gradient. Illumination changes are compensated by normalizing the mean and standard deviation of the pixels of the gray values within each patch [16].

Structuring element W

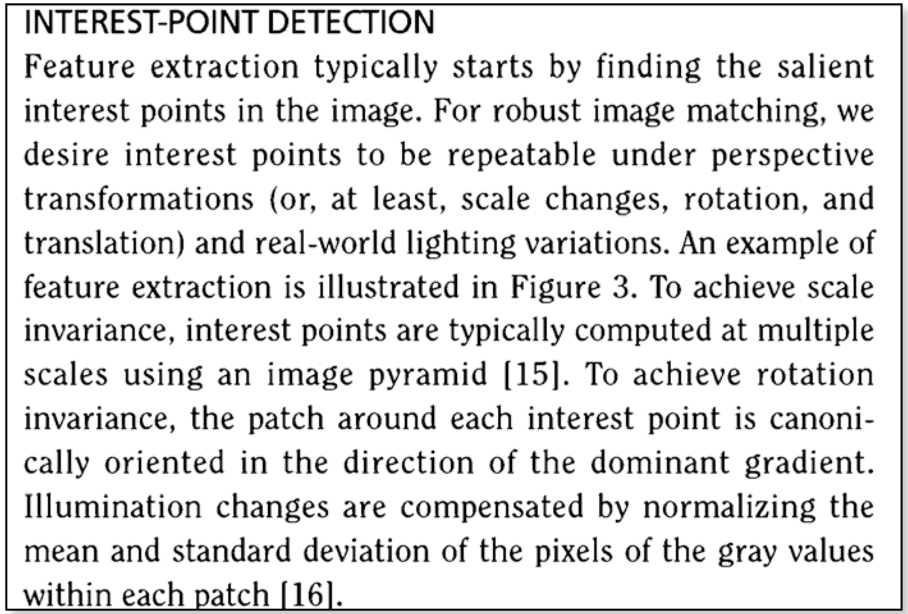


Hit-miss filter

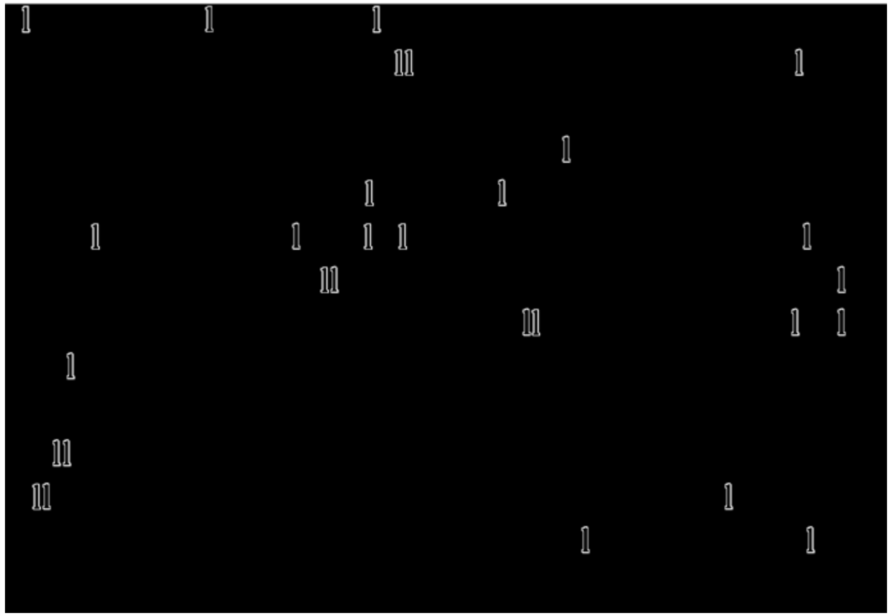
Binary image f

$$dilate\left(erode\left(NOT\left[f\right],V\right)\&erode\left(f,W\right),W\right)$$

1400


INTEREST-POINT DETECTION

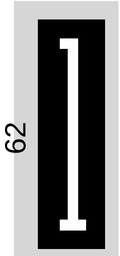
Feature extraction typically starts by finding the salient interest points in the image. For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. An example of feature extraction is illustrated in Figure 3. To achieve scale invariance, interest points are typically computed at multiple scales using an image pyramid [15]. To achieve rotation invariance, the patch around each interest point is canonically oriented in the direction of the dominant gradient. Illumination changes are compensated by normalizing the mean and standard deviation of the pixels of the gray values within each patch [16].



2000

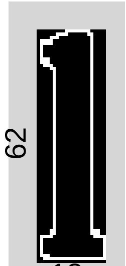
Structuring element V

62



18

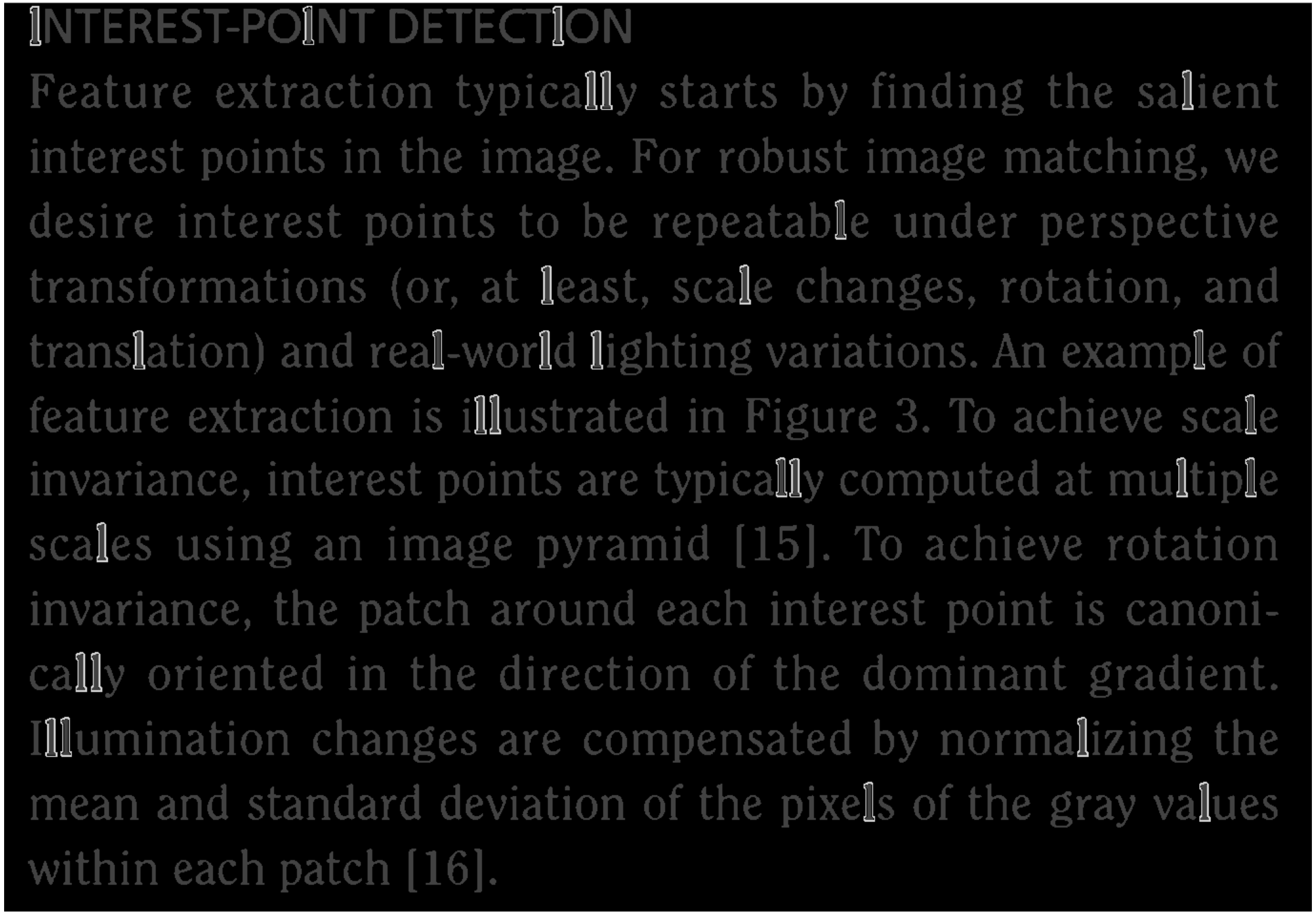
62



18

Structuring element W

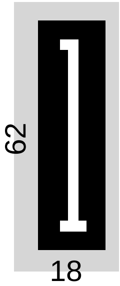
Hit-miss filter



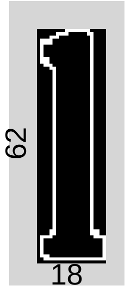
INTEREST-POINT DETECTION

Feature extraction typically starts by finding the salient interest points in the image. For robust image matching, we desire interest points to be repeatable under perspective transformations (or, at least, scale changes, rotation, and translation) and real-world lighting variations. An example of feature extraction is illustrated in Figure 3. To achieve scale invariance, interest points are typically computed at multiple scales using an image pyramid [15]. To achieve rotation invariance, the patch around each interest point is canonically oriented in the direction of the dominant gradient. Illumination changes are compensated by normalizing the mean and standard deviation of the pixels of the gray values within each patch [16].

Structuring element V



Structuring element W



Morphological filters for gray-level images

- Threshold sets of a gray-level image $f[x,y]$

$$\mathrm{T}_\theta\left(f\left[x,y\right]\right)=\left\{\left[x,y\right]:f\left[x,y\right]\geq\theta\right\},\qquad -\infty<\theta<+\infty$$

- Reconstruction of original image from threshold sets

$$f\left[x,y\right]=\sup\left\{\theta:\left[x,y\right]\in\mathrm{T}_\theta\left(f\left[x,y\right]\right)\right\}$$

- Idea of morphological operators for multi-level (or continuous-amplitude) signals
  - Decompose into threshold sets
  - Apply binary morphological operator to each threshold set
  - Reconstruct via supremum operation
  - Gray-level operators thus obtained: ***flat operators***

➔ *Flat morphological operators and thresholding are commutative*

Dilation/erosion for gray-level images

- Explicit decomposition into threshold sets not required in practice
- Flat dilation operator: local maximum over window W

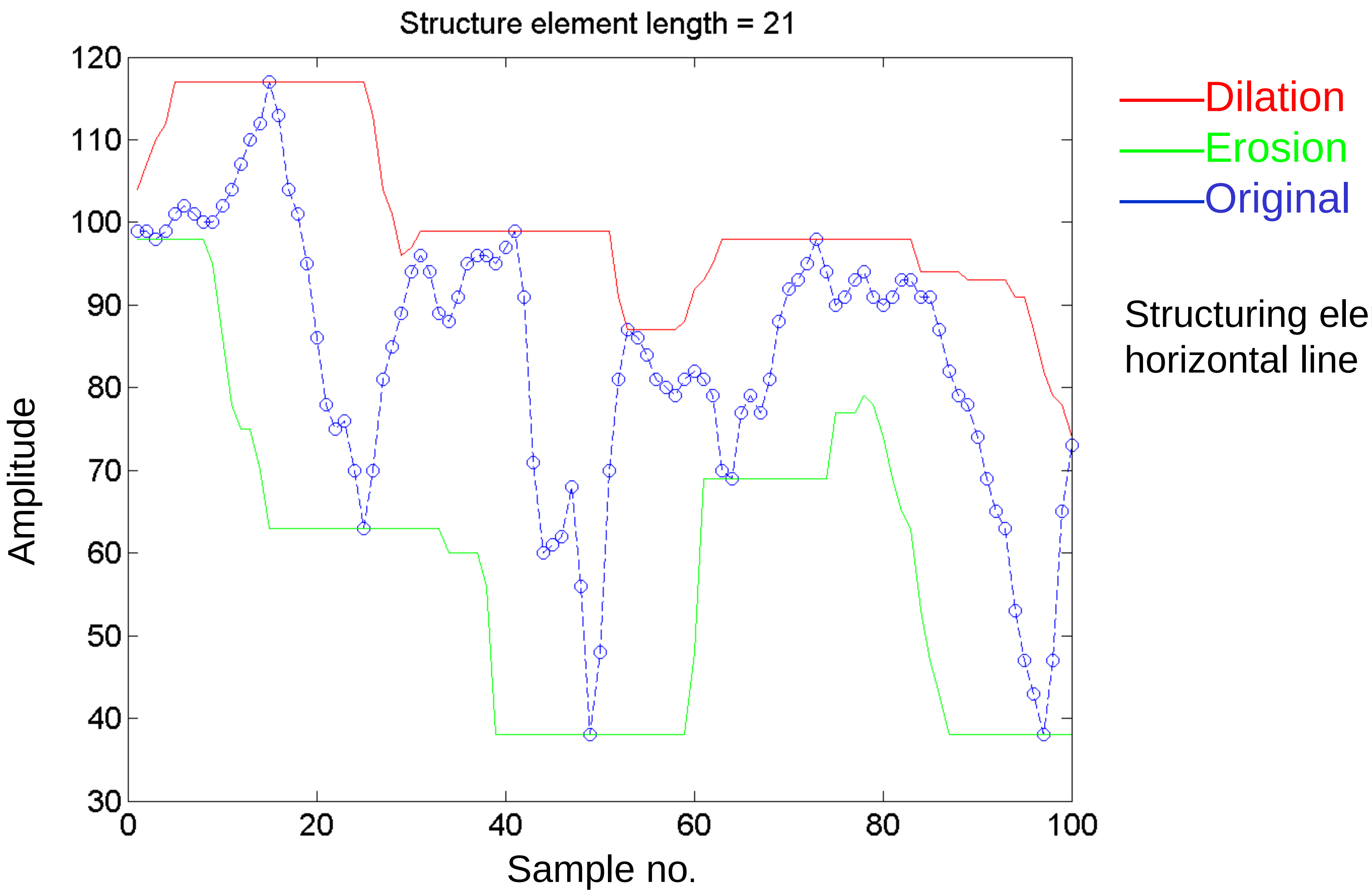
$$g[x,y] = \max\left\{W\left\{f[x,y]\right\}\right\} := dilate(f,W)$$

- Flat erosion operator: local minimum over window W

$$g[x,y] = \min\left\{W\left\{f[x,y]\right\}\right\} := erode(f,W)$$

- Binary dilation/erosion operators contained as special case

1-d illustration of erosion and dilation



Structuring element:
horizontal line

Image example

Original

Dilation

Erosion

Flat dilation with different structuring elements

Original

Diamond

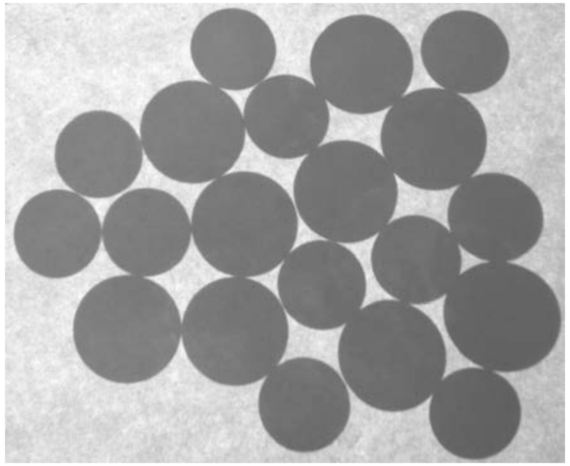
Disk

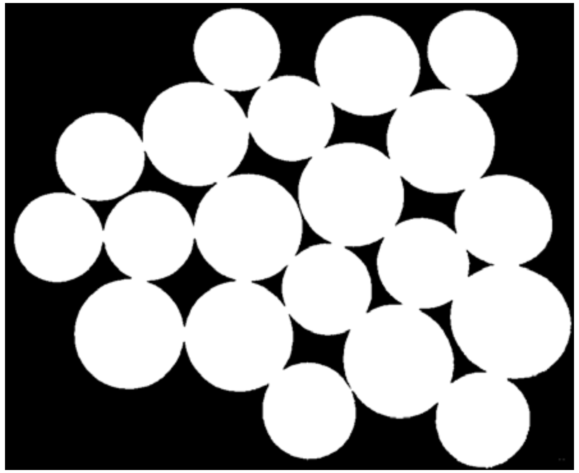
20 degree line

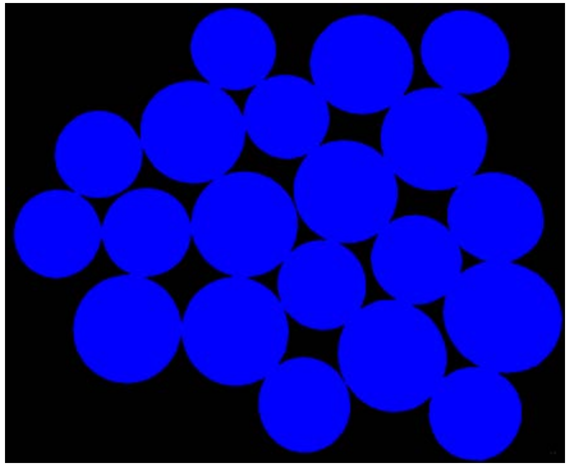
9 points

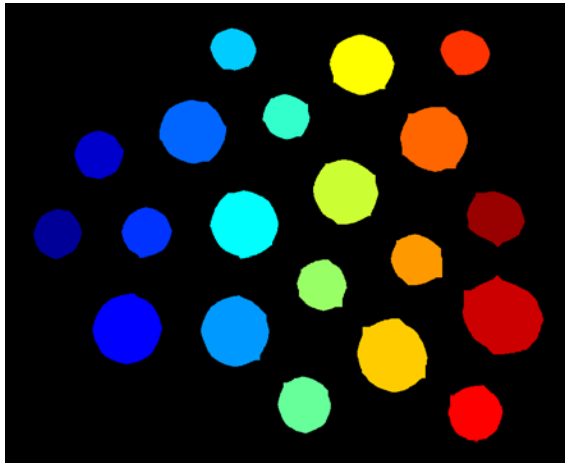
2 horizontal lines

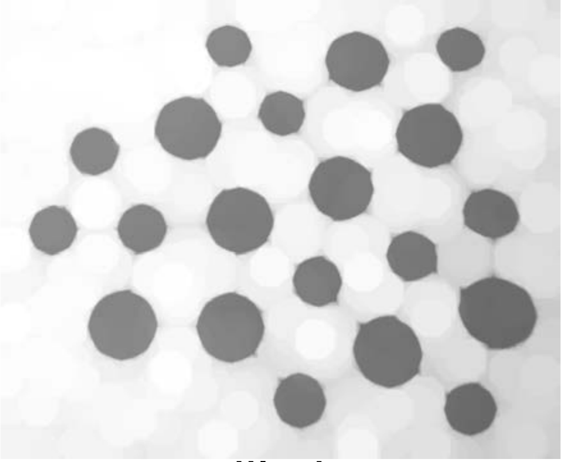
Example: counting coins

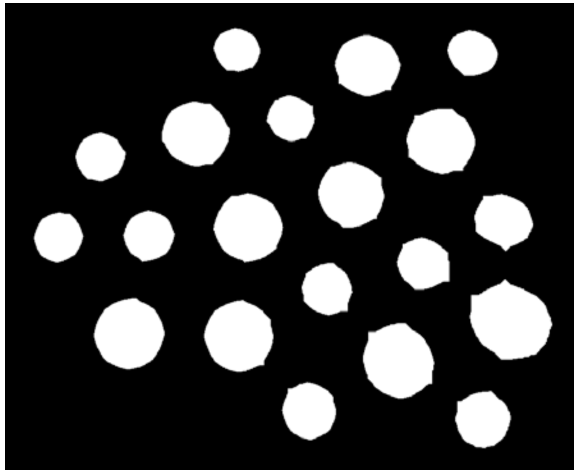

Original


thresholded

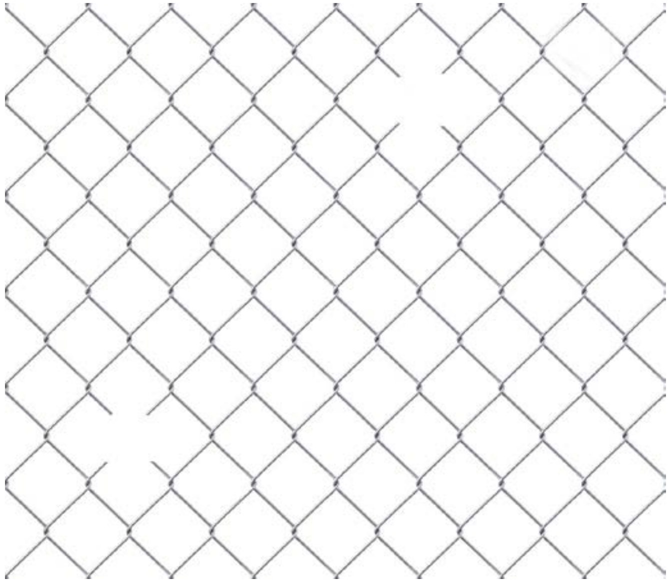

1 connected component


20 connected components

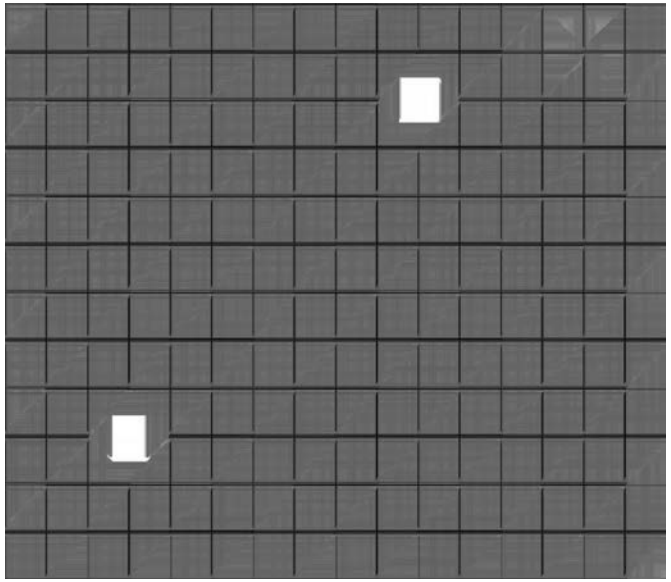

dilation


thresholded after dilation

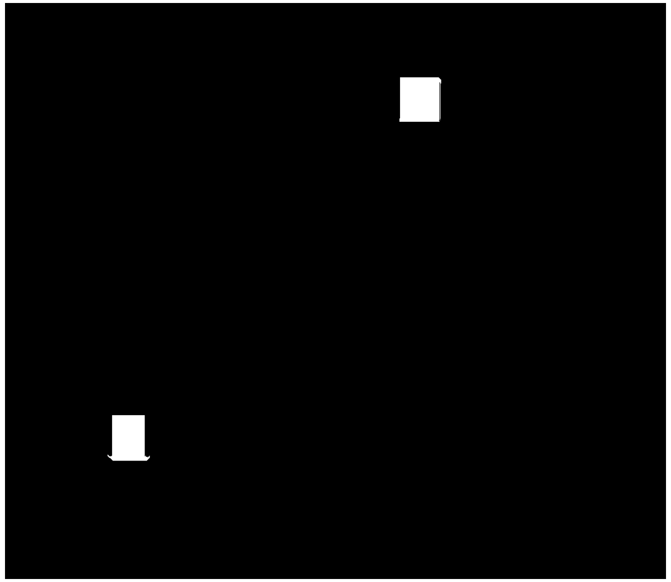
Example: chain link fence hole detection



Original grayscale image
Fence (1023 x 1173*)*



Flat erosion with 151x151
“cross” structuring element



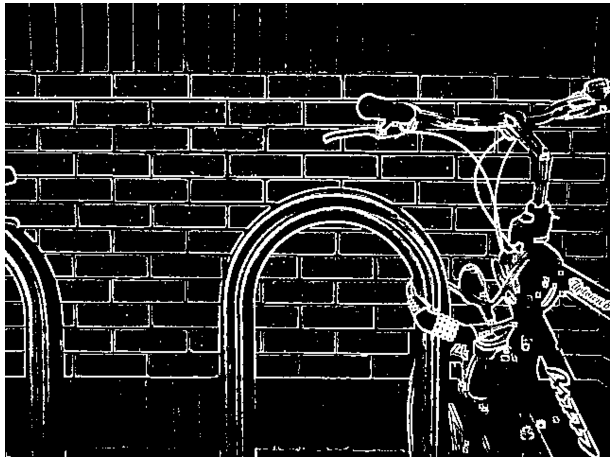
Binarized by Thresholding

Morphological edge detector


original f


dilation g


g-f


$(g$-$f)$ thresholded

Beyond flat morphological operators

- General dilation operator

$$g[x,y] = \sup_{\alpha,\beta}\{f[x-\alpha, y-\beta] + w[\alpha,\beta]\} = \sup_{\alpha,\beta}\{w[x-\alpha, y-\beta] + f[\alpha,\beta]\}$$

- Like linear convolution, with sup replacing summation, addition replacing multiplication
- Dilation with “unit impulse”

$$d[\alpha,\beta] = \begin{cases} 0 & \alpha = \beta = 0 \\ -\infty & \text{else} \end{cases}$$

does not change input signal:

$$f[x,y] = \sup_{\alpha,\beta}\{f[x-\alpha, y-\beta] + d[\alpha,\beta]\}$$

Flat dilation as a special case

- Find $w[\alpha,\beta]$ such that

$$f[x,y] = \sup_{\alpha,\beta}\left\{f[x-\alpha,y-\beta]+w[\alpha,\beta]\right\} = dilate(f,W)$$

- Answer:

$$w[\alpha,\beta] = \begin{cases} 0 & [\alpha,\beta]\in\Pi_{xy} \\ -\infty & \text{else} \end{cases}$$

- Hence, write in general

$$g[x,y] = \sup_{\alpha,\beta}\left\{f[x-\alpha,y-\beta]+w[\alpha,\beta]\right\}$$
$$= dilate(f,w) = dilate(w,f)$$

General erosion for gray-level images

- General erosion operator

$$g[x,y] = \inf_{\alpha,\beta}\left\{f[x-\alpha, y-\beta] - w[\alpha,\beta]\right\} = erode(f,w)$$

- Dual of dilation

$$g[x,y] = \inf_{\alpha,\beta}\left\{f[x-\alpha, y-\beta] - w[\alpha,\beta]\right\}$$

$$= -\sup_{\alpha,\beta}\left\{-f[x-\alpha, y-\beta] + w[\alpha,\beta]\right\} = -dilate(-f,w)$$

- Flat erosion contained as a special case

Cascaded dilations

f

$dilate(f, w_1)$

$dilate(f, w_1, w_2)$

$dilate(f, w_1, w_2, w_3)$

$$dilate\left[dilate(f, w_1), w_2\right] = dilate(f, w)$$

$$\text{where } w = dilate(w_1, w_2)$$

Cascaded erosions

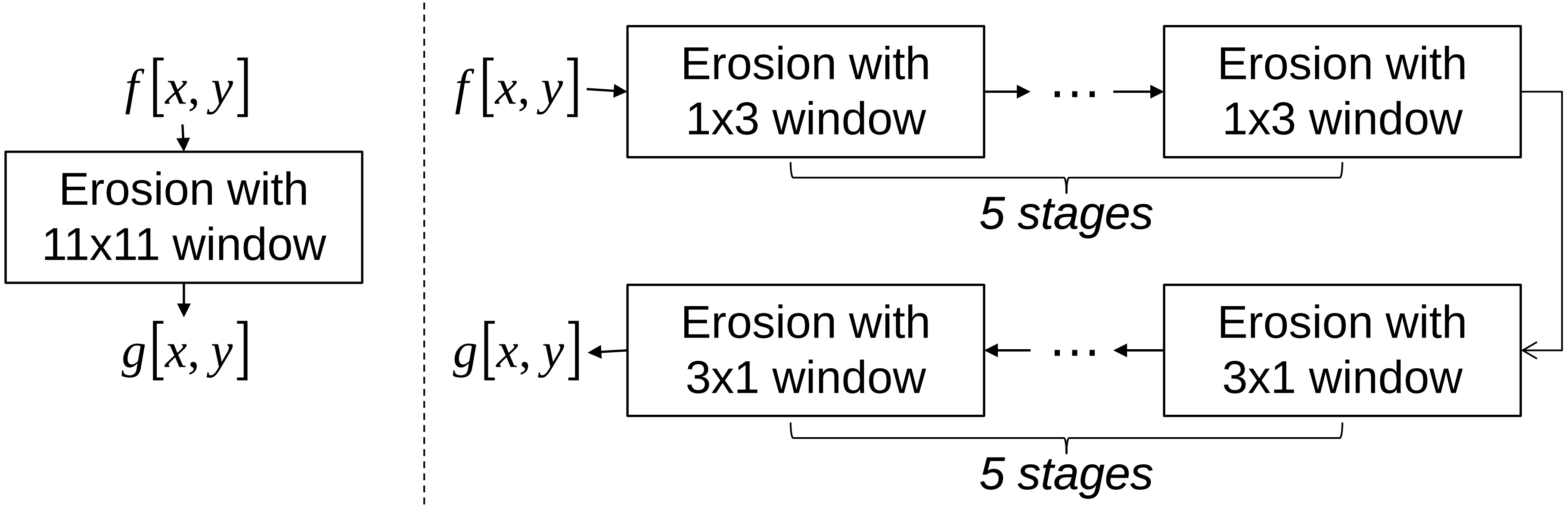
- Cascaded erosions can be lumped into single erosion

$$\begin{aligned} erode\left[erode\left(f,w_1\right),w_2\right] &= erode\left[-dilate\left(-f,w_1\right),w_2\right] \\ &= -dilate\left[dilate\left(-f,w_1\right),w_2\right] \\ &= -dilate\left(-f,w\right) \\ &= erode\left(f,w\right) \\ \text{where} \quad w &= dilate\left(w_1,w_2\right) \end{aligned}$$

- New structuring element (SE) is <u>not</u> the erosion of one SE by the other, but dilation.

Fast dilation and erosion

- Idea: build larger dilation and erosion operators by cascading simple, small operators
- Example: binary erosion by 11x11 window



120 AND per pixel

2x10 = 20 AND per pixel

Rank filters

- Generalisation of flat dilation/erosion: in lieu of min or max value in window, use the p-th ranked value
- Increases robustness against noise
- Best-known example: median filter for noise reduction
- Concept useful for both gray-level and binary images
- All rank filters are commutative with thresholding

Median filter

- Gray-level median filter

$$g\left[x,y\right]=median\left[W\left\{f\left[x,y\right]\right\}\right]:=median\left(f,W\right)$$

- Binary images: majority filter

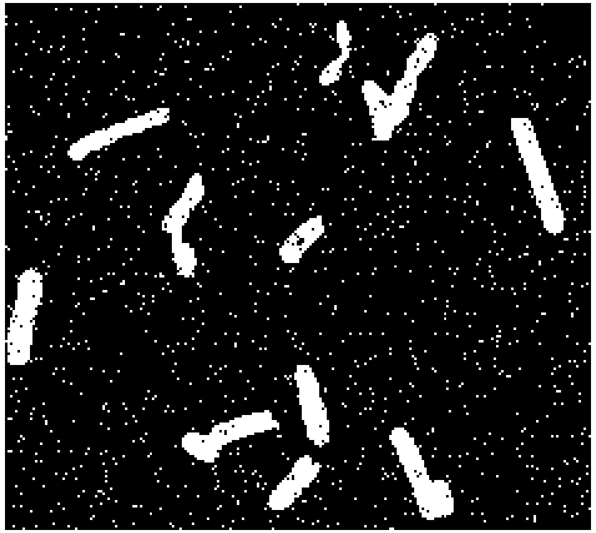
$$g\left[x,y\right]=MAJ\left[W\left\{f\left[x,y\right]\right\}\right]:=majority\left(f,W\right)$$

- Self-duality

$$median\left(f,W\right)=-\left[median\left(-f,W\right)\right]$$

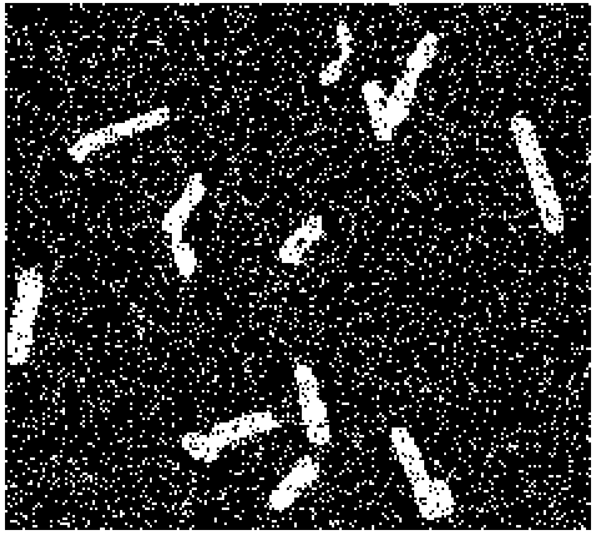
$$majority\left(f,W\right)=NOT\left[majority\left(NOT\left[f\right],W\right)\right]$$

Majority filter: example



Binary image with
5% ‘Salt&Pepper’ noise

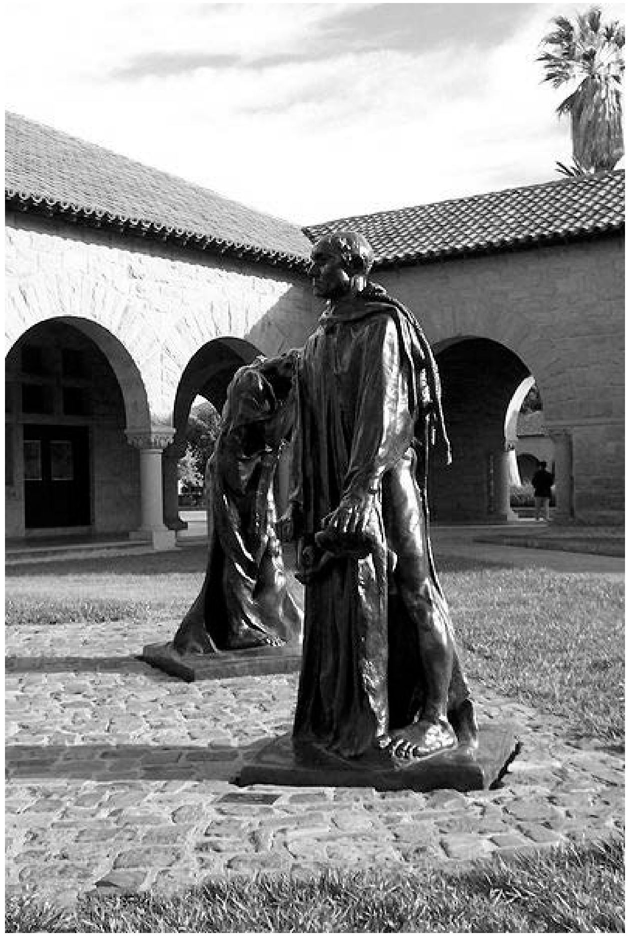
3x3 majority filter



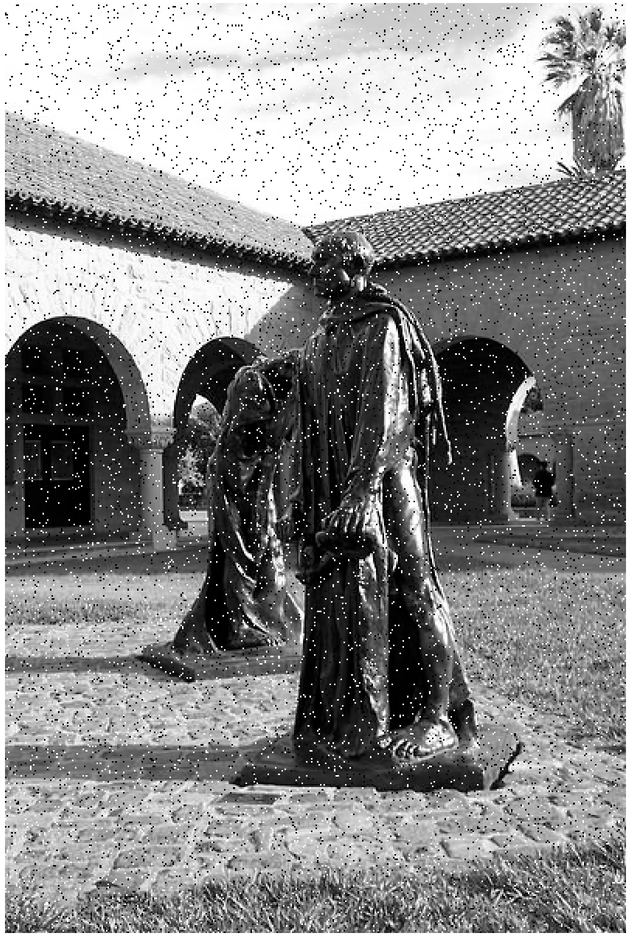
20% ‘Salt&Pepper’ noise

3x3 majority filter

Median filter: example



Original image

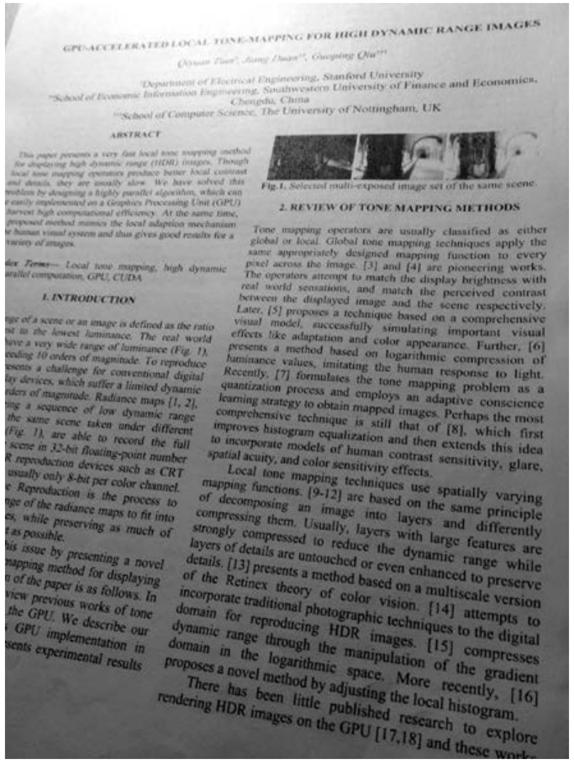


5% ‘Salt&Pepper’ noise

3x3 median filtering

7x7 median filtering

Example: non-uniform lighting compensation



GPU-ACCELERATED LOCAL TONE-MAPPING FOR HIGH DYNAMIC RANGE IMAGES

Department of Electrical Engineering, Stanford University
School of Economic Information Engineering, Southwestern University of Finance and Economics, Chengdu, China
School of Computer Science, The University of Nottingham, UK

ABSTRACT

This paper presents a very fast local tone mapping method for displaying high dynamic range (HDR) images. Though local tone mapping operators produce better local contrast and details, they are usually slow. We have solved this

Fig.1. Selected multi-exposed image set of the same scene.

2. REVIEW OF TONE MAPPING METHODS

Tone mapping operators are usually classified as either global or local. Global tone mapping techniques apply the same appropriately designed mapping function to every pixel across the image. [3] and [4] are pioneering works. The operators attempt to match the display brightness with real world sensations, and match the perceived contrast between the displayed image and the scene respectively. Later, [5] proposes a technique based on a comprehensive visual model, successfully simulating important visual effects like adaptation and color appearance. Further, [6] presents a method based on logarithmic compression of luminance values, imitating the human response to light. Recently, [7] formulates the tone mapping problem as a quantization process and employs an adaptive conscience learning strategy to obtain mapped images. Perhaps the most comprehensive technique is still that of [8], which first improves histogram equalization and then extends this idea to incorporate models of human contrast sensitivity, glare, spatial acuity, and color sensitivity effects.

Local tone mapping techniques use spatially varying mapping functions. [9-12] are based on the same principle of decomposing an image into layers and differently compressing them. Usually, layers with large features are strongly compressed to reduce the dynamic range while layers of details are untouched or even enhanced to preserve details. [13] presents a method based on a multiscale version of the Retinex theory of color vision. [14] attempts to incorporate traditional photographic techniques to the digital domain for reproducing HDR images. [15] compresses dynamic range through the manipulation of the gradient domain in the logarithmic space. More recently, [16] proposes a novel method by adjusting the local histogram.

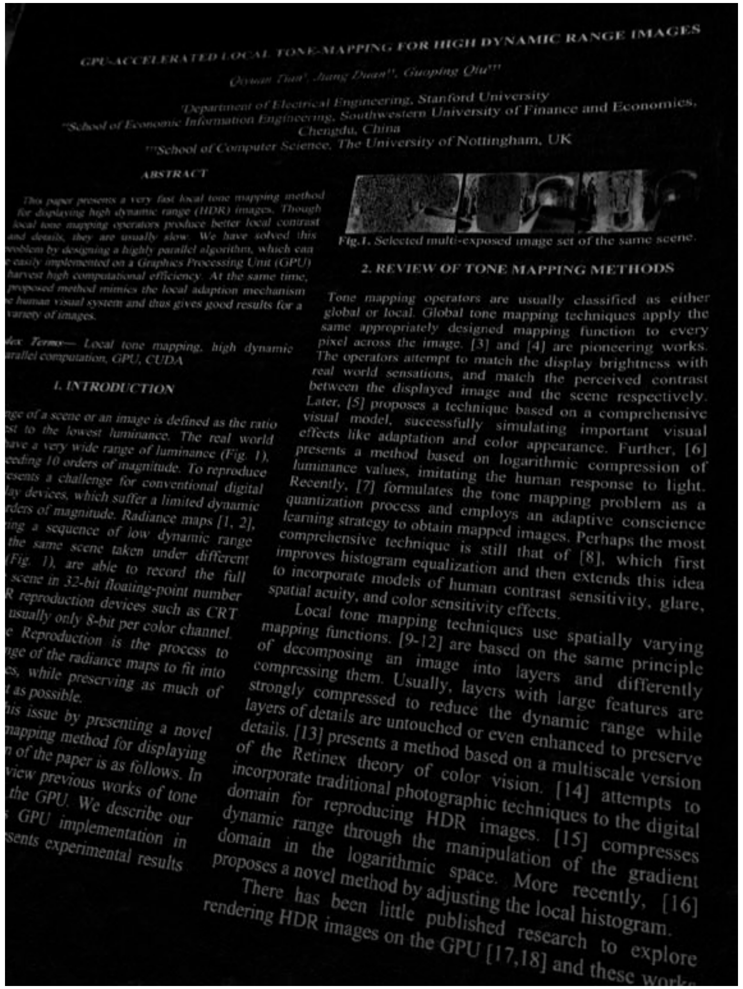
There has been little published research to explore rendering HDR images on the GPU [17,18] and these works

Original image
1632x1216 pixels

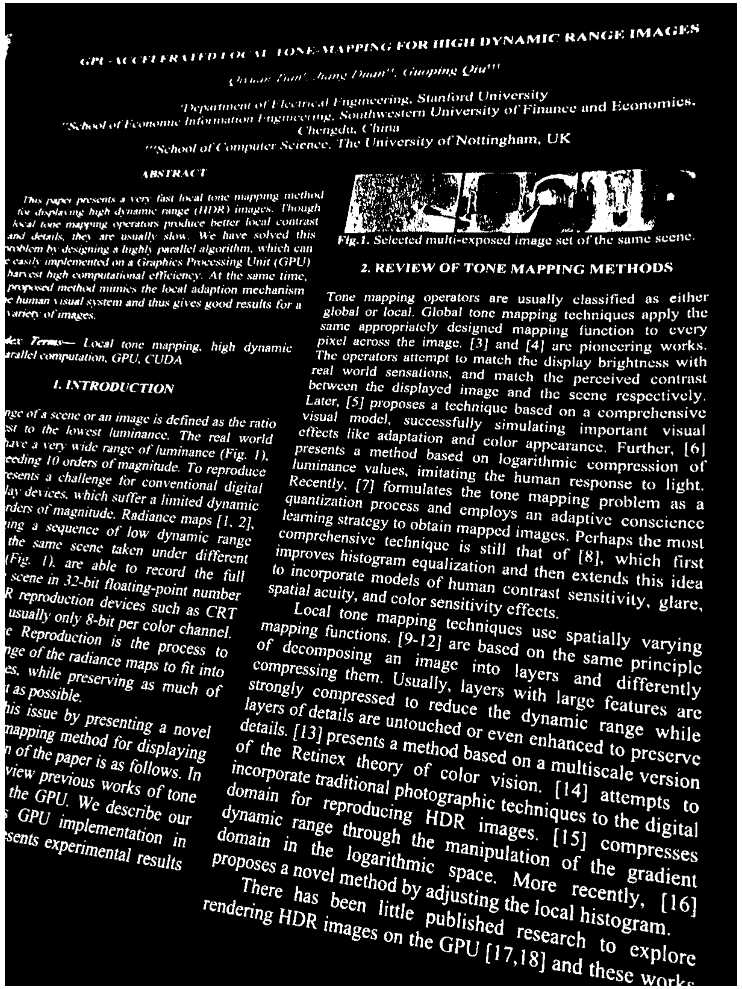
Dilation (local max)
61x61 structuring element

Rank filter
10^{st} brightest pixel
61x61 structuring element

Example: non-uniform lighting compensation



Background – original image



After global thresholding