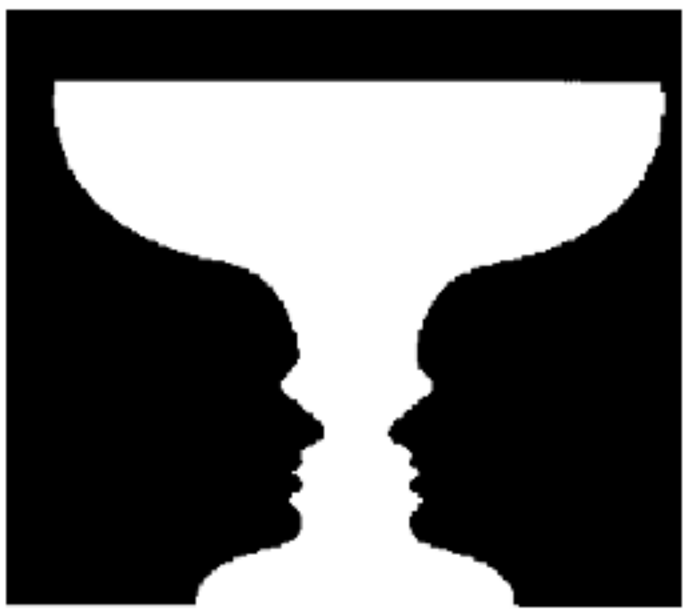
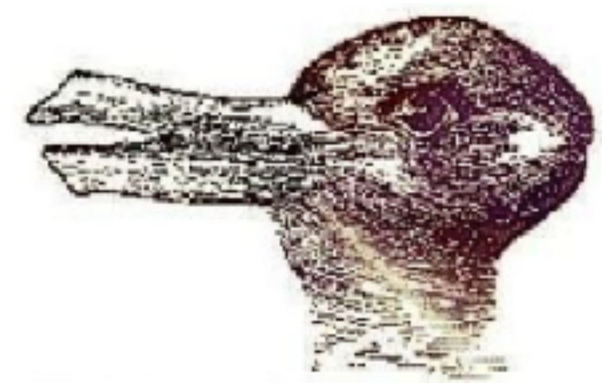
Binary Morphology

Sumohana Channappayya

Binary Images

Illusions





Binary Images
Definition

- Recall: A digital image is an array of sampled and quantized values
- For gray scale images, scale defined by $K = 2^B$ levels and B bits
- For binary images, $K = 2,\ B = 1$

Binary Images
Interpretation

- Common binary image meanings:
  - Intensity differentiator: low vs high
  - Presence or absence of object
  - Presence or absence of a property
- Why work with binary images?
  - Contain useful information: shape, structure, form
  - Compression (depending on application)

Binary Images

Generation

- Several ways to create binary images:
  - Specialised inputs: stylus (light pen), tablet etc.
  - Gray level thresholding:
    - Simple thresholding: pick a threshold T and make a binary decision
      - For an image $I(i,j)$ with K levels, pick $0 \leq T \leq (K-1)$
    - $J(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq T, \\ 0, & \text{otherwise} \end{cases}$

Binary Images

Threshold Selection

- Why is threshold selection important?
  - Quality of binary image directly depends on it
  - Different thresholds may lead to different insights
  - It may not always be able to produce useful binary images for any threshold
- Questions:
  - Is thresholding useful/possible?
  - How do we pick a good threshold T?

Binary Images

Connected Components

- The Connected Components Algorithm or “blob colouring” or “region labelling”
- Why?
  - Thresholding leads to imperfect binary images
  - Extraneous blobs or holes due to noise or low-interest regions
- Blob colouring is a method for labelling/colouring/indexing objects

Binary Images

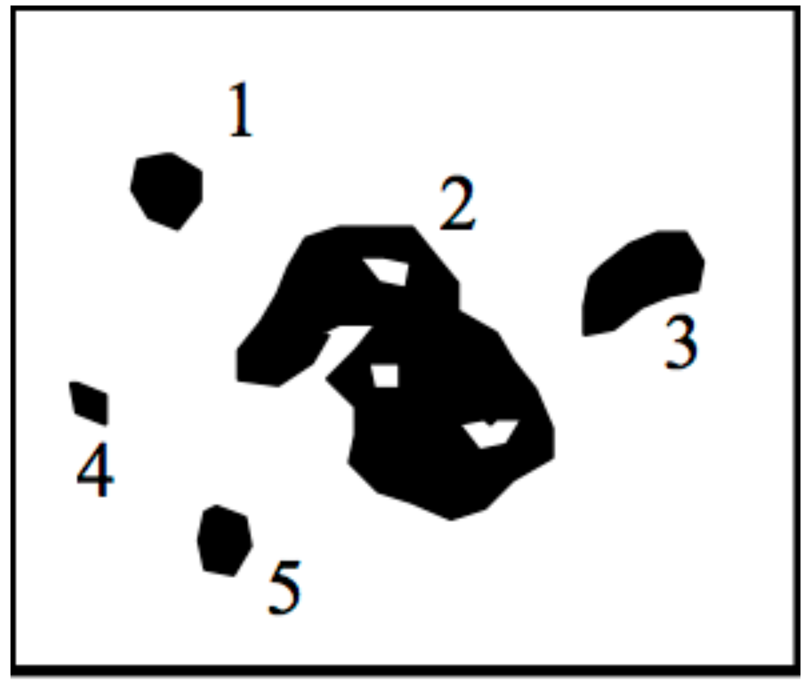
Connected Components Algorithm (4 connectivity)

- For a binary image I, define a region colour array R, where $R(i, j)$ is the region number of pixel $I(i, j)$
- Set $R = 0$ (all zeros) and region number counter $k = 1$
- **Assumption:** border pixels are *background* and have the *same value*
- While scanning the image from the top left to the bottom right, do the following:
  -  If $I(i, j) = 0$ and $I(i, j - 1) = 1$ and $I(i - 1, j) = 1$, then set $R(i, j) = k$ and $k = k + 1$
  -  If $I(i, j) = 0$ and $I(i, j - 1) = 1$ and $I(i - 1, j) = 0$, then set $R(i, j) = R(i - 1, j)$
  -  If $I(i, j) = 0$ and $I(i, j - 1) = 0$ and $I(i - 1, j) = 1$, then set $R(i, j) = R(i, j - 1)$
  -  If $I(i, j) = 0$ and $I(i, j - 1) = 0$ and $I(i - 1, j) = 0$, then set $R(i, j) = \min(R(i, j - 1), R(i - 1, j))$; if $R(i, j - 1) \neq R(i - 1, j)$ link the regions

Binary Images

Blob Colouring Example

- Blob colouring result

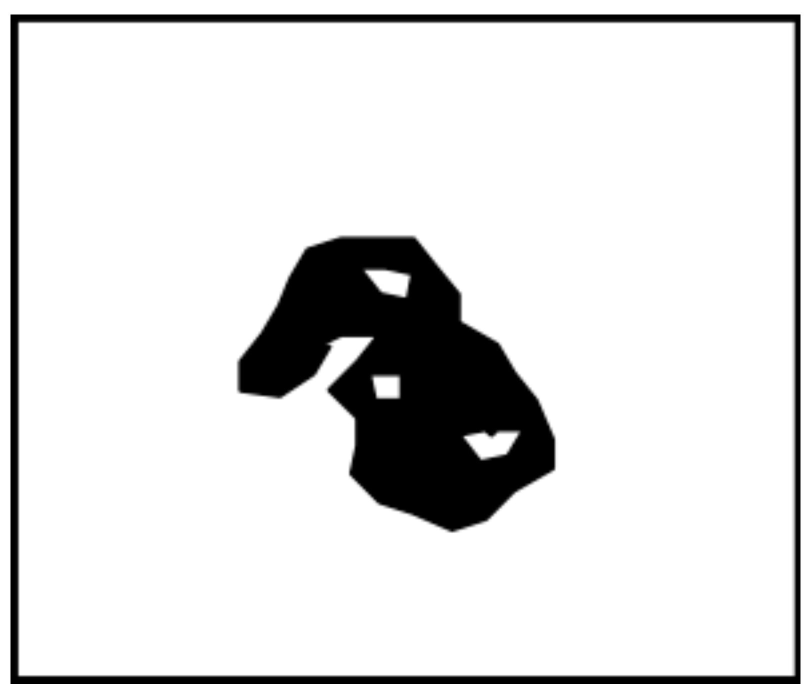


- Blob counting result

Binary Images

Minor Blob Removal

- Let m be the label of the largest blob
- While scanning the image from the top left to the bottom right: if $I(i,j) = 0$ and $R(i,j) \neq m$, set $I(i,j) = 1$



Binary Images

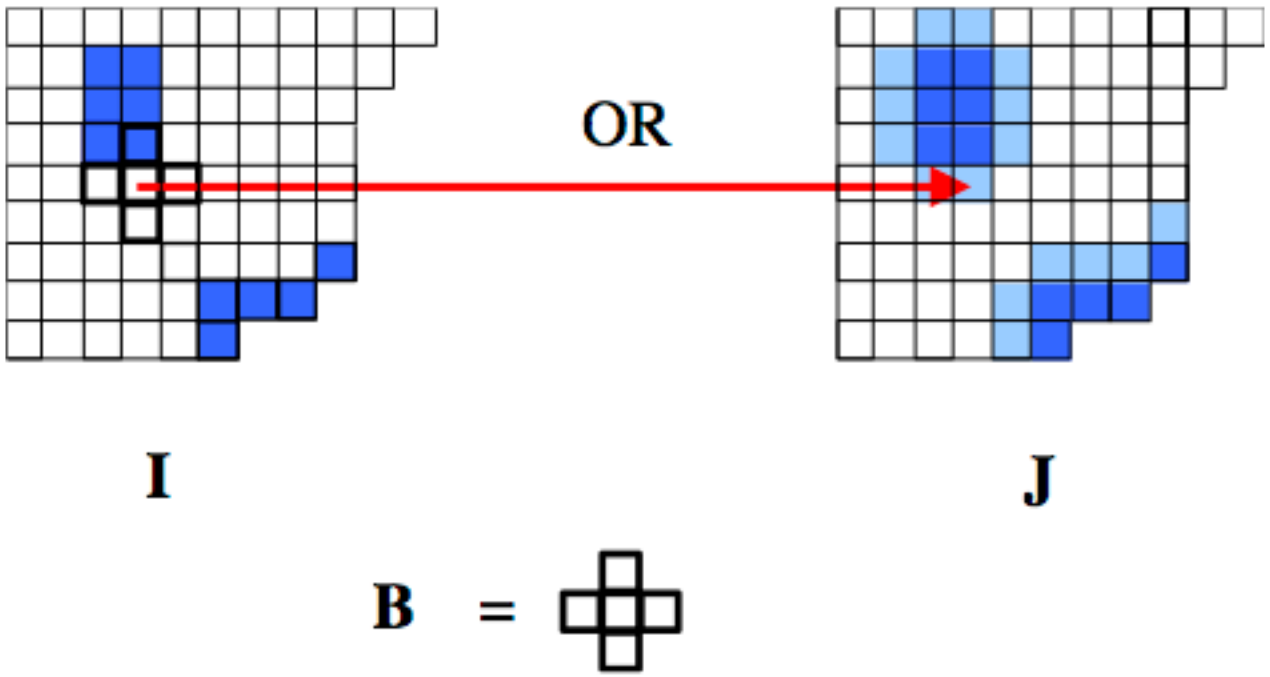
Minor Blob Removal

- To clean up:
    - Complement
    - Count blobs
    - Minor blob removal
    - Complement

Binary Morphology

Definition

- Morphology: the study of form and structure
- Mathematical morphology: tool for extracting image components for describing shapes like boundaries, skeletons, convex hulls
- Binary morphology: a class of binary image operators



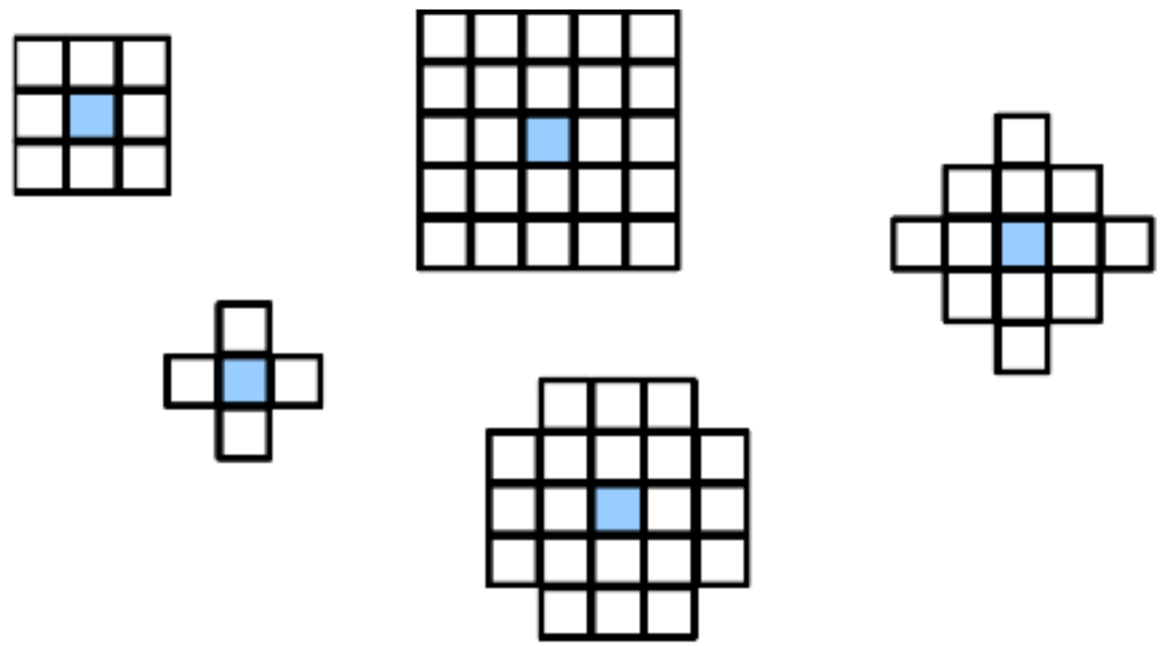
Binary Morphology

Morphological Operations

- Morphological operations:
    - **affect the shape** of objects and regions of binary images
    - operate on a local basis i.e., on **local neighbourhoods**
- Morphological operators:
    - expand or **dilate** objects
    - shrink or **erode** objects
    - smooth object boundaries
    - eliminate holes
    - fill gaps and eliminate convex hulls
    - are logical operations

Binary Morphology

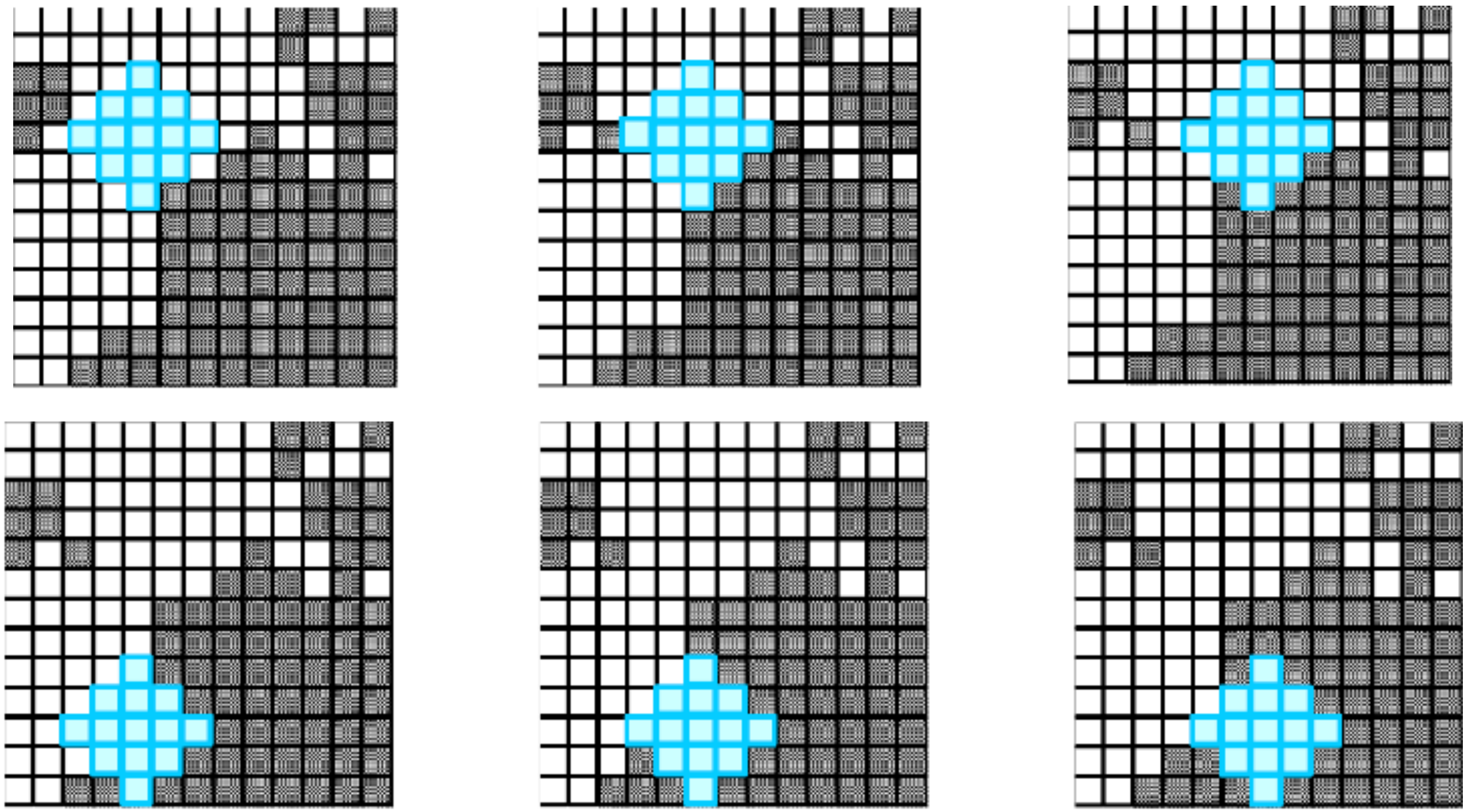
Structuring Element



- Definition: A structuring element defines a relationship between a pixel and its neighbours
- Window:
    - is a method of collecting pixels according to a geometric rule
    - is a structuring element
    - almost always contains an odd number of elements along each dimension - why?

Binary Morphology

Windows



- Using a window to perform local operations over an image

Binary Morphology

Windows

- Definition: A window $\mathbf{B}$ is a set of coordinate shifts $\mathbf{B}_i = (p_i, q_i)$ centred around $(0,0)$ i.e., $\mathbf{B} = \{\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{2P+1}\} = \{(p_1, q_1), (p_2, q_2), \ldots, (p_{2P+1}, q_{2P+1})\}$

- Examples:

  - $\mathbf{B} = \text{ROW}(2P+1) = \{(0, -P), \ldots(0,P)\}$

  - $\mathbf{B} = \text{COL}(2P+1) = \{(-P,0), \ldots(P,0)\}$

  - $\mathbf{B} = \text{CROSS}(2P+1) = \text{ROW}(2P+1) \cup \text{COL}(2P+1)$

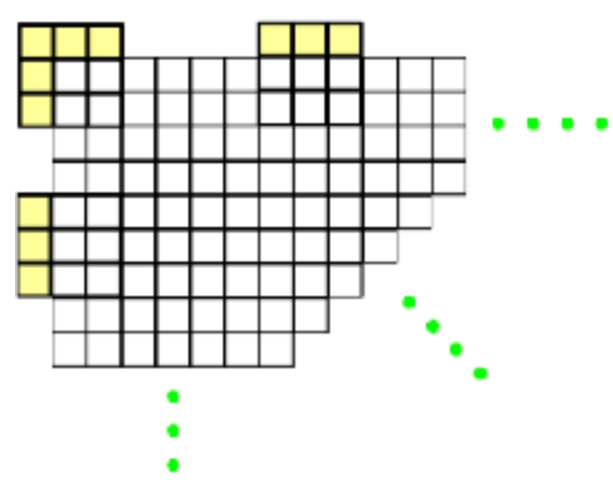
Binary Morphology
The Windowed Set

- Definition: For a binary image $\mathbf{I}$ and window $\mathbf{B}$, the **windowed set** at location (i, j) is given defined as
  $\mathbf{B} \diamond \mathbf{I}(i, j) = \{\mathbf{I}(i - p, j - q); (p, q) \in \mathbf{B}\}$
- Interpreted as the set of pixels covered by $\mathbf{B}$ centred at (i, j)
- Helps make simple and flexible design of binary filters
- Examples:
  - $\mathbf{B} = \text{ROW}(3); \mathbf{B} \diamond \mathbf{I}(i, j) = \{\mathbf{I}(i, j - 1), \mathbf{I}(i, j), \mathbf{I}(i, j + 1)\}$
  - $\mathbf{B} = \text{COL}(3); \mathbf{B} \diamond \mathbf{I}(i, j) = \{\mathbf{I}(i - 1, j), \mathbf{I}(i, j), \mathbf{I}(i + 1, j)\}$

Binary Morphology

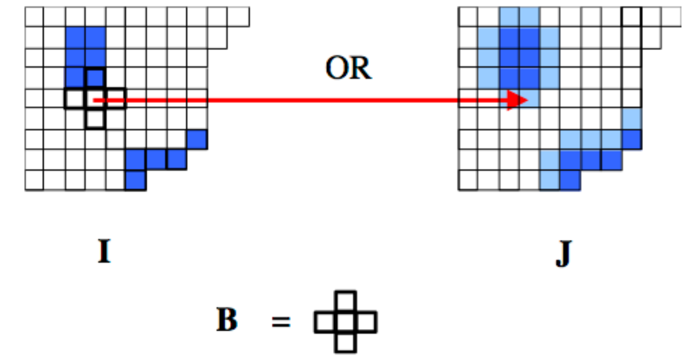
General Binary Filter

- Notation: A binary operator $\mathbf{G}$ on a windowed set $\mathbf{B} \diamond \mathbf{I}(i, j)$ is denoted as
  $\mathbf{J}(i, j) = \mathbf{G}\{\mathbf{B} \diamond \mathbf{I}(i, j)\} = \mathbf{G}\{\{\mathbf{I}(i - p, j - q); (p, q) \in \mathbf{B}\}\}$
- Performing the operation at every pixel gives us the filtered image $\mathbf{J} = \mathbf{G}[\mathbf{I}, \mathbf{B}] = [\mathbf{J}(i, j); 0 \leq i \leq N - 1, 0 \leq j \leq M - 1]$
- How about image boundary?
  - Replication: use nearest neighbors to fill empty slots

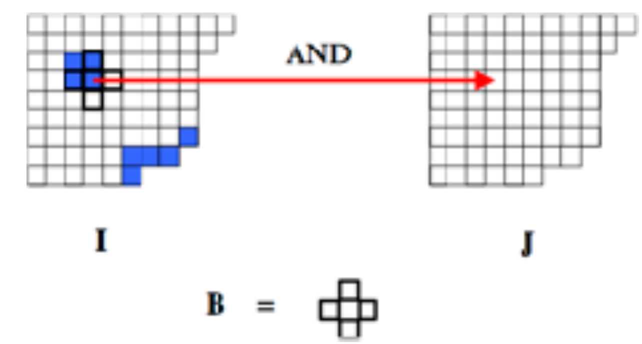


Binary Morphology

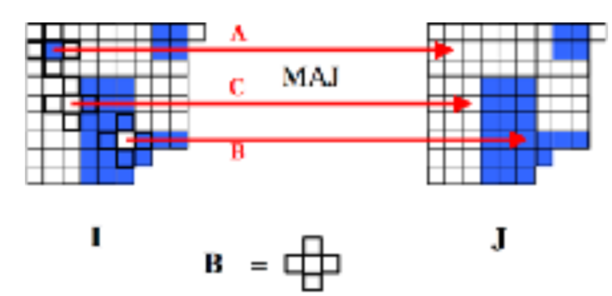
Dilation, Erosion and Median Filters



- **Dilation**: Given a window $\mathbf{B}$ and a binary image $\mathbf{I}$,
  $\mathbf{J} = \text{DILATE}(\mathbf{I}, \mathbf{B})$ if
  $\mathbf{J}(i,j) = \text{OR}\{\mathbf{B} \diamond \mathbf{I}(i,j)\} = \text{OR}\{\{\mathbf{I}(i-p, j-q); (p,q) \in \mathbf{B}\}\}$



- **Erosion**: Given a window $\mathbf{B}$ and a binary image $\mathbf{I}$,
  $\mathbf{J} = \text{ERODE}(\mathbf{I}, \mathbf{B})$ if
  $\mathbf{J}(i,j) = \text{AND}\{\mathbf{B} \diamond \mathbf{I}(i,j)\} = \text{AND}\{\{\mathbf{I}(i-p, j-q); (p,q) \in \mathbf{B}\}\}$

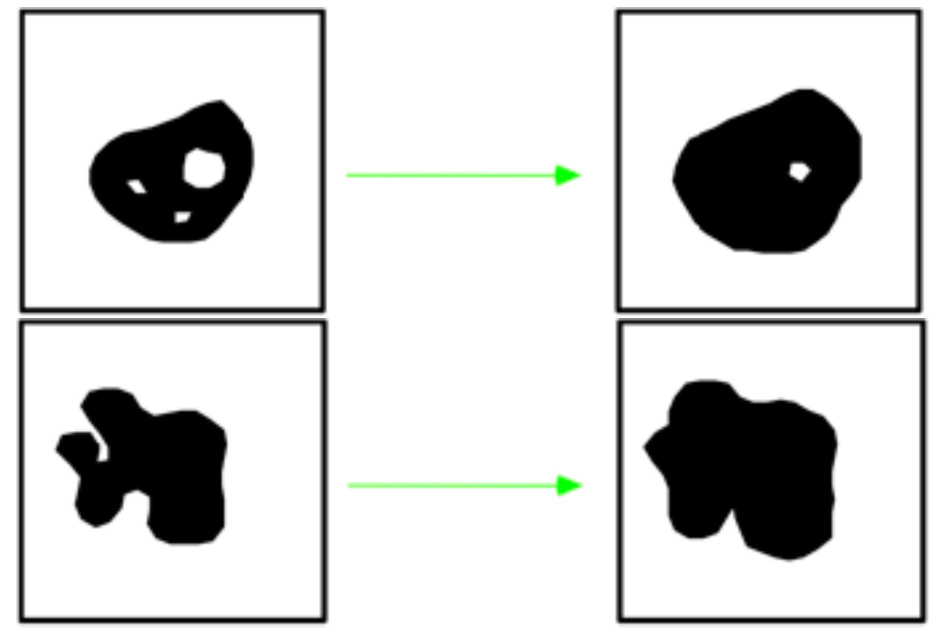


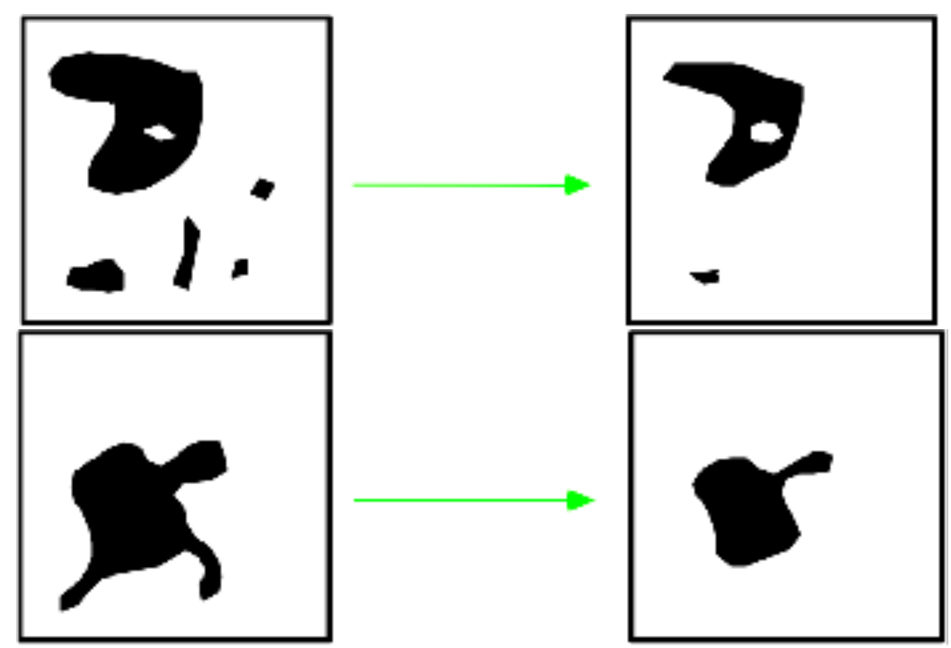
- **Median**: Given a window $\mathbf{B}$ and a binary image $\mathbf{I}$,
  $\mathbf{J} = \text{MEDIAN}(\mathbf{I}, \mathbf{B})$ if
  $\mathbf{J}(i,j) = \text{MAJ}\{\mathbf{B} \diamond \mathbf{I}(i,j)\} = \text{MAJ}\{\{\mathbf{I}(i-p, j-q); (p,q) \in \mathbf{B}\}\}$

Binary Morphology

Dilation and Erosion Properties

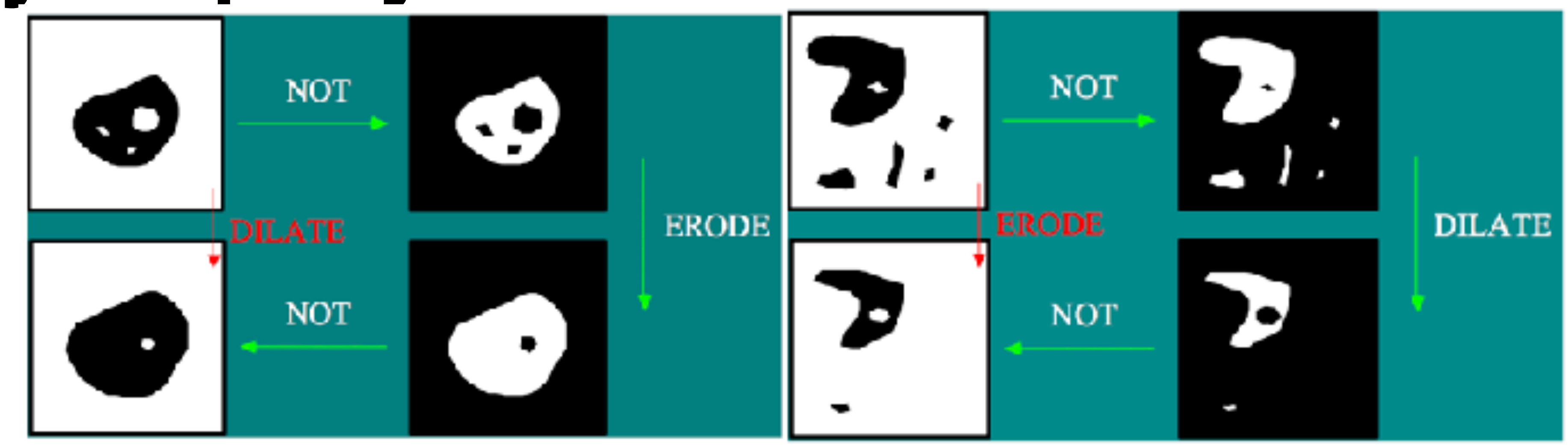
- Dilation
    - Fills small gaps or holes
    - Fills bays
- Erosion
    - Eliminates small objects
    - Eliminates peninsulas





Binary Morphology

Duality Property



- Dilation and Erosion are **duals** with respect to complementation
- Median is its **own dual** with respect to complementation
- Dilation and Erosion are *approximate inverses* of one another
  - Peninsulas *eliminated* by erosion cannot be recreated
  - Small object *eliminated* by erosion cannot be recreated
  - Holes *filled* by dilation cannot be recreated
  - Gaps or bays *filled* by dilation cannot be recreated
- Median filter generally does not change object size (boundary) but alters them

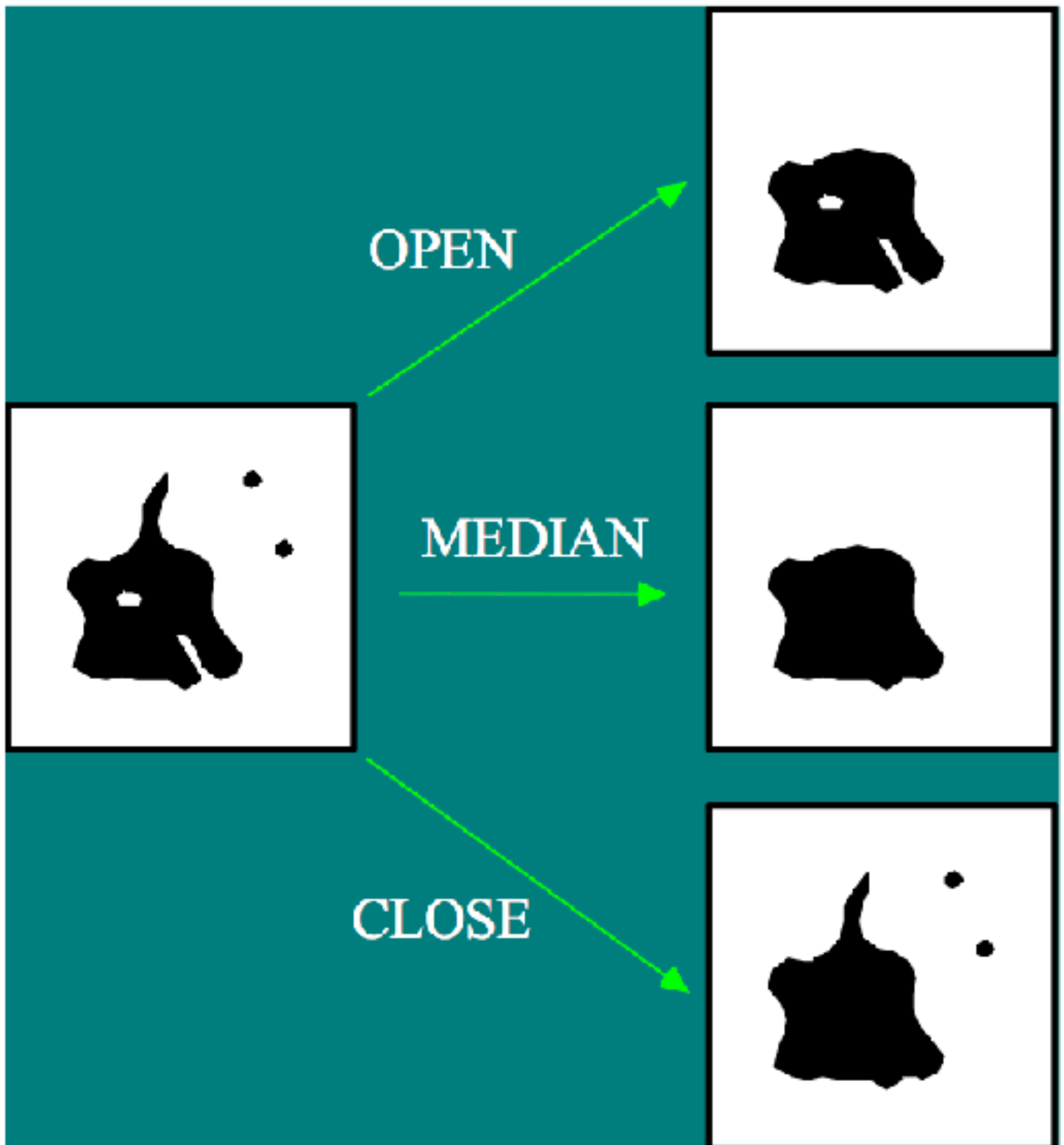
Binary Morphology

OPEN and CLOSE operators

- Definition of new operators by applying basic operators in sequence
- Given a binary image $\mathbf{I}$ and a window $\mathbf{B}$,
  - OPEN$(\mathbf{I}, \mathbf{B}) =$ DILATE[ERODE$(\mathbf{I}, \mathbf{B}), \mathbf{B})]]$
  - CLOSE$(\mathbf{I}, \mathbf{B}) =$ ERODE[DILATE$(\mathbf{I}, \mathbf{B}), \mathbf{B})]]$
- Similar to MEDIAN filter
- OPEN removes small objects better than MEDIAN but not holes, gaps or bays
- CLOSE removes small holes and gaps better than MEDIAN but not small objects
- In general, OPEN and CLOSE do not affect object size

Binary Morphology

Comparing OPEN, CLOSE and MEDIAN



Binary Morphology

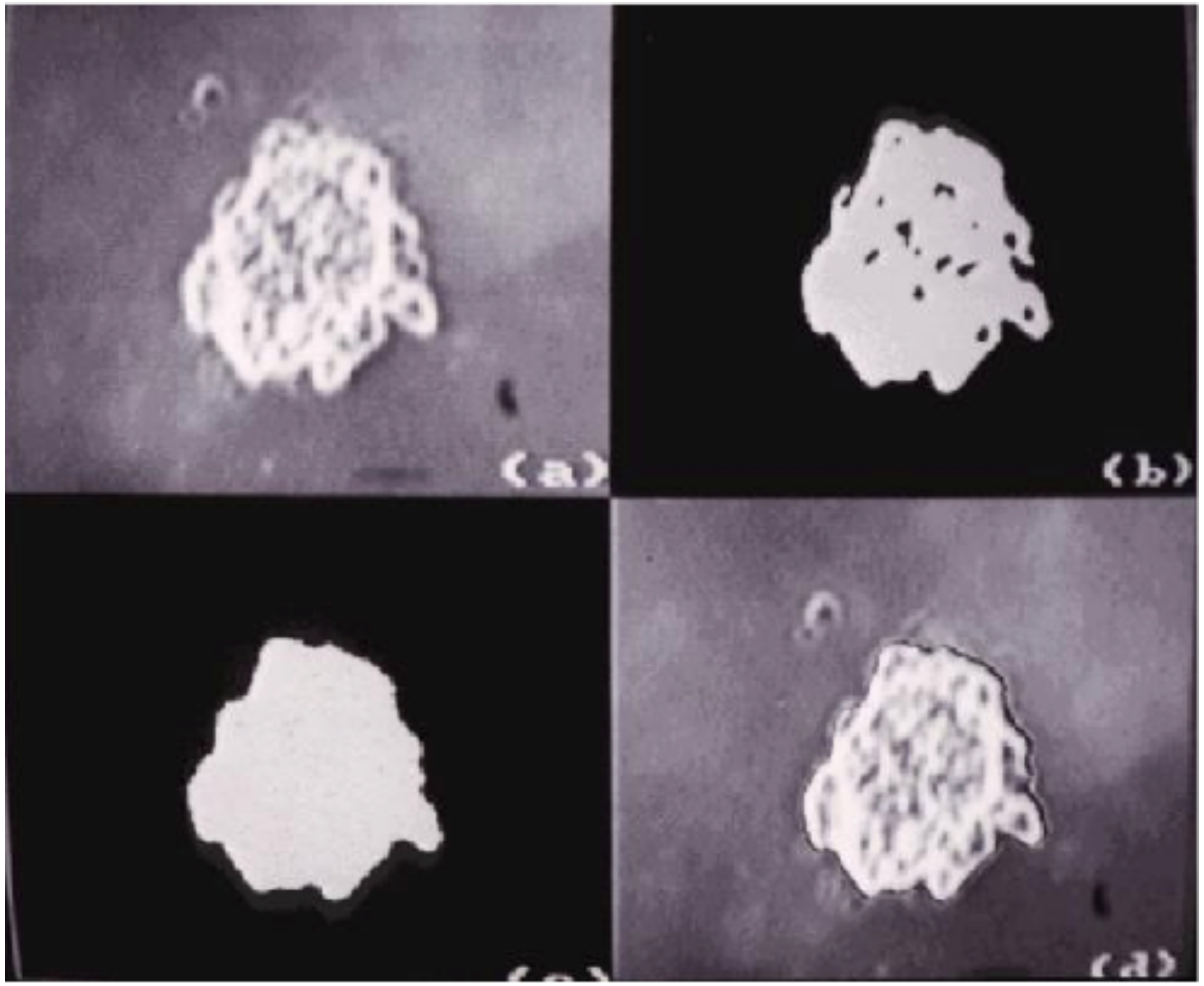
OPEN-CLOS and CLOS-OPEN operators

- Continue to cascade basic operators:
  - OPEN-CLOS$(\mathbf{I}, \mathbf{B}) =$ OPEN[CLOSE$(\mathbf{I}, \mathbf{B}), \mathbf{B})$]]
  - CLOS-OPEN$(\mathbf{I}, \mathbf{B}) =$ CLOSE[OPEN$(\mathbf{I}, \mathbf{B}), \mathbf{B})$]]
- Properties:
  - Good smoothing operators
  - Remove small objects without affecting size
  - Similar to MEDIAN filter but more smoothing
  - OPEN-CLOS tends to link neighboring objects together
  - CLOS-OPEN tends to link neighboring holes together

Binary Morphology

Application

- **Measuring cell area**
- Binarise image using thresholding
- Apply region correction
    - Blob colouring
    - Minor blob removal
    - CLOS-OPEN
- Display result for verifying operator
- Count pixels for cell area calculation
- True cell area computed using perspective projection



Binary Morphology
Summary

- Binary images are a very useful class of gray scale images
- Binary morphology provides techniques for accomplishing several useful tasks