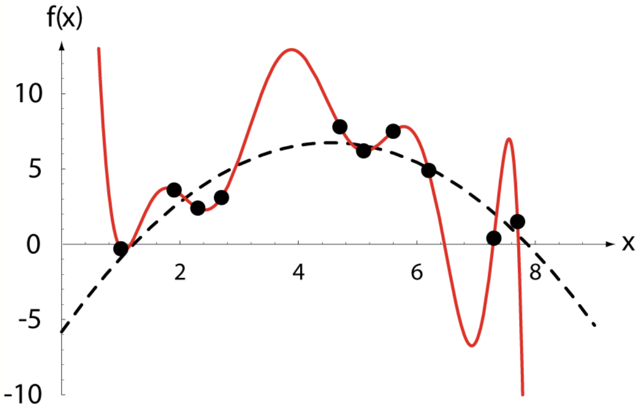
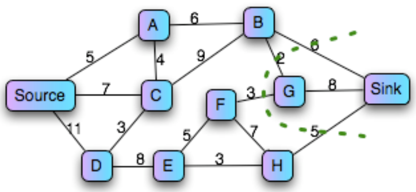
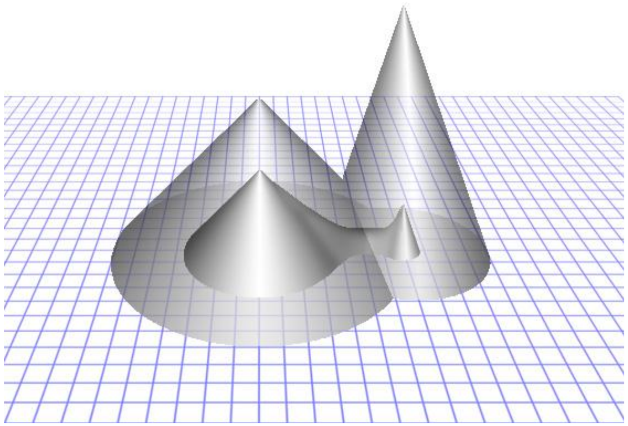






INF4300
Regularization and smoothing

Asbjørn Berge 03-10-2010

Readings for this lecture

- Morphology
  - R.C. Gonzales and R.E. Woods: *Digital Image Processing, 3rd ed*, 2008. Prentice Hall. ISBN: 978-0-13-168728-8. Chapter 9, 9.4,9.5 very cursory
- Graph cuts and other regularization approaches
  - Exercise/tutorial text
  - Cursory reading for interested students
    - http://www.cs.cornell.edu/~rdz/graphcuts.html
    - Classic paper: What Energy Functions can be Minimized via Graph Cuts? (Kolmogorov and Zabih, ECCV '02/PAMI '04)

What is regularization?

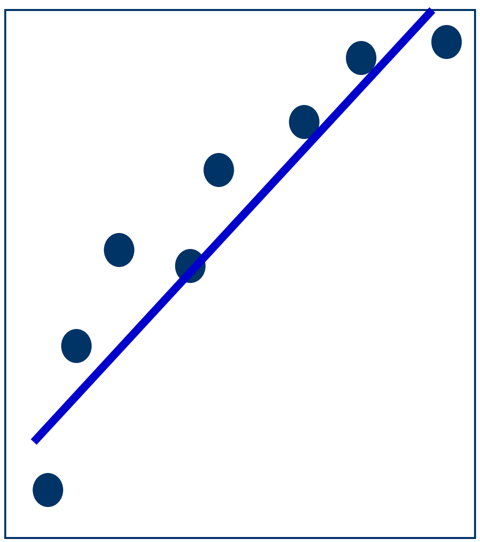
- «Cleaning things up» / *smoothing*
  - Classification results (category output or probabilities)
  - Models (for example linear regressions or decision boundaries)
  - Detections (edge detectors, etc.)
  - Even raw images!
- How do we regularize
  - Smoothing results in 2D, by comparing with neighborhood
  - Penalties on «non-regular» behavior on results
  - Smoothing parameters, images or inputs

What will this lecture cover?

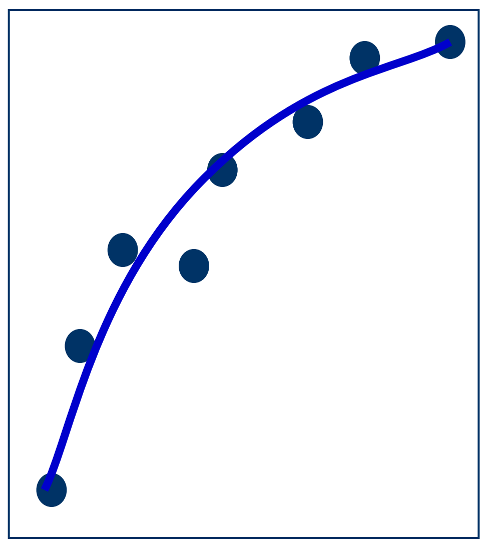
- General terminology on regularization
- The regularization inherent in Support Vector Machines
- Morphology
- Graph cuts

Summary: what is regularization?

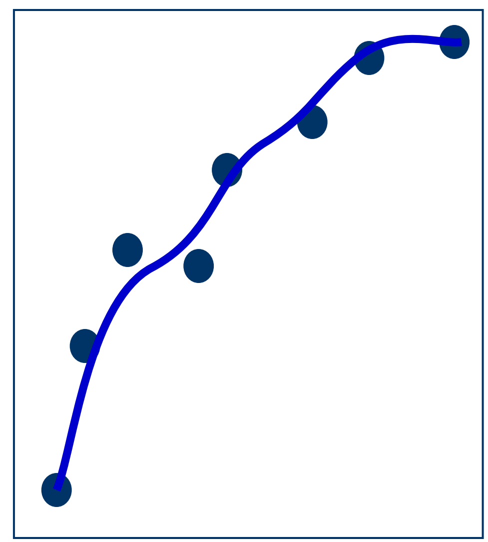
- **Regularization is smoothing of *parameters* or *output* of a classification method**
- Rationale is because we believe smooth descriptions to be less noisy
- Regularization is essentially the same across different PR approaches
  - Example: Regularization penalizes bending energy
    - What is the best way to show graph points with a smooth line? Heavy regularization is a poor fit, light regularization causes local distortion



Heavy Regularization



Medium Regularization



Light Regularization

Why regularize?

Measurements are noisy, so we don’t want to learn the training data perfectly, because then we learn the noise as well.

Avoid overfitting!

Choose the simplest possible model, but not *too* simple.

What is regularization?

Modification of our estimation procedure for *f(X)*

- Goal: get reasonable answers in unstable situations
- Approach: use simpler models or restrict models

"A class of methods of **avoiding over-fitting to the training set** by penalizing the fit by a measure of 'smoothness' of the fitted function."
- B. D. Ripley, *Pattern Recognition and Neural Networks*

What leads to an "unstable situation"?

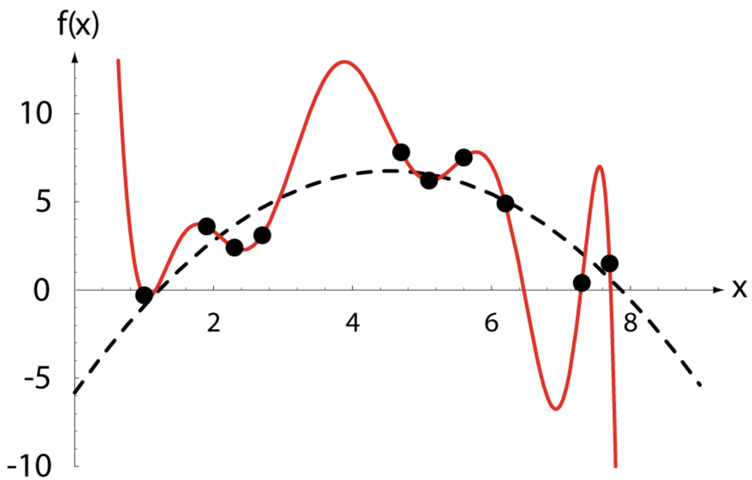
- High dimension of *X* (# measurements for each example)
- Few examples in *T* (size of training data set)
- Measurements in *X* very similar (high correlation / collinearity)
- Choice of *f(X)* (high flexibility in model)

Trading flexibility for stability

- Analog to bias – variance tradeoff in regression

- Simple models vary less over repeated experiments

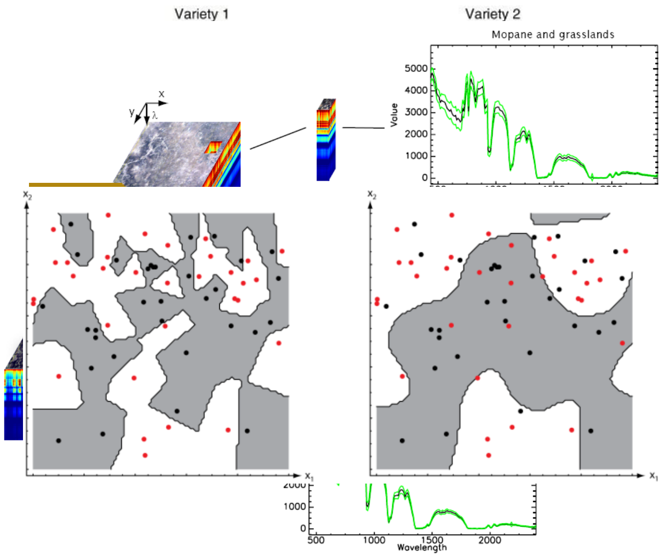
- But simple models may have poorer fit



Examples of regularization in a classification context

Statistical models for classes

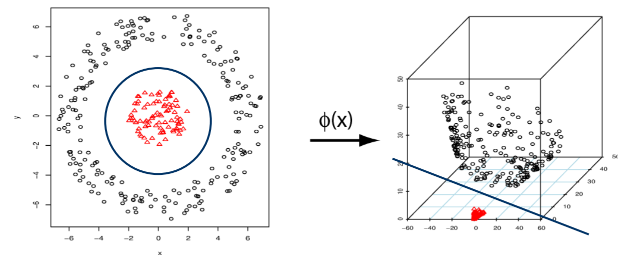
- Parametric models
  - **Regularization** : Stabilization of model parameters
  - High dimensional data
    - Hyperspectral images
    - Spectrometer data
    - Biostatistics / Microarrays
    - Text classification
- Nonparametric models (density estimate)
  - **Regularization** : Smoothness in density estimate

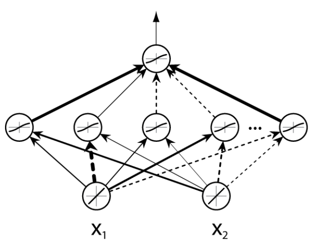


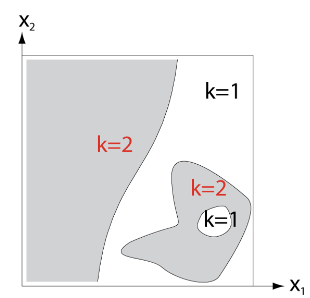
Examples of regularization in a classification context

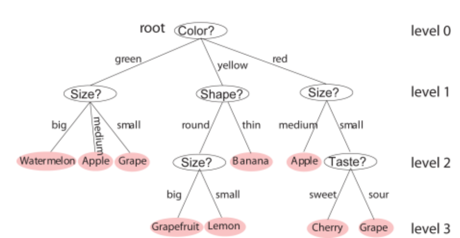
Decision boundary estimation

- Support vector machines
  - Mapping of data into higher dimensional space
  - **Regularization** : Reduction in degrees of freedom of solution
- Neural networks
  - Complex interaction in network of neurons
  - **Regularization** : Reduction or removal of neurons
- Tree classifiers
  - Complex decision trees
  - **Regularization** : Reduction in the number of branches





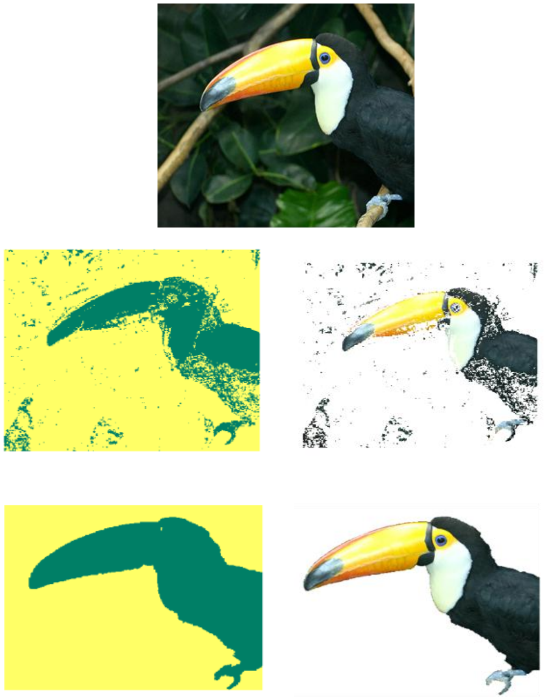




Examples of regularization in a classification context

Regularization as prior belief

- Prior on class model parameters
  - We believe parameters have some known structure
  - **Regularization** : Assume some initial structure (distribution) on classifier parameters.
    - Implicit in some regularization procedures
- Spatial classification
  - Classification labels noisy in a spatial sense
  - **Regularization** : Smoothing classification labels to obtain contigouous regions
  - Prior belief that neighbor pixels same class
    - Medical imaging, tissue classification
    - Remote sensing, mapping applications
    - Video image segmentation



SINTEF

Statistical model for classes

Statistical model for the distribution of each class

$$p(X \mid k; \mu_k, \Sigma_k) = (2\pi)^{-p/2} \,|\Sigma_k|^{-1/2} \exp[-\tfrac{1}{2}(X-\mu_k)^T \Sigma_k^{-1}(X-\mu_k)]$$

Rule for finding the statistical model for classes (Bayes' rule)

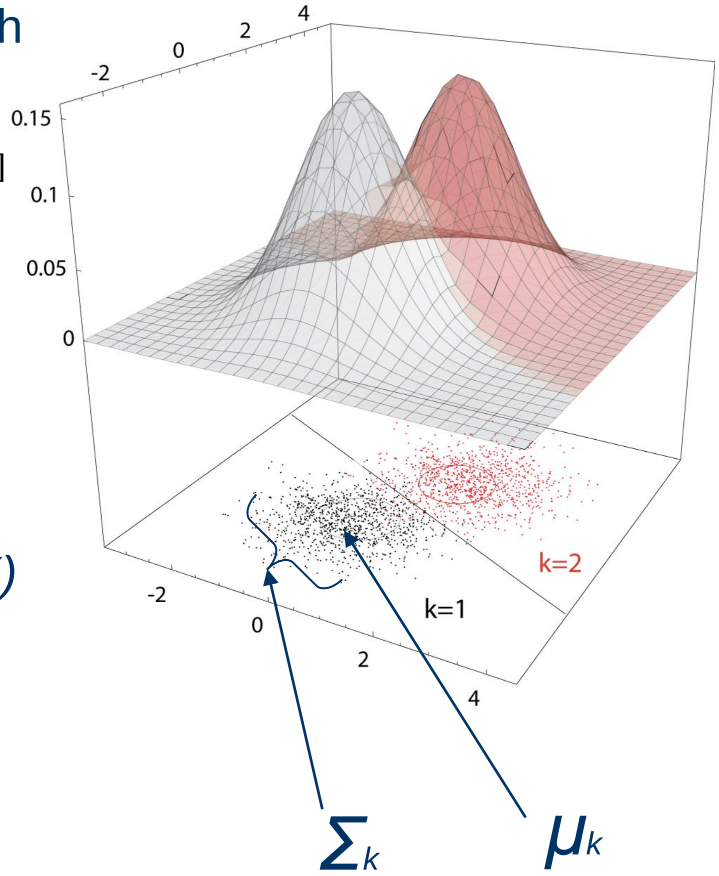
$$p(k|X) = \frac{p(X \mid k)\, p(k)}{p(X)}$$

Classification is done by maximizing *p(k|X)*

$$f(X) = k : \max_{1 \le k \le K} p(k \mid X)$$

by finding the largest discriminant function

$$d_k(X; \mu_k, \Sigma_k) = (X-\mu_k)^T \Sigma_k^{-1}(X-\mu_k) + \ln|\Sigma_k| \; -2\ln \pi_k$$

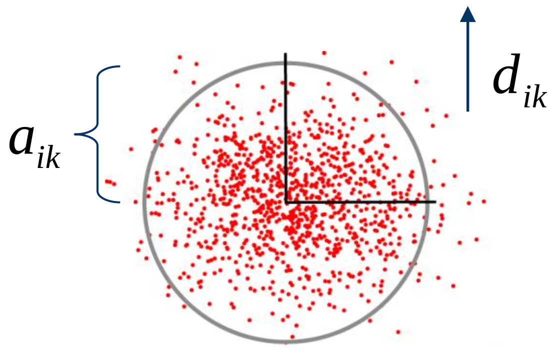


Covariance matrix stabilization

The covariance matrix can be eigendecomposed:

$$\Sigma_k = D_k A_k D_k^T = \sum_{i=1}^{p} a_{ik} d_{ik} d_{ik}^T$$

$$\Sigma_k^{-1} = \sum_{i=1}^{p} \frac{d_{ik} d_{ik}^T}{a_{ik}}$$



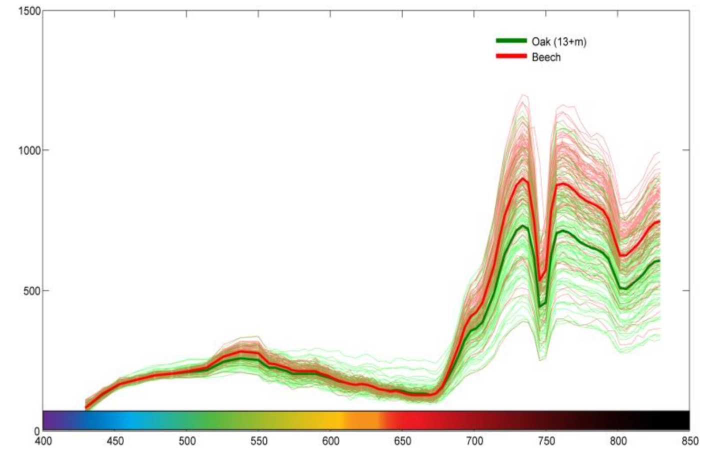
The discriminant function is mainly influenced by the eigenvectors in *directions of small eigenvalues*, which can be *noisy*.

$$d_k(X; \mu_k, \Sigma_k) = (X - \mu_k)^T \Sigma_k^{-1} (X - \mu_k) + \ln|\Sigma_k| \; -2\ln \pi_k$$

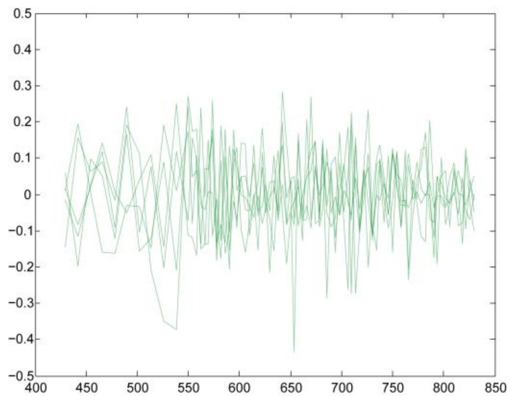
$$d_k(X) = \sum_{i=1}^{p} \frac{[d_{ik}^T (X - \mu_k)]^2}{a_{ik}} + \sum_{i=1}^{p} \ln(a_{ik}) - 2\ln \pi_k$$

A short example of the variability in the discriminant function

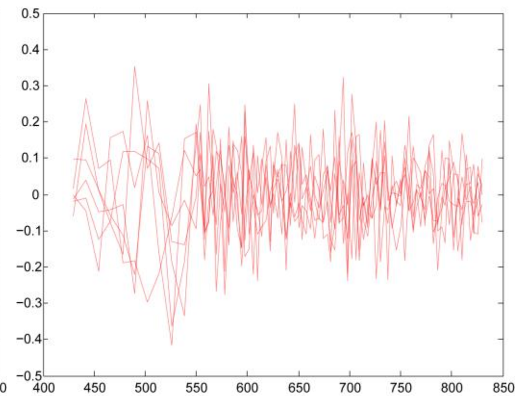
- Do 5 random draws from the data from both classes. How do the eigenvectors look?
- The eigenvectors corresponding to the smallest eigenvalue give very rough vectors and realizations are not similar



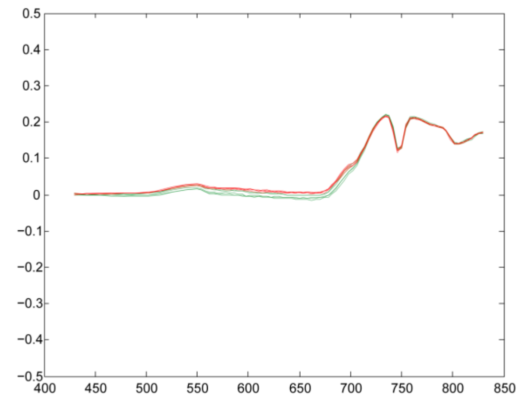
Smallest eigenvector in oak group



Smallest eigenvector in beech group



Largest eigenvector in both groups



Covariance matrix stabilization

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1p} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{p1} & \sigma_{p2} & \cdots & \sigma_p^2 \end{bmatrix} \qquad \Sigma = DAD^T = \begin{bmatrix} \uparrow & & \uparrow \\ d_1 & \cdots & d_p \\ \downarrow & & \downarrow \end{bmatrix} \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & a_{pp} \end{bmatrix} \begin{bmatrix} \leftarrow & d_1 & \rightarrow \\ & \vdots & \\ \leftarrow & d_p & \rightarrow \end{bmatrix}$$

Modification of eigenvalues a_{ii}

- Replace the smallest eigenvalues with their average
  (Discriminant Analysis with Shrunken Covariance)
- Add a small diagonal matrix to the covariance estimate
  (Ridge penalty)

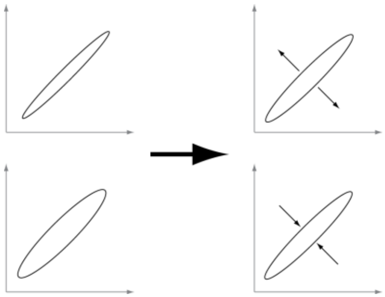
Shrinking Σ_k towards some simpler matrix

- Shrinking towards the common covariance matrix Σ
- Shrinking towards the identity matrix I
  (Regularized Discriminant Analysis)
- Discrete steps towards common covariance matrix Σ
  (Eigendecomposition Discriminant Analysis)

Modification of eigenvalues a_{ii}

Keep the *q* largest eigenvalues and average the smallest (DASCO)

$$\Sigma_k = D_k A_k D_k^{\;T} = D_k \begin{bmatrix} a_{11}^k & 0 & 0 & 0 & 0 \\ 0 & a_{22}^k & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & \bar{a}^k & \\ 0 & 0 & 0 & 0 & \bar{a}^k \end{bmatrix} D_k^{\;T}$$

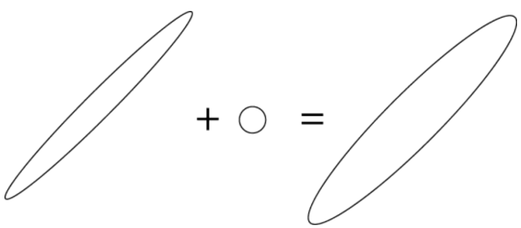


Motivation:

- directions with small eigenvalues useful for classification
- average (smooth) away the negative effect of the smallest directions

Add a small value to each of the eigenvalues (Ridge penalty)

$$\Sigma_k = D_k A_k D_k^{\;T} = D_k \begin{bmatrix} a_{11}^k + \lambda & 0 & 0 & 0 \\ 0 & a_{22}^k + \lambda & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & a_{pp}^k + \lambda \end{bmatrix} D_k^{\;T}$$



Motivation:

- gradually equalize the amount of impact of the small and large eigenvalues

SINTEF

Covariance shrinkage

- Covariance matrix estimates with less parameters are more stable
  - Use all samples from all classes to estimate covariance

- Simpler estimates are less flexible
  - A common estimate gives linear decision rules

- A linear combination gives a regularized estimate of flexible model

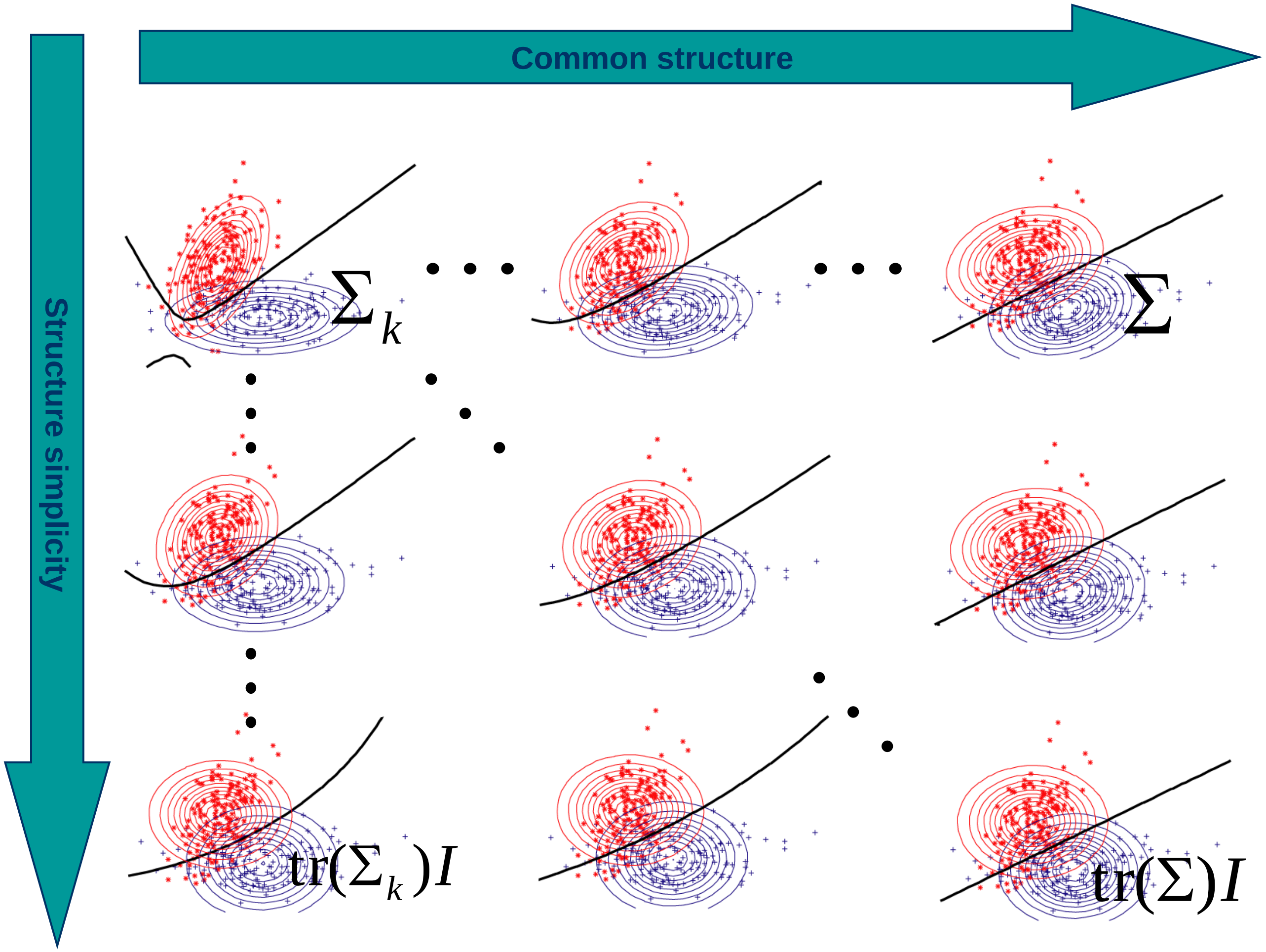
$$\boldsymbol{\Sigma}_k(\alpha) = (1-\alpha)\boldsymbol{\Sigma}_k + \alpha\boldsymbol{\Sigma}, 0 < \alpha < 1$$

Shrink towards common covariance

$$\boldsymbol{\Sigma}_k(\alpha,\beta) = (1-\beta)\boldsymbol{\Sigma}_k(\alpha) + \beta \mathrm{tr}[\boldsymbol{\Sigma}_k(\alpha)]\mathrm{I}, \quad 0 < \beta < 1$$

Shrink towards diagonal matrix

Regularized discriminant analysis



Estimating the boundary directly

Support Vector Machines (SVM)

- Optimization problem

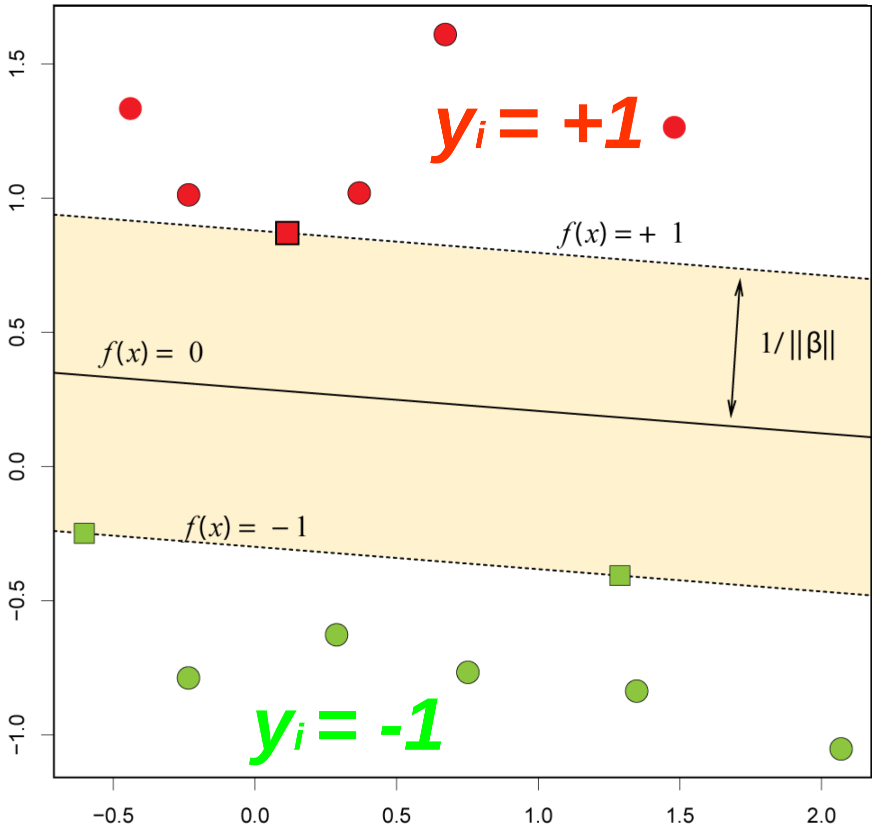
$$f(X) = \beta^T X + \beta_0$$

$$\min_{\beta,\beta_0} \sum_{i=1}^{n} [1 - Y_i(\beta^T X_i + \beta_0)]_+ + \frac{1}{2C}\|\beta\|^2$$

- Maximize margin
- Minimize error

- Error *0* for points outside margin
- Error ε for points inside margin

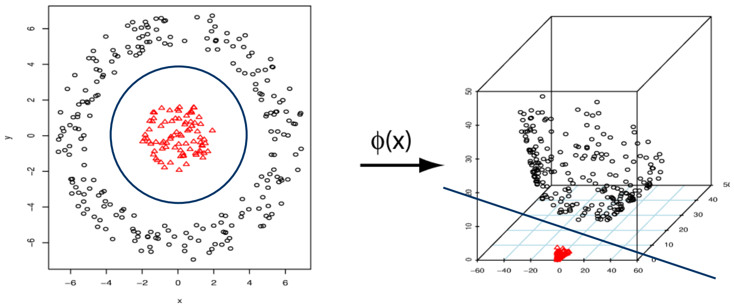
- C = cost of misclassification
- Last term **regularizer**



Non-linear SVM

$$f(X) = \phi(X)^T \beta + \beta_0$$

- Map data onto another basis of higher dimension and find linear solution
- More flexible classifier, regularization necessary!
- Regularization term $||\beta||^2$ shrinks coefficients of $\Phi(X)$ in the original space
- This usually leads to more ”smooth” boundaries in original space



$$\Re^2 \Rightarrow \Re^3$$

$$(x_1, x_2) \Rightarrow (x_1^2, x_2^2, \sqrt{2} x_1 x_2)$$

SVM regularization

Example: expanded basis *Φ(X)* with much higher degree of freedom

- High C (regularization low)
  - no misclassification on training data
  - non-smooth boundary
  - test error high

- Low C (high regularization)
  - some misclassification on training data
  - smooth desicion boundary
  - test error low

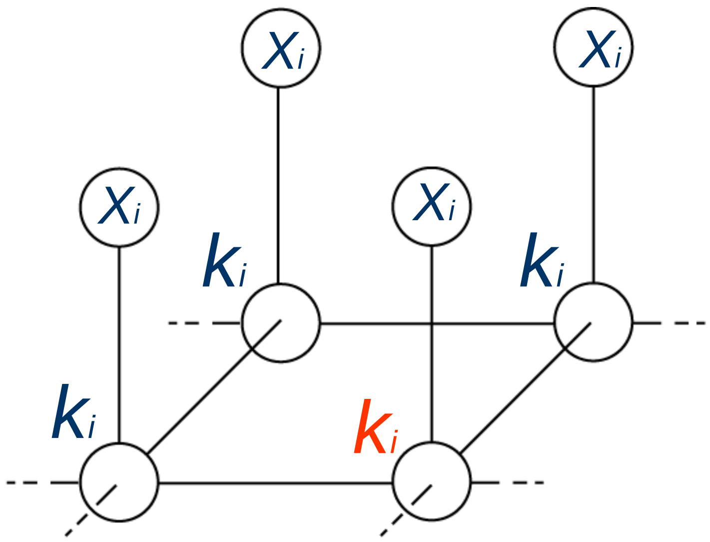
SINTEF

Smooth labels

- Classification as segmentation of an image
    - *contiguous regions*
- Bayes rule for entire image

$$p(K|X) = \frac{p(X \mid K)\,p(K)}{p(X)}$$

- *Regularize* classification *output* by prior belief *P(K)* that class of a pixel is likely to be similar to the classes of neighbors



ENERGY = DATA + SMOOTHNESS

$$p(K) = p(k_i \mid k_j; j \neq i) = p(k_i \mid k_j; j \in N(i)) \propto \exp(-\beta \sum_j \partial(k_i, k_j))$$

•Thus, the Bayes rule is on the form

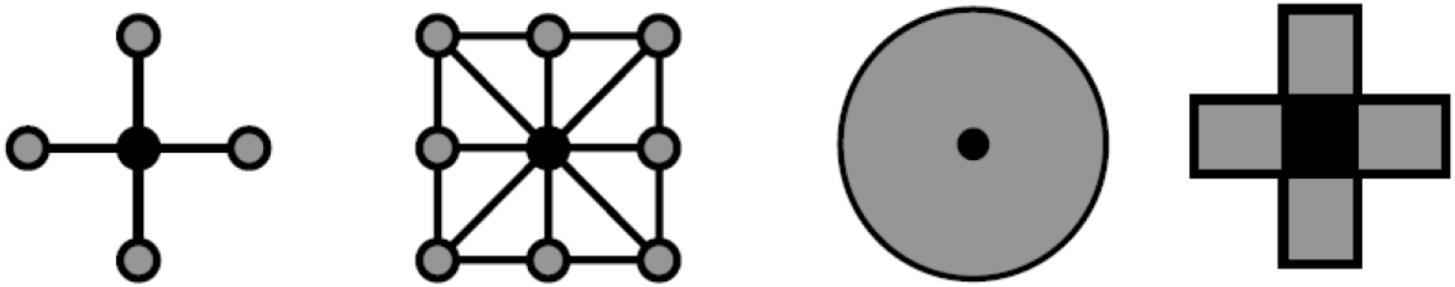
$$p(K|X) \propto p(X \mid K)\,p(K) \propto \exp(-d_k)\exp(-\beta \sum_j \partial(k_i, k_j))$$

Smooth labels

$\beta=0.5$
$\beta=1$
$\beta=4$

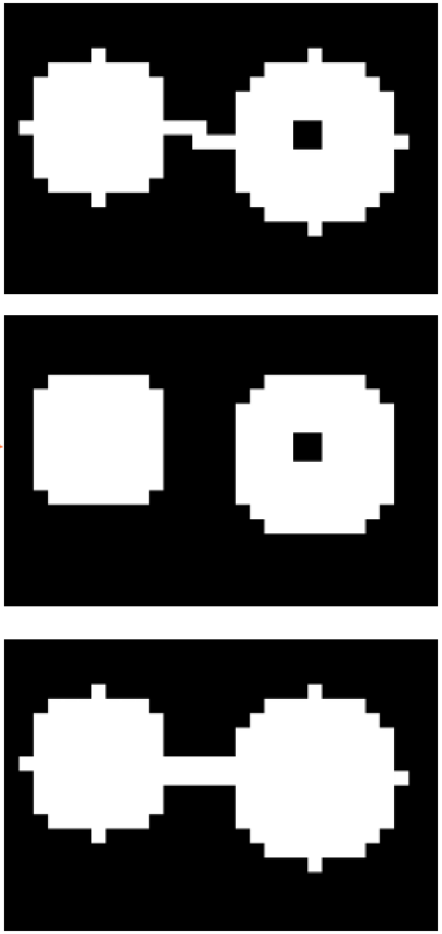
Morphology

- Structuring element (SE)
- Small set to probe the image under study.
- For each SE, define an origin, usually centre
- Shape and size must be adapted to geometric properties for the objects.



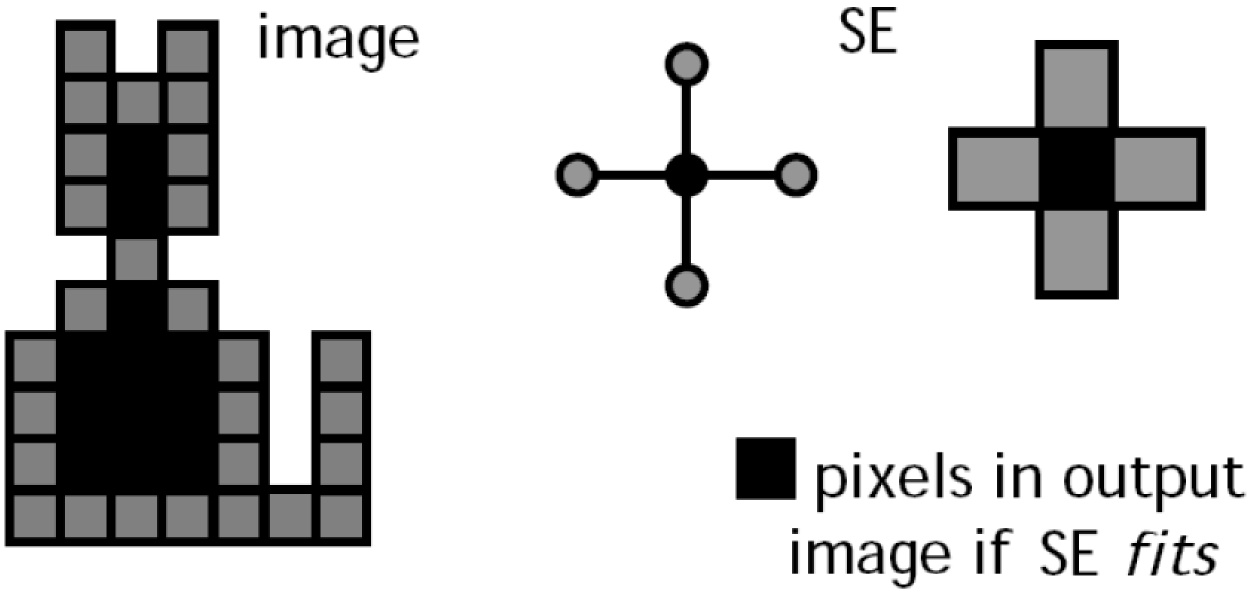
Five basic morphological transforms

- ⊖ ε erosion
  - shrinking
- ⊕ δ dilation
  - growing
- ○ γ opening
  - erosion followed by dilation (same SE)
- ● ϕ closing
  - dilation followed by erosion (same SE)
- ⊗ hit-or-miss transform

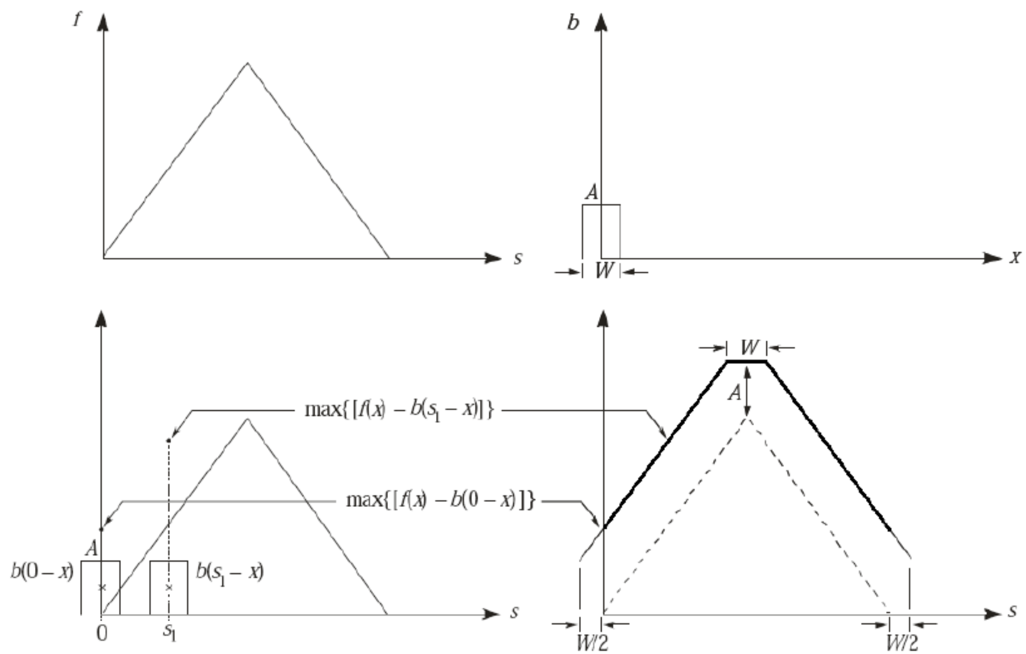


Basic idea

- In parallel for each pixel in binary image:
- Check if SE is satisfied.
- Output pixel is set to 0 or 1 depending on used operation.



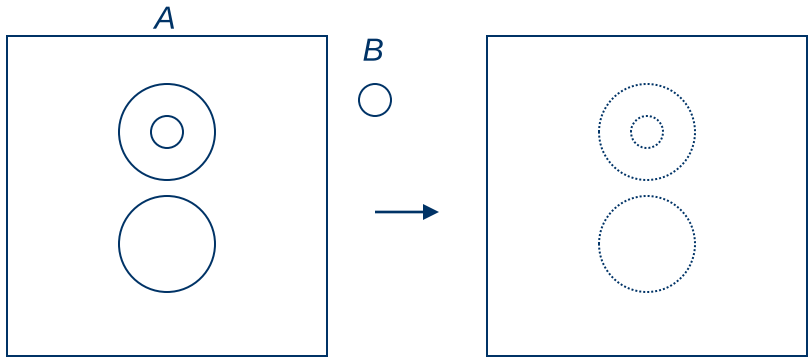
1D grayscale image $f(s)$ dilated by structuring element $b(x)$.

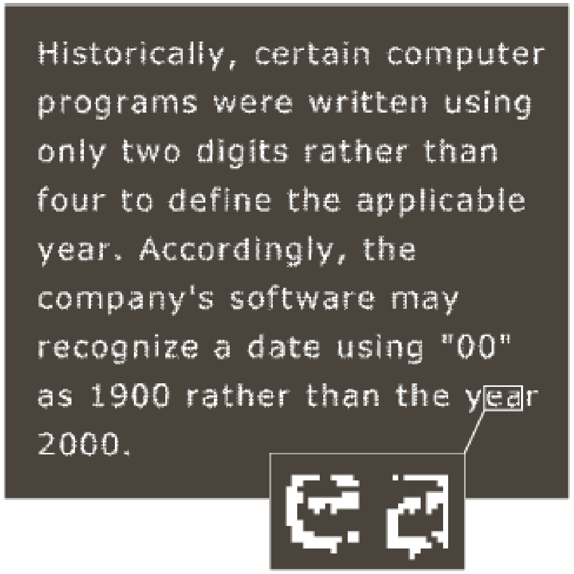


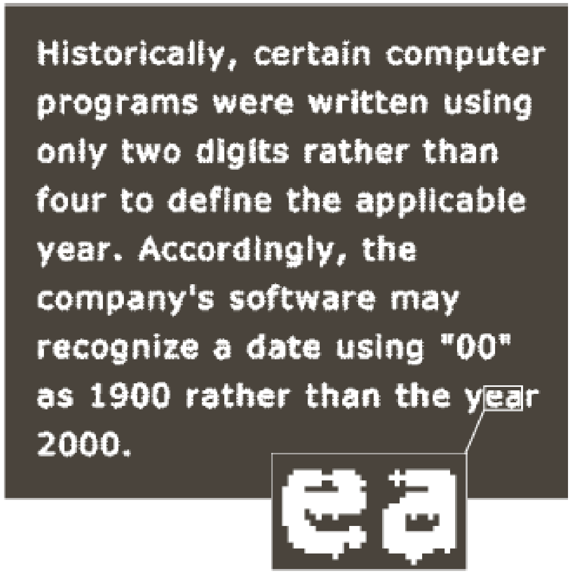
$$[f \oplus b](s) = \max_{x \in b} \{f(s-x) + b(x)\}$$

Dilation

- Dilation fills in holes, thickens thin parts, grows object







0	1	0
1	1	1
0	1	0

a c
b

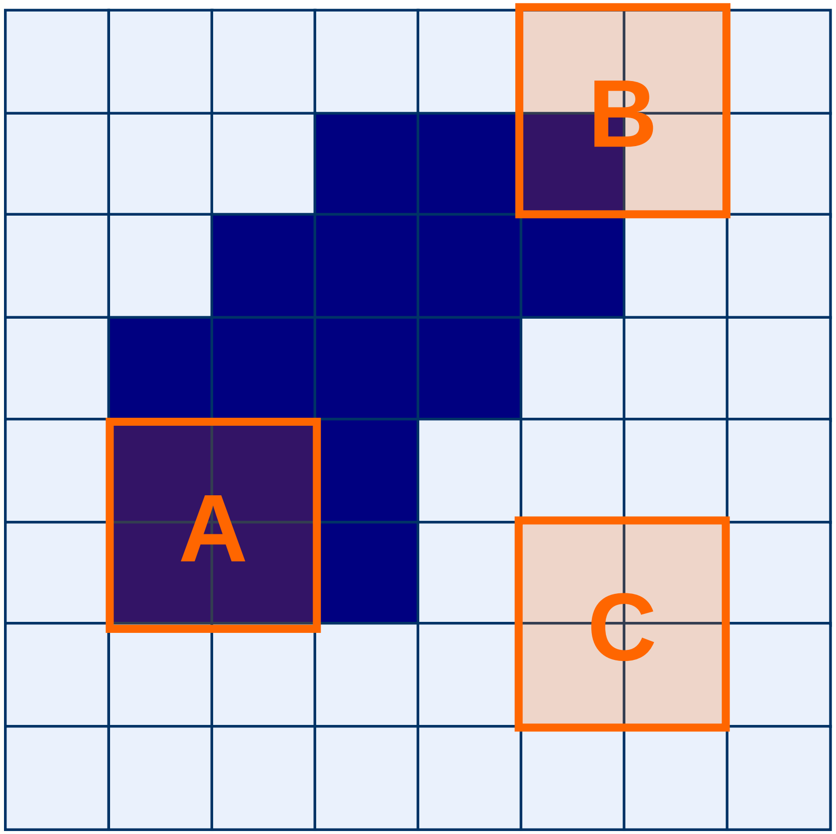
FIGURE 9.7 (a) Sample text of poor resolution with broken characters (see magnified view). (b) Structuring element. (c) Dilation of (a) by (b). Broken segments were joined.

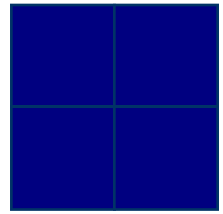
Quick Example

Image after segmentation

Image after segmentation and morphological processing

Structuring Elements, Hits & Fits





Structuring Element

Fit: All *on pixels* in the structuring element cover *on pixels* in the image

Hit: Any *on pixel* in the structuring element covers an *on pixel* in the image

All morphological processing operations are based on these simple ideas

Structuring Elements

Structuring elements can be any size and make any shape

However, for simplicity we will use rectangular structuring elements with their origin at the middle pixel

1	1	1
1	**1**	1
1	1	1

0	1	0
1	**1**	1
0	1	0

0	0	1	0	0
0	1	1	1	0
1	1	**1**	1	1
0	1	1	1	0
0	0	1	0	0

Fitting & Hitting

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0
0	0	1	1 **B**	1	1	1	0	0 **C**	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1 **A**	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0

1	1	1
1	1	1
1	1	1

Structuring Element 1

0	1	0
1	1	1
0	1	0

Structuring Element 2

Fundamental Operations

Fundamentally morphological image processing is very like spatial filtering

The structuring element is moved across every pixel in the original image to give a pixel in a new processed image

The value of this new pixel depends on the operation performed

There are two basic morphological operations: **erosion** and **dilation**

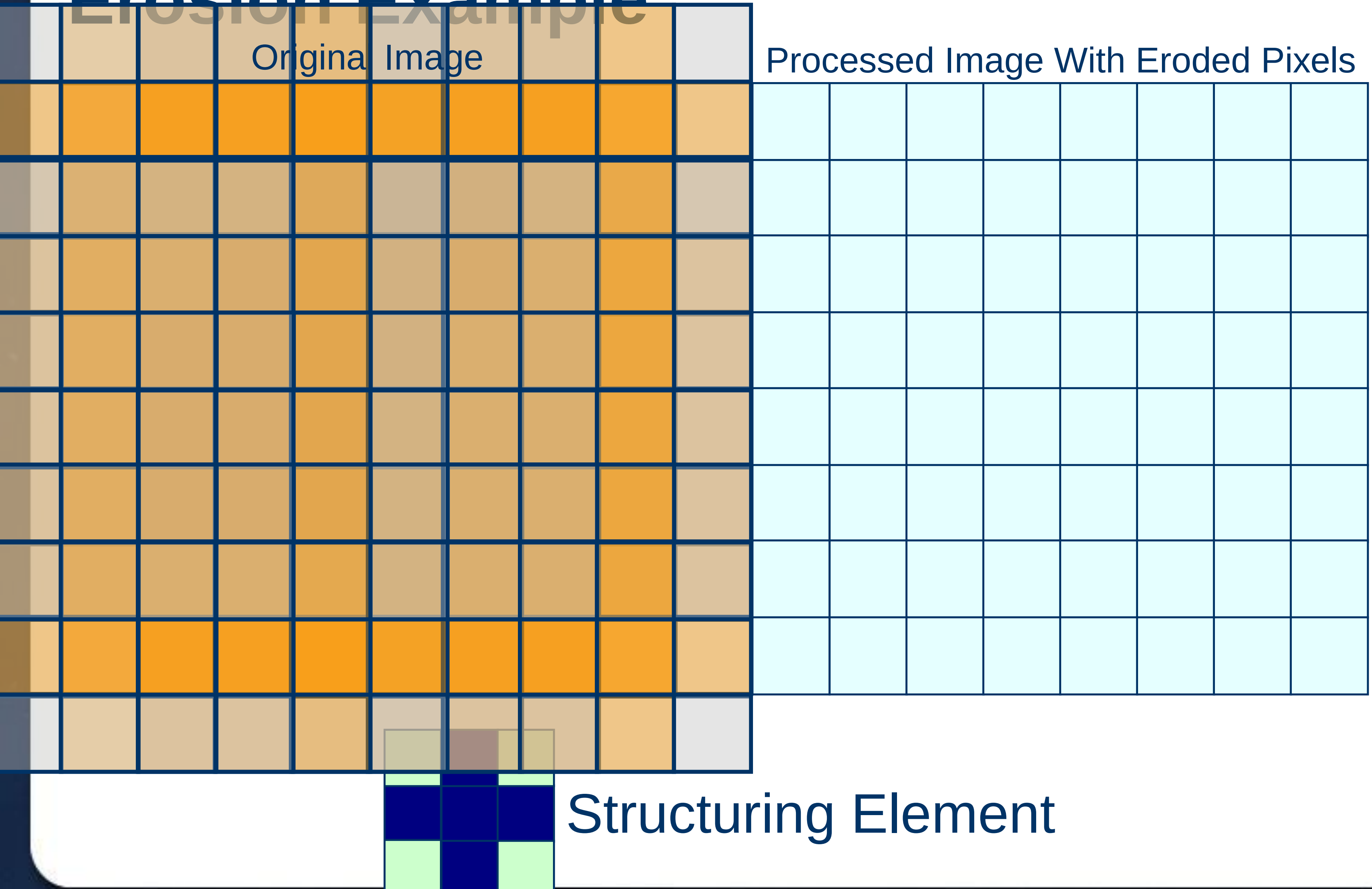
Erosion

Erosion of image f by structuring element s is given by $f \ominus s$

The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

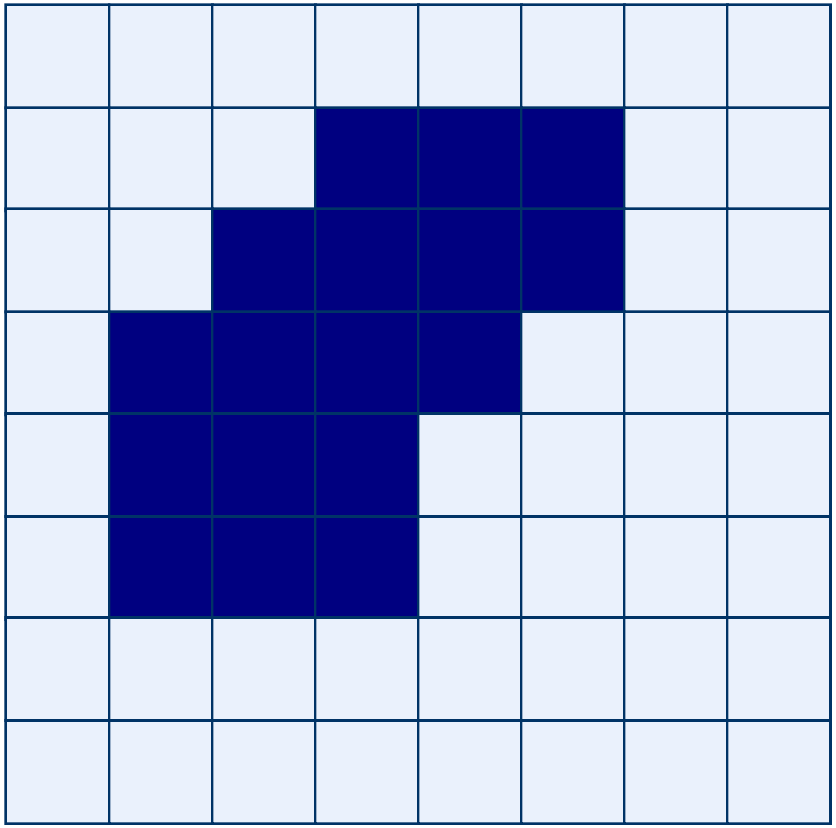
$$g(x, y) = \begin{cases} 1 \text{ if } s \text{ fits } f \\ 0 \text{ otherwise} \end{cases}$$

Erosion Example

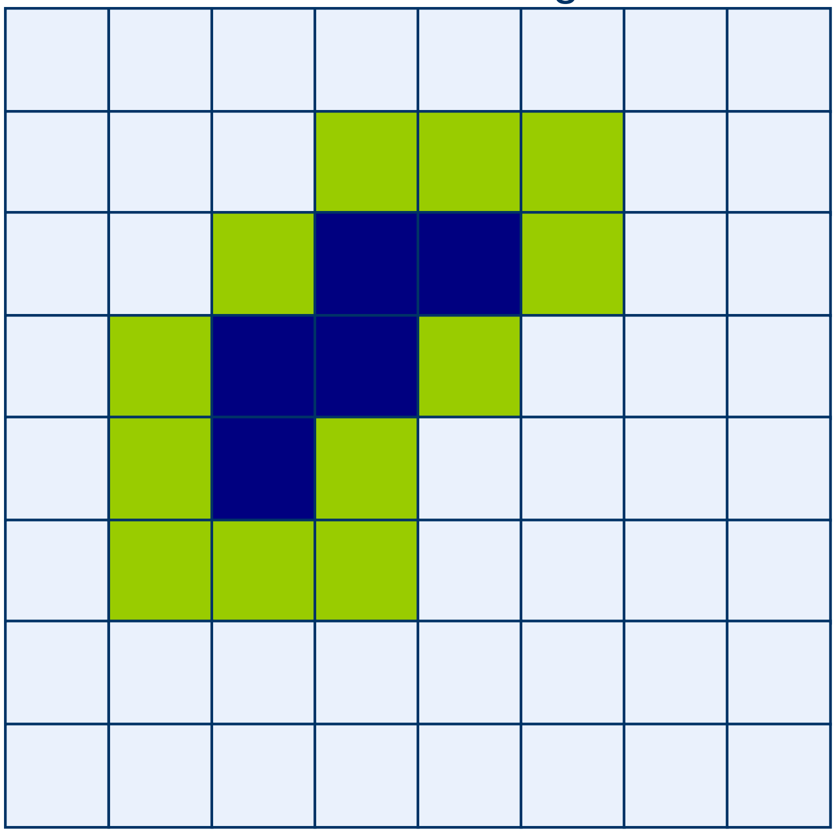


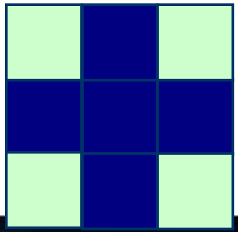
Erosion Example

Original Image



Processed Image

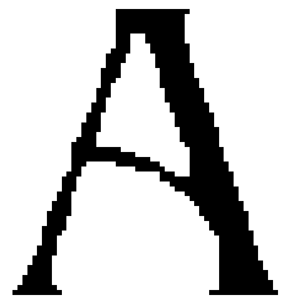




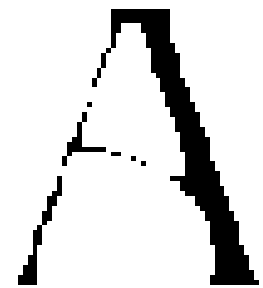
Structuring Element

Erosion Example 1

Original image



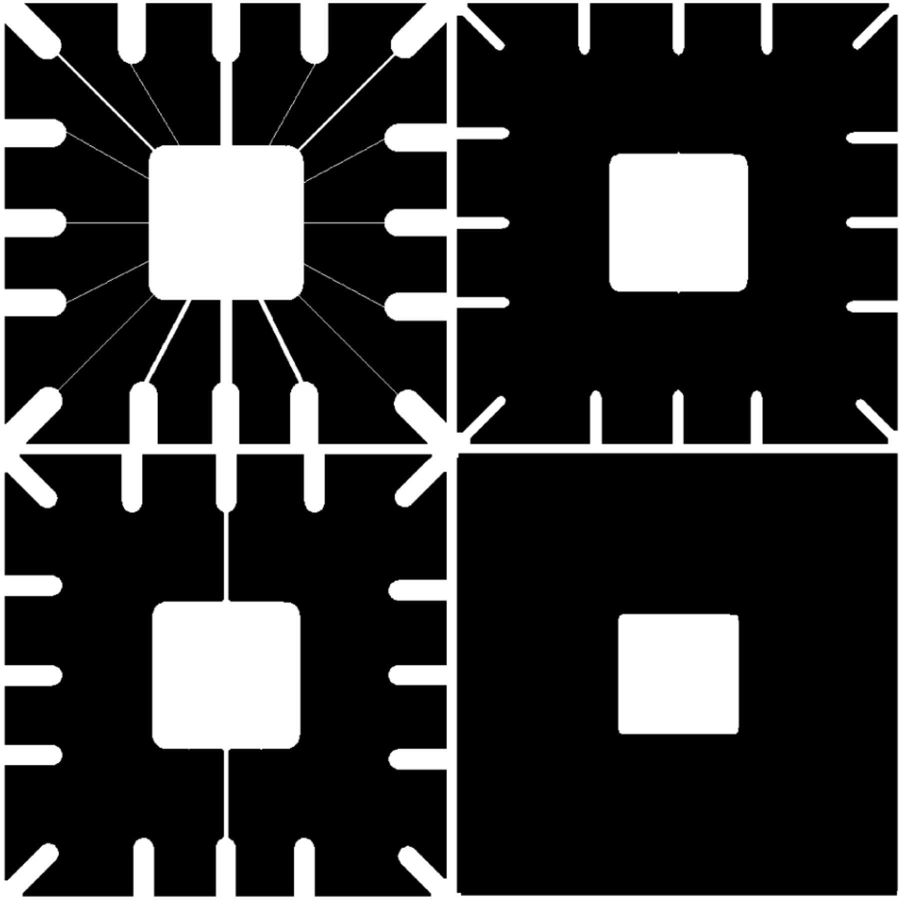
Erosion by 3*3 square structuring element



Erosion by 5*5 square structuring element

SINTEF

Erosion Example 2



Original image

After erosion with a disc of radius 10

After erosion with a disc of radius 5

After erosion with a disc of radius 20

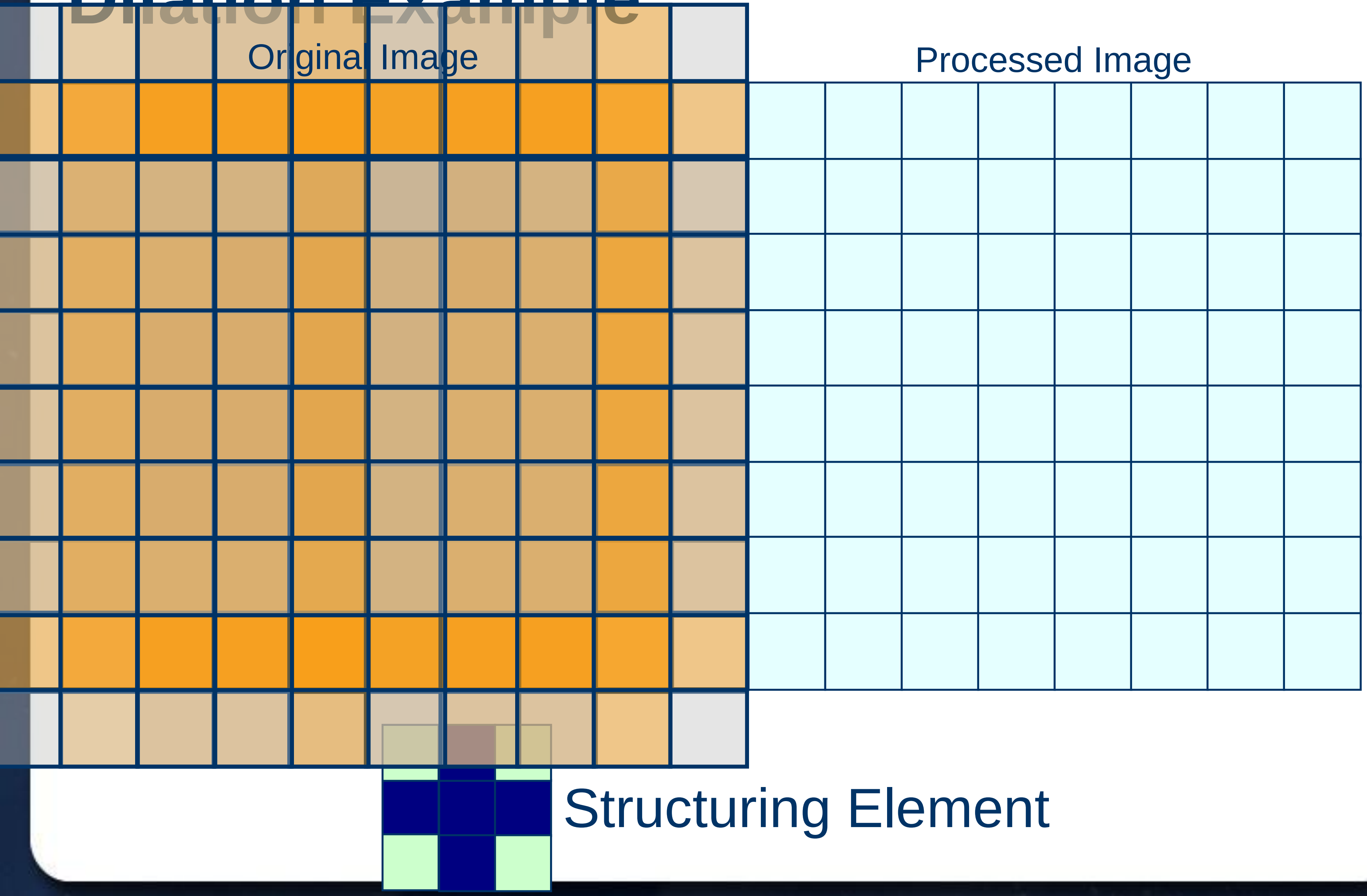
Dilation

Dilation of image f by structuring element s is given by $f \oplus s$

The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

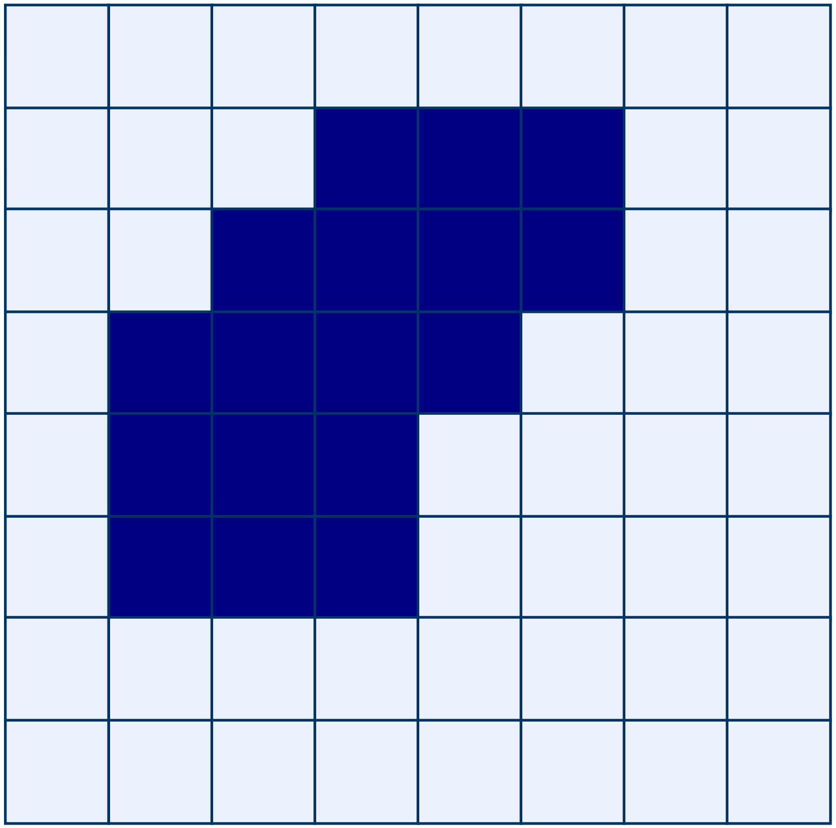
$$g(x, y) = \begin{cases} 1 \text{ if } s \text{ hits } f \\ 0 \text{ otherwise} \end{cases}$$

Dilation Example

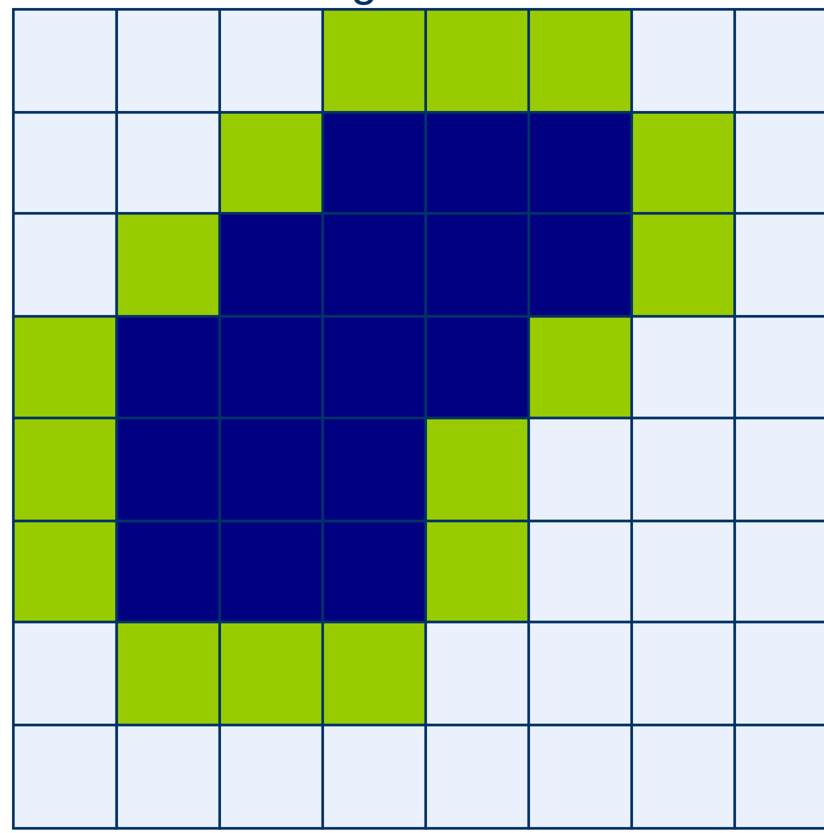


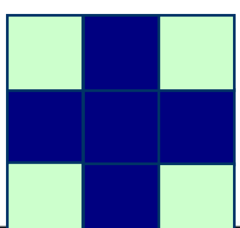
Dilation Example

Original Image



Processed Image With Dilated Pixels





Structuring Element

Dilation Example 1

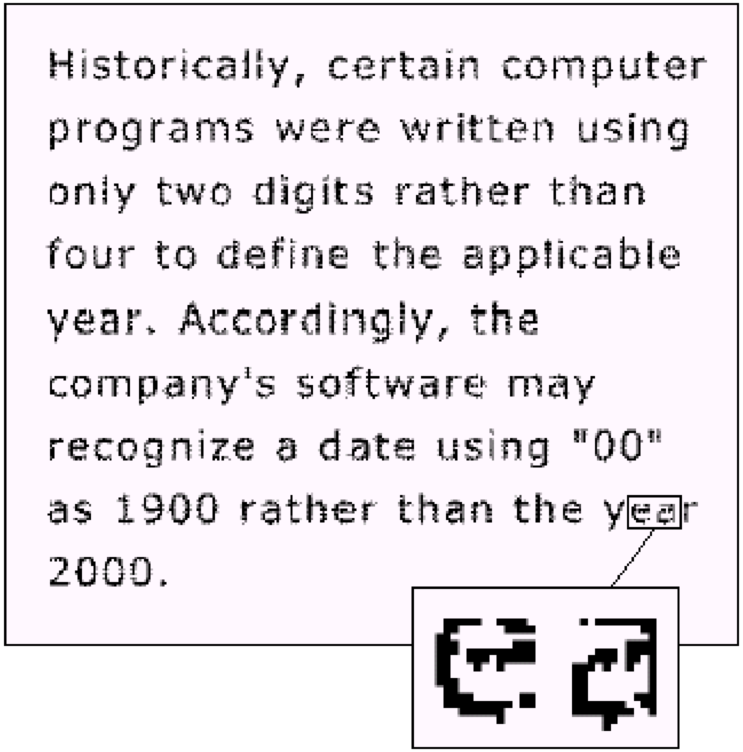
Original image

Dilation by 3*3 square structuring element

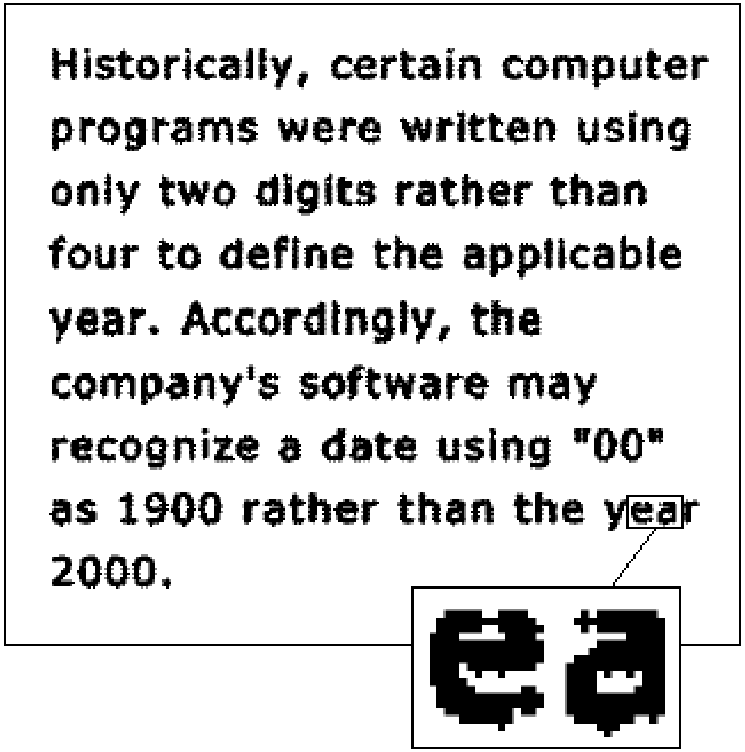
Dilation by 5*5 square structuring element

Dilation Example 2

Original image


Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

After dilation


Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

0	1	0
1	1	1
0	1	0

Structuring element

Compound Operations

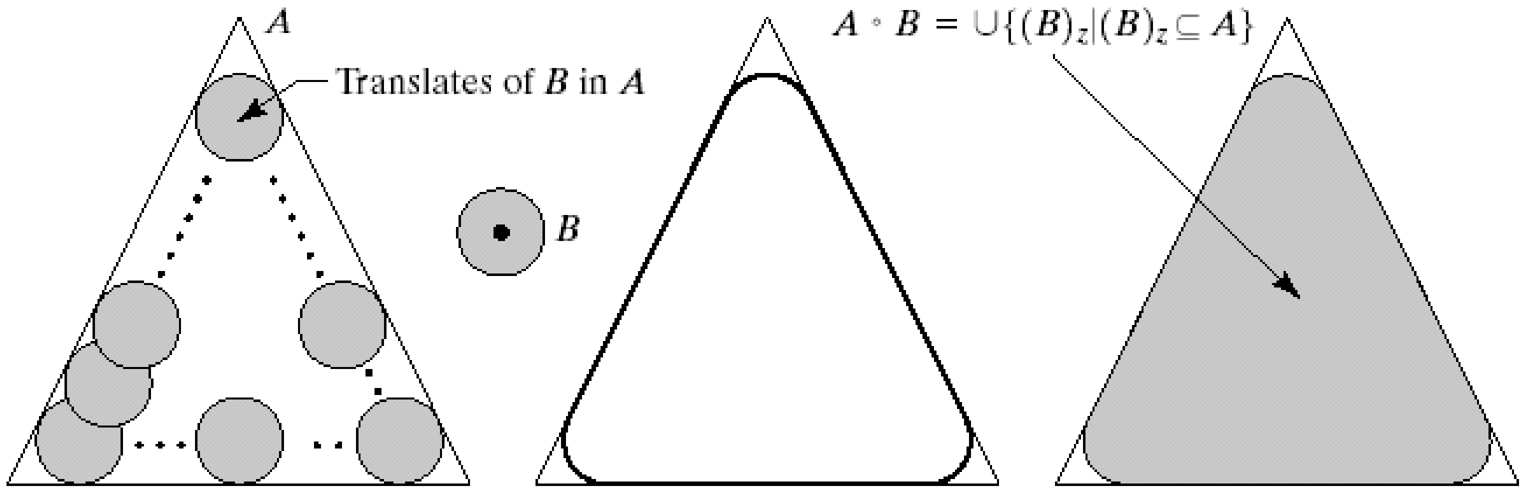
More interesting morphological operations can be performed by performing combinations of erosions and dilations

The most widely used of these *compound operations* are:

- Opening
- Closing

Opening

- Intuitive description
  - Let B be a disk
  - The boundary of the opening is the points in B that reach the farthest into A as B is rolled around inside of A



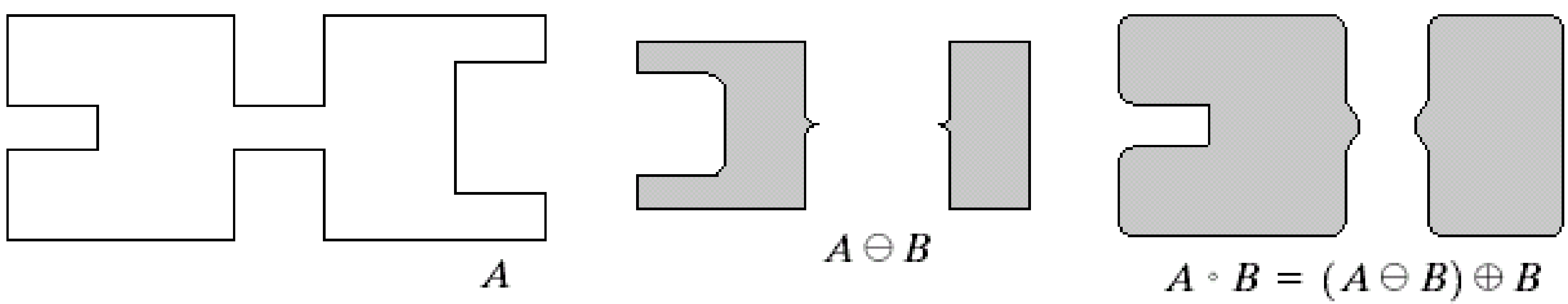
a b c d

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

The opening of image f by structuring element s, denoted $f \circ s$ is simply an erosion followed by a dilation

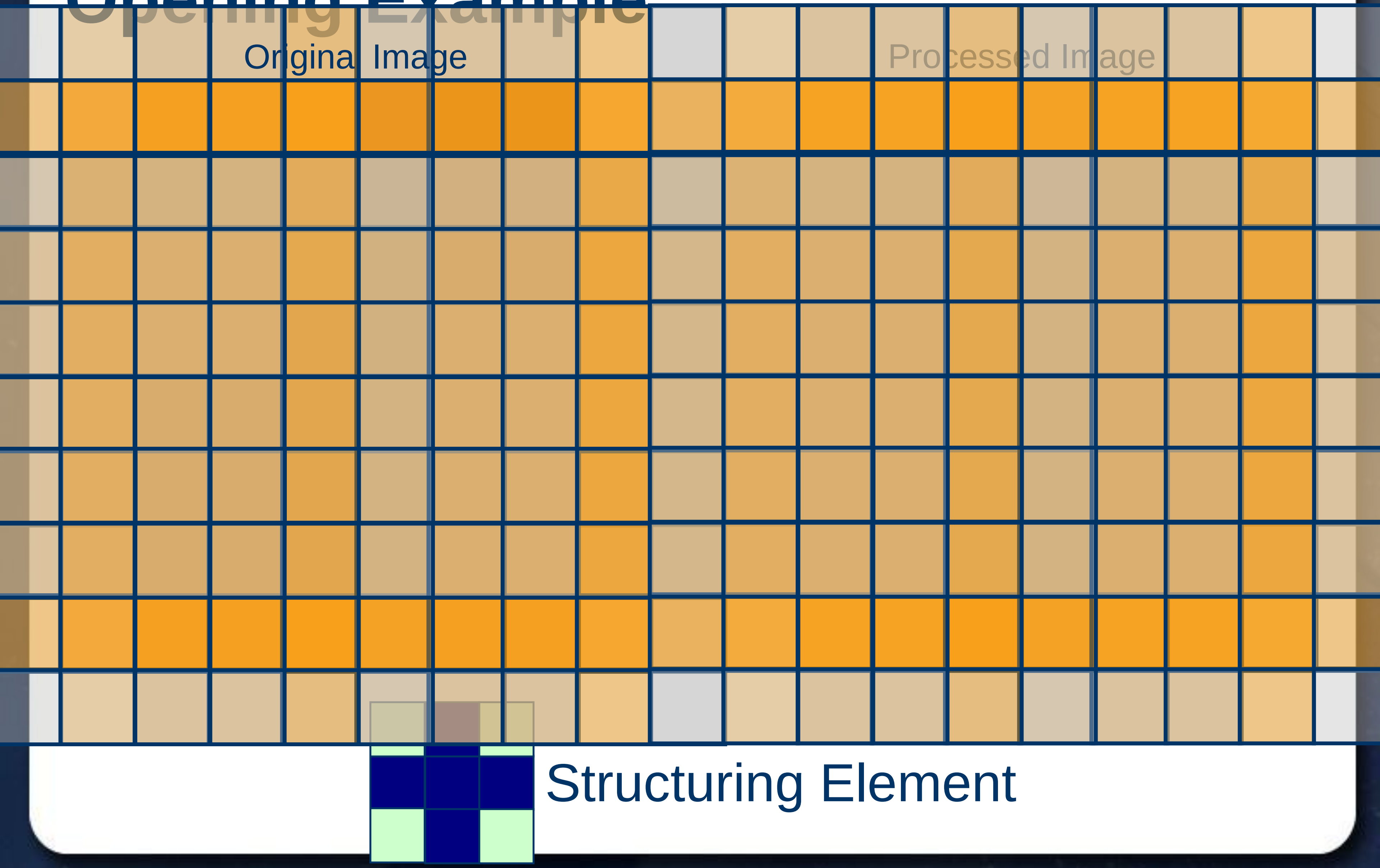
$$f \circ s = (f \ominus s) \oplus s$$



Original shape After erosion After dilation (opening)

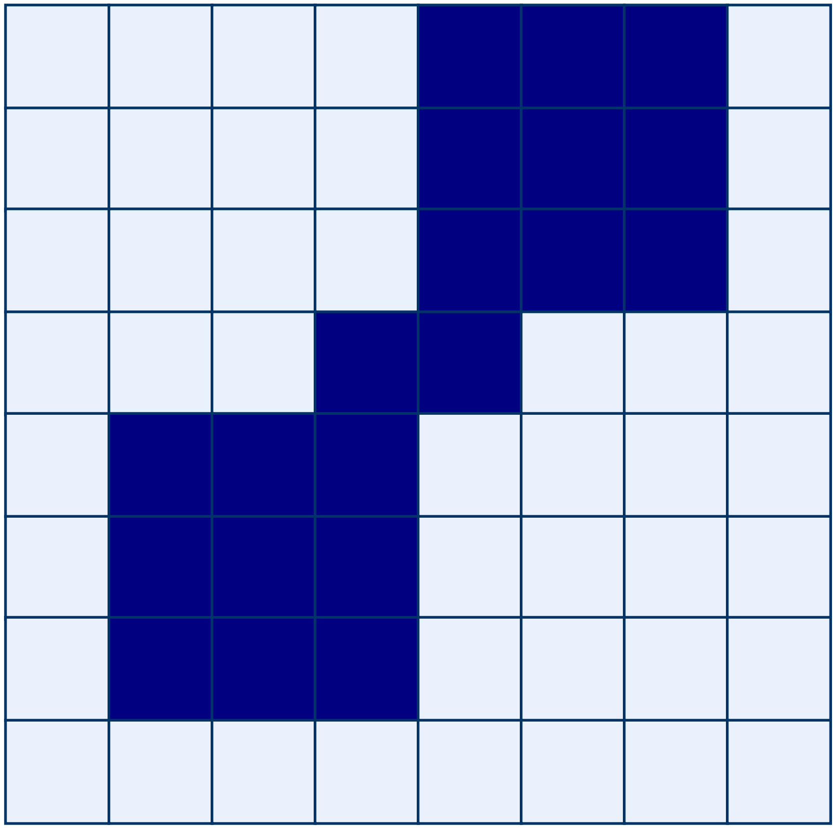
Note a disc shaped structuring element is used

Opening Example

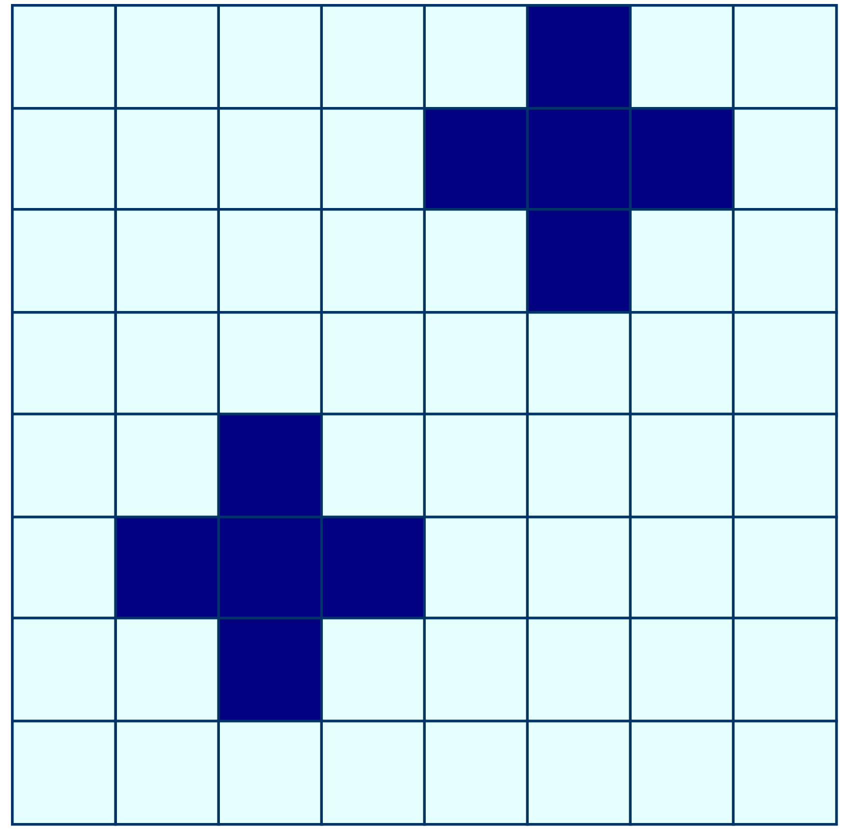


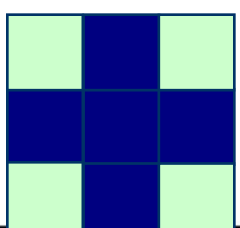
Opening Example

Original Image



Processed Image





Structuring Element

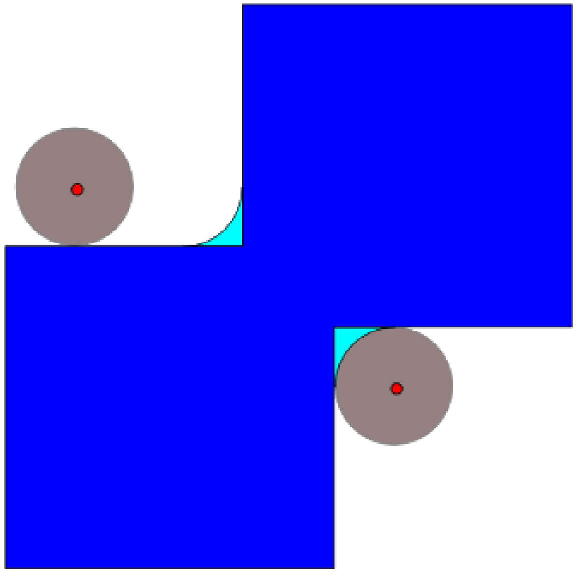
Closings

- Closing – a dilation followed by an erosion
  - Use the same structuring element B for both

$$A \bullet B = \left(A \oplus B\right) \theta\, B$$

- Take the union of all the translates of B that do not intersect A; the closing is the complement of that

The closing of the dark-blue shape (union of two squares) by a disk, resulting in the union of the dark-blue shape and the light-blue areas.



http://en.wikipedia.org/wiki/Mathematical_morphology

Closings

- Intuitive description
  - Let B be a disk
  - We roll B around the outside of A
  - The boundary of the closing is the points of B that just touch A

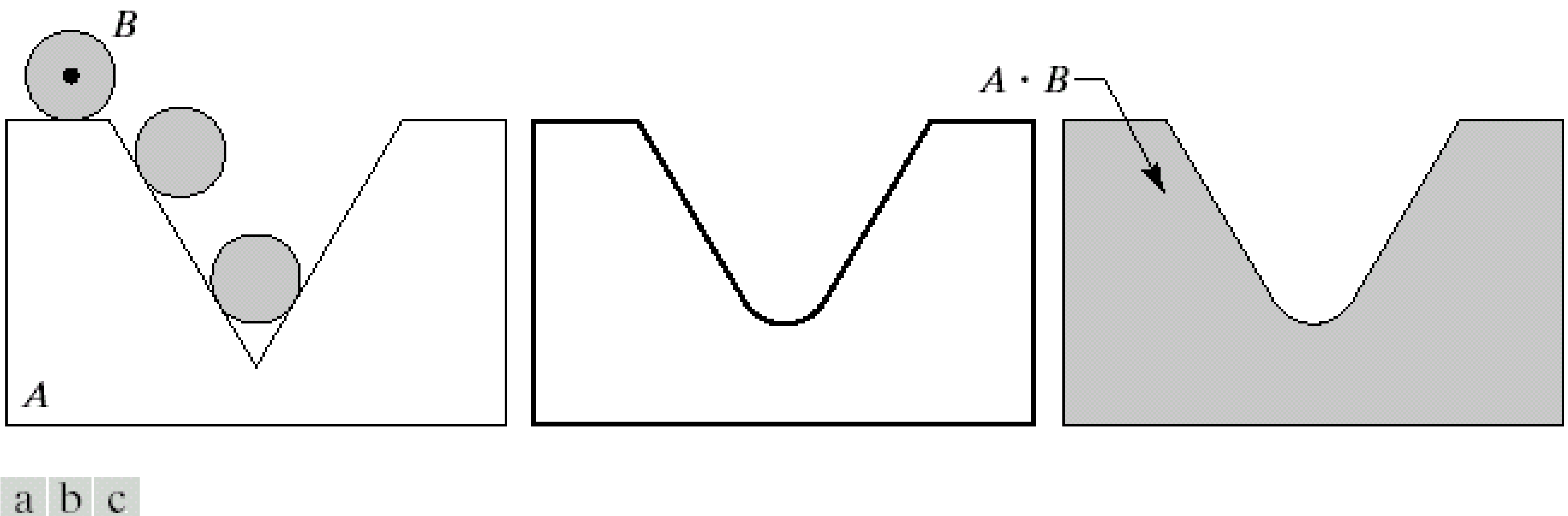
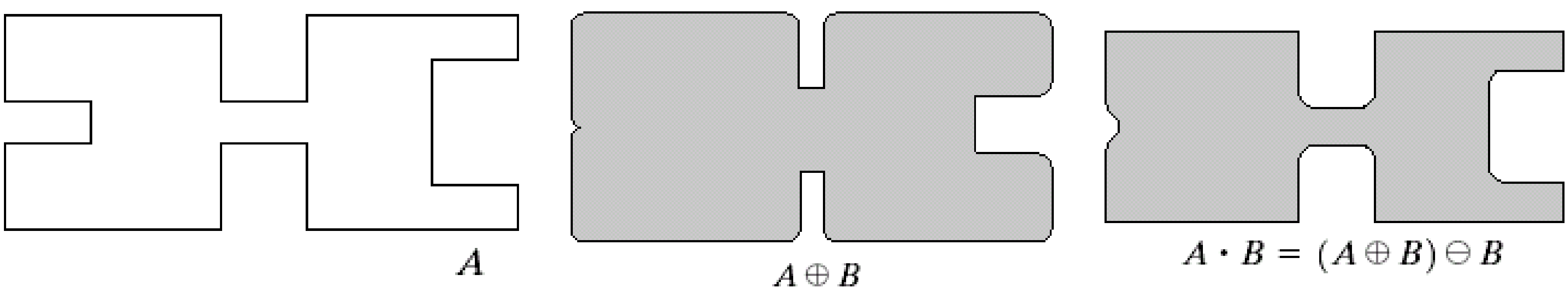


FIGURE 9.9 (a) Structuring element B "rolling" on the outer boundary of set A. (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).

Closing

The closing of image *f* by structuring element *s*, denoted $f \bullet s$ is simply a dilation followed by an erosion

$$f \bullet s = (f \oplus s) \ominus s$$



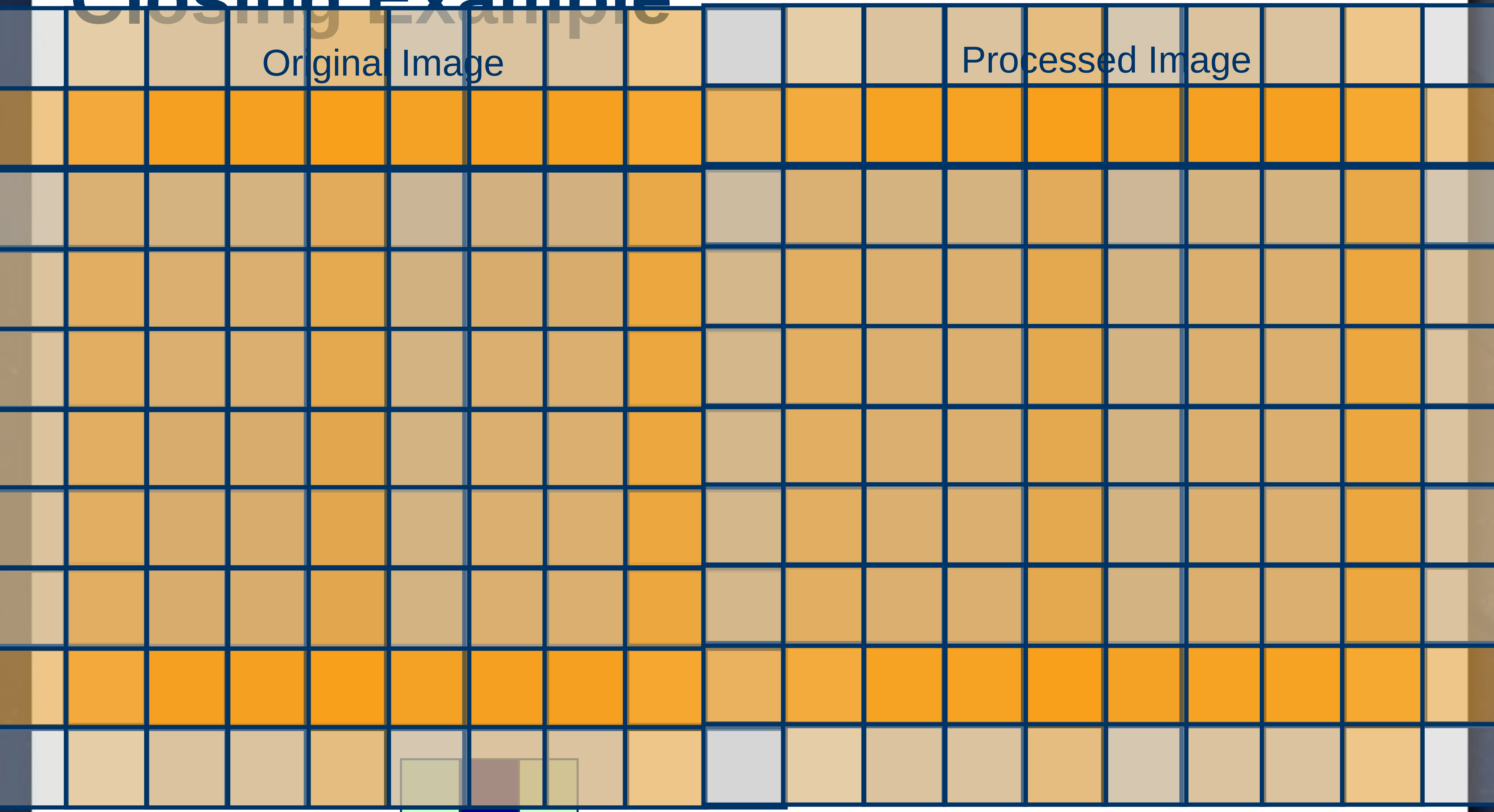
Original shape

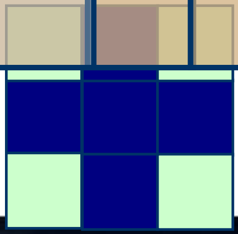
After dilation

After erosion (closing)

Note a disc shaped structuring element is used

Closing Example

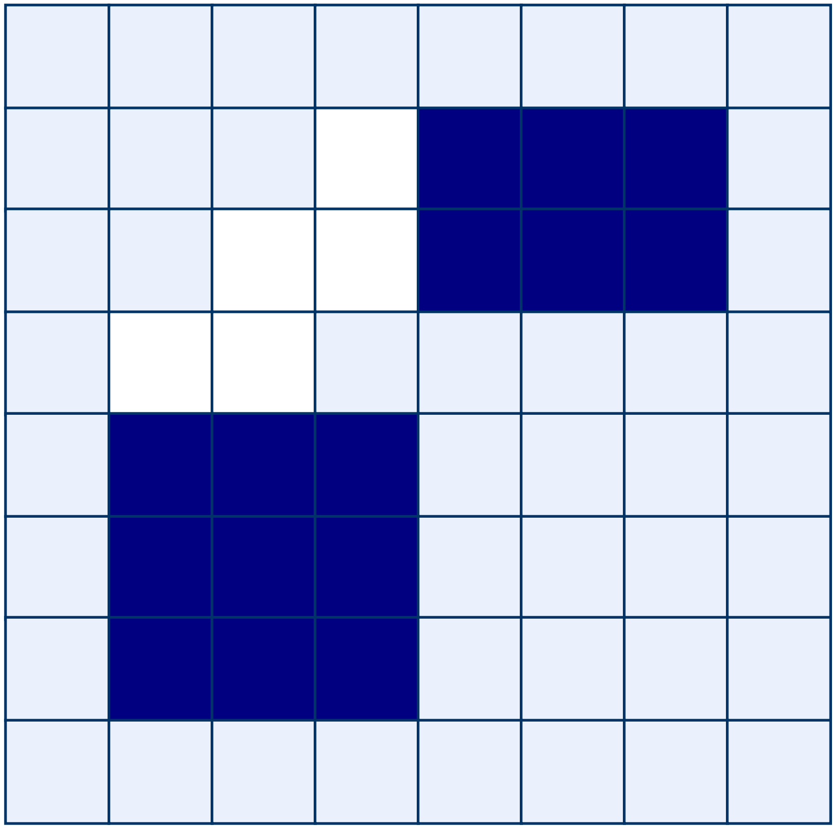




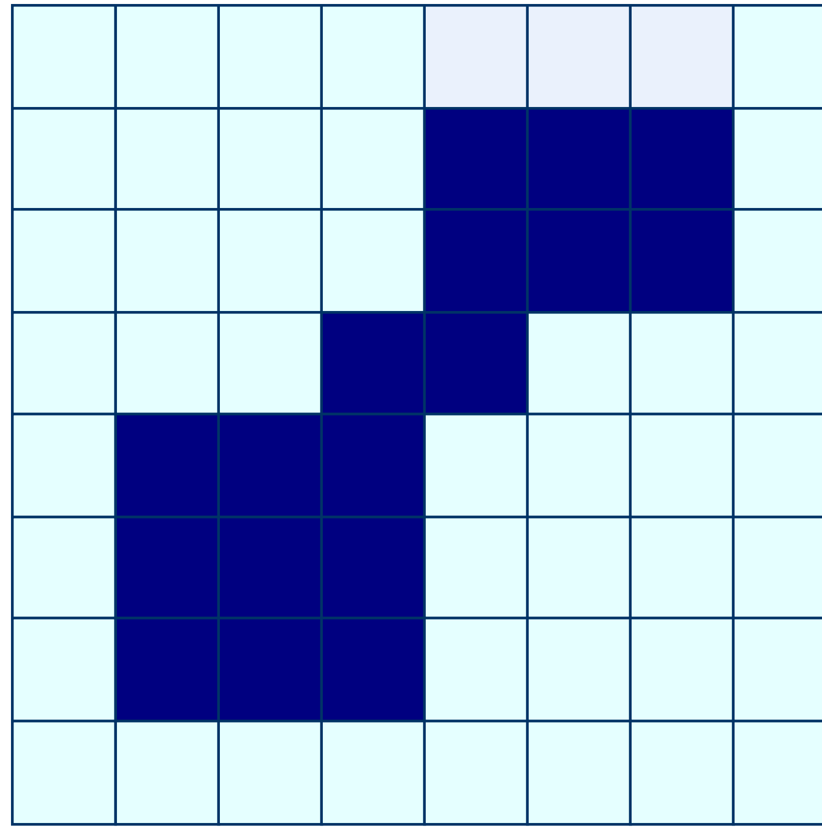
Structuring Element

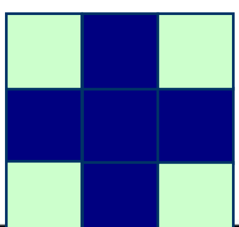
Closing Example

Original Image



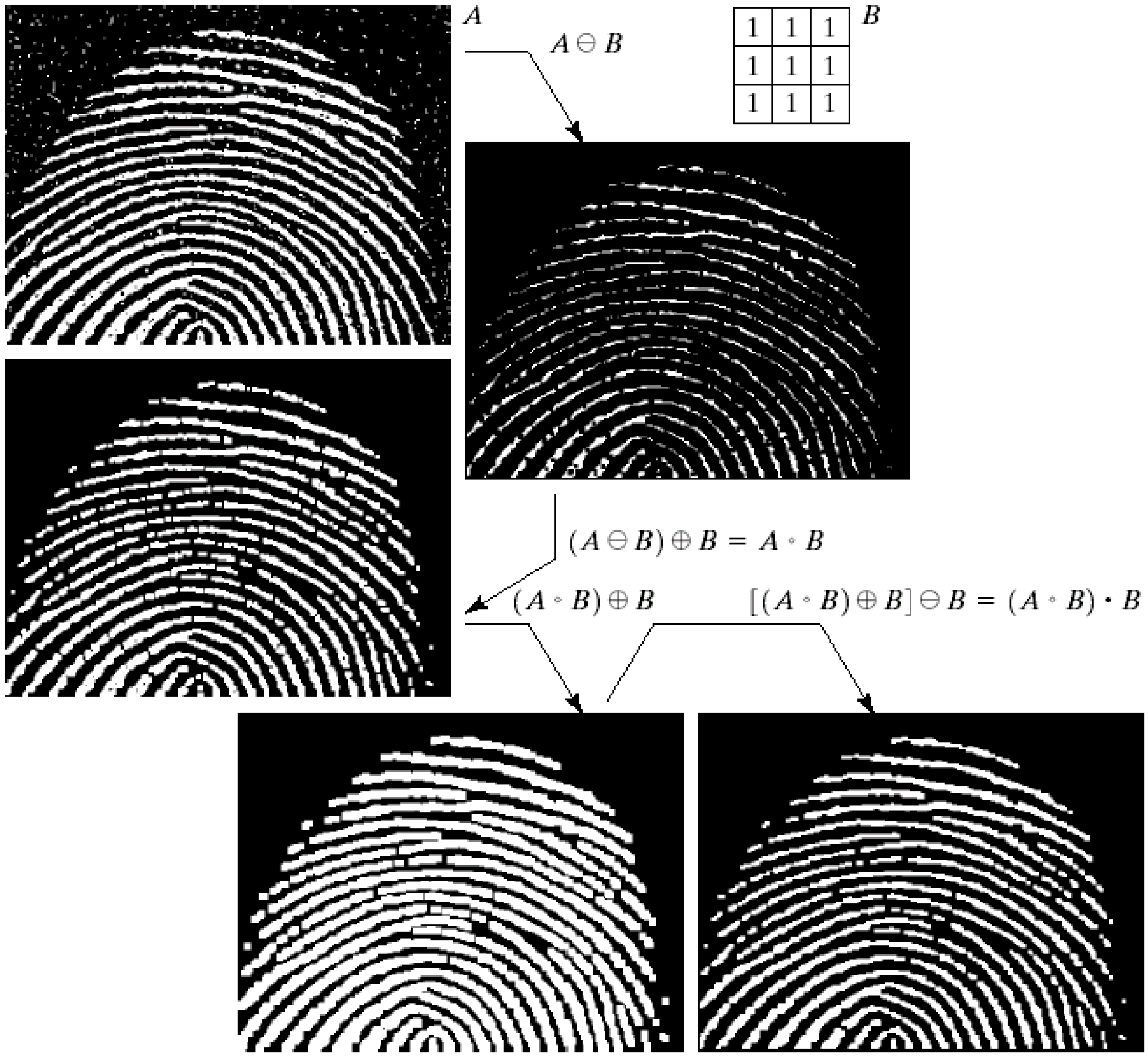
Processed Image



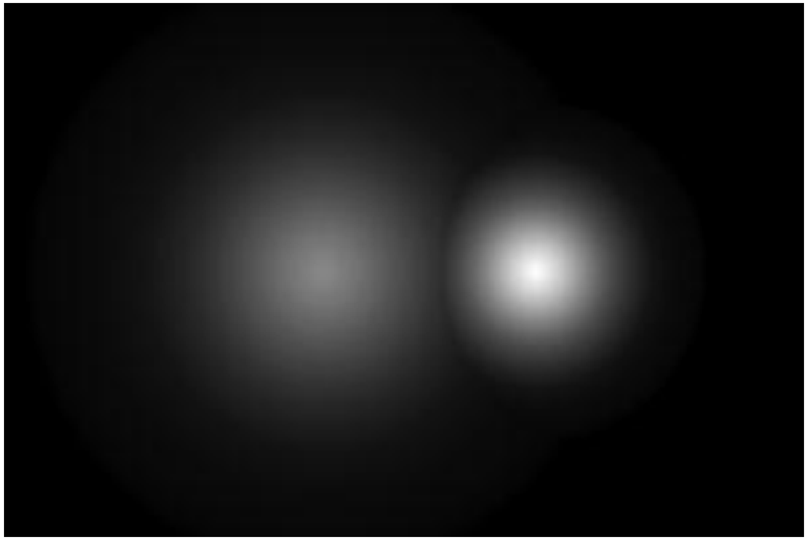


Structuring Element

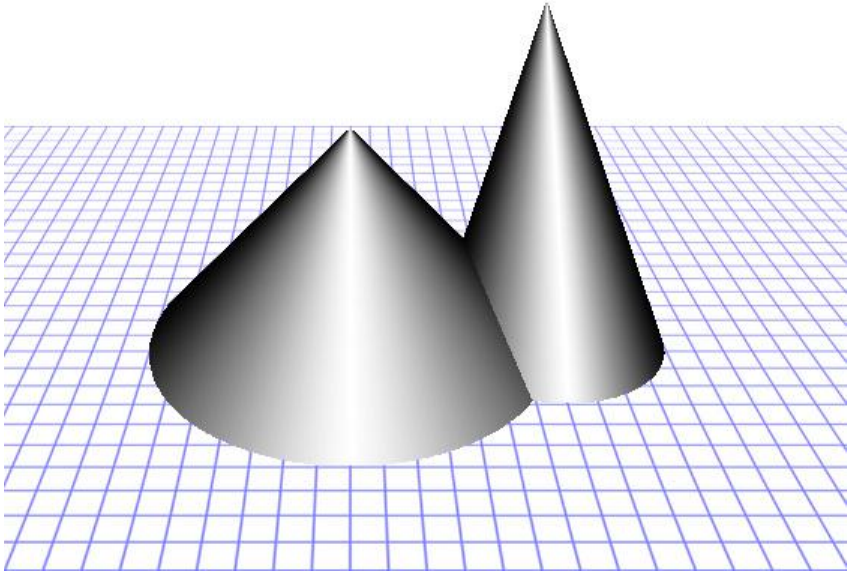
Morphological Processing Example



Grayscale Morphology: image



image



landscape

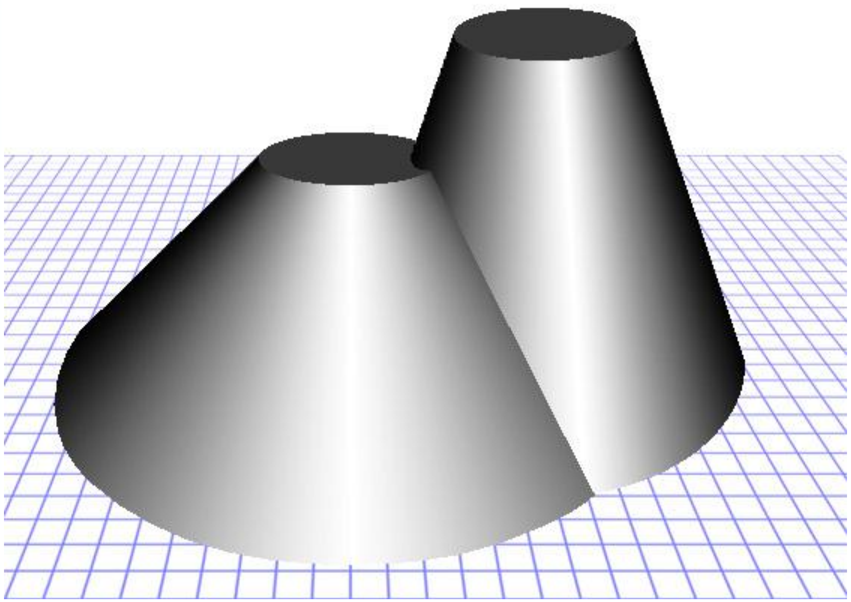
Grayscale image and 3D solid representation

Grayscale Dilation

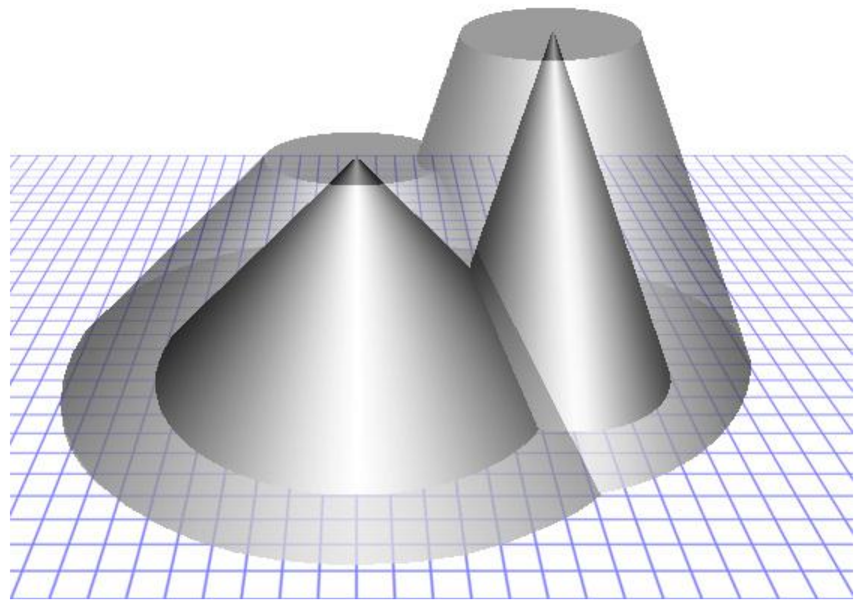
- Grayscale Dilation: A grayscale image *F* dilated by a grayscale SE *K* is defined as:

$$D_G(F,K) = F \oplus_g K = \max_{[a,b]\in K}\left\{F(m-a,n-b)+K(a,b)\right\}$$

- It generally brightens the source image.



dilation



dilation over original

Grayscale Dilation

Source image

Dilated image

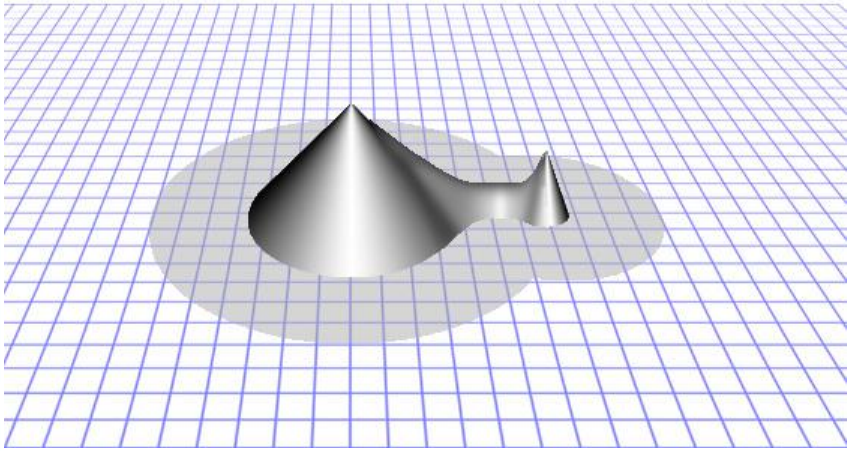
SINTEF

Grayscale Erosion

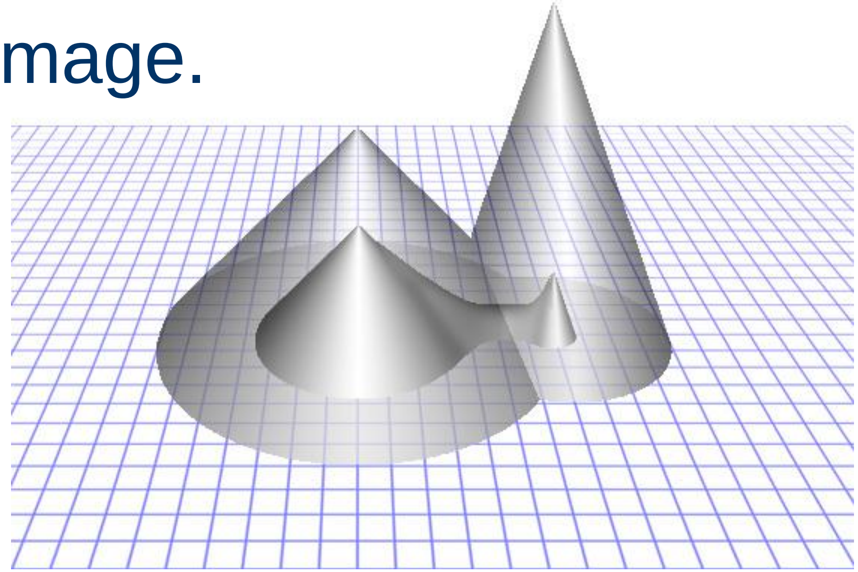
- Grayscale Erosion: A grayscale image *F* eroded by a grayscale SE *K* is defined as:

$$E_G(F,K) = F \ominus_g K = \min_{[a,b]\in K}\{F(m+a,n+b)-K(a,b)\}$$

- It generally darken the image.



erosion



erosion under original

Grayscale Erosion

Source image

Eroded image

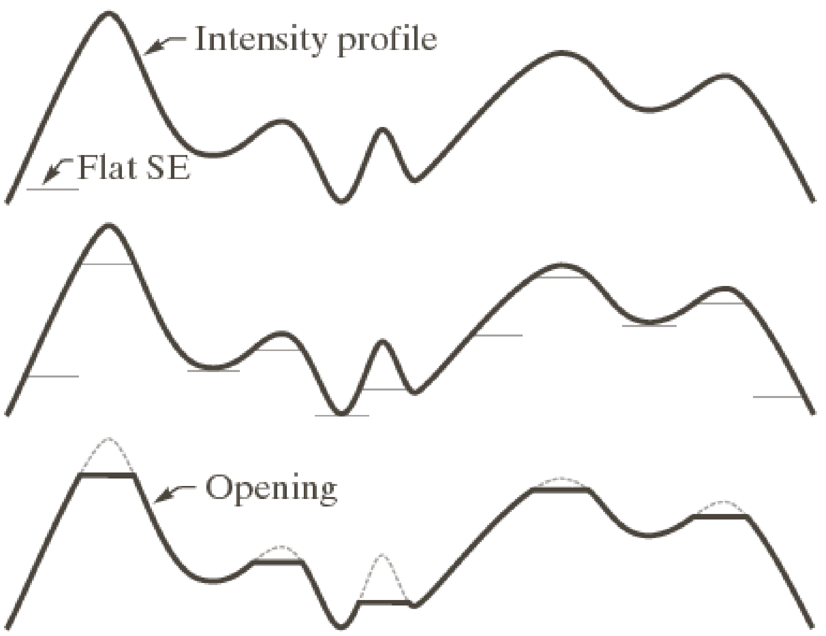
Opening and Closing

- Similar to binary case
  - Opening is erosion followed by dilation
  - Closing is dilation followed by erosion

$$f \circ b = (f \ominus b) \oplus b$$
$$f \bullet b = (f \oplus b) \ominus b$$

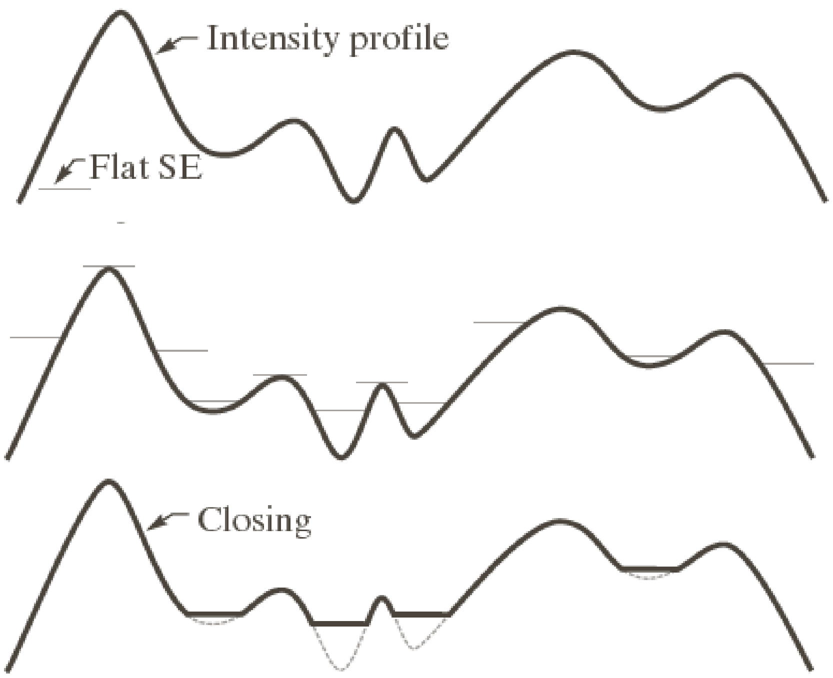
- Geometric interpretation of **opening**
  - Push the SE up from below against the underside of f
  - Take the highest values achieved at every point

Opening removes small, bright details



Opening and Closing

- Geometric interpretation of **closing**
  - Push the SE down from above against the topside of f
  - Take the lowest values achieved at every point



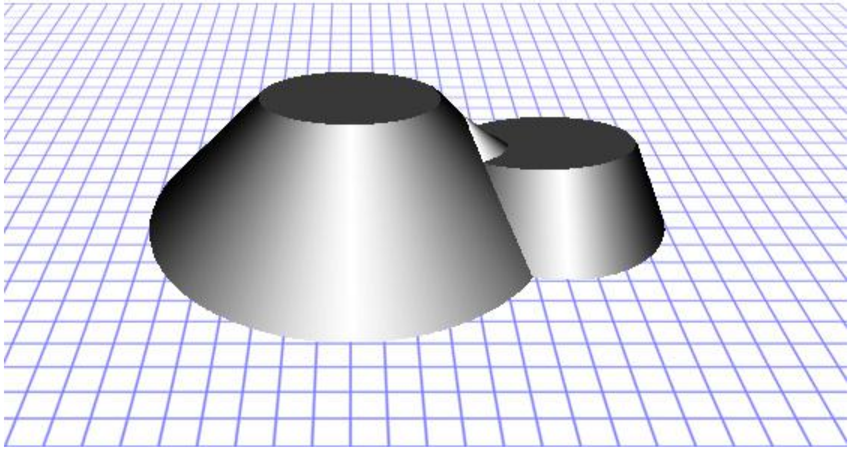
Closing removes small, dark details

Grayscale Opening

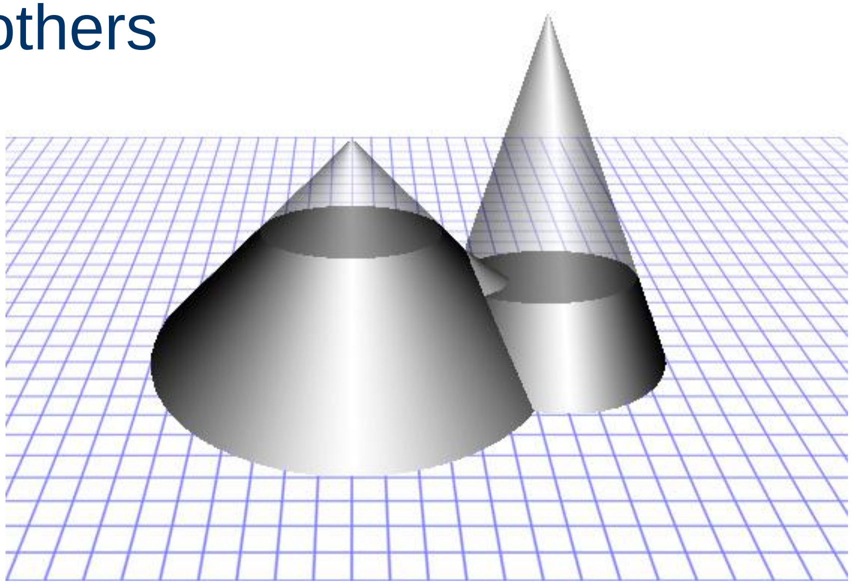
- Grayscale Opening: A grayscale image *F* opened by a grayscale SE *K* is defined as:

$$O_G(F,K) = F \circ_g K = (F \ominus_g K) \oplus_g K$$

- It can be used to select and preserve particular intensity patterns while attenuating others



opening



opened & original

Grayscale Opening

Source image

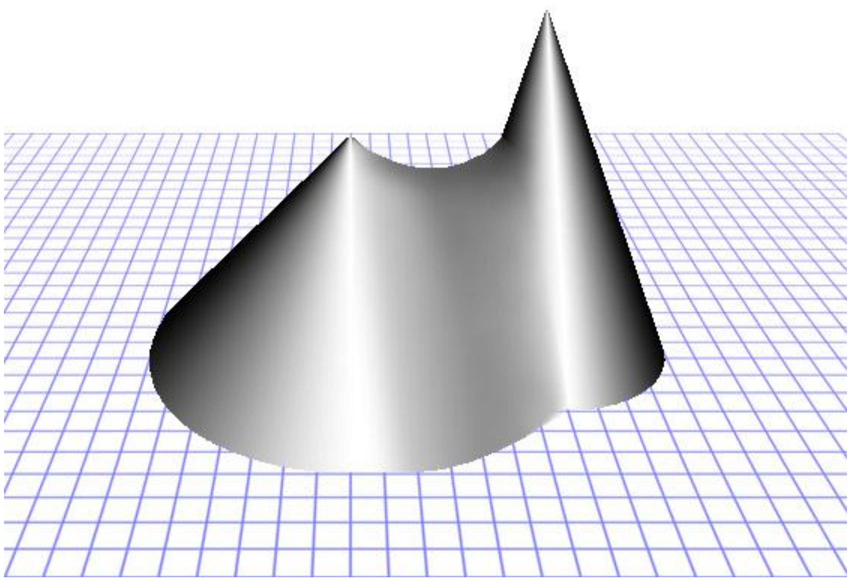
Opened image

Grayscale Closing

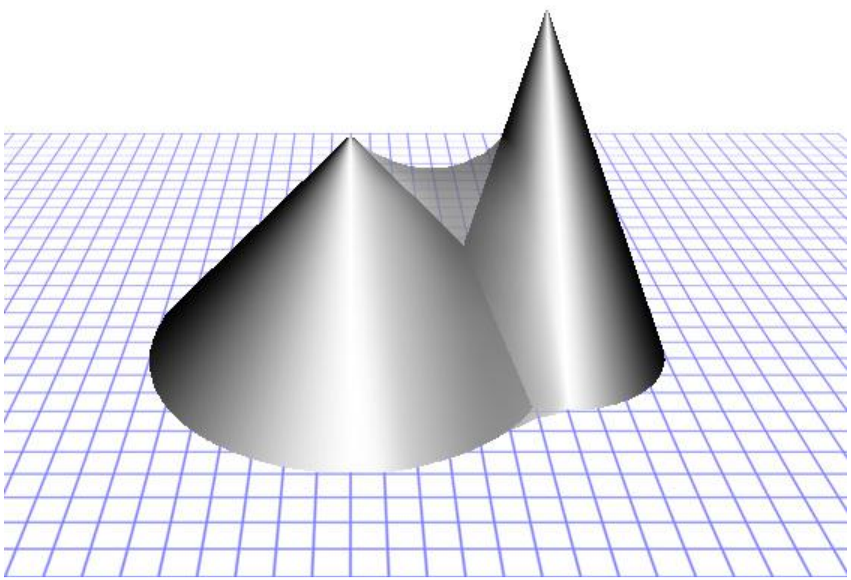
- Grayscale Closing: A grayscale image *F* closed by a grayscale SE *K* is defined as:

$$O_G(F,K) = F \bullet_g K = (F \oplus_g K) \ominus_g K$$

- It is another way to select and preserve particular intensity patterns while attenuating others.



closing



closing & original

Grayscale Closing

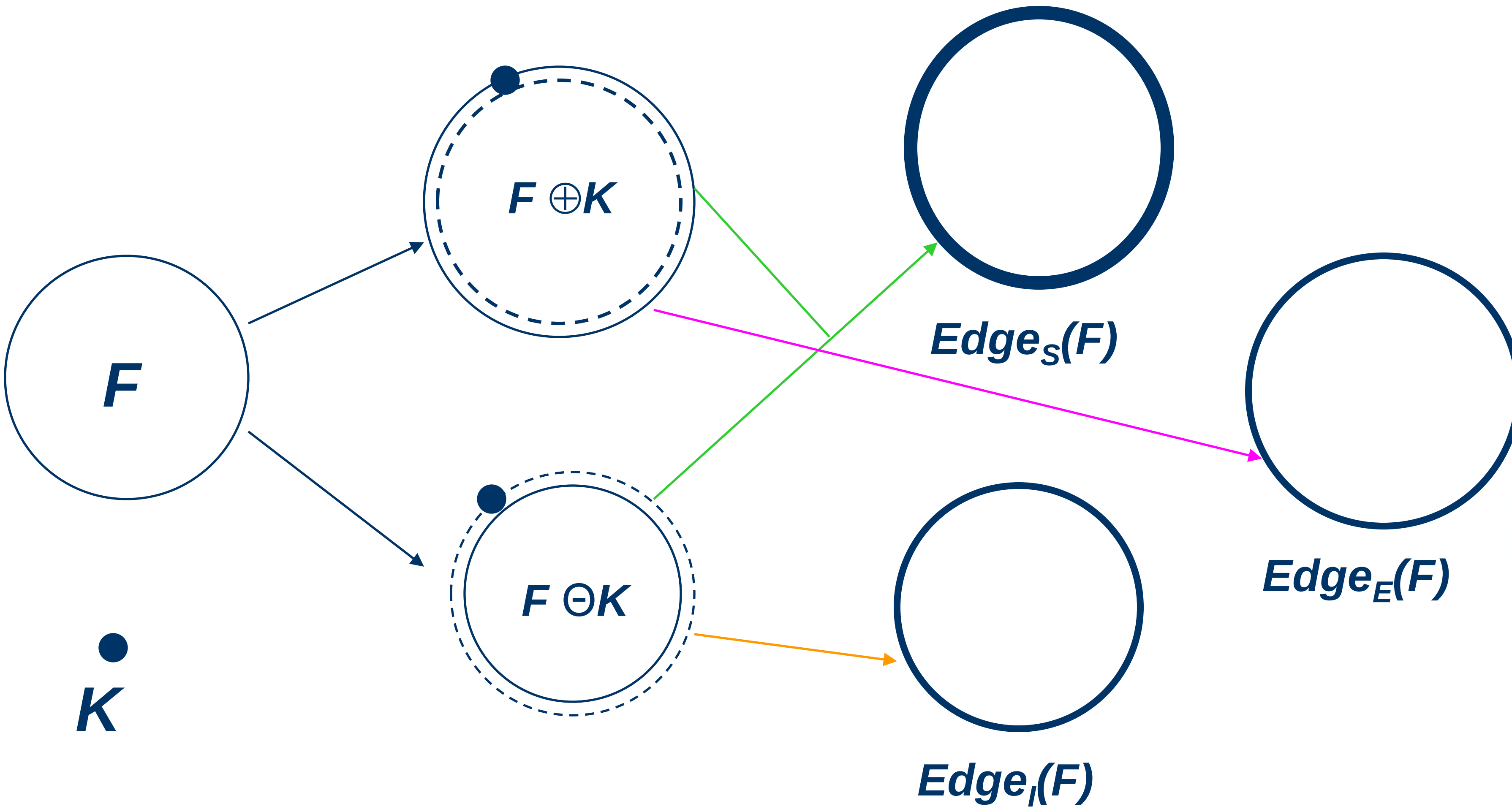
Source image

Closed image

Morphological Edge Detection

- Morphological Edge Detection is based on Binary Dilation, Binary Erosion and Image Subtraction.
- Morphological Edge Detection Algorithms:
  - Standard: $Edge_S(F) = (F \oplus K) - (F \ominus K)$
  - External: $Edge_E(F) = (F \oplus K) - F$
  - Internal: $Edge_I(F) = F - (F \ominus K)$

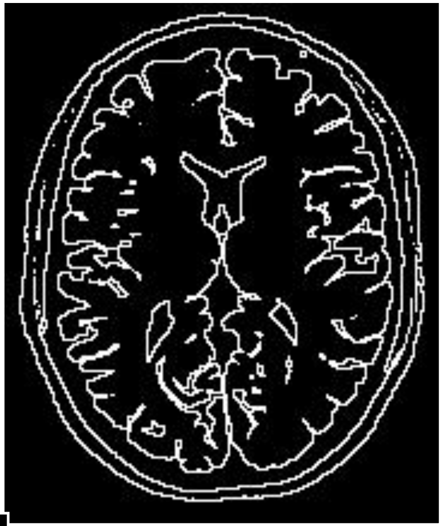
Morphological Edge Detection



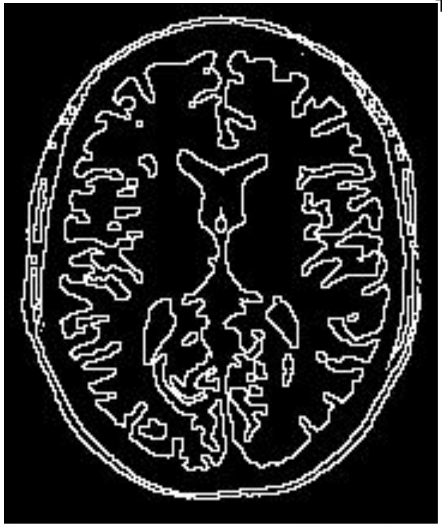
Morphological Edge Detection

F

$Edge_S(F)$



$Edge_E(F)$



$Edge_I(F)$

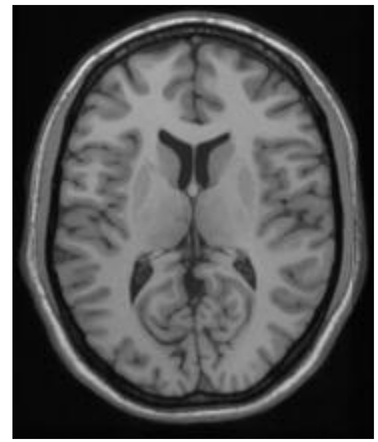
Morphological Gradient

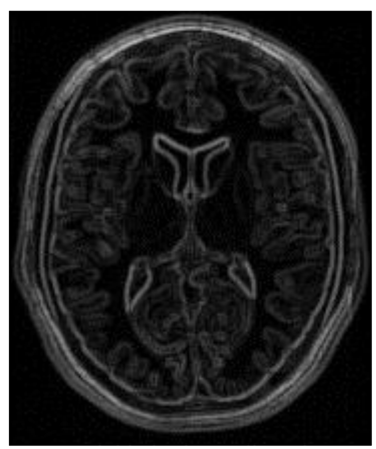
- Morphological Gradient is calculated by grayscale dilation and grayscale erosion

$$Gradient(F) = \frac{1}{2} D_g(F,K) - E_g(F,K)$$

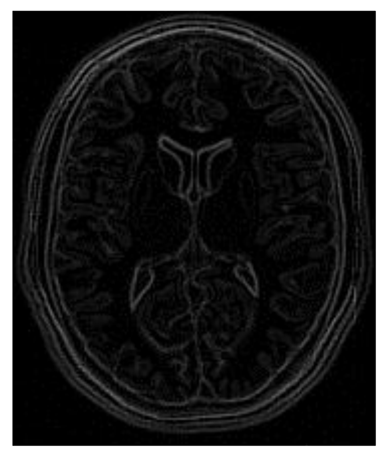
- It is quite similar to the standard edge detection
- We also have external and internal gradient

Morphological Gradient

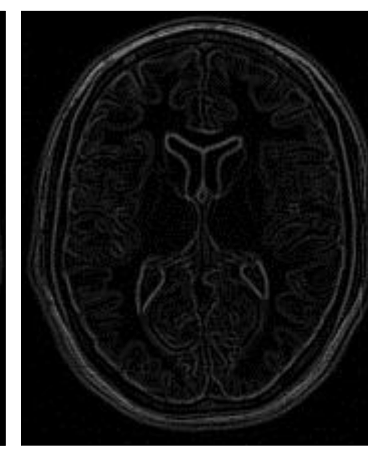




Standard



External



Internal

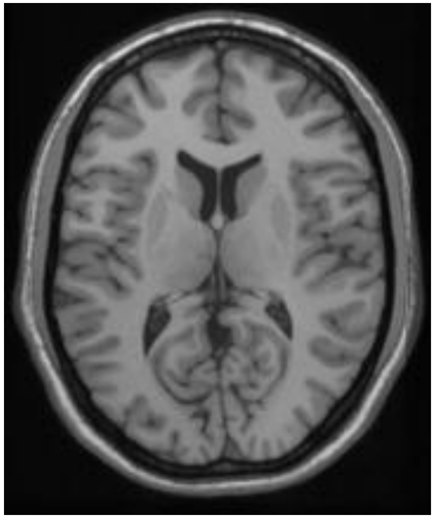
Morphological Smoothing

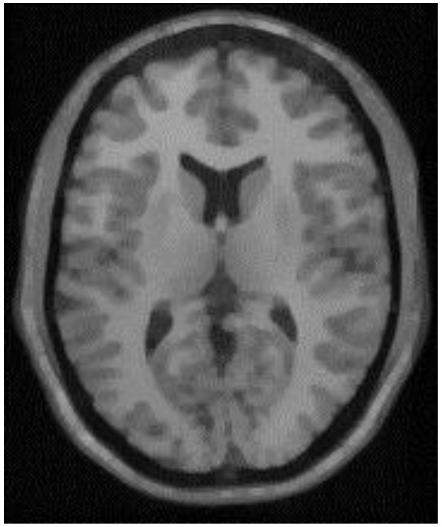
- Morphological Smoothing is based on the observation that a *grayscale opening* smoothes a grayscale image from above the brightness surface and the *grayscale closing* smoothes from below. So, the "smoothing sandwich" is:

$$Smooth(F) = C_g(O_g(F,K),K)$$

$$= ((F \circ_g K) \bullet_g K)$$

Morphological Smoothing





Top-hat Transform

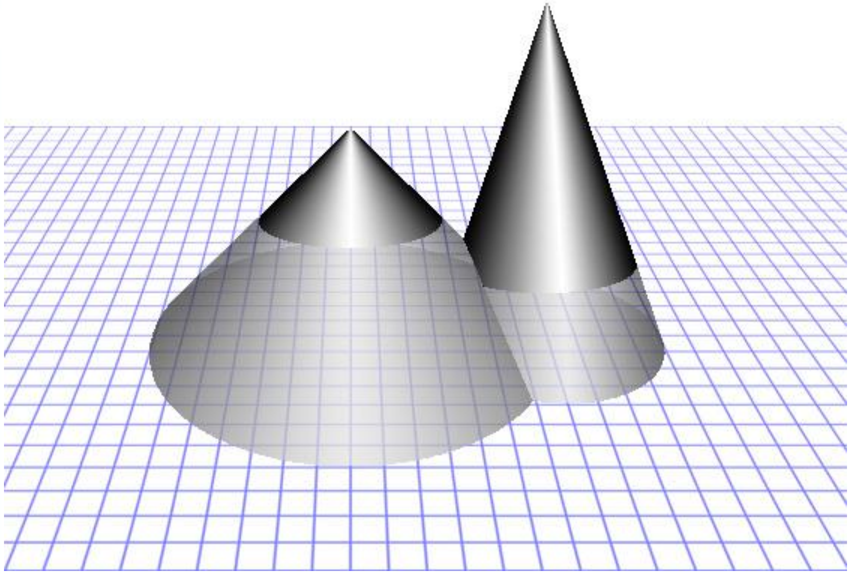
- Top-hat Transform (TT): An efficient segmentation tool for extracting bright (respectively dark) objects from uneven background.
  - White Top-hat Transform (WTT):

$$WTT = F - F \circ_g K$$

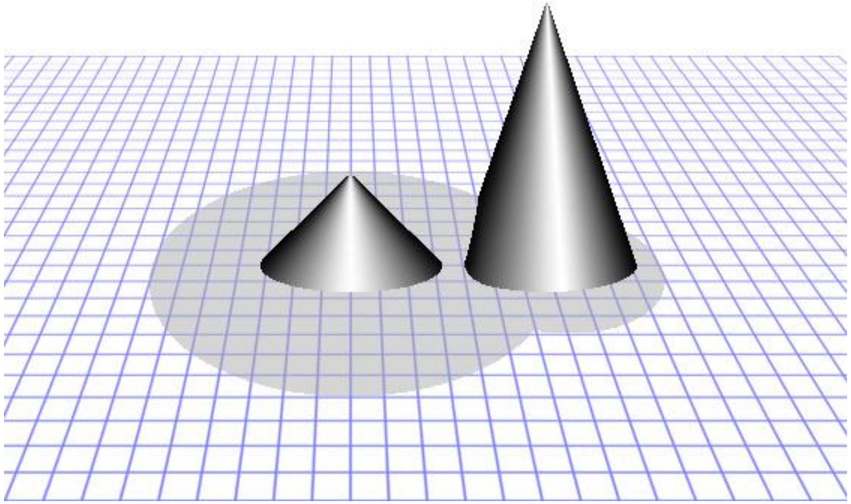
  - Black Top-hat Transform (BTT):

$$BTT = F \bullet_g K - F$$

Top-hat Transform

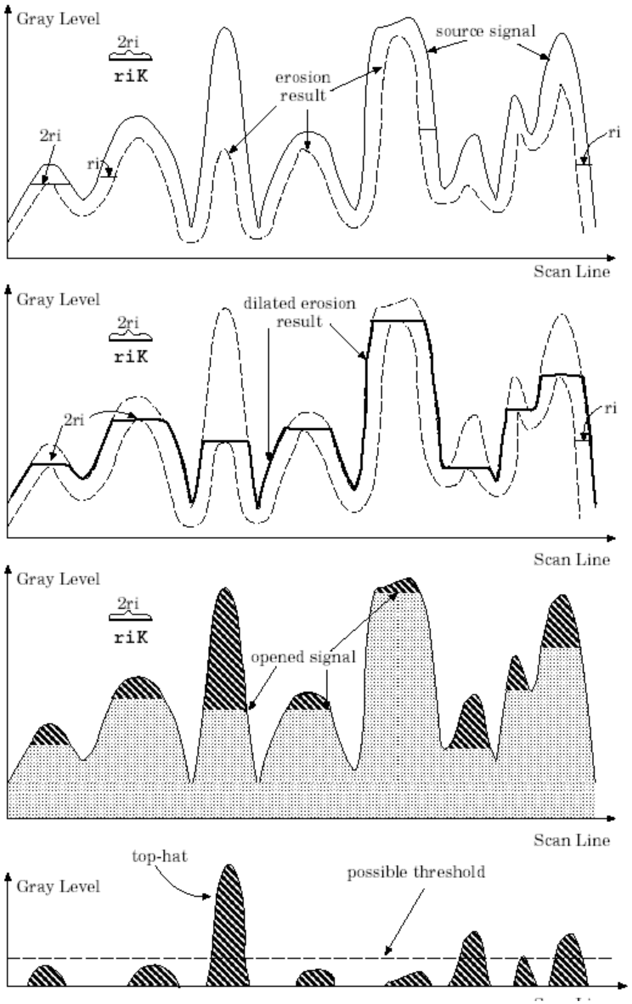


tophat + opened = original



tophat: original - opening

Top-hat Transform



(a) Source Singnal and its erosion result

(b) Dilated erosion result

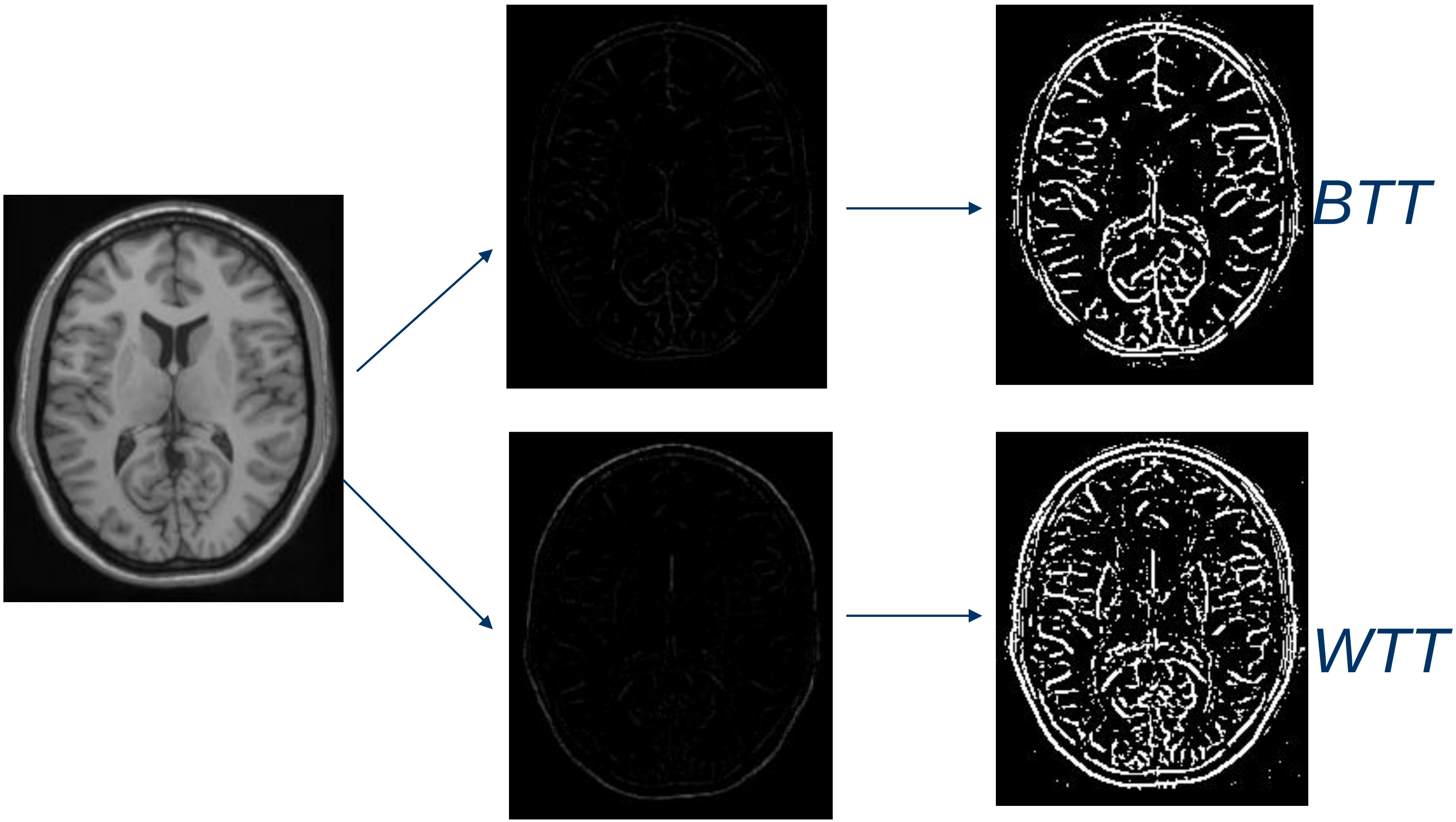
(c) The result of opened signal

(d) Tophat: Ti

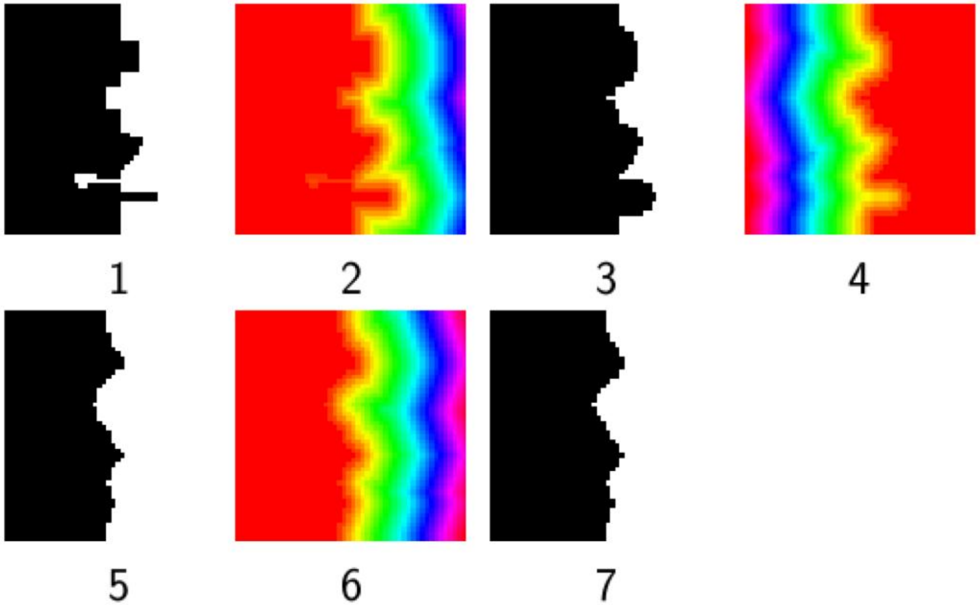
BTT

WTT

Top-hat Transform



Smoothing binary boundaries



- Original border (image 1)
- Distance Transform (images 2,4,6) and thresholding instead of
  - dilation with square SE of size 7x7 (□ image 3)
  - erosion with square SE of size 13x13 (□ image 5)
  - dilation with square SE of size 7x7 (□ image 7)

Energy Minimization Problems

- In general terms, given some problem, we:
  - Formulate the known constraints
  - Build an "energy function" (aka "cost function")
  - Look for a solution that minimizes it
- If we have no further knowledge:
  - The problem can be **NP-Hard** (requires exponential solution time)
  - Use **slow**, generic **approximation** algorithms for optimization problems (such as simulated annealing)

EM in Computer Vision

- Consider a broad class of problems called **Pixel Labeling**
  - Given some images we want to “say something about the pixels”
  - For each **pixel** $\mathbf{p}$, give it a **label** $\mathbf{f_p}$ from a **finite set of labels** $\mathbf{L}$, such that we minimize some energy function.
- Many applications
  - Image Segmentation
  - Image Restoration
  - Stereo and Motion
  - Medical Imaging
  - Multicamera Scene Reconstruction

Example binary segmentation

- Suppose we want to segment an image into foreground and background

Example binary segmentation

- Suppose we want to segment an image into foreground and background

User sketches out a few strokes on foreground and background…

How do we classify the rest of the pixels?

Binary segmentation as energy minimization

- Define a labeling *L* as an assignment of each pixel with a 0-1 label (background or foreground)

- Problem statement: find the labeling *L* that minimizes

$$E(L) = \underbrace{E_d(L)}_{\text{match cost}} + \lambda \underbrace{E_s(L)}_{\text{smoothness cost}}$$

("how similar is each labeled pixel to the foreground / background?")

EM in Computer Vision

- Consider a specific family of Energy Functions
  - Powerful enough to formulate many useful problems
  - Can be reduced to solving a graph min-cut problem
- Problems defined with these functions:
  - Can be solved quickly (using max-flow algorithms)
  - In many cases – optimal solution or within a known factor of the optimum

SINTEF

Energy Function Definition

$$E(f) = \underbrace{\sum_{p \in P} D_p(f_p)}_{E_{data}(f)} + \underbrace{\sum_{p,q \in N} V_{p,q}(f_p, f_q)}_{E_{smooth}(f)}$$

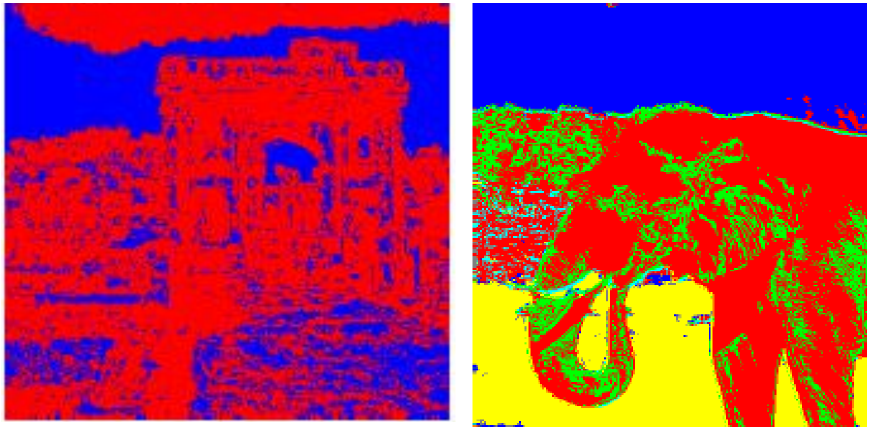
- Input: set of pixels P, set of labels L, $N \subset P \times P$ is a neighbourhood system on pixels.
- Goal: find a labeling $f : P \rightarrow L$ that minimizes $E(f)$.
- $D_p(f_p)$ is a function derived from the observed data that measures the cost of assigning label f_p to pixel p.
- $V_{p,q}(f_p, f_q)$ measures the cost of assigning the labels f_p, f_q to adjacent pixels p, q. Used to impose spatial smoothness.

Energy Function - E_{data} Component

- The $E_{data}(f)$ component:
  - Look at each pixel independently
  - Given it’s current value, what would it cost to label it with each of the labels?
  - Examples:
    - Cost based on a-priori known pixel intensity or color distribution
    - $\left(f_p - i_p\right)^2, i_p$ is the observed intensity of pixel p
- What if we used **only** this component in $E(f)$?
  - Label each pixel independently with the most likely (cheap) label

Optimizing E_{data} Only – Illustration

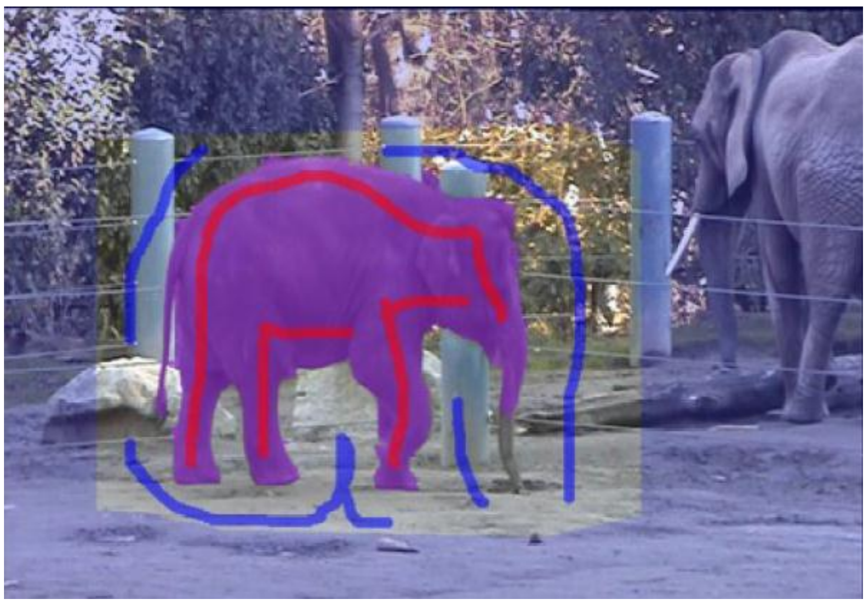
- What would be the problem?
- For example (object segmentation):



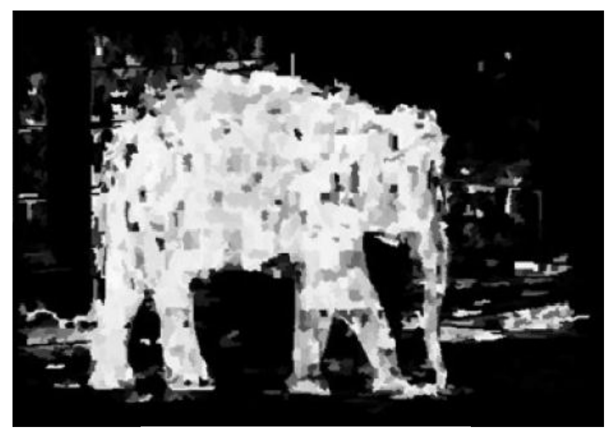
Typical k-means classifier outputs

- We need to add a "smoothness" cost

$$E(L) = E_d(L) + \lambda E_s(L)$$



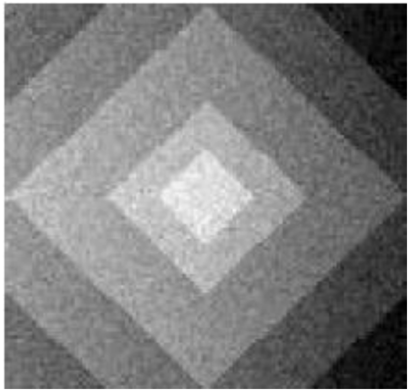
$C'(x, y, 0)$



$C'(x, y, 1)$

Energy Function - E_{smooth} Component

- Look at all pairs of neighbor pixels
- Penalize adjacent pixels with different labels
- What smoothness cost function to use?

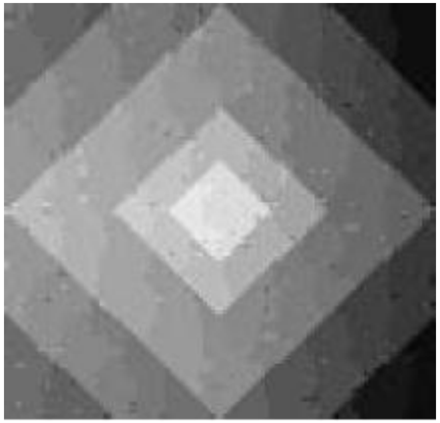


Noised diamond image

Fast Approximate Energy Minimization via Graph Cuts
Yuri Boykov, Olga Veksler, Ramin Zabih

Smoothness Cost Functions

- Potts Interaction Penalty:
  - $$E_{smooth}(f) = \sum_{p,q \in N} K * T\left(f_p \neq f_q\right)$$
  - T – indicator function, K – constant
  - The solution will be **piecewise constant**, with discontinuities at the boundaries



Fast Approximate Energy Minimization via Graph Cuts

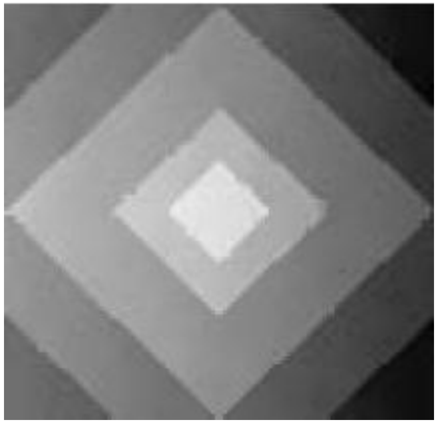
Yuri Boykov, Olga Veksler, Ramin Zabih

Smoothness Cost Functions

- L_2 distance:
  - $E_{smooth}(f) = \sum_{p,q \in N} \| f_p - f_q \|$
  - What is the problem?
  - High penalties at object boundaries
  - We want smooth objects, but allow different labels at object boundary – a ***discontinuity-preserving*** function.

Smoothness Cost Functions

- Truncated L_2 distance:
  - $$E_{smooth}(f) = \sum_{p,q \in N} \min\left(K, \left\| f_p - f_q \right\|\right)$$
  - K – constant
  - The solution will be **piecewise smooth**, with discontinuities at the boundaries



Fast Approximate Energy Minimization via Graph Cuts

Yuri Boykov, Olga Veksler, Ramin Zabih

Smoothness Cost Functions

- Normalize for neighbor distance, image contrast:
  - $$E_{smooth}(f) = \gamma \sum_{p,q \in N} \frac{T(f_p \neq f_q)}{dist(p,q)} \cdot \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right)$$

    BOYKOV, Y., AND JOLLY, M.-P. 2001. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *Proc. IEEE Int. Conf. on Computer Vision*, CD–ROM.
  - This function penalizes a lot for discontinuities between pixels of similar intensities when $|Ip - Iq| < \sigma$.
  - However, if pixels are very different, $|Ip - Iq| > \sigma$, then the penalty is small.

$$E(L) = E_d(L) + \lambda E_s(L)$$

- Neighboring pixels should generally have the same labels
  - Unless the pixels have very different intensities

$$E_s(L) = \sum_{\text{neighbors } (p,q)} w_{pq} |L(p) - L(q)|$$

w_{pq} : similarity in intensity of *p* and *q*

w_{pq} = 10.0

w_{pq} = 0.1

(can use the same trick for stereo)

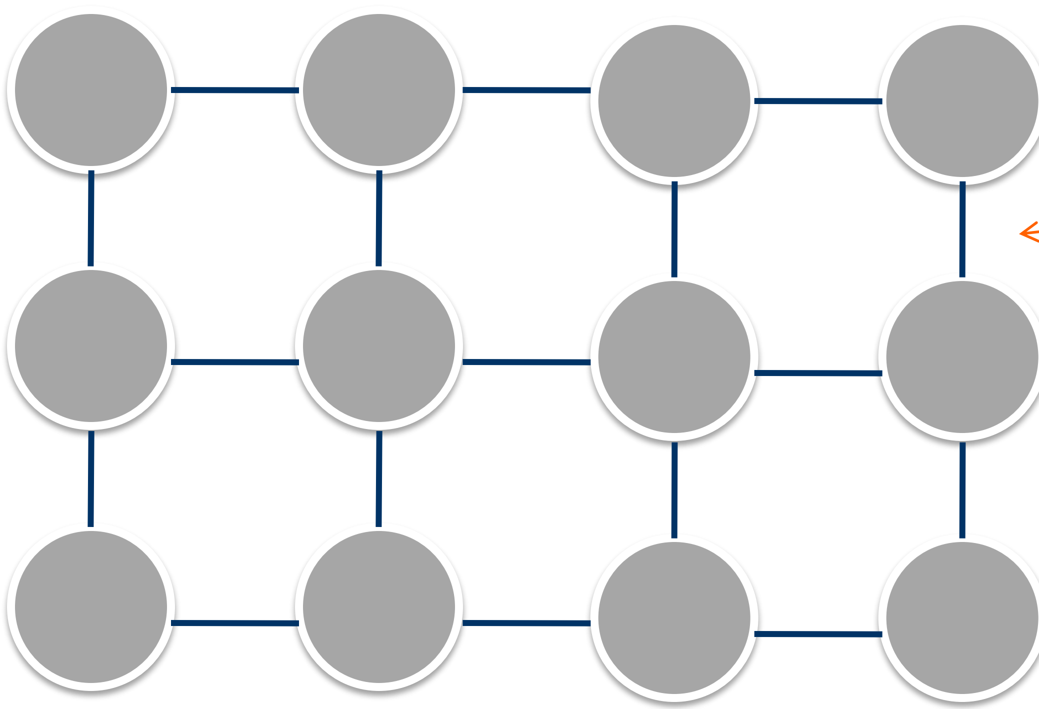
Binary segmentation as energy minimization

$$E(L) = E_d(L) + \lambda E_s(L)$$

- For this problem, we can easily find the global minimum!
- Use max flow / min cut algorithm

Markov Random Fields

Node y_i: pixel la



Edge: constrained pairs

Cost to assign a label to each pixel

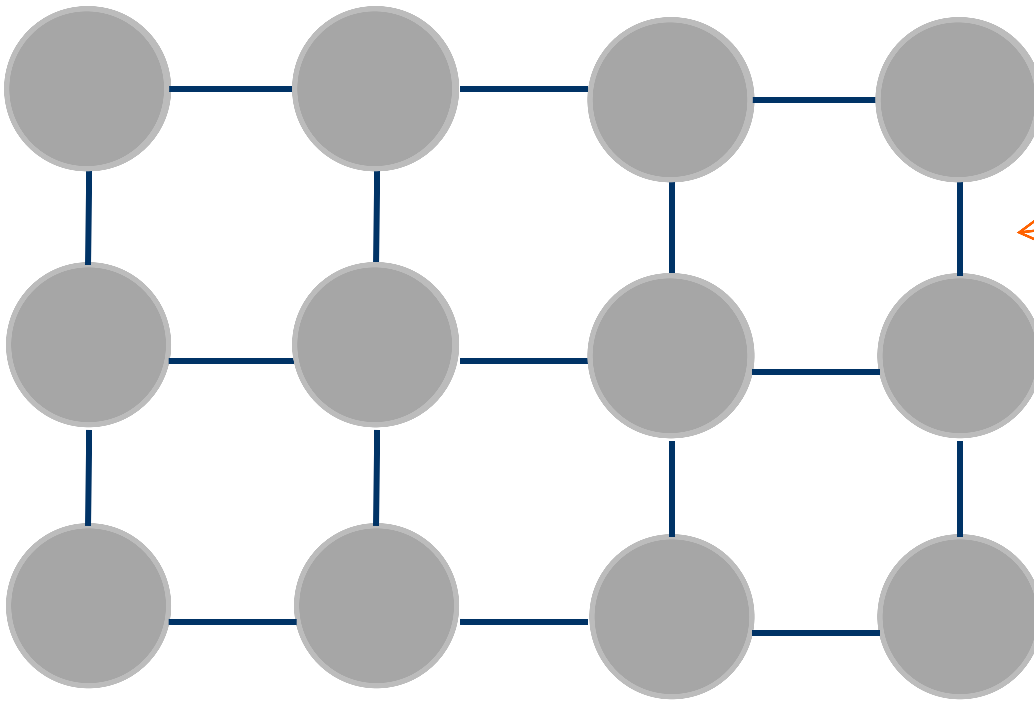
Cost to assign a pair of labels to connected pixels

$$Energy(\mathbf{y};\theta,data)=\sum_{i}\psi_1(y_i;\theta,data)\ \sum_{i,j\in edges}\psi_2(y_i,y_j;\theta,data)$$

SINTEF

Markov Random Fields

- Example: “label smoothing” grid



Unary potential

0: -logP(y_i = 0 ; data)
1: -logP(y_i = 1 ; data)

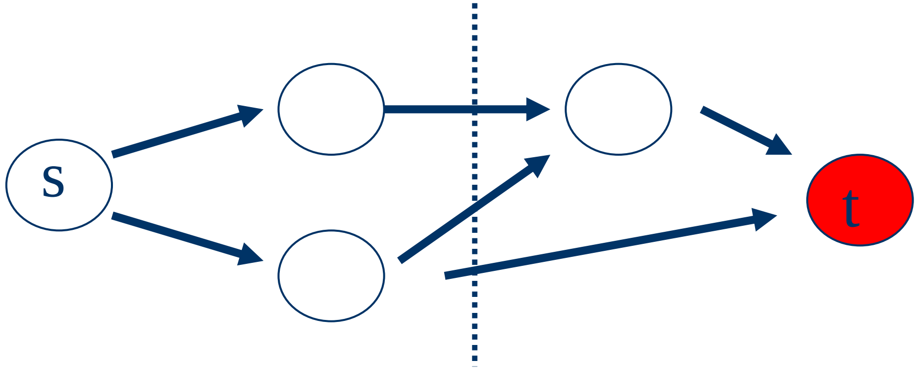
Pairwise Potential

	0	1
0	0	K
1	K	0

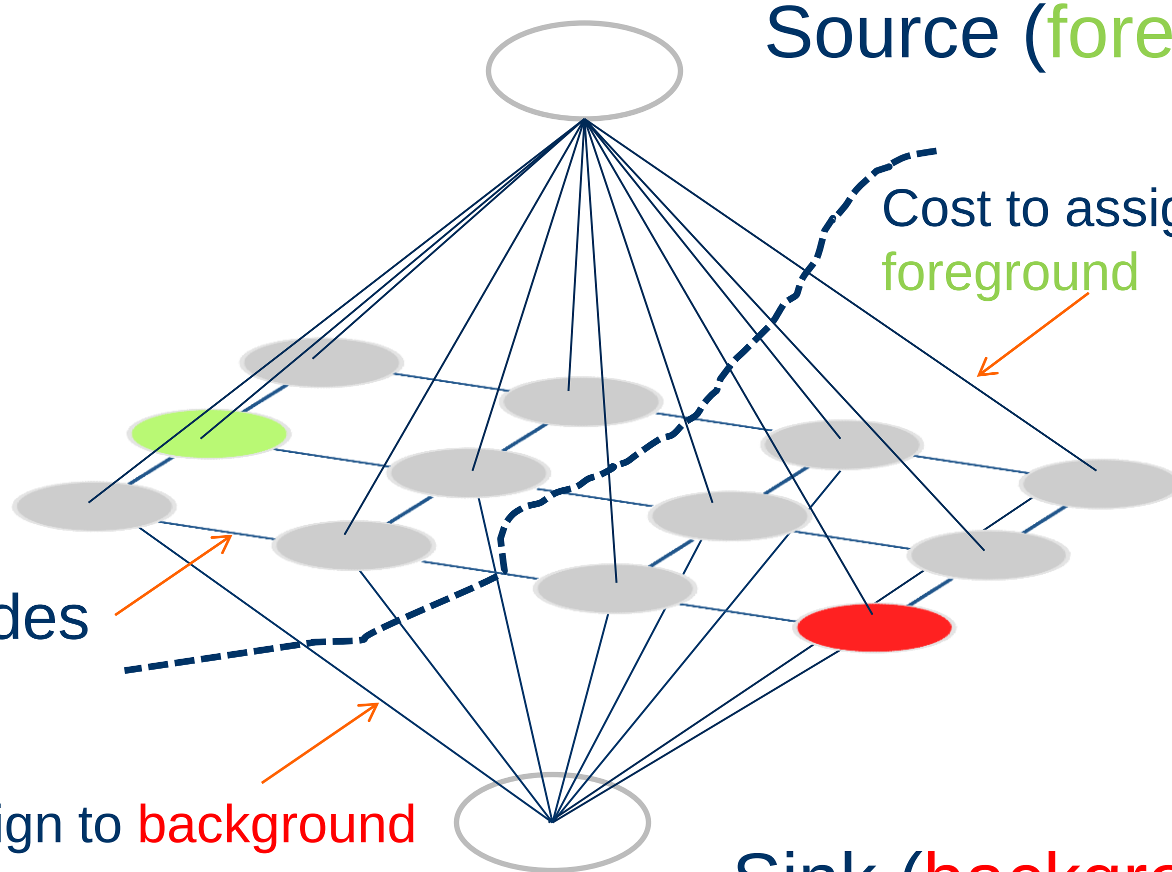
$$Energy(\mathbf{y};\theta,data)=\sum_{i}\psi_1(y_i;\theta,data)\ \sum_{i,j\in edges}\psi_2(y_i,y_j;\theta,data)$$

Graph Cut

- *G*(*V*,*E*) is a finite directed graph and every edge (*u*,*v*) has a capacity *c*(*u*,*v*) (a non-negative real number).
- Assume two vertices, the source *s* and the sink *t*, have been distinguished.
- A **cut** is a split of the nodes into two sets S and T, such that s is in S and t is in T.

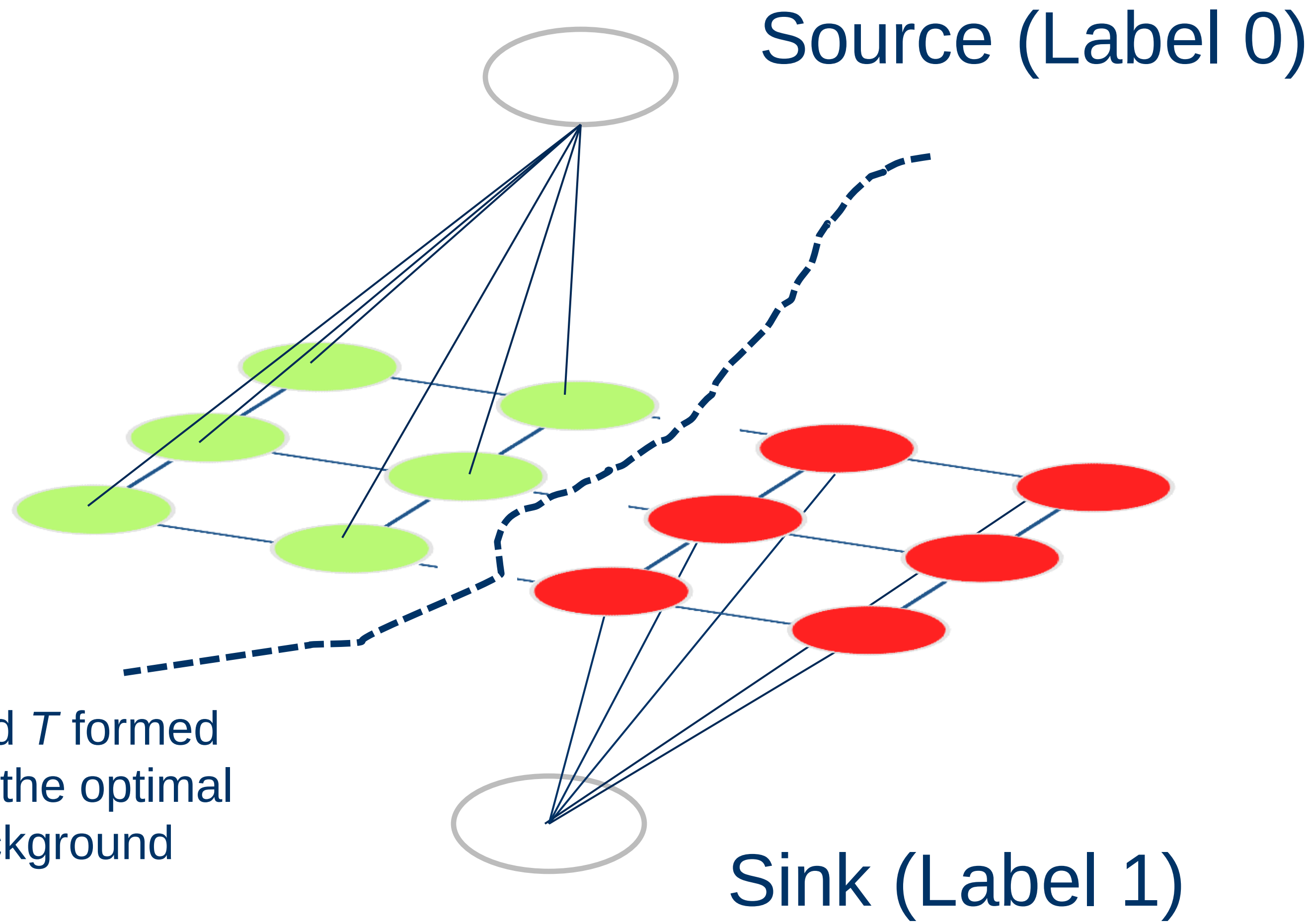


Solving MRFs with graph cuts



$$Energy(\mathbf{y};\theta,data)=\sum_{i}\psi_1(y_i;\theta,data)\ \sum_{i,j\in edges}\psi_2(y_i,y_j;\theta,data)$$

Solving MRFs with graph cuts



The partitions *S* and *T* formed by the *min cut* give the optimal foreground and background segmentation
I.e., the resulting labels minimize $E(d) = E_d(d) + \lambda E_s(d)$

Min-Cut

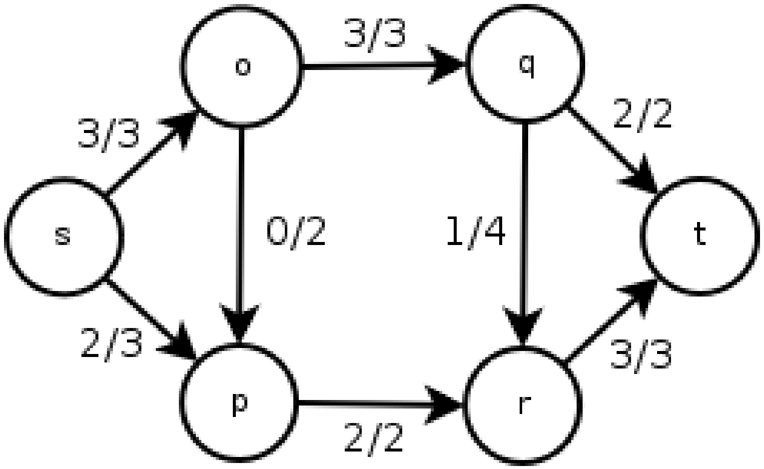
- The capacity of a cut (S,T) is defined as

$$c(S,T) = \sum_{x \in S} \sum_{y \in T} c(x,y)$$

- Min-Cut – finding the cut with the minimal capacity

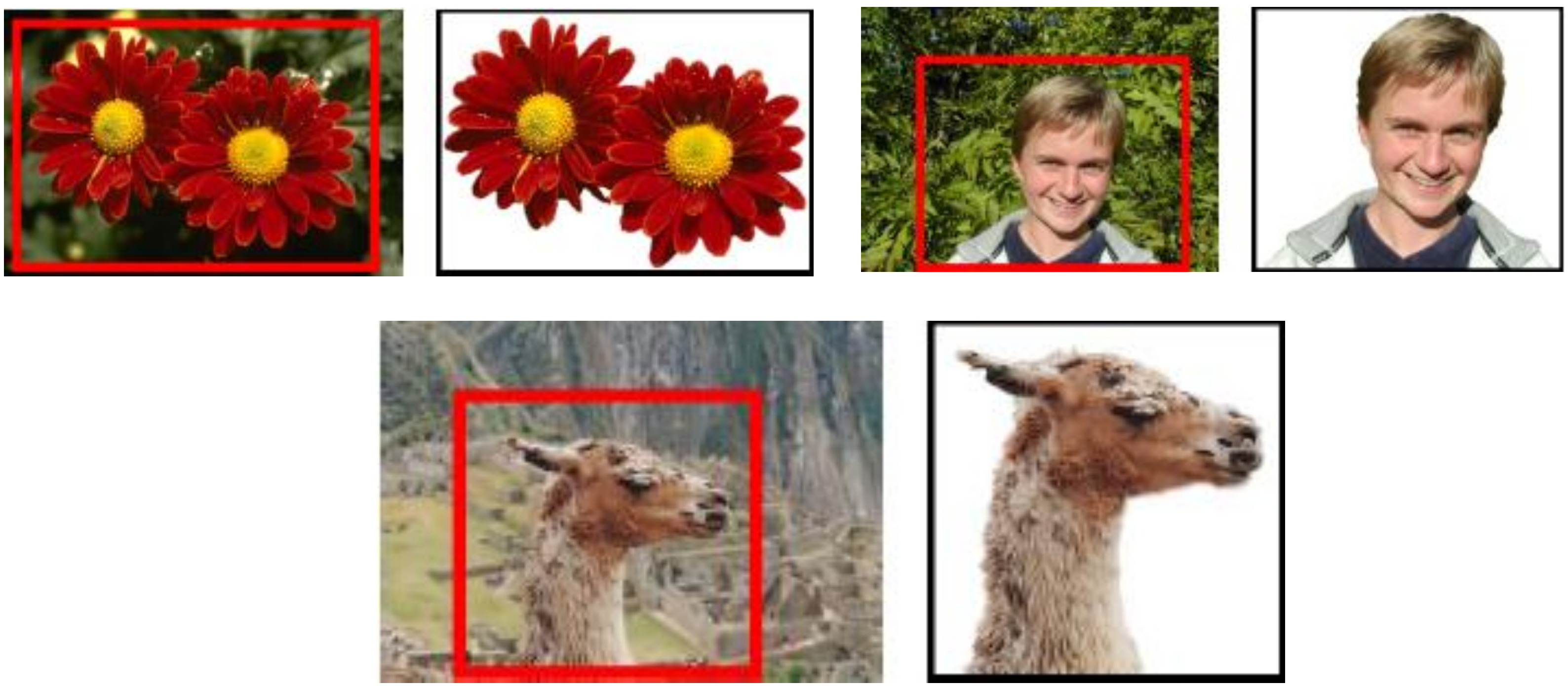
Max Flow

- Find the maximum flow from *s* to *t*

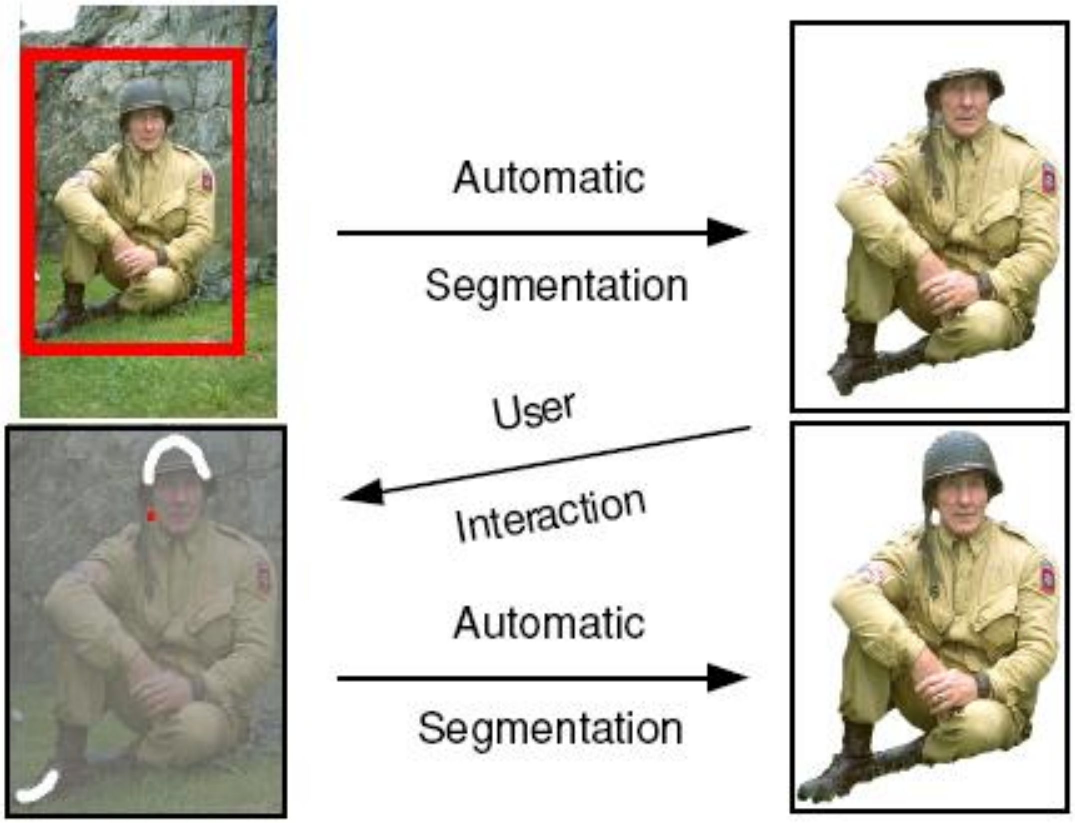


GrabCut

"Interactive Foreground Extraction using Iterated Graph Cuts"
Carsten Rother, Vladimir Kolmogorov, Andrew Blake, 2004
Microsoft Research Cambridge, UK

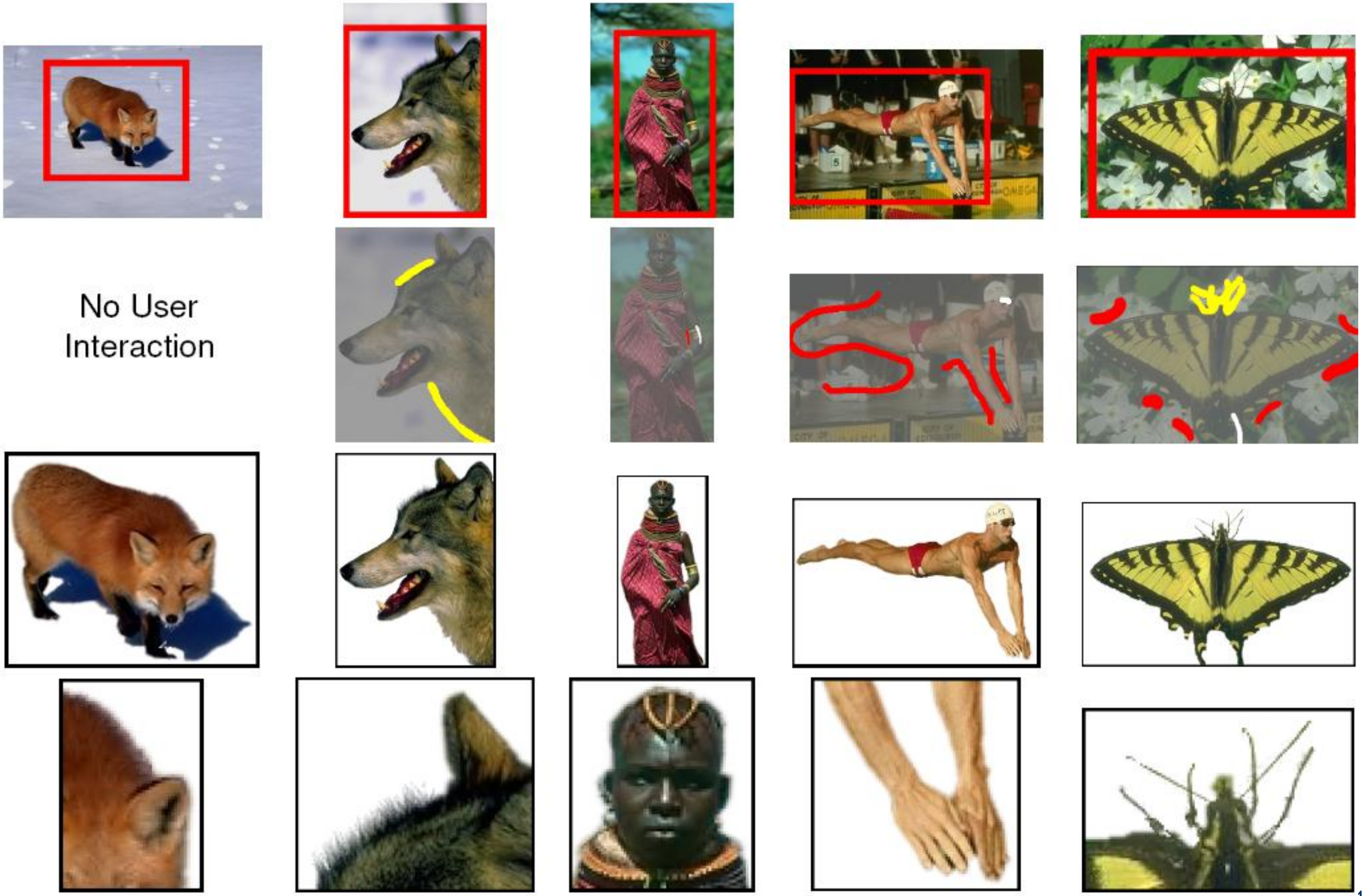


GrabCut


Automatic
Segmentation
User
Interaction
Automatic
Segmentation

SINTEF

GrabCut



Graph cuts segmentation

1. Define graph
   – usually 4-connected or 8-connected
2. Define unary potentials
   – Color histogram or mixture of Gaussians for background and foreground

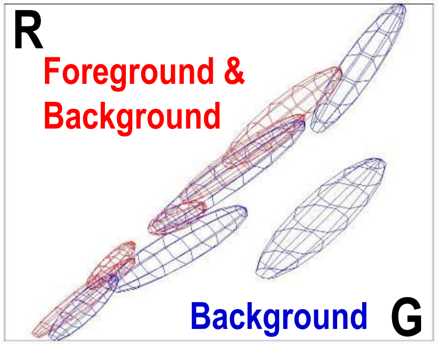
$$unary\_potential(x) = -\log\left(\frac{P(c(x);\theta_{foreground})}{P(c(x);\theta_{background})}\right)$$

3. Define pairwise potentials

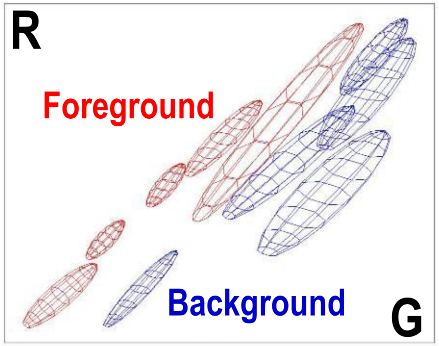
$$edge\_potential(x,y) = k_1 + k_2 \exp\left\{\frac{-\|c(x)-c(y)\|^2}{2\sigma^2}\right\}$$

4. Apply graph cuts
5. Return to 2, using current labels to compute foreground, background models

Colour Model



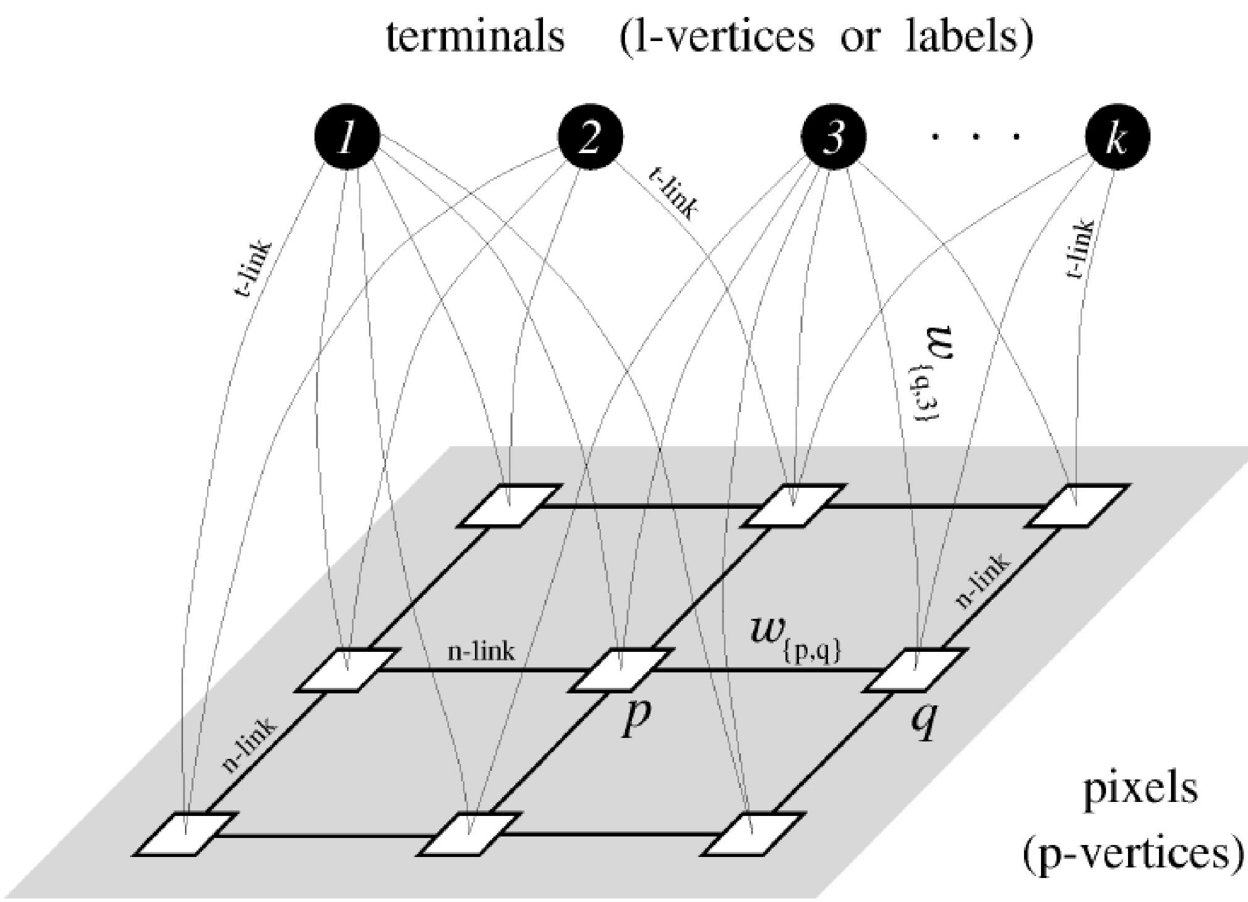
Iterated graph cut



Gaussian Mixture Model (typically 5-8 components)

SINTEF

Multi-Label Case

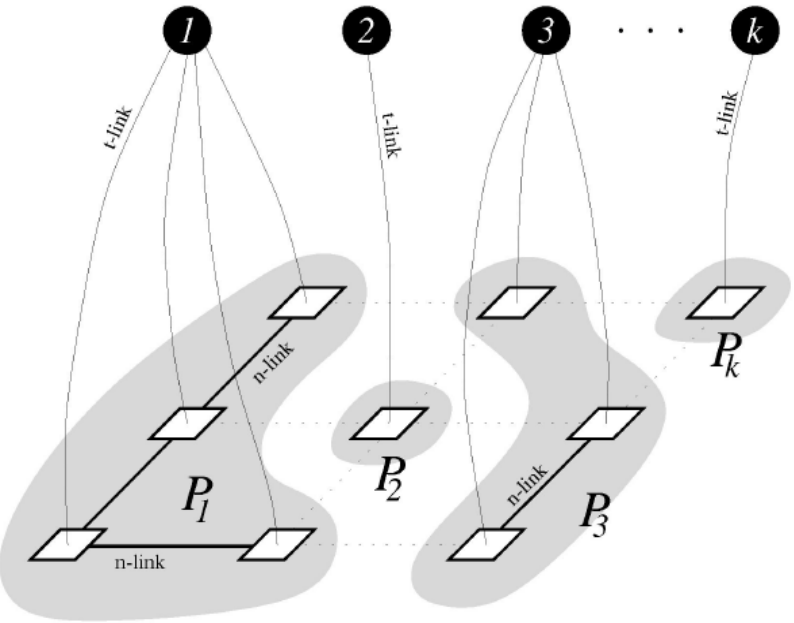


Multi-Label Case

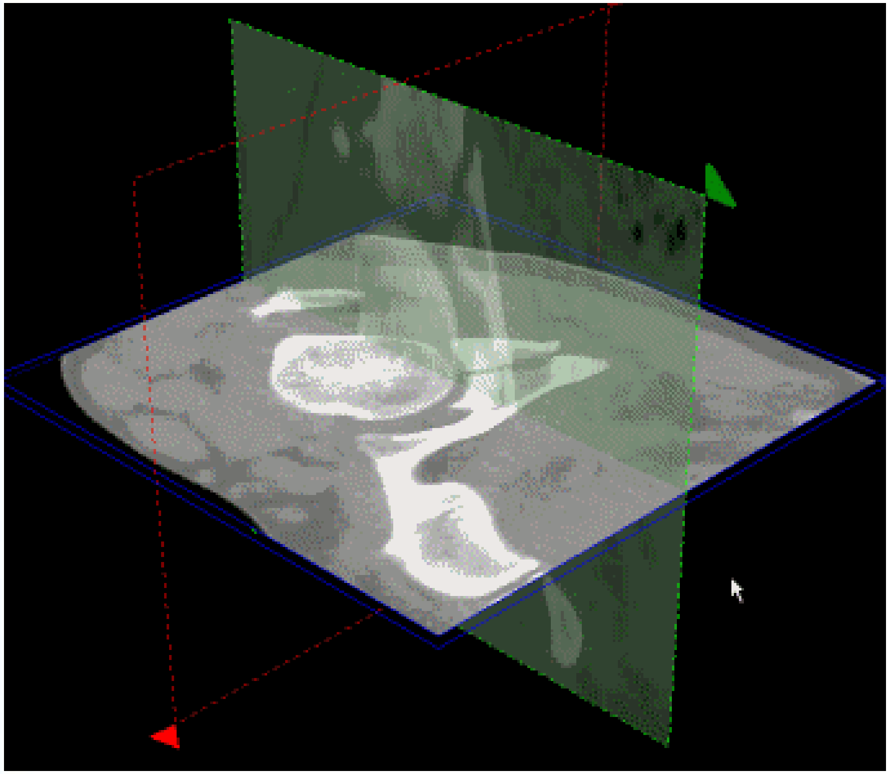
Solve multiple-labels problems with *binary decisions*

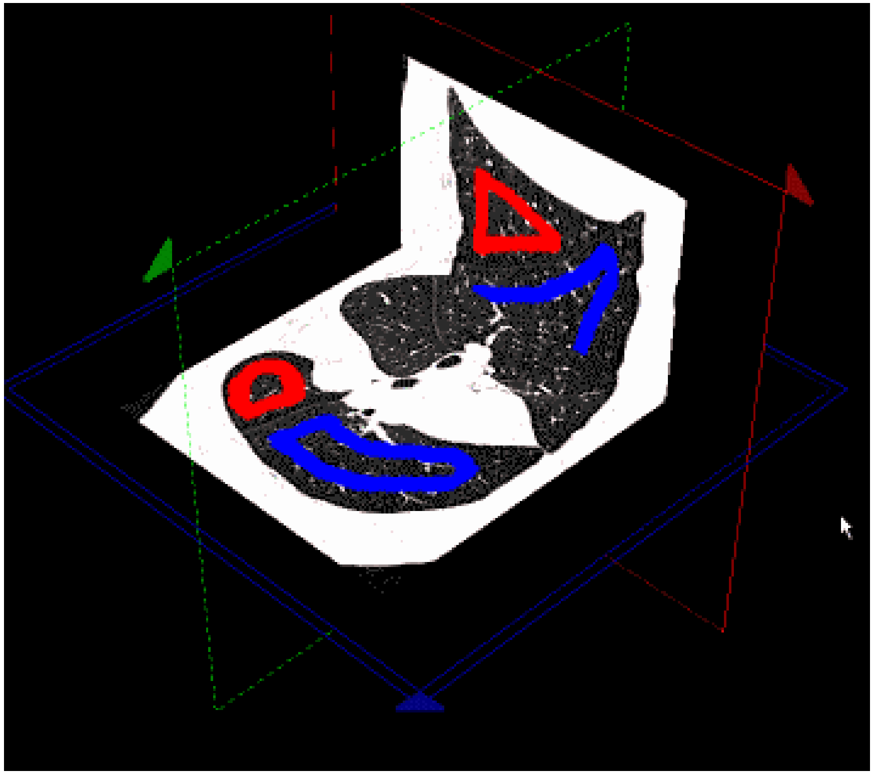
I.e., try to relabel (expand) one label against the rest, and *compare* total energy

Solution is an ***approximation***



Interactive Segmentations





Images: Boykov