5\ What are the 4 main binary mathematical morphology ?

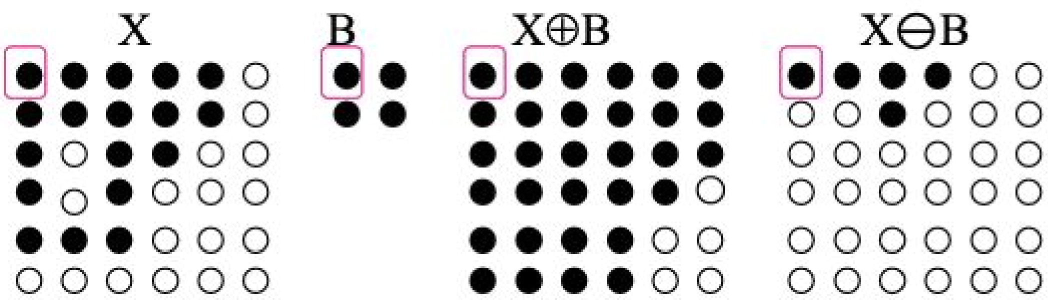
<u>1\ Dilation:</u> In binary images, dilation is an operation that expands the size of foreground objects and reduces the size of holes in an image

$$X \oplus B = \{c \in E \text{ t.q. } c = a + b \text{ avec } a \in X,\ b \in B\}$$

<u>2\ Erosion:</u> is a process that increases the size of background objects and shrinks the foreground objects in binary images.

- So what does it do? The kernel slides through the image (as in 2D convolution). A pixel in the original image (either 1 or 0) will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero).

$$X - B = \{c \in E \text{ t.q. } \forall b \in B,\ c + b \in X\}$$



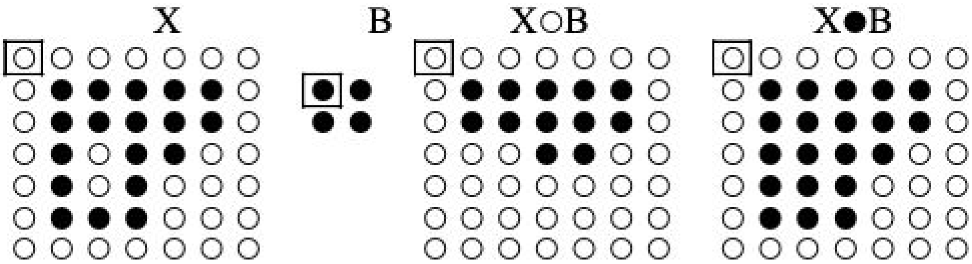
<u>3\ Opening:</u> is defined as erosion followed by dilation using the same structuring element for both operations. The erosion part removes some foreground pixels from the edges of regions of foreground pixels, while the dilation part adds foreground pixels. The foreground features remain roughly the same size, but their contours are smoother.

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

- The effect of opening depends on the shape of the structuring element. Opening preserves foreground regions that have a similar shape to the structuring element.
- If the structural element is included in the object at the time of processing than the whole structural element will appear in the output of the transformation, otherwise not all points will appear.
- Because, erosion removes white noises, but it also shrinks our object. So we dilate it. Since noise is gone, they won’t come back, but our object area increases.
- Opening tends to smooth an image, break narrow joins, and remove thin protrusions.

4\ Closing: is defined as the dilation process followed by erosion, using the same structuring element for both operations. Closing smoothes the contours of foreground objects, merges narrow breaks or gaps and eliminates small holes.

$$X \times B \ (X \oplus B) - B$$

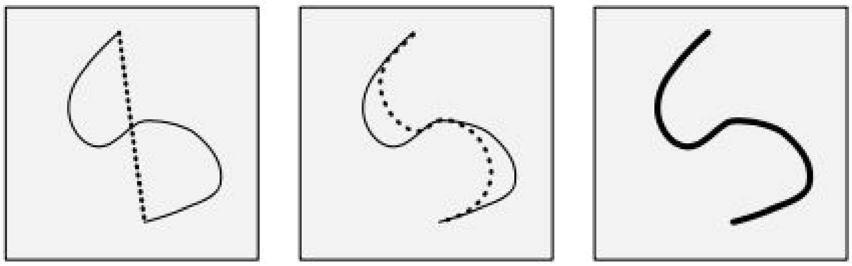


6\ What is the principle of an active contour ?

Snakes: is defined as a model which deforms a curve C through the object contours by maximizing an energy functional - Done by Kass-Witkin-Terzopolous 1987

How it works ?

- A higher level process or a user initializes any curve close to the object boundary.
- The snake then starts deforming and moving towards the desired object boundary.
- In the end it completely “shrink-wraps ” around the object.



Snakes may be understood as a special case of a more general technique of matching a deformable model to an image by means of energy minimization.

1st Model of Active Contours:

So, the goal is find a contour C that lies in the position in which the snake reaches a local energy minimum. Then, from Calculus of Variations, the Euler-Lagrange condition states that the spline F which minimizes Esnake must satisfy:

The contour is defined in the (x, y) plane of an image as a parametric curve : **v(s) = (x(s),y(s))**

Contour is said to possess an energy (**Esnake**) which is defined as the sum of the three energy terms.

- Parametric representation: ***v(s)=(x(s),y(s))***

$$E_{snake} = \int_0^1 E_{\text{int}}(v(s)) + E_{image}(v(s)) + E_{con}(v(s))ds$$

- ***Eint*** : internal energy due to bending. Serves to impose = internal energy due to bending. Serves to impose piecewise smoothness constraint. piecewise smoothness constraint.
- ***Eimage*** : image forces pushing the snake toward image = image forces pushing the snake toward image features (edges, etc…). features (edges, etc…).
- ***Econ*** : external constraints are responsible for putting = external constraints are responsible for putting the snake near the desired local minimum. It may come from: - Higher level interpretation Higher level interpretation. - User interaction.

__The internal forces__ are designed to keep the model smooth during deformation.

__The external forces__ are defined to move the model toward an object boundary or other desired features within an image.

__Objective:__ Therefore our problem of detecting contours reduces to an energy minimization problem.

Internal Energy:

The smoothness energy at contour point ***v(s)*** could be evaluated as

$$E_{in}(v(s)) = \underbrace{\boxed{\alpha(s)\left|\frac{dv}{ds}\right|^2}}_{\text{Elasticity/stretching}} + \underbrace{\boxed{\beta(s)\left|\frac{d^2v}{d^2s}\right|^2}}_{\text{Stiffness/bending}}$$

α(s) and **β(s)** controls the relative importance of stretching and stiffness

External Energy: The external energy describes how well the curve matches the image data locally

$$E_{ex}(\mathbf{v}) = -|\nabla I(\mathbf{v})|^2 = -|\nabla I(x,y)|^2$$

∇I represent the gradient at pixel **I**(x, y) in the image.

According to Calculus of Variations in order to have minimum energy of a function C that fits exactly the curve we want it must satisfy **Euler-Lagrage Equation** :

Euler-Lagrange Equation for 2[d] order derivative:

$$\left(\frac{\partial}{\partial C} - \frac{d}{dp}\frac{\partial}{\partial C_p} + \frac{d^2}{dp^2}\frac{\partial}{\partial C_{pp}}\right) f(C, C_p, C_{pp}) = 0$$

By discretizing the curve **C(p)=(x(ih), y(ih))** : where derivatives are approximated by finite differences if they differences if they cannot be computed cannot be computed analytically.

In matrix form where *A* is a pentadiagonal banded matrix:

Microsoft PowerPoint - 02-Snake

$$Ax + f_x(x,y) = 0$$
$$Ay + f_y(x,y) = 0$$

__*Drawbacks:*__

- Snakes don’t solve the entire problem of finding contours in image, rather they depend on the mechanisms such as interaction with a user, starting position for the snake somewhere near the desired contour.
- A particular limitation of the snake approach is that the segmentation delivers a closed contour which is unsatisfactory in many applications.
- Snake may over-smooth the boundary.
- Not trivial to prevent curve self intersecting

2nd Model of Active Contours "Geodesic active contour ":

- If the shape changes dramatically, curve reparameterization may also be required, there this proposed model is parameterization invariant.

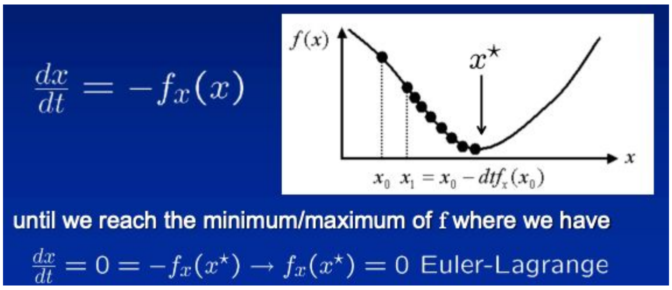
This model puts a curve C on the object contours of the image by minimizing the following energy functional.

$$F(C) = \int_0^1 g(I(C))|C_p|dp$$

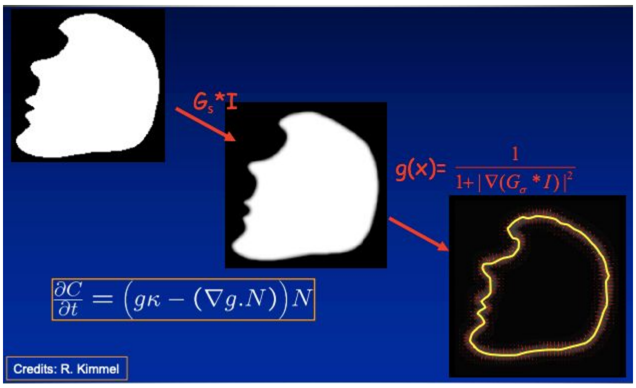
By applying **Euler-Lagrage Equation** :

$$\Big((\nabla g.N) - g\underbrace{\kappa}_{\text{curvature of } C}\Big)\underbrace{N}_{\text{normal of } C} = 0$$

In order to solve this equation, we use gradient descent to find the optimal value



- Negative gradient at point (x) gives direction of the steepest descent towards lower values of function F.



** Ct =k N : smoothing,   Ct =gk.N : border detection, C = (∇g.N).N : attractive force through the contours*

Level Sets:

An alternative representation for such closed contours is to use level sets (LS).

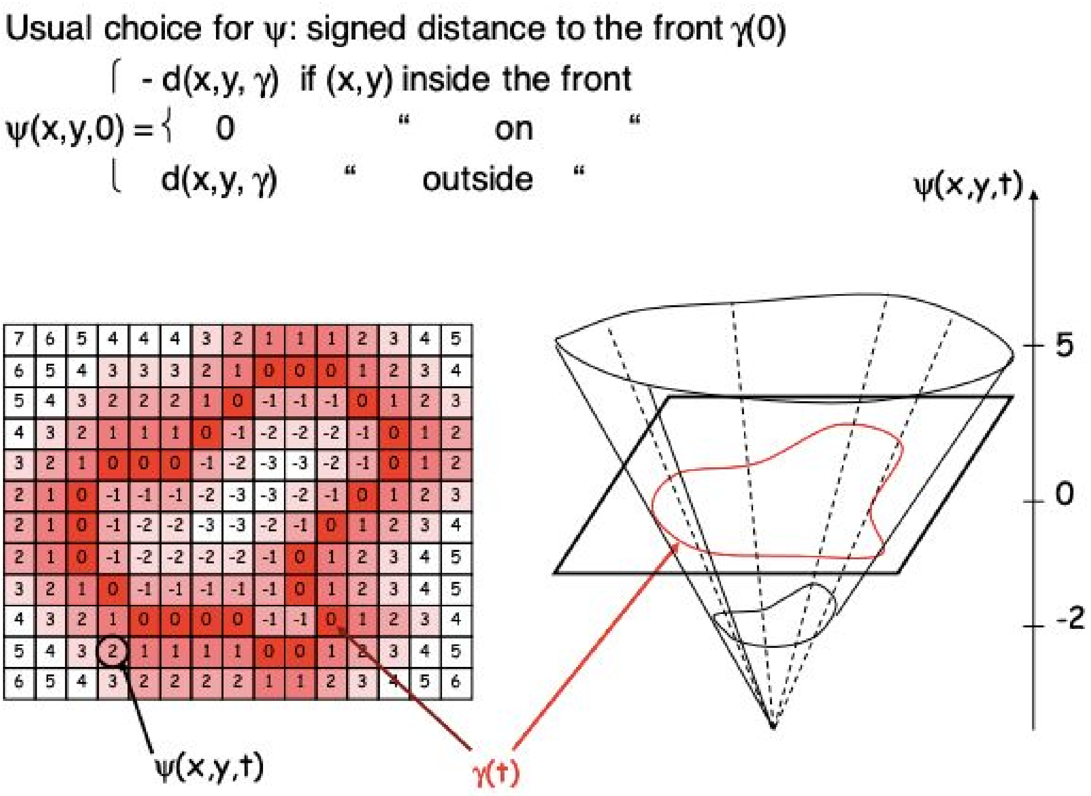
- LS evolves to fit and track objects of interest by modifying the underlying embedding function instead of curve function f(s).
- Zero-crossing(s) of a characteristic function define the curve.

Idea:

The level set C0 at time t of a function **ψ(x,y,t)** is the set of arguments **{ (x,y) , ψ(x,y,t) = c0 }**

Idea: define a function **ψ(x,y,t)** so that at any time, **γ(t) = { (x,y) , ψ(x,y,t) = 0 }**

- There are many such ψ
- ψ has many other level sets, more or less parallel to γ
- Only γ has a meaning for segmentation, not any other level set of ψ



Drawbacks:

- Shape constraint based LS is good, but not easy to construct shape constraint

7\ What are Fourier Descriptors ?

Fourier descriptors are a method of description of objects using features globally calculated on the whole contour.

Suppose a boundary of a particular shape has N pixels, each has position (Xk, Yk). We can describe the contour as two parametric equations: Let’s suppose that we take Fourier Transform of each function.

For a finite number of discrete contours pixels we simply use the Discrete Fourier Transform. The DFT treats the signal as periodic “contour itself is periodic”.

We can consider a set of points (X, Y), be the coordinates of N successive points of a contour. For each point, we can define them as a complex number: U = X + j Y

- Then, we calculate the Fourier Transform for the set of points.
- The two result spectra are called descriptors.

We will get a set components known as descriptors as a decomposition of the contour as a set of frequencies (high, and low) where the low-frequencies determine the overall shape and high frequencies account for fine details about the shape.

Properties:

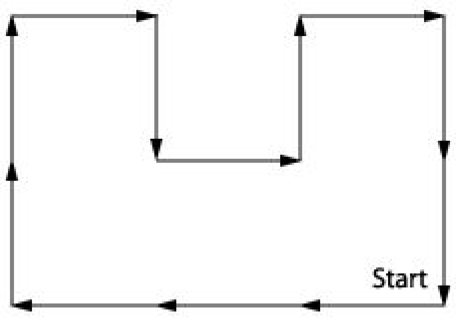
- Translation: means adding components to values of x(k), y(k) therefore it will only change the zero component frequency ⇒ Translation invariant.
- Rotation: means multiplying by $exp(i\Theta)$ ⇒ Fourier descriptors rotation invariant.
- Starting Point: means translation which FD invariant to translation

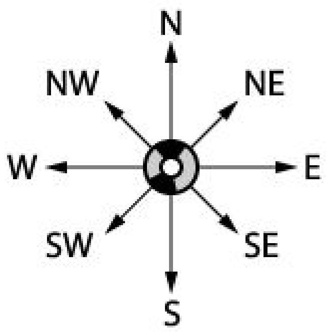
9\ What is Freeman Code or “Chain code” ?

An object is described by a sequence of small unit size line segments with a set of defined orientations.

<u>How it works ?</u>

A set of connected components, then starting from one point of a segment, according to 4 / 8 connectivity, the direction of the next segment is determined. The direction from one unit size segment to the next unit-size segment becomes an element of chain code.





Answer: W,W,W,N,N,E,S,E,N,E,S,S

After calculating the chain code of an object, we can use it to compare it with other objects to calculate the similarity between the two objects. An example of that would be spell-checking.

<u>Disadvantage:</u>

- The chain code will be different for different starting points

We can use chain code to correct the closed word to another by calculating the difference between the two.

- ***The edition distance δ(x,y)*** between two chains x and y is the <u>minimum</u> number of elementary operations (insertion, suppression or substitution) necessary to transform x into y

  Example: x = « ababa » and y = « abba »

  - Replace the « a » in the middle by « b » and suppress the next « b » : 2 operations
  - Suppress the « a » in the middle : 1 operation
  - Thus the edition distance: δ(x,y) = 1

10\ What is A morphological skeleton ?

Morphological skeleton is a skeleton (or medial axis) representation of a shape or binary image, computed by means of morphological operators.

- The skeleton is useful because it provides a simple and compact representation of a shape that preserves many of the topological and size characteristics of the original shape.

Mathematically, a skeleton can be written as a series of openings and closing where k indicates k successive erosions of A by B.

$$S(A) = \bigcup_{k=0}^{K} S_k(A)$$
$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B$$

Medial Axis Transform: is a gray-level image where each point on the skeleton has an intensity which represents its distance to a boundary in the original object.

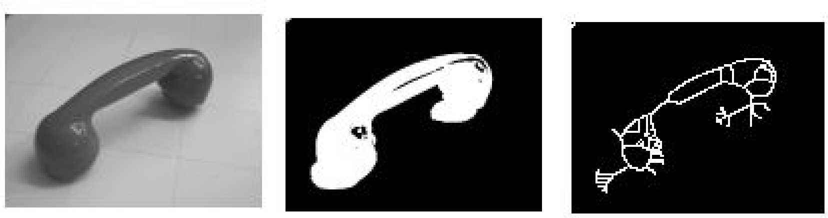
Hit-or-miss transform: The result of the hit-or-miss transform is the set of positions where the first structuring element fits in the foreground of the input image, and the second structuring element misses it completely.

- Used to extract pixels with specific neighbourhood configurations from an image.
- We can do this using erosion alone, because we have to be able to detect when deletion of a pixel would make the object disconnected. So, we have to use hit-and-miss operators (thinning).

How it works ?

The skeleton/MAT can be produced in two main ways:

- The first is to use some kind of morphological thinning that successively erodes away pixels from the boundary (while preserving the end points of line segments) until no more thinning is possible, at which point what is left approximates the skeleton.
- The alternative method is to first calculate the distance transform of the image. The skeleton then lies along the *singularities* (*i.e.* creases or curvature discontinuities) in the distance transform.



- Morphological Skeletonization can be considered as a controlled erosion process. This involves shrinking the image until the area of interest is 1 pixel wide.

11\ What is a Bayesian Classifier ?

Principle: General idea is to think about the world as a random generator which maps features with associated object classes (m, w).

Objective: is to build a classifier that exploits this prior information about maps features (assuming they come from *a probability distribution function PDF*) and their associated classes.

- Given a new unseen instance, we (1) find its probability of it belonging to each class, and (2) pick the most probable.

How it works ?

- Let us consider the two class problem, with known prior probabilities P(w1) and P(w2).
- The conditional PDFs p(x |wi) are also assumed known (be identified from the training set).

Bayes Rule:
$$P(w_i \mid x) = \frac{p(x \mid w_i)\, P(w_i)}{p(x)}$$

Bayesian classification (maximum a posteriori)

$$P(w_1 \mid x) \overset{?}{<>} P(w_2 \mid x)$$
$$p(x \mid w_1)P(w_1) \overset{?}{<>} p(x \mid w_2)P(w_2)$$

-

The decision function between classes 1 and 2 is the equation : **P(x, w1) = P(x, w2)**

Advantages:

- Bayesian technique allows bringing in prior knowledge about the parameter vector.
- Best linear classifier (assuming the hypothesis that distribution is indeed Gaussian).
- Taking advantage of Gaussian Probability : because the sum of independent Gaussian random variables is Guassian.

Limitations:

- It depends really on whether our hypothesis about the dataset is Guassian indeed or not.
- Assumption that class-specific feature distribution does not depend on sample a different class.
- Overlapping class-specific feature distribution. The features are not perfectly discriminative. Solution: find better features that are more powerful.
- Estimation error: This error occurs because there are too few samples to the number of parameters to give a suitable estimation. Solution : can be diminished by gathering more samples, or reducing the number of parameters.

Generalization to n classes:

- The decision surface between classes i and j has the equation **P(x, wi) = P(x, wi).**
- Assume there is a function **f** which is discriminant function that is monotonically increasing function, it assigns each feature vector to its class as: $fi\,(x) > fj\,(x)\, for\ all\ j \neq i$.

- Therefore, we can get the decision surface by: $f_i(x) - f_j(x) = 0$

Normal law : the pdf follows a Gaussian law:

- 1D: $$p(x \mid w_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$
- *l*D : $$p(x \mid w_i) = \frac{1}{(2\pi)^{l/2} \left|\Sigma_i\right|^{1/2}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)}$$
  - *μ_i is the mean of class w_i*
  - *Σ_i is the covariance matrix of size l x l, defined by*

$$\Sigma_i = E\left[(x-\mu_i)(x-\mu_i)^T\right]$$

Since PDFs depend on the Mean and covariance matrix for each class, we can predict the type of the decision surface based on those parameters.

Case I : The covariance is equal for all classes.

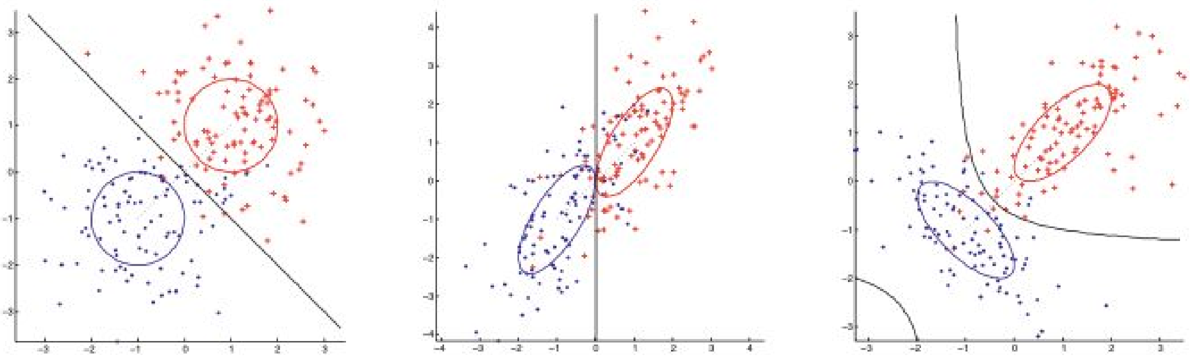
- We will get a hyperplane decision surface.
- The discriminant function is a linear function can be written as : $f_i(x) = W_i . X + W_{i0}$

Sub-case I : Covariance is Diagonal with equal values on diagonal Σ =σ2I

- The decision hyper plane passes by X0.
- The decision hyperplane is orthogonal to µ1-µ2 and X - X0 is also on the hyperplane, and since (µ1-µ2)T(x-x0)=0

Sub-case II : Σ =/= σ2I

- The decision hyperplane passes by X0
- The decision hyperplane is not orthogonal to µ1-µ2, but to a linear transformation of it: Σ-1(µ1-µ2).
- With non-equal class covariances the decision boundary will be hyperbolic.



(left) If the features are uncorrelated and have the same variance, maximum-likelihood classification leads to the basic linear classifier, whose decision boundary is orthogonal to the line connecting the means .

(middle) As long as the per-class covariance matrices are identical, the Bayes-optimal decision boundary is linear – if we were to decorrelate the features by rotation and scaling, we would again obtain the basic linear classifier.

(right) Unequal covariance matrices lead to hyperbolic decision boundaries.

The multivariate normal distribution essentially translates distances into probabilities. This becomes obvious when we plug the definition of **Mahalanobis distance**:

$$d_m = \left((x-\mu_i)\Sigma^{-1}(x-\mu_i)\right)^{1/2}$$

- If Σ is not diagonal: the most probable class is the one that maximizes discriminant function, i.e. which minimizes the Mahalanobis distance.
- Can be applied also if we don't know much about our data (distribution, covariance ..etc)

The standard normal distribution translates **Euclidean distance**s into probabilities.

- if Σ =σ2I : the most probable class is the one that maximizes gi (x), i.e. which minimizes the Euclidean distance