Computer Vision
Image segmentation - lecture 13

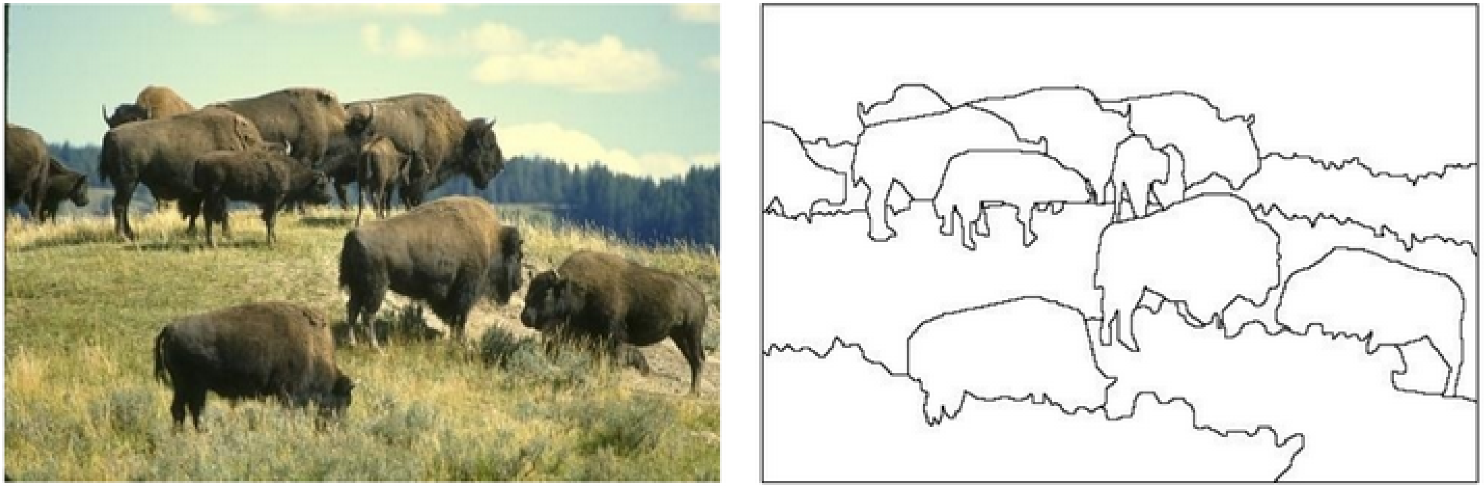
Adam Szmigielski
aszmigie@pjwstk.edu.pl

materials: $ftp(public)://aszmigie/WMAEnglish$

Image segmentation

- Image segmentation is the task of finding groups of pixels that "go together".
- In statistics, this problem is known as *cluster analysis* and is a widely studied area with hundreds of different algorithms

Image segmentation

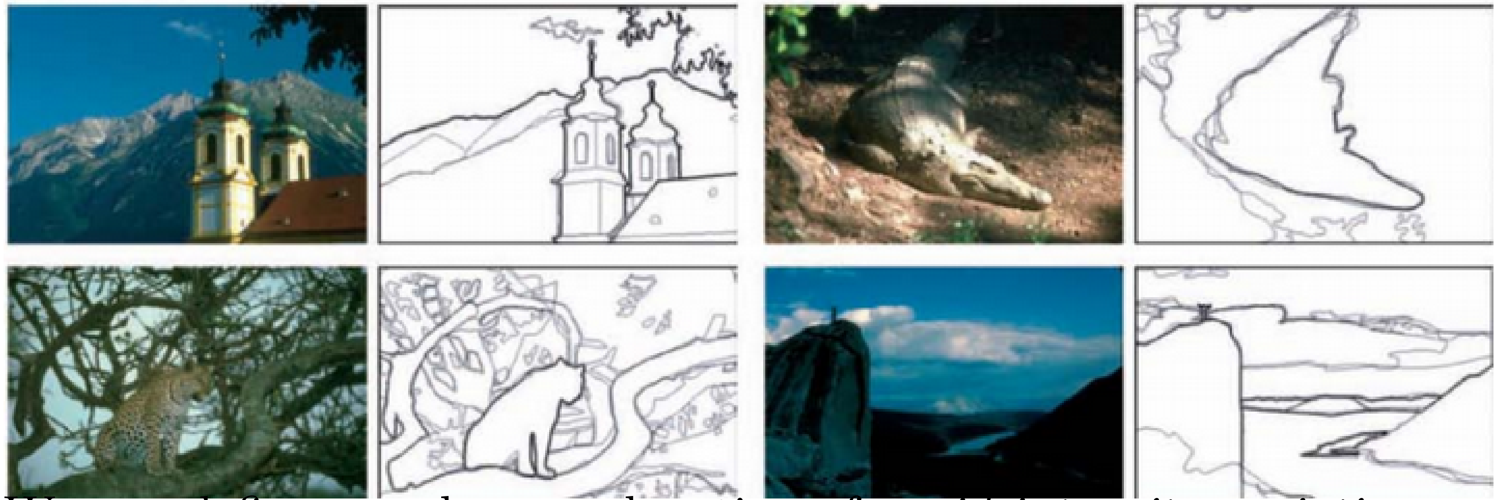


- Image segmentation methods will look for objects that either have some measure of homogeneity within themselves, or have some measure of contrast with the objects on their border
- The homogeneity and contrast measures can include features such as gray level, color, and texture

Edge Detection: First Step to Image Segmentation

- The goal of image segmentation is to find regions that represent objects or meaningful parts of objects
- Division of the image into regions corresponding to objects of interest is necessary for scene interpretation and understanding
- Identification of real objects, pseudo-objects, shadows, or actually finding anything of interest within the image, requires some form of segmentation

Edge detection



We can define an edge as a location of *rapid intensity variation.*

- A mathematical way to define the slope and direction of a surface is through its gradient,

$$J(x) = \bigtriangledown I(x) = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})(x).$$

- The local gradient vector J points in the direction of steepest ascent in the intensity function,
- Its magnitude is an indication of the slope or strength of the variation.

Edges calculation

The gradient of the smoothed image can therefore be written as

$$J_\sigma(x) = \bigtriangledown[G_\sigma(x) \star I(x)] = [\bigtriangledown G_\sigma](x) \star I(x)$$

- We can convolve the image with the horizontal and vertical derivatives of the Gaussian kernel function,

  $$\bigtriangledown G_\sigma(x) = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})(x) = [-x - y]\frac{1}{\sigma^3}\exp(-\frac{x^2+y^2}{2\sigma^2})$$

  The parameter σ indicates the width of the Gaussian

- Desired directional derivative is equivalent to second gradient $J_\sigma(x)$:

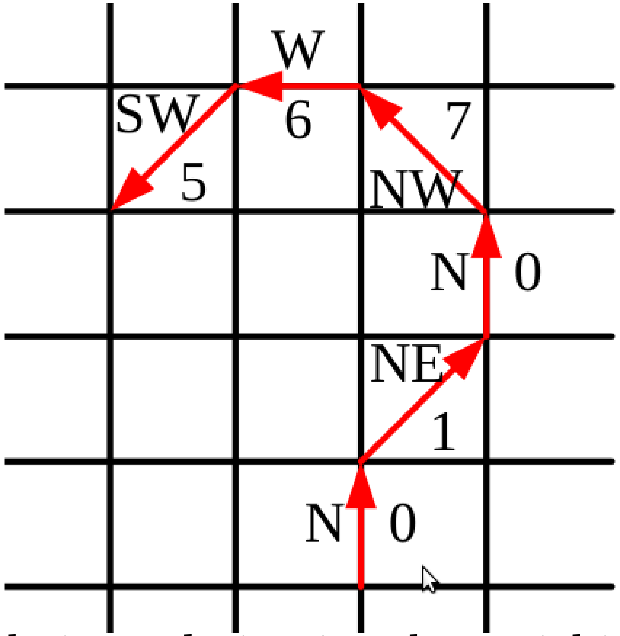
  $$S\sigma(x) = J_\sigma(x) = [\bigtriangledown^2 G_\sigma](x) \star I(x)$$

  The gradient operator dot product with the gradient is called the *Laplacian of Gaussian* (LoG)

- *The Laplacian of Gaussian* can be replaced with *Difference of*

Gaussian (DoG).

Edge linking



- Linking the edgels into chains involves picking up an unlinked edgel and following its neighbors in both directions.
- More compactly - A chain code encodes a list of connected points lying on an $N8$ grid using a three-bit code corresponding to the eight cardinal directions (N, NE, E, SE, S, SW, W, NW)
- Once the edgels have been linked into chains, we can apply an optional thresholding with hysteresis to remove low-strength contour segments (Canny).

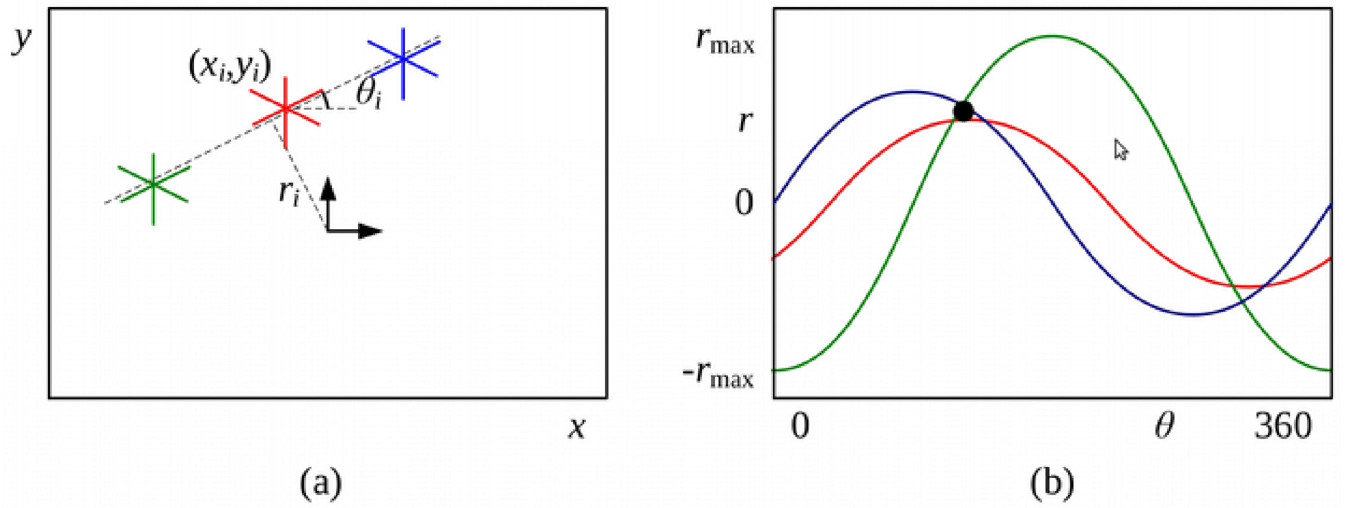
Edge smoothing

- A more useful representation is the *arc length parameterization* of a contour, $x(s)$, where s denotes the arc length along a curve,
- Arc-length parameterization can also be used to smooth curves in order to remove digitization noise,
- An alternative approach, based on selectively modifying different frequencies in a wavelet decomposition,
- Curve smoothing with a Gaussian kernel.

Lines

- Detecting and matching these lines can be useful in a variety of applications (pose estimation, in indoor environments, road lines, buildings etc.).
- In many applications it is preferable to approximate curve with a simpler representation (e.g., as a piecewise-linear polyline)
- If a smoother representation or visualization is desired, either approximating or interpolating splines or curves can be used.

Hough transforms



- (a) Each point votes for a complete family of potential lines $r_i(\theta) = x_i \cos\theta + y_i sin\theta$
- (b) Each pencil of lines sweeps out a sinusoid in (r, θ); their intersection provides the desired line equation
- Each edge point votes for all possible lines passing through it, and lines corresponding to high accumulator are examined for potential line fits.

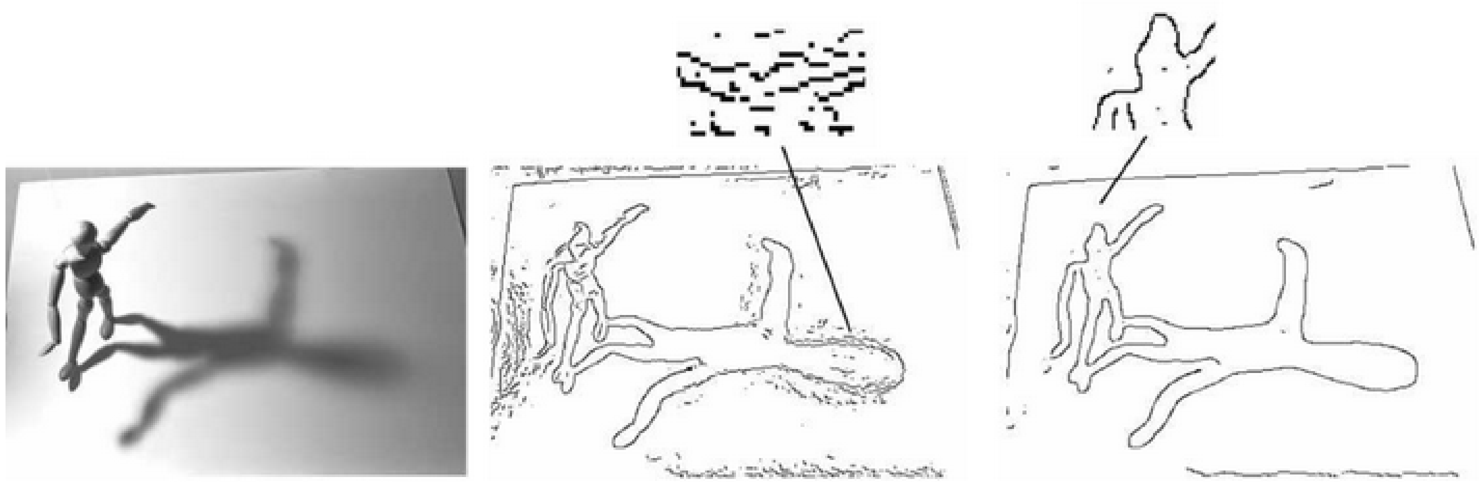
Edge Detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (artist is also using object-level knowledge)

Edge Detection Methods

- Gradient operators
  - Roberts
  - Prewitt
  - Sobel
- Gradient of Gaussian (Canny)
- Laplacian of Gaussian (Marr-Hildreth)
- Facet Model Based Edge Detector (Haralick)

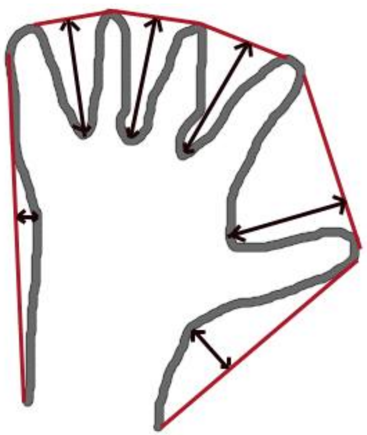
What are contours?



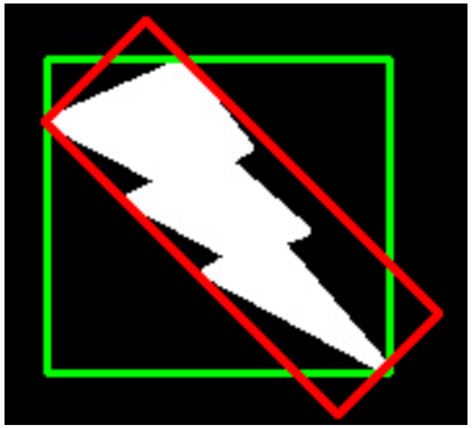
- Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity.
- The contours are a useful tool for shape analysis and object detection and recognition.

Contour Features - opencv

- **Moments**
- **Contour Area**
- **Contour Perimeter** - also called arc length.
- **Contour Approximation** - it approximates a contour shape to another, simpler shape
- **Convex Hull** - looks similar to contour approximation,

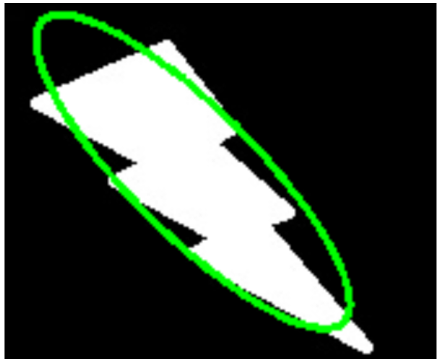


- **Bounding Rectangle** - *Straight Bounding Rectangle* and *Rotated Rectangle*

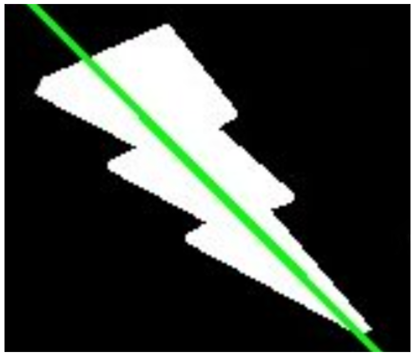


- **Minimum Enclosing Circle**

- **Fitting an Ellipse**



- **Fitting a Line**



Contour Properties

- **Aspect Ratio** - ratio of width to height of bounding rectangle

$$AspectRatio = \frac{Width}{Heigh}$$

- **Extent** ratio of contour area to bounding rectangle area.

$$Extent = \frac{ObjectArea}{BoundingRectangleArea}$$

- **Solidity** ratio of contour area to its convex hull area.

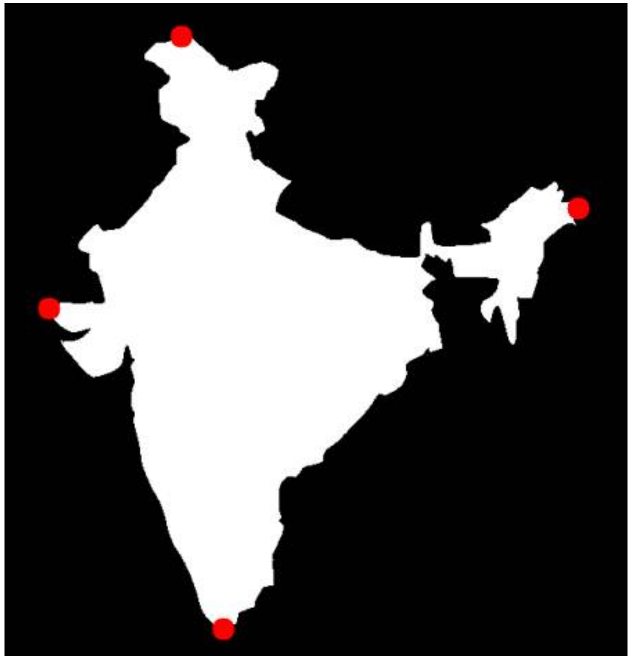
$$Solidity = \frac{ContourArea}{ConvexHullArea}$$

- **Equivalent Diameter** - diameter of the circle whose area is same as the contour area:

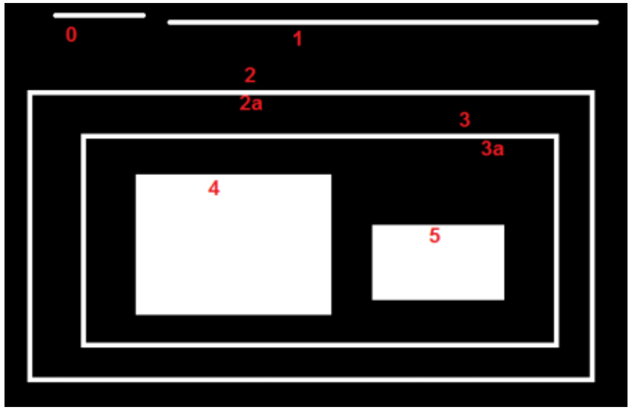
$$EquivalentDiameter = \sqrt{\frac{4 \times ContourArea}{\pi}}$$

- **Orientation** is the angle at which object is directed,

- Maximum Value, Minimum Value and their locations,
- Mean Color or Mean Intensity,
- **Extreme Points** means topmost, bottommost, rightmost and leftmost points of the object.



Contours Hierarchy - opencv



- When some shapes are inside other shapes (outer one as parent and inner one as child) then contours in an image has some relationship to each other - this relationship is called the **hierarchy**.
- Contours 0,1,2 are external or outermost. Contour-2a can be considered as a child of contour-2. Contour-3 is child of contour-2 and it comes in next hierarchy. Contours 4,5 are the children of contour-3a, and they come in the last hierarchy level.

Most popular segmentation techniques

- *active contours*,
- *level sets*,
- *region splitting and merging*,
- *mean shift* (mode finding),
- *normalized cuts* - splitting based on pixel similarity metrics,
- *binary Markov random fields* - solved using graph cuts.

Active contours

- **Snakes** is an energy-minimizing, two-dimensional spline curve that evolves (moves) towards image features such as strong edges,
- **Intelligent scissors** allow the user to sketch in real time a curve that clings to object boundaries,
- **Level set techniques** evolve the curve as the zero-set of a characteristic function, which allows them to easily change topology and incorporate region-based statistics.

Snakes

- Snakes are a two-dimensional generalization of the 1D energy-minimizing splines:

$$E_{int} = \int \alpha(s)||f_s(s)||^2 + \beta(s)||f_{ss}(s)||^2 ds,$$

where s is the arc-length along the curve $f(s) = (x(s), y(s))$ and $\alpha(s)$ and $\beta(s)$ are first and second-order weighting functions

- We can discretize this energy by sampling the initial curve position evenly along its length to obtain

$$E_{int} = \sum_i \alpha(i)||f(i+1)-f(i)||^2/h^2+\beta(i)||f(i+1)-2f(i)+f(i-1)||^2/h^4,$$

where h is the step size of resample the curve along its arc-length iteration.

Internal spline energy

- Snake simultaneously minimizes external image-based and constraint-based potentials.
- The image-based potentials are the sum of several terms:

$$E_{image} = w_{line}E_{line} + w_{edge}E_{edge} + w_{term}E_{term},$$

where the *line* term attracts the snake to dark ridges, the *edge* term attracts it to strong gradients (edges), and the *term* term attracts it to line terminations.

Snakes

- Most systems only use the edge term, which can either be directly proportional to the image gradients,

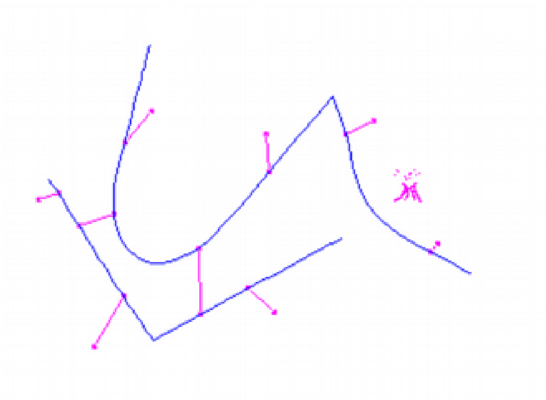
$$E_{edge} = \sum_i -|| \bigtriangledown I(f(i))||^2,$$

or to a smoothed version of the image Laplacian,

$$Eedge = \sum_i |(G_\sigma \star \bigtriangledown^2 I)(f(i))|^2.$$

People also sometimes extract edges and then use a distance map to the edges as an alternative to these two originally proposed potentials.

Snakes - example

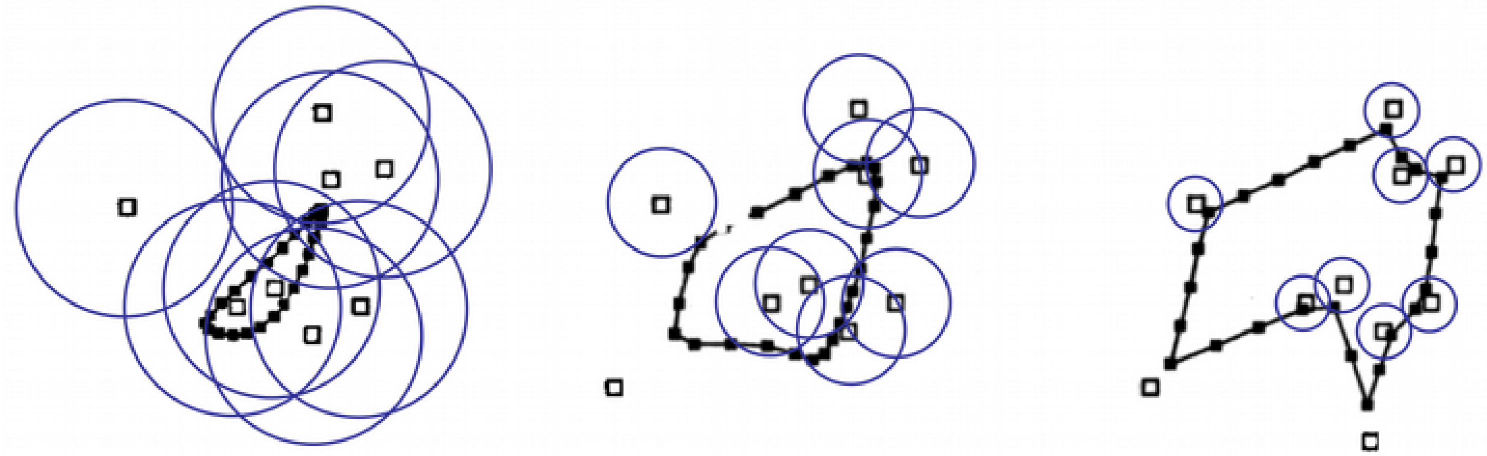


- In interactive applications, a variety of user-placed constraints can also be added, e.g., attractive (spring) forces towards anchor points $d(i)$,

$$E_{spring} = k_i||f(i) - d(i)||^2,$$

As the snakes evolve by minimizing their energy, they often "wiggle" and "slither", which accounts for their popular name.

Elastic net



- Energy-minimizing framework is based on the Traveling Salesman Problem,
- A snake that is constrained to pass through each city could solve this problem (without any optimality guarantees)
- Closed squares linked by straight line segments are the tour points.
- The blue circles indicate the approximate extent of the attraction force of each city.

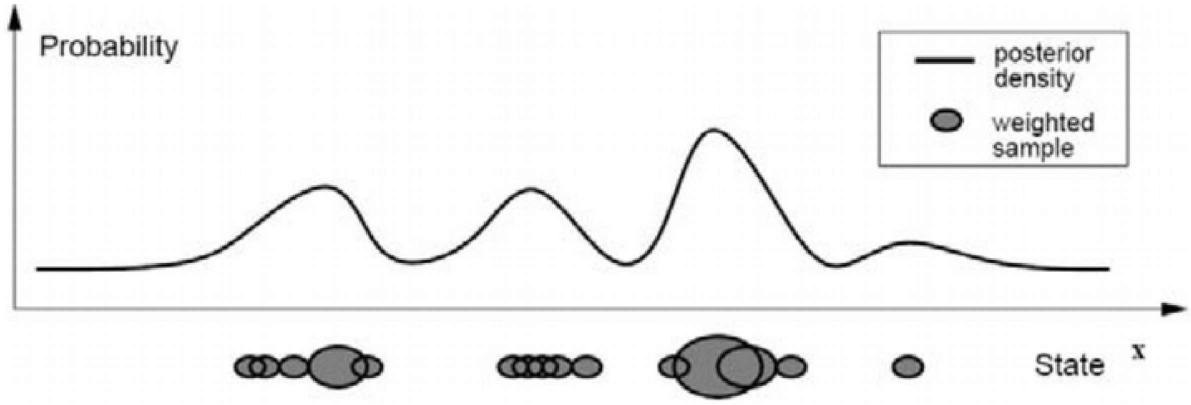
Dynamic snakes

- Object of interest is being tracked from frame to frame as it deforms and evolves.
- It makes sense to use estimates from the previous frame to predict and constrain the new estimates.
- One way to do this is to use Kalman filtering, which results in a formulation called **Kalman snakes**.
- The Kalman filter is based on a linear dynamic model of shape parameter evolution,

$$x_t = Ax_{t-1} + w_t,$$

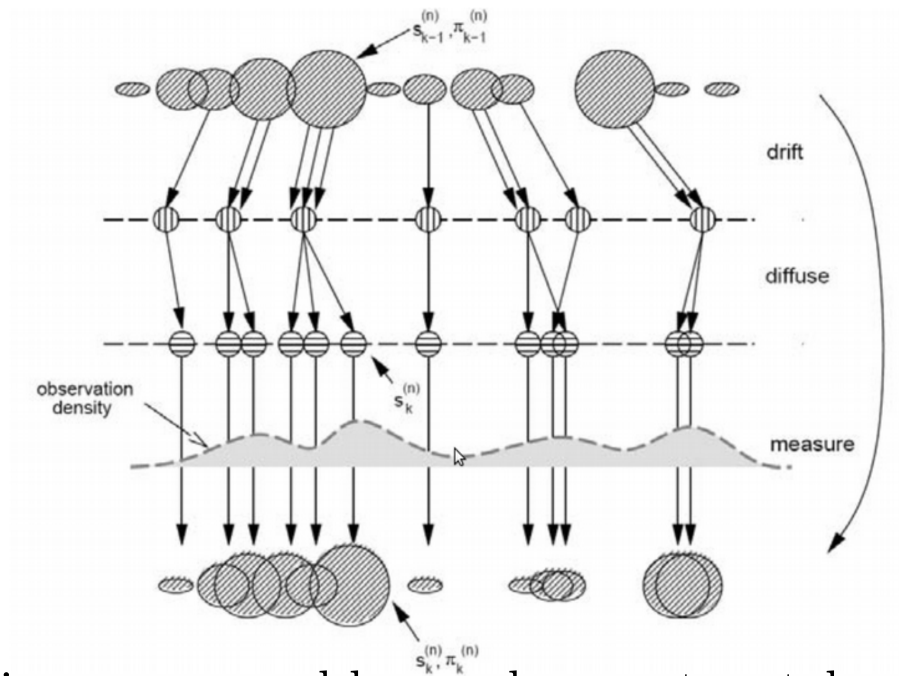
where x_t and x_{t-1} are the current and previous state variables, A is the linear transition matrix, and w is a noise (often modeled as a Gaussian).

Particle filtering



- Particle filtering techniques represent a probability distribution using a collection of weighted point samples,
- To update the locations of the samples according to the linear dynamics (deterministic drift),
- The centers of the samples are updated and multiple samples are generated for each point.

Condensation by using particle filter



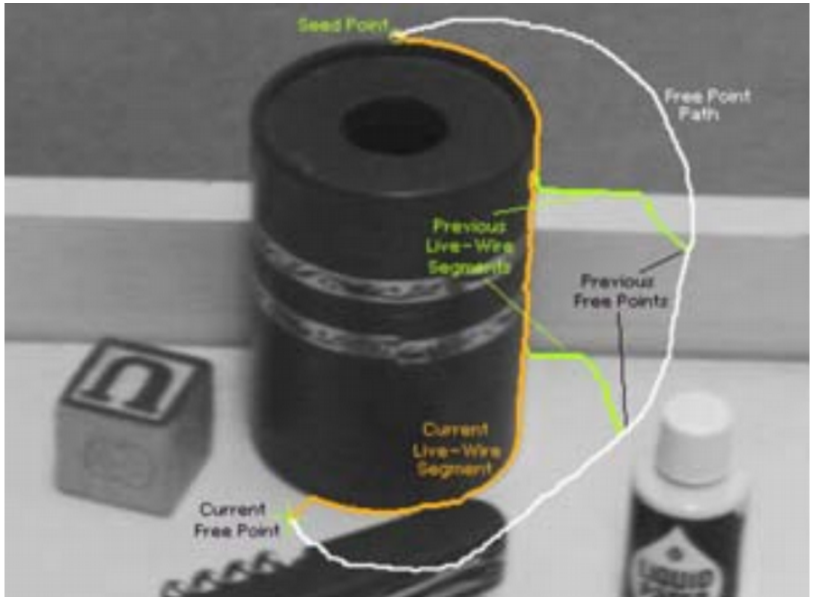
- Locations are moved by random vectors taken from the distribution,
- Finally, the weights of these samples are multiplied by the measurement probability density,
- We take each sample and measure its likelihood given the current (new) measurements.

Scissors

Active contours allow a user to roughly specify a boundary of interest and have the system evolve the contour towards a more accurate location.

- As the user draws a rough outline, the system computes and draws a better curve that clings to high-contrast edges,
- Image is first pre-processed to associate low costs with edges (combination of zero-crossing, gradient magnitudes, and gradient orientations to compute these costs).
- Next, as the user traces a rough curve, the system continuously recomputes the lowest- cost path.

Scissors - example



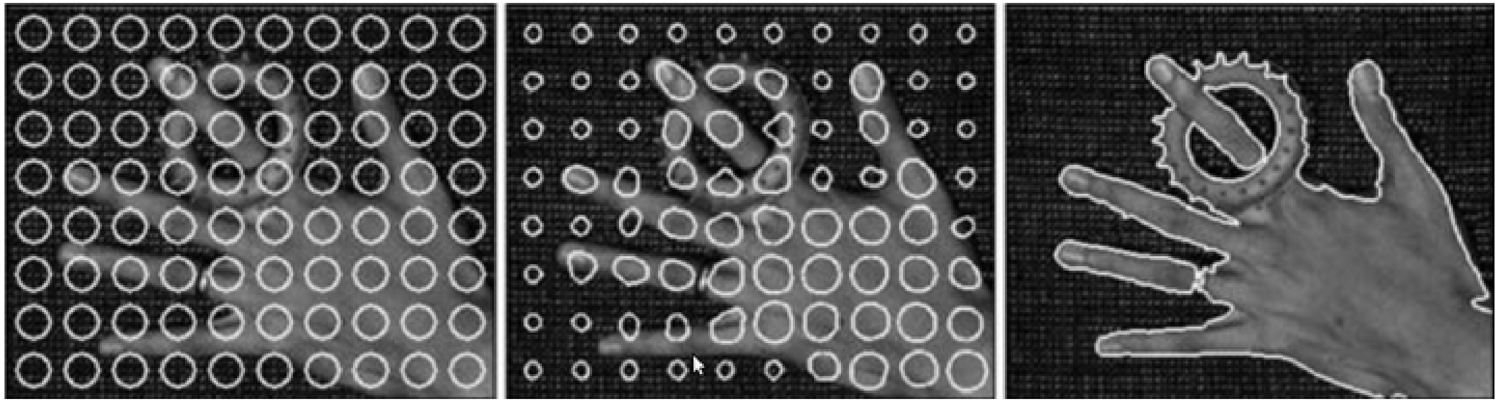
- As the mouse traces the white path, the scissors follow the orange path along the object boundary,
- Green curves show intermediate positions.

Level Sets

An alternative representation for such closed contours is to use a level set, where the zero-crossing of a characteristic function define the curve.

- Level sets evolve to fit and track objects of interest by modifying the underlying **embedding function** (2D function) $\phi(x, y)$ instead of the curve $f(s)$,
- An alternative approach is to re-cast the problem in a segmentation framework, where the energy measures the consistency of the image statistics inside and outside the segmented regions.

Level Sets - example



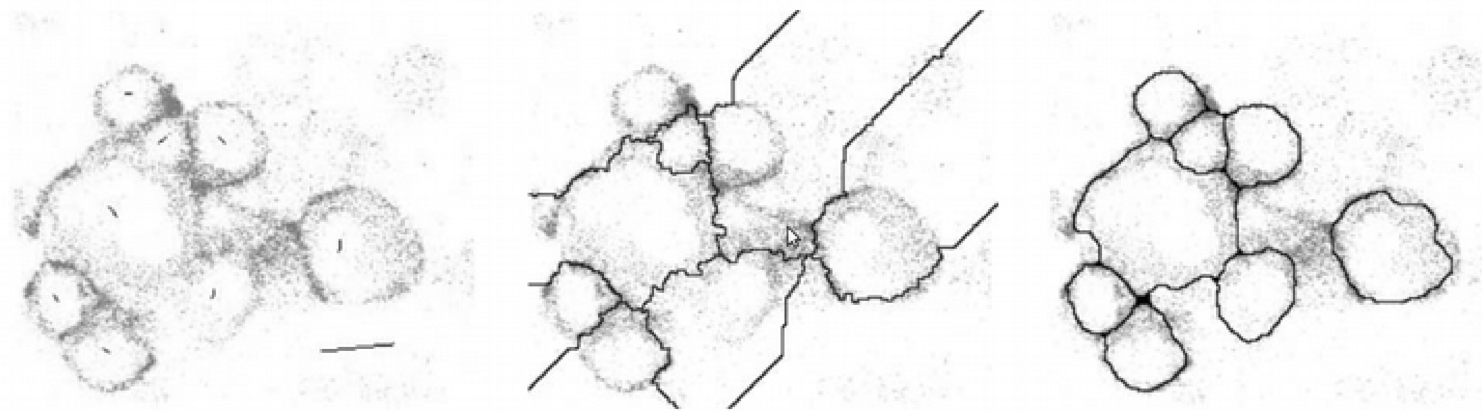
- Gaussians are used to model the foreground and background pixel distributions.
- The initial circles evolve towards an accurate segmentation of foreground and background, adapting their topology as they evolve.

Watershed

A technique related to thresholding, since it operates on a grayscale image, is watershed computation.

- the simplest possible technique for segmenting a grayscale image is to select a threshold and then compute connected components.
- This technique segments an image into several catchment basins, which are the regions of an image (interpreted as a height field or landscape),
- Since images rarely have dark regions separated by lighter ridges, watershed segmentation is usually applied to a smoothed version of the gradient magnitude image.

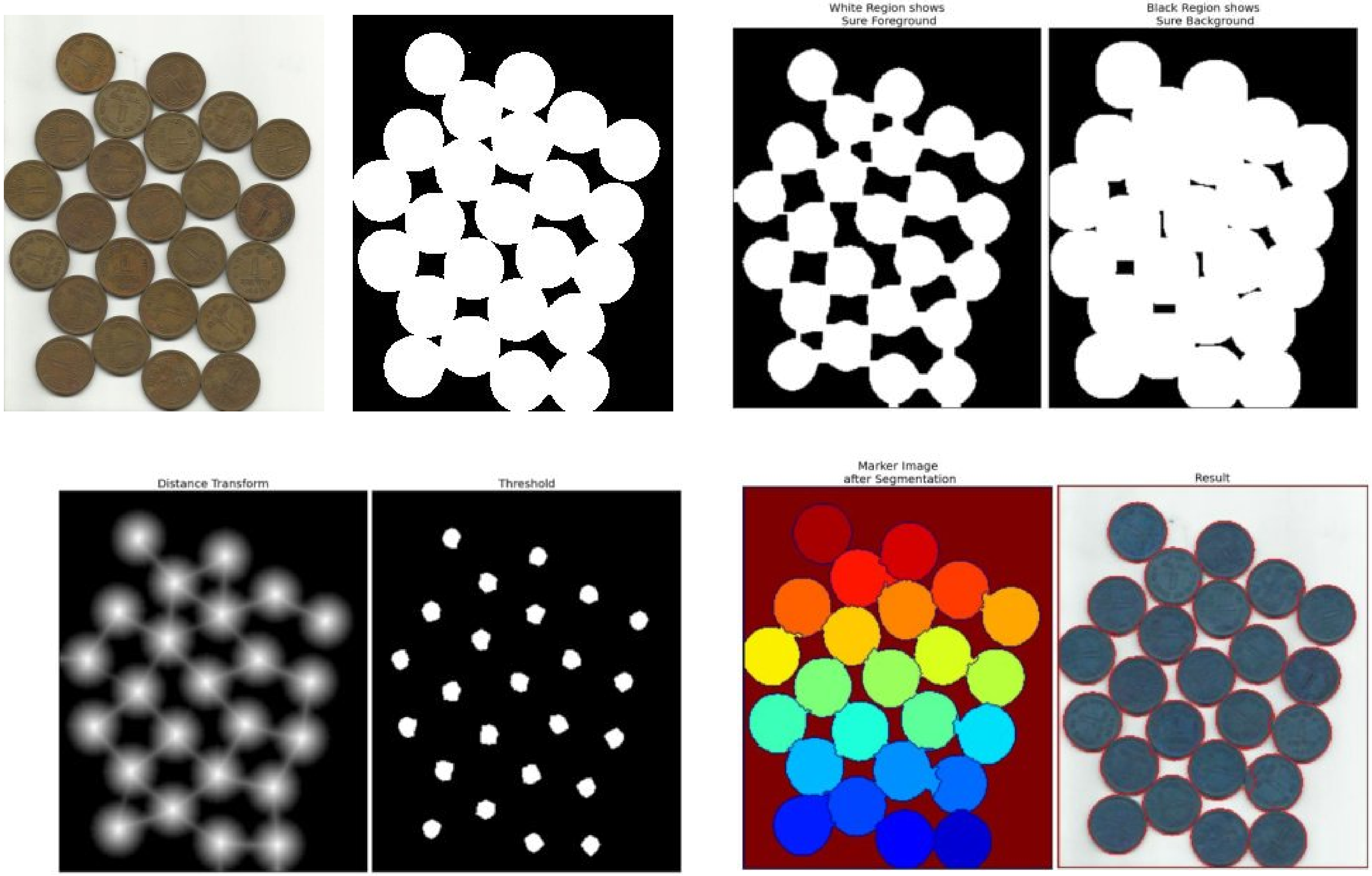
Watershed - example



(a) (b) (c)

- (a) Original confocal microscopy image with marked seeds (line segments);
- (b) Standard watershed segmentation;
- (c) Locally constrained watershed segmentation.

Image Segmentation with Watershed Algorithm - opencv



Graph-based segmentation

That algorithm uses relative dissimilarities between regions to determine which ones should be merged.

- It starts with a pixel to pixel dissimilarity measure $w(e)$ that measures, for example, intensity differences between $N8$ neighbors.
- For any region R, its internal difference is defined as the largest edge weight in the region's minimum spanning tree,

$$Int(R) = \min_{e \in R} w(e).$$

- For any two adjacent regions with at least one edge connecting their vertices, the difference between these regions is defined as the minimum weight edge connecting the two regions,

$$Dif(R_1, R_2) = \min_{e=(v_1 \in R_1, v_2 \in R_2)} w(e).$$

- Their algorithm merges any two adjacent regions whose difference is smaller than the minimum internal difference of these two regions,

$$MInt(R_1, R_2) = \min(Int(R_1) + \tau(R_1), Int(R_2) + \tau(R_2)),$$

where $\tau(R)$ is a heuristic region penalty

Mean shift and mode finding

Mean-shift and mode finding techniques, such as k-means and mixtures of Gaussians.

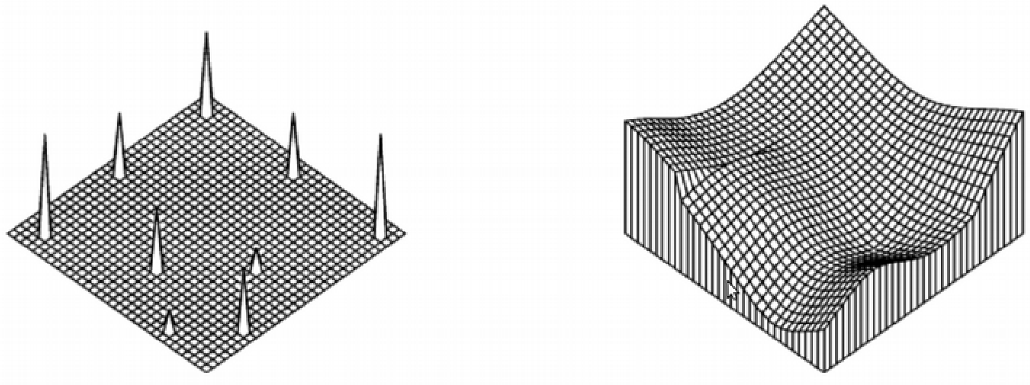
- The k-means and mixtures of Gaussians techniques use a parametric model of the density function,
- Density is the superposition of a small number of simpler distributions (e.g., Gaussians) whose locations (centers) and shape (covariance) can be estimated.
- Mean shift, on the other hand, smoothes the distribution and finds its peaks,
- Since a complete density is being modeled, this approach is called *non-parametric*.

Global optimization

We can first formulate the goals of the desired transformation using some optimization criterion and then find the solution that best meets this criterion.

- *Regularization* or *variational methods*, constructs a continuous global energy function that describes the desired characteristics and then finds a minimum energy solution,
- Formulates the problem using Bayesian statistics, modeling both the noisy measurement process that produced the input images as well as *prior assumptions* about the solution space (often encoded using a *Markov Random Field*)

Regularization - ill-conditioned problems



(a) (b)

- If we use polynomial interpolation, such kind of problems (a) are ill-conditioned,
- Since we are trying to recover the unknown function $f(x, y)$ from which the data point $d(x_i, y_i)$ were sampled, such problems are also often called *inverse problems*,
- Since we are trying to recover a full description from a limited set of samples.

Energy measures of function

- For one-dimensional functions $f(x)$, we can integrate the squared first derivative of the function,

$$\varepsilon_1 = \int f_x^2(x)dx$$

or perhaps integrate the squared second derivative,

$$\varepsilon_2 = \int f_{xx}^2(x)dx$$

we use subscripts to denote differentiation.

- Such energy measures are examples of *functionals*, which are operators that map functions to scalar values. They are also often called *variational methods*.

Data penalty

- In the smoothness term, regularization also requires some kind of *data penalty*,
- For scattered data interpolation, the data term measures the distance between the function $f(x, y)$ and a set of data points $d_i = d(x_i, y_i)$,

$$\varepsilon_d = \sum_i [f(x_i, y_i) - d_i]^2.$$

- To obtain a global energy that can be minimized, the two energy terms are usually added together,

$$\varepsilon = \varepsilon_d + \lambda \cdot \varepsilon_s,$$

  where ε_s is the *smoothness penalty* and λ is the *regularization parameter*.

Two-dimensional discrete data energy

- The two-dimensional discrete data energy is written as

$$E_d = \sum_{i,j} w(i,j)[f(i,j) - d(i,j)]^2,$$

where the local weights $w(i,j)$ control how strongly the data constraint is enforced.

- These values are set to zero where there is no data and can be set to the inverse variance of the data measurements when there is data
- The total energy of the discretized problem can now be written as a quadratic form

$$E = E_d + \lambda E_s = x^T A x - 2x^T b + c,$$

where $x = [f(0,0) \dots f(m-1,n-1)]$ is called the *state vector*.

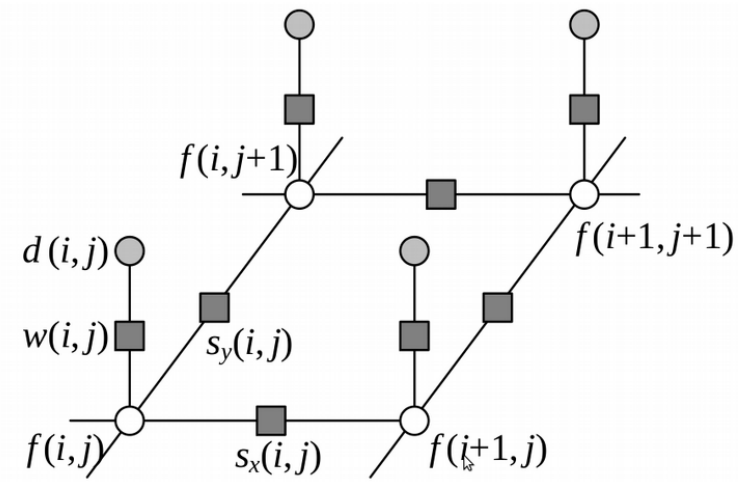
Minimizing the quadratic form of energy

Goal is to minimize energy $E_s = x^T Ax - 2x^T b + c$,

- The sparse symmetric positive-definite matrix A is called the *Hessian* since it encodes the second derivative of the energy function,
- We call b the weighted data vector. Minimizing the above quadratic form is equivalent to solving the sparse linear system

$$Ax = b,$$

First-order regularization



The discrete smoothness energy functions become is $E_1 =$

$$= \sum_{i,j} s_x(i,j)[f(i+1,j)-f(i,j)-g_x(i,j)]^2 + s_y(i,j)[f(i,j+1)-f(i,j)-g_y(i,j)]^2$$

- The white circles are the unknowns $f(i,j)$ while the dark circles are the input data $d(i,j)$.
- In the resistive grid interpretation, the d and f values encode input and output voltages and the black squares denote resistors whose conductance is set to $s_x(i,j)$, $s_y(i,j)$, and $w(i,j)$.

Markov random fields

- Bayesian model can separately model the noisy image formation (measurement) process, and statistical prior model over the solution space.
- Markov random field models can be defined over discrete variables, such as image labels (where the variables have no proper ordering), for which regularization does not apply.

Bayes' Rule

- The posterior distribution for a given set of measurements y, $p(y|x)$, combined with a prior $p(x)$ over the unknowns x, is given by

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

where $p(y) = \sum_x p(y|x)p(x)$ is a normalizing constant used to make the $p(x|y)$ distribution proper (integrate to 1).

- Taking negative logarithm of both sides of, we get

$$-\log p(x|y) = -\log p(y|x) - \log p(x) + C$$

which is the *negative posterior log likelihood*

Maximum a posteriori or MAP

- To find the most likely (maximum a posteriori) solution x given some measurements y, we simply minimize this negative log likelihood, which can also be thought of as an energy,

$$E(x, y) = E_d(x, y) + E_p(x).$$

- The first term $E_d(x, y)$ is the **data energy** or data penalty - it measures the negative log likelihood that the data were observed given the unknown state x.
- The second term $E_p(x)$ is the **prior energy** - it plays a role analogous to the smoothness energy in regularization.

Image processing applications - Markov random field

- The unknowns x are the set of output pixels

$$x = [f(0,0)...f(m-1, n-1)],$$

and the data are (in the simplest case) the input pixels

$$y = [d(0,0)...d(m-1, n-1)]$$

- The probability $p(x)$ is a Gibbs or Boltzmann distribution, whose negative log likelihood can be written as a sum of pairwise interaction potentials,

$$E_p(x) = \sum_{\{(i,j),(k,l)\}\in N} V_{i,j,k,l}(f(i,j), f(k,l)),$$

where $N(i,j)$ denotes the neighbors of pixel (i,j).

- The energy may have to be evaluated over a larger set of cliques, which depend on the order of the Markov random field

Markov random field - binary example

- Examples of such fields include 1-bit (black and white) scanned document
- To denoise a scanned image, we set the data penalty to reflect the agreement between the scanned and final images,

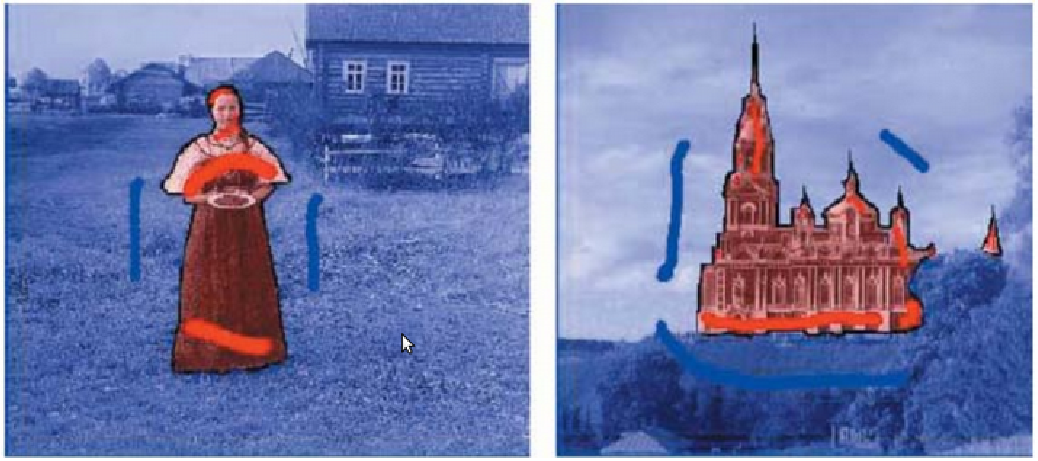
$$E_d(i,j) = w\delta((i,j), d(i,j))$$

- The smoothness penalty to reflect the agreement between neighboring pixels

$$E_p(i,j) = E_x(i,j) + E_y(i,j) = s\delta(f(i,j), f(i+1,j)) + s\delta(f(i,j), f(i,j+1)).$$

- The simplest approach is to perform gradient descent. This approach is known as contextual classification.

Image segmentation - MRF



- The user draws a few red strokes in the foreground object and a few blue ones in the background.
- The system computes color distributions for the foreground and background and solves a binary MRF.
- The smoothness weights are modulated by the intensity gradients (edges), which makes this a conditional random field (CRF).

Graph cuts and energy-based methods

That segmentation algorithms is the desire to group pixels that have similar statistics and to have the boundaries between pixels in different regions.

Graph cuts and energy-based methods

the energy corresponding to a segmentation problem can be written as:

$$E(f) = \sum_{i,j} E_r(i,j) + E_b(i,j),$$

where the region term

$$E_r(i,j) = E_S(I(i,j); R(f(i,j)))$$

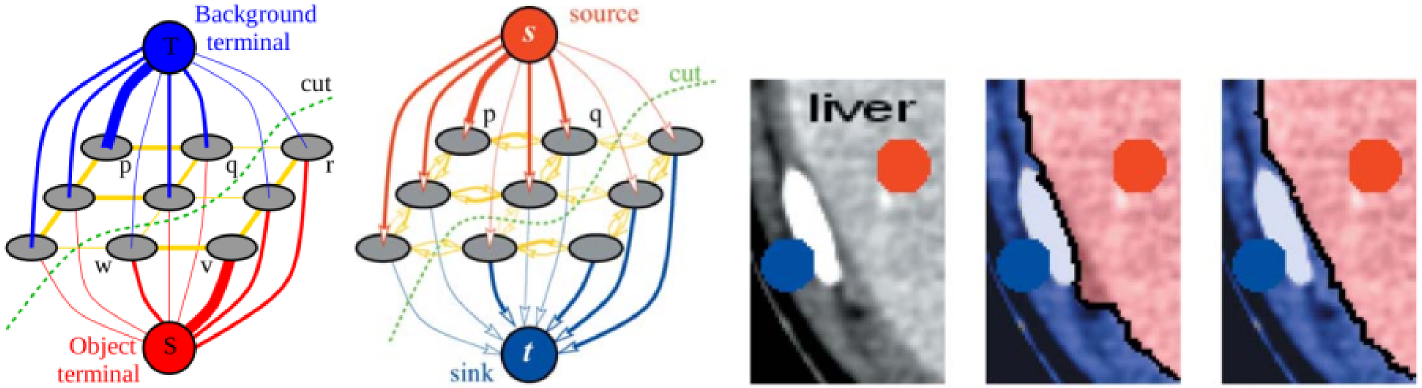
is the negative log likelihood that pixel intensity (or color) $I(i,j)$ is consistent with the statistics of region $R(f(i,j))$ and the boundary term:

$$E_b(i,j) = s_x(i,j)\delta(f(i,j) - f(i+1,j)) + s_y(i,j)\delta(f(i,j) - f(i,j+1))$$

measures the inconsistency between $N4$ neighbors modulated by local horizontal and vertical smoothness terms $s_x(i,j)$ and $s_y(i,j)$. Region statistics can be something as:

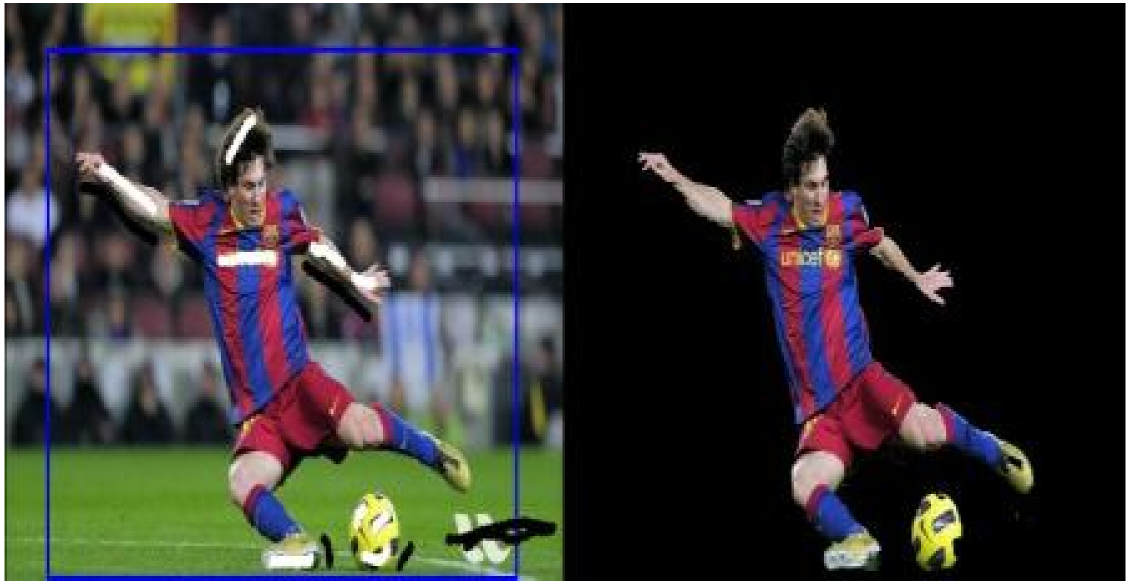
$$E_S(I; \mu_k) = ||I - \mu_k||^2.$$

Graph cuts and energy-based methods - example



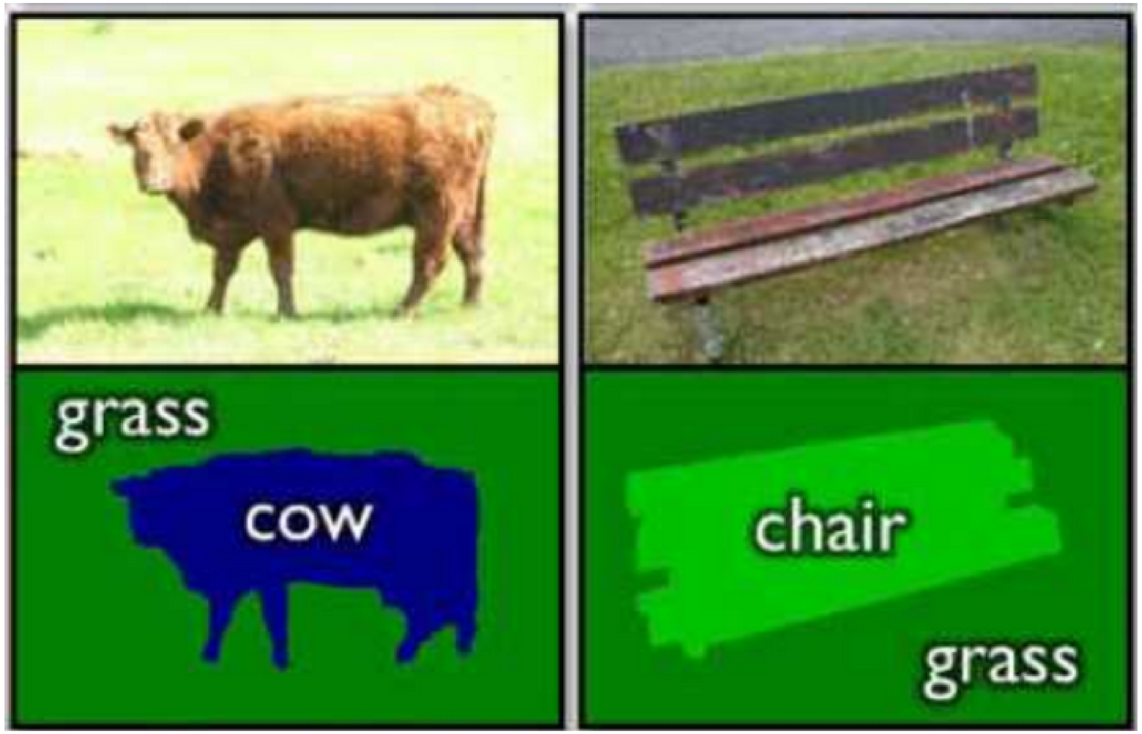
- User first marks pixels in the background and object regions,
- These pixels then become the seeds that tie nodes in the $S^{\smile}T$ graph to the source and sink labels S and T,
- Edges in the graph are derived from the region and boundary energy terms.

Interactive Foreground Extraction using GrabCut Algorithm - opencv



- First player and football is enclosed in a blue rectangle.
- Then some final touchups with white strokes (denoting foreground) and black strokes (denoting background) is made.

Semantic segmentation



- A challenging version of general object recognition and scene understanding is to simultaneously perform recognition and accurate boundary segmentation,
- To be continued...

Tasks for labs.

Implement (follow an example from opencv) the *Watershed* segmentation algorithm in two versions

- Where focus areas are selected automatically (coin photos can be used),
- Where segmentation areas are manually marked (other image than in case of 1)
- Try apply this technique to task of counting coins on the tray - previously task.