Centre for Medical Image Computing

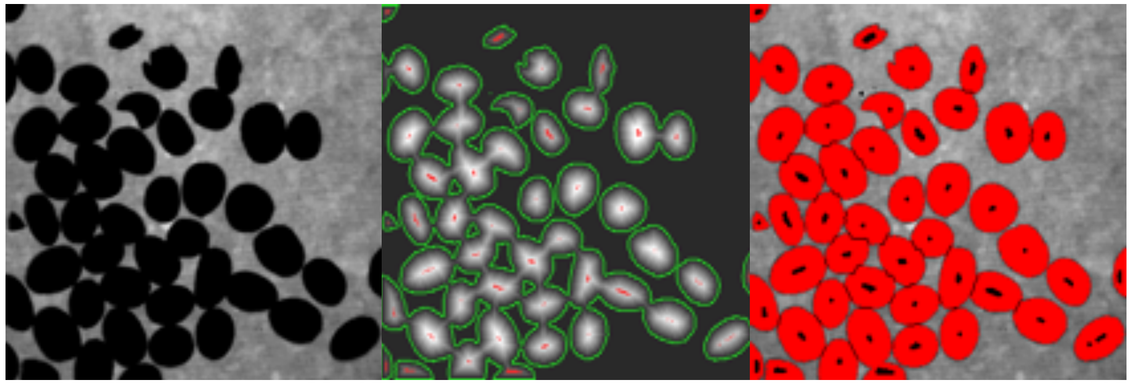
Segmentation

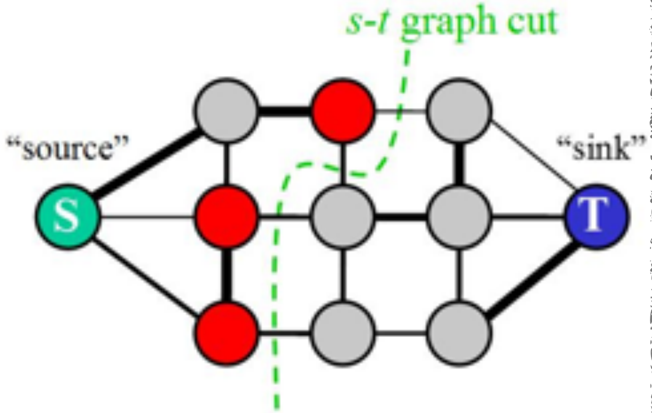
Clustering/EM/Active Contours

M. Jorge Cardoso
manuel.cardoso@ucl.ac.uk

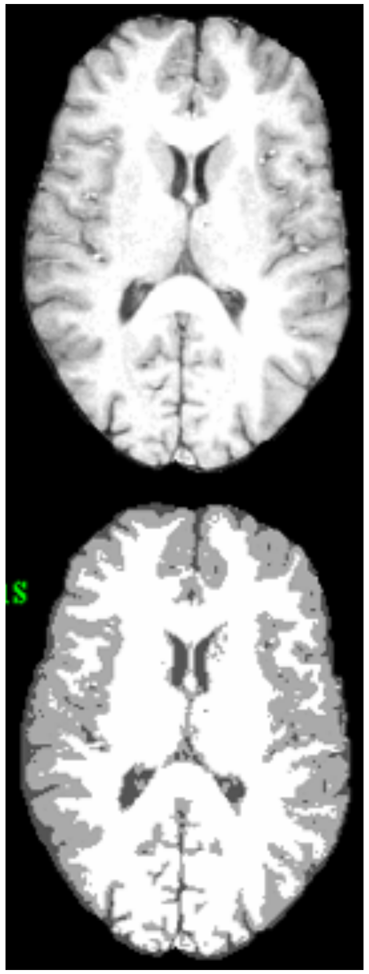
Segmentation

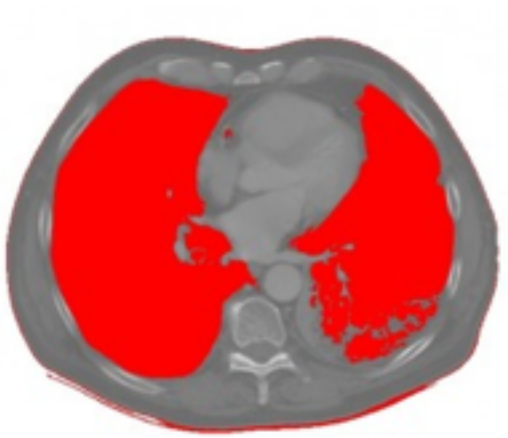
Region/Edges

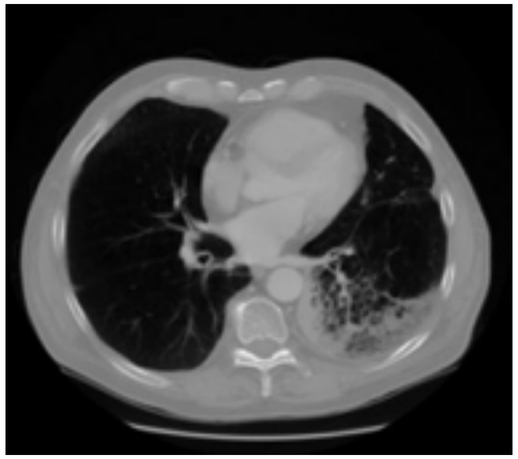


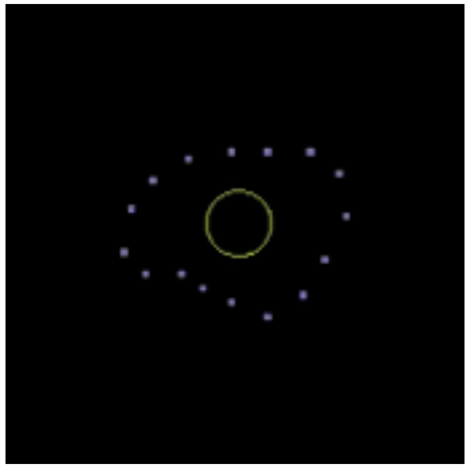


Intensity/Histogram/Clustering



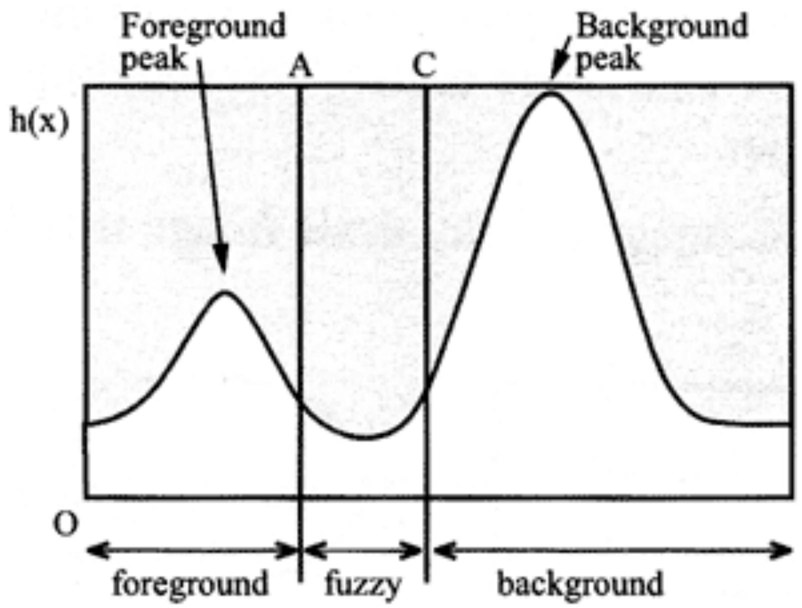


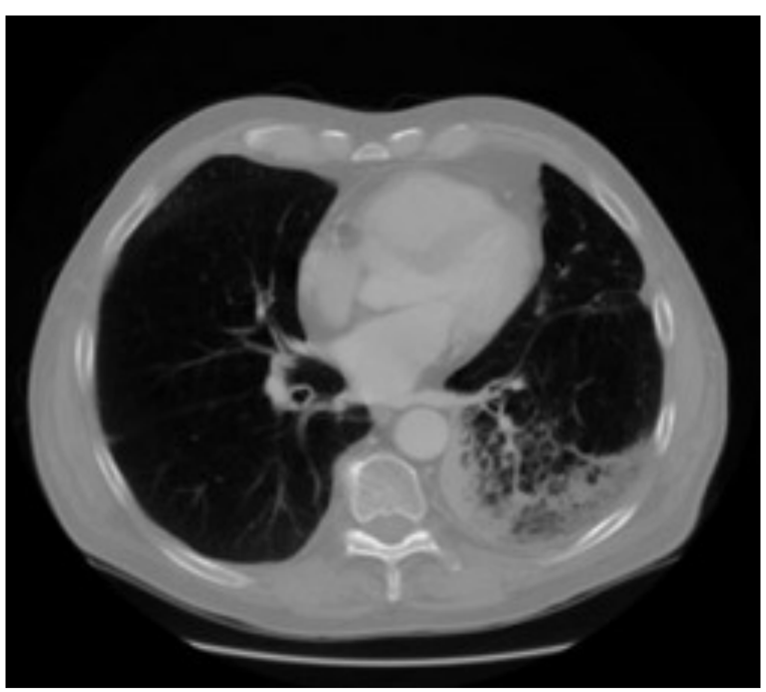


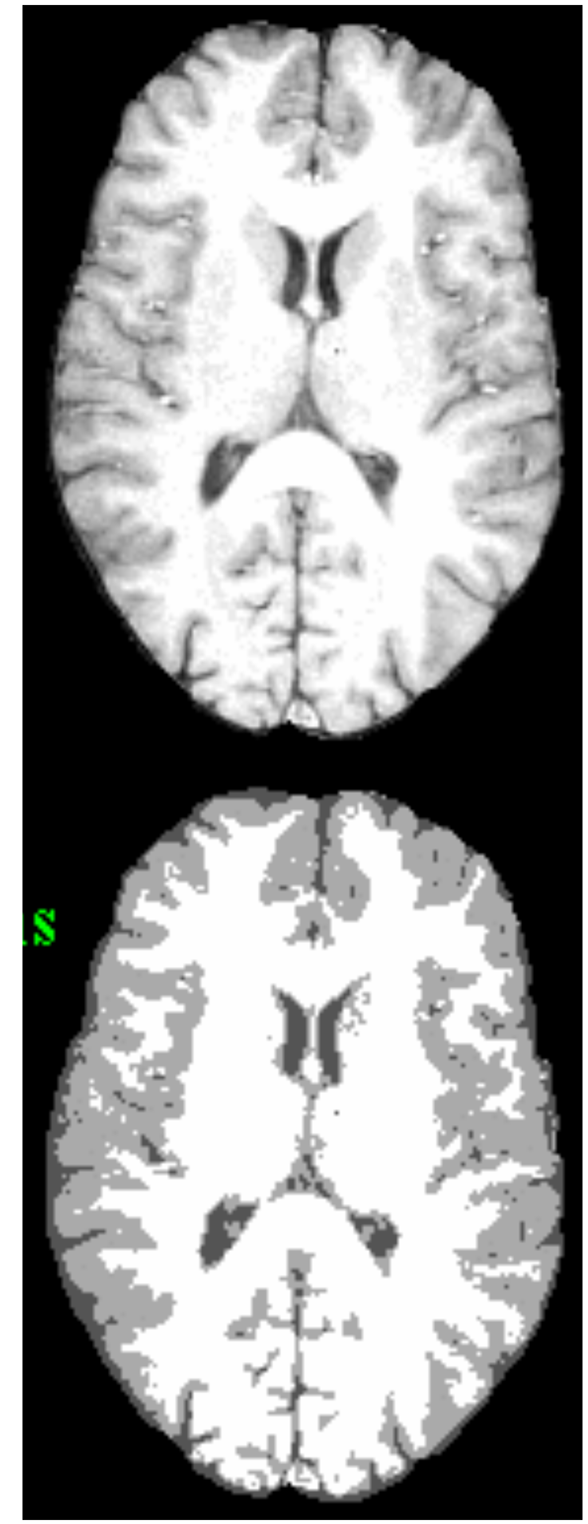


INTENSITY/HISTOGRAM/CLUSTERING - DATA CLUSTERING

Segmentation - Histogram Based







What is Clustering?

“Also called unsupervised learning, sometimes called classification by statisticians and sorting by psychologists and segmentation by people in marketing”

- Organising data into classes such that there is
  - high intra-class similarity
  - low inter-class similarity
- Finding the class labels and the number of classes directly from the data

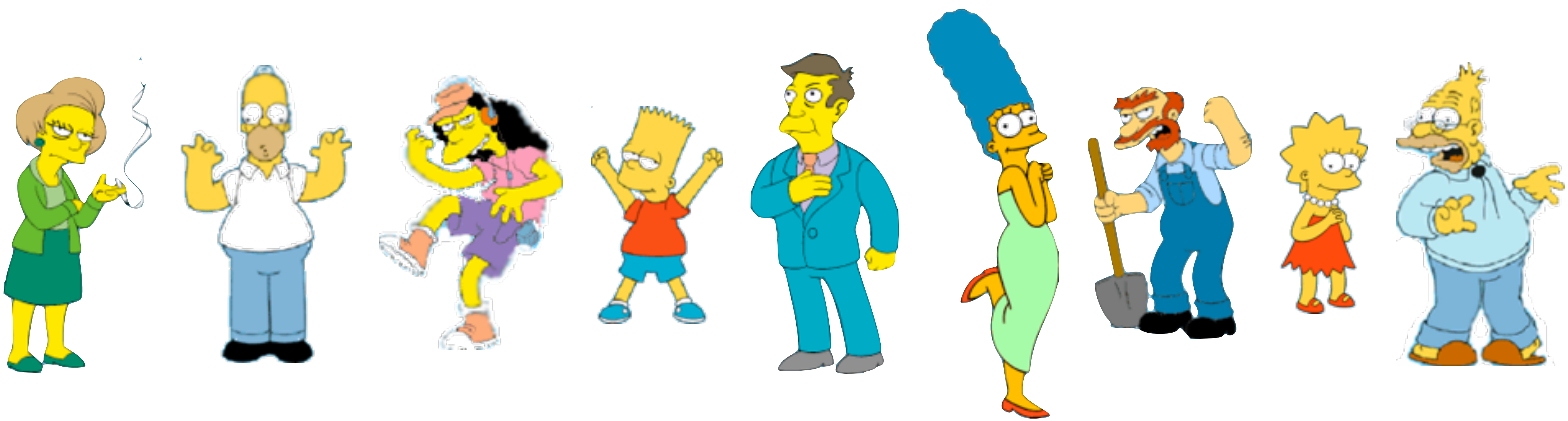
(in contrast to classification).

- More informally, finding natural groupings among objects.

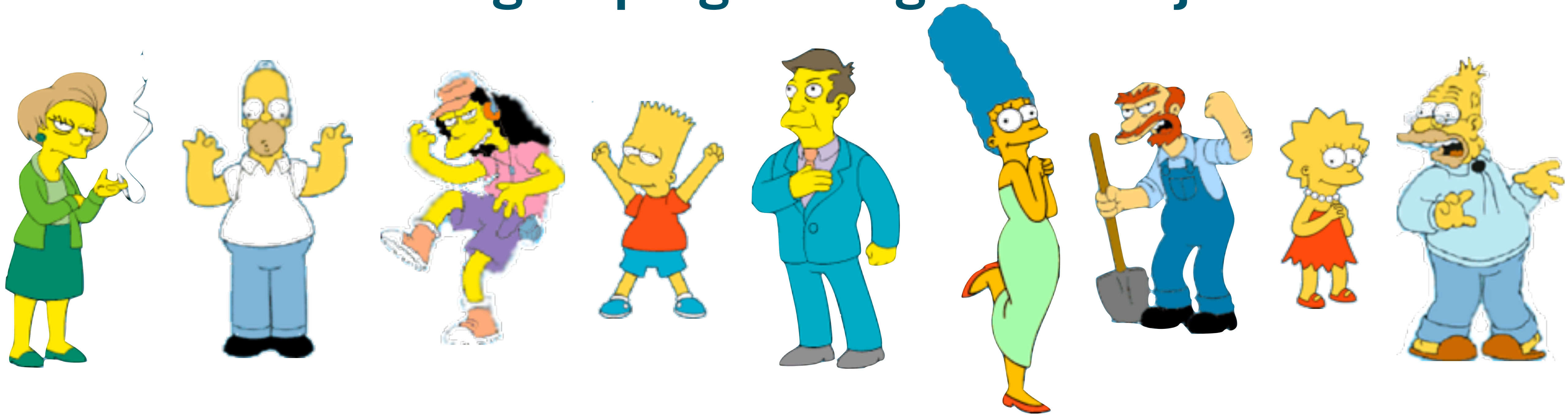
Clustering applied to image segmentation:

- What is segmentation?
  - Refers to the process of partitioning a digital image into multiple regions to simplify and/or change the representation of an image into something that is more meaningful and easier to analyse.
- Image segmentation is typically used to locate objects and boundaries in images.
- The result of image segmentation techniques is a set of regions or a set of contours extracted from the image

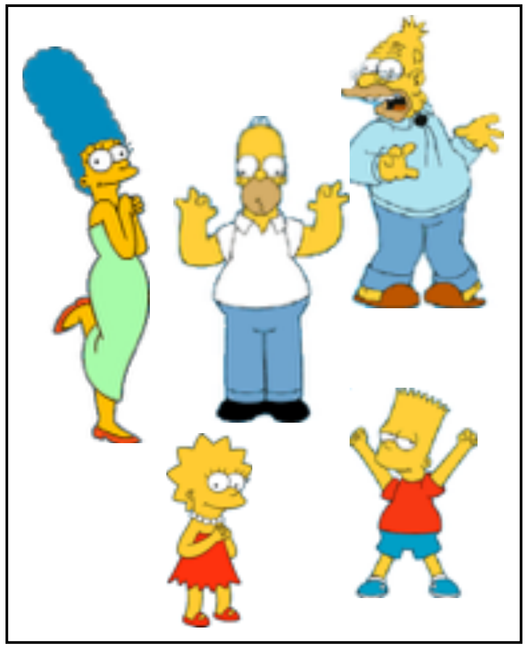
What is a natural grouping among these



What is a natural grouping among these objects?

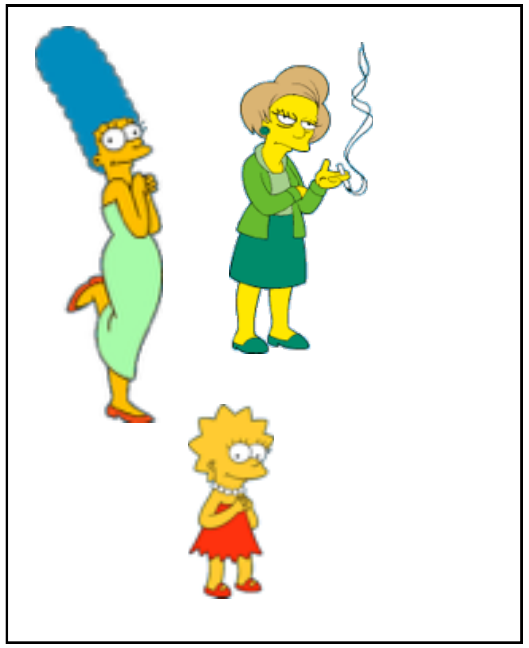


Clustering is subjective



Simpson's Family

School Employees



Females

Males

Clustering - image processing

- Features
  - Intensity
    - Grey
    - RGB
  - Other
    - Patterns
    - Shapes
    - Kernel similarities

Non Iterative methods

- Threshold
  - Manual
  - Known Distribution/Fraction - using the cumulative histogram
  - Peaks and Valleys
- Statistical
  - Otsu - The combined intra-class variance is minimal
  - Trussel - Largest Student's T-test
- Entropy
  - Shanon - Minimise Shanon's Entropy

Iterative clustering algorithms:

- Clustering:
  - K-Means
  - Fuzzy K-Means
  - Others (K-NN ,SVM)

- Clustering by modelling:
  - EM

Desirable Properties of a Clustering Algorithm

- Scalability
- Able to deal with noise and outliers
- Input order insensitive
- Incorporation of user-specified constraints and priors
- **Interpretability** and **usability**

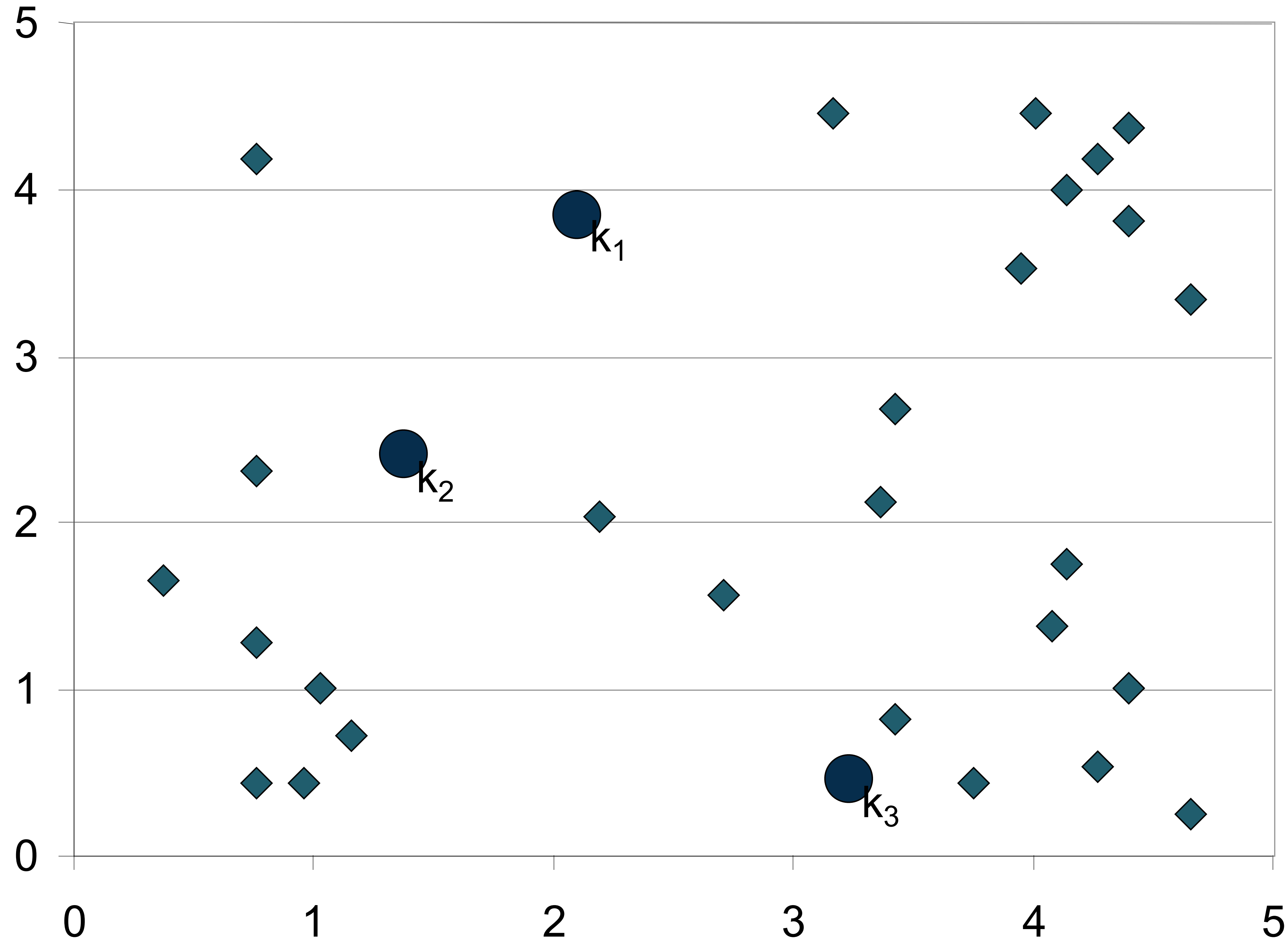
K-MEANS

Algorithm k-means

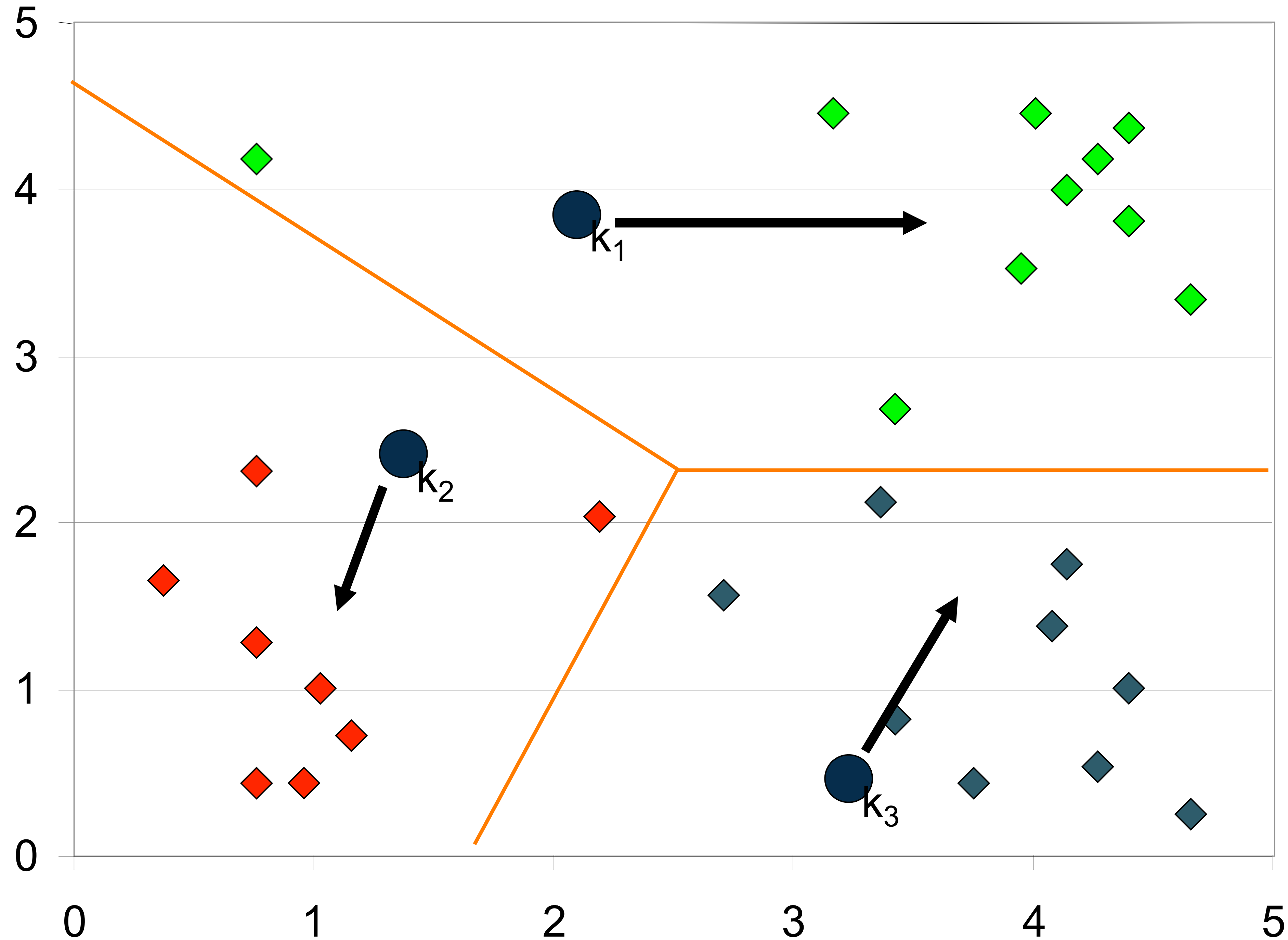
$$arg\ min_s \sum_{i=1}^{k} \sum_{x_j \in S_i} ||x_j - \mu_i||^2$$

- S_i is the set of features
- μ_i is the average position of cluster i
- $x_j \in Si$ is true when the distance between x_j and a specific cluster centre *i* is the smallest

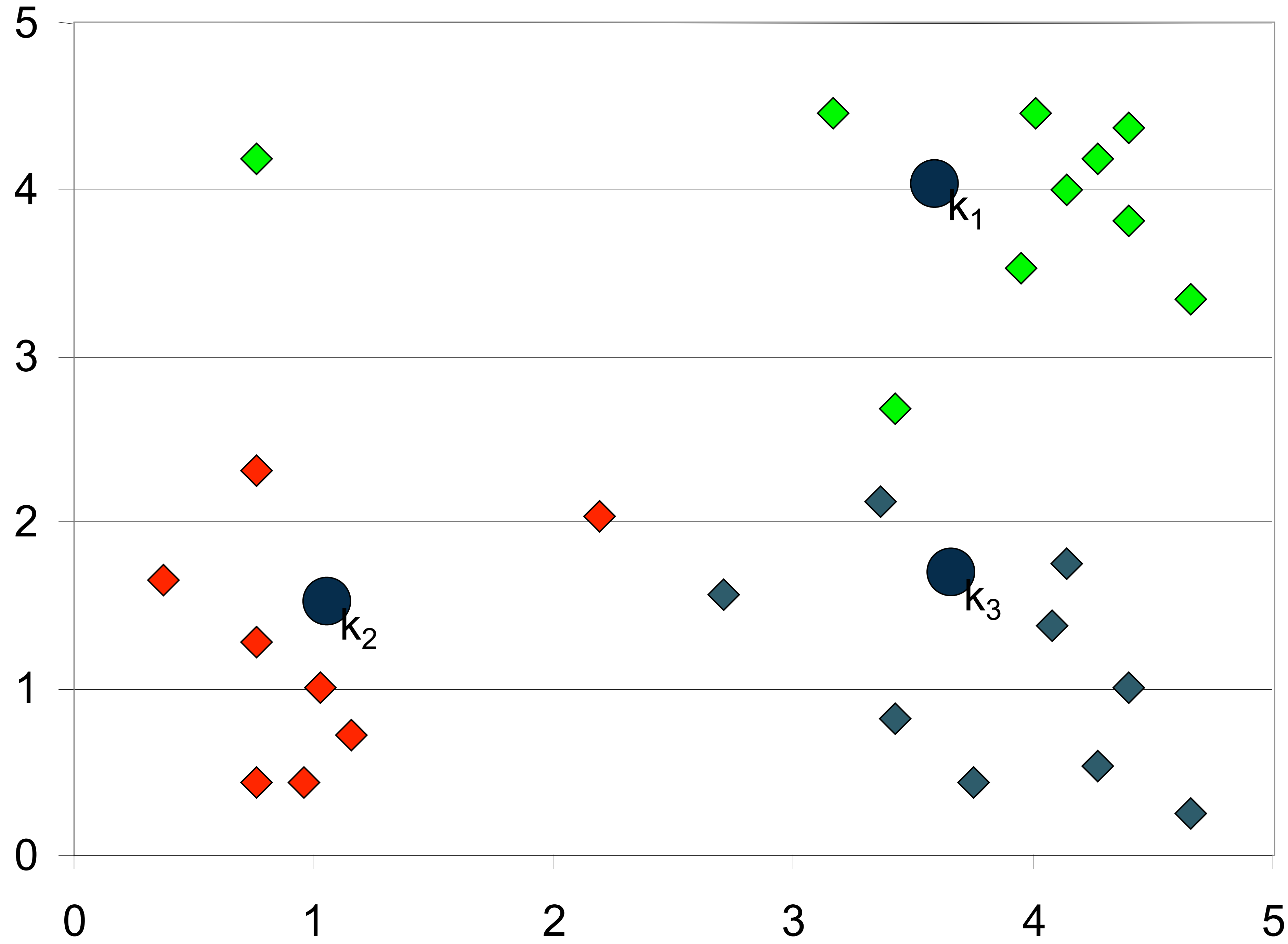
K-means - Distance Metric: Euclidean



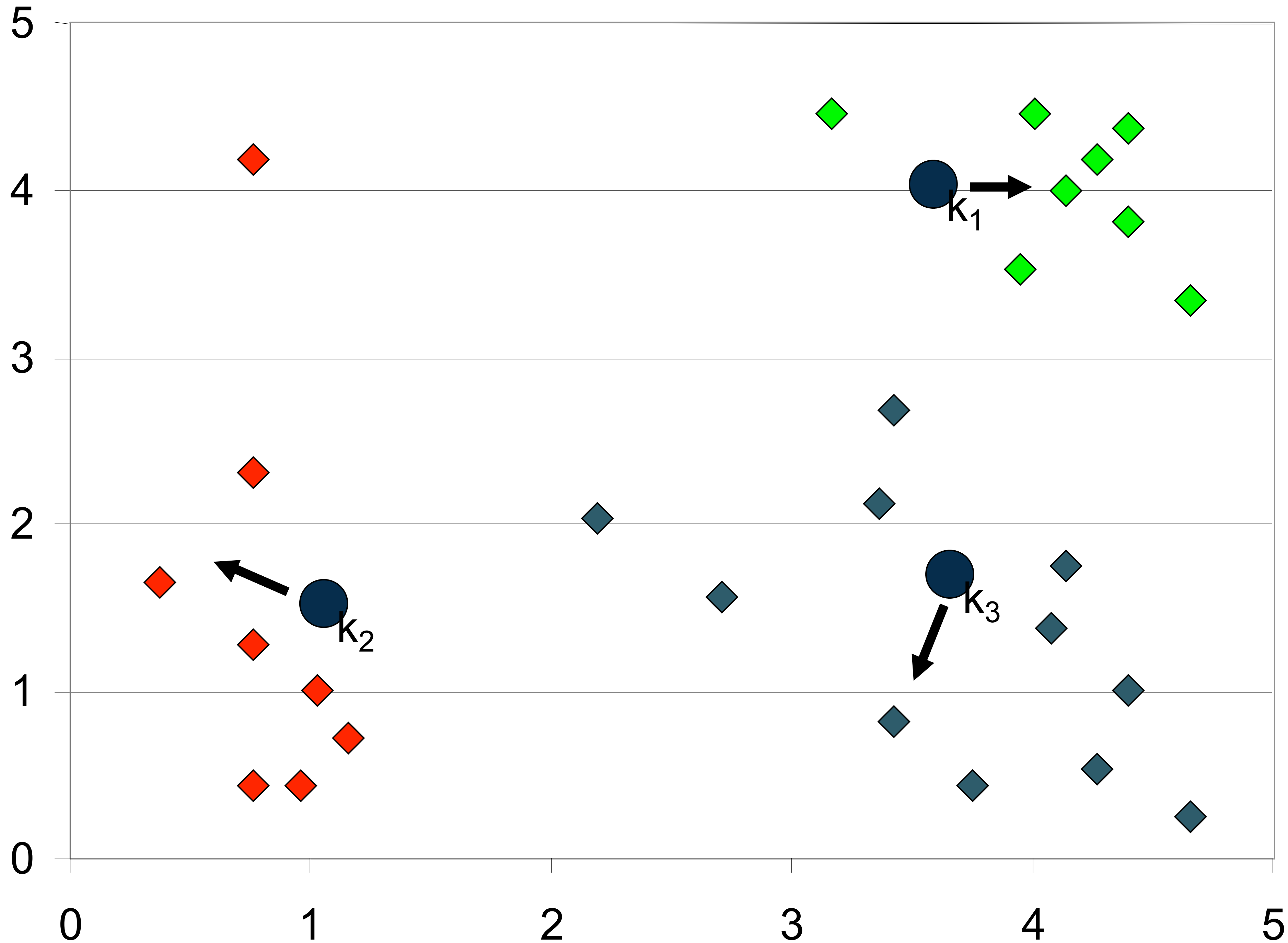
K-means - Distance Metric: Euclidean



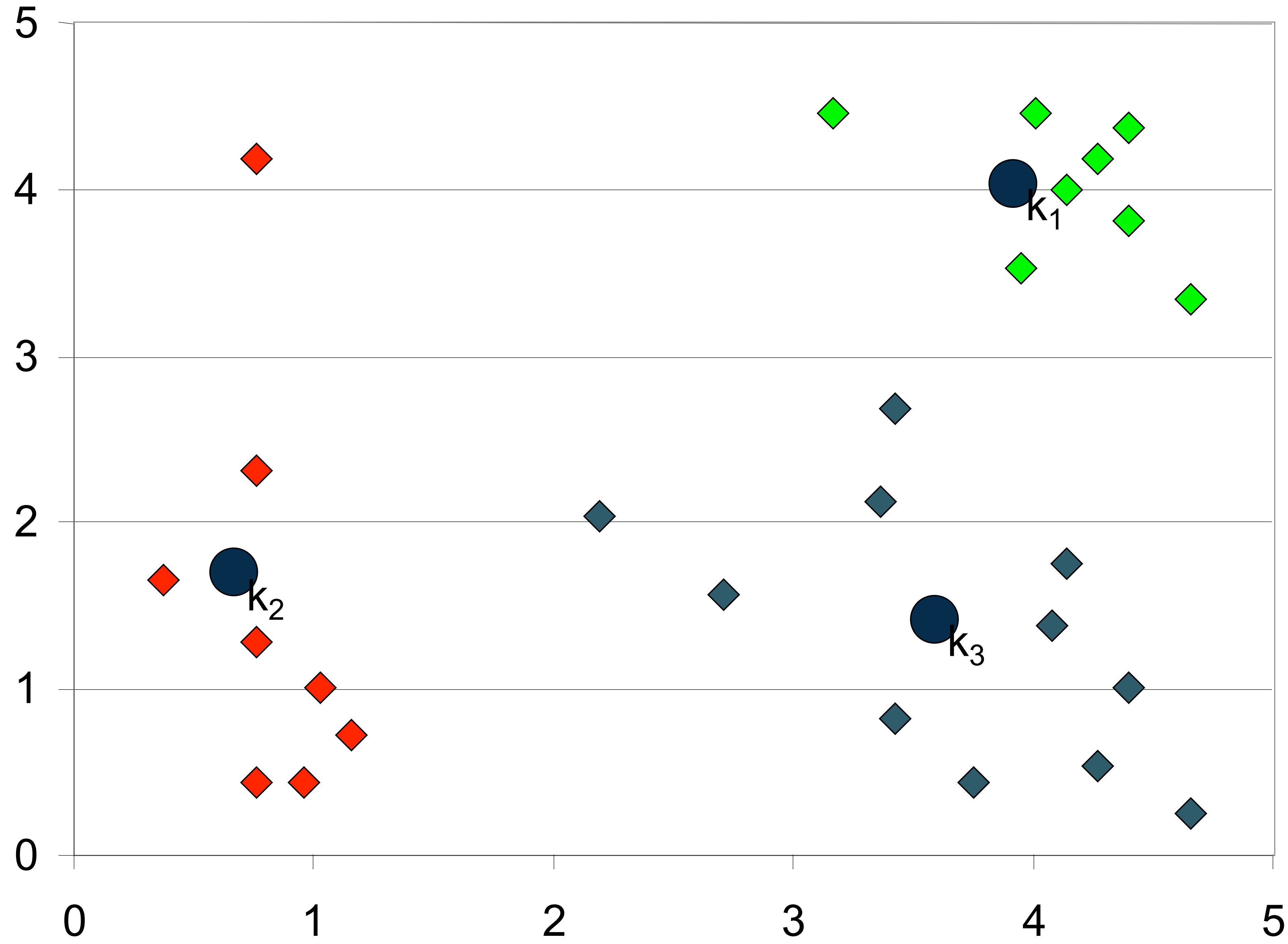
K-means - Distance Metric: Euclidean



K-means - Distance Metric: Euclidean



K-means - Distance Metric: Euclidean



Algorithm k-means

1. Decide on a value for *k*.

2. Initialise the *k* cluster centres (randomly, if necessary).

3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster centre.

4. Re-estimate the *k* cluster centres, by assuming the memberships found above are correct.

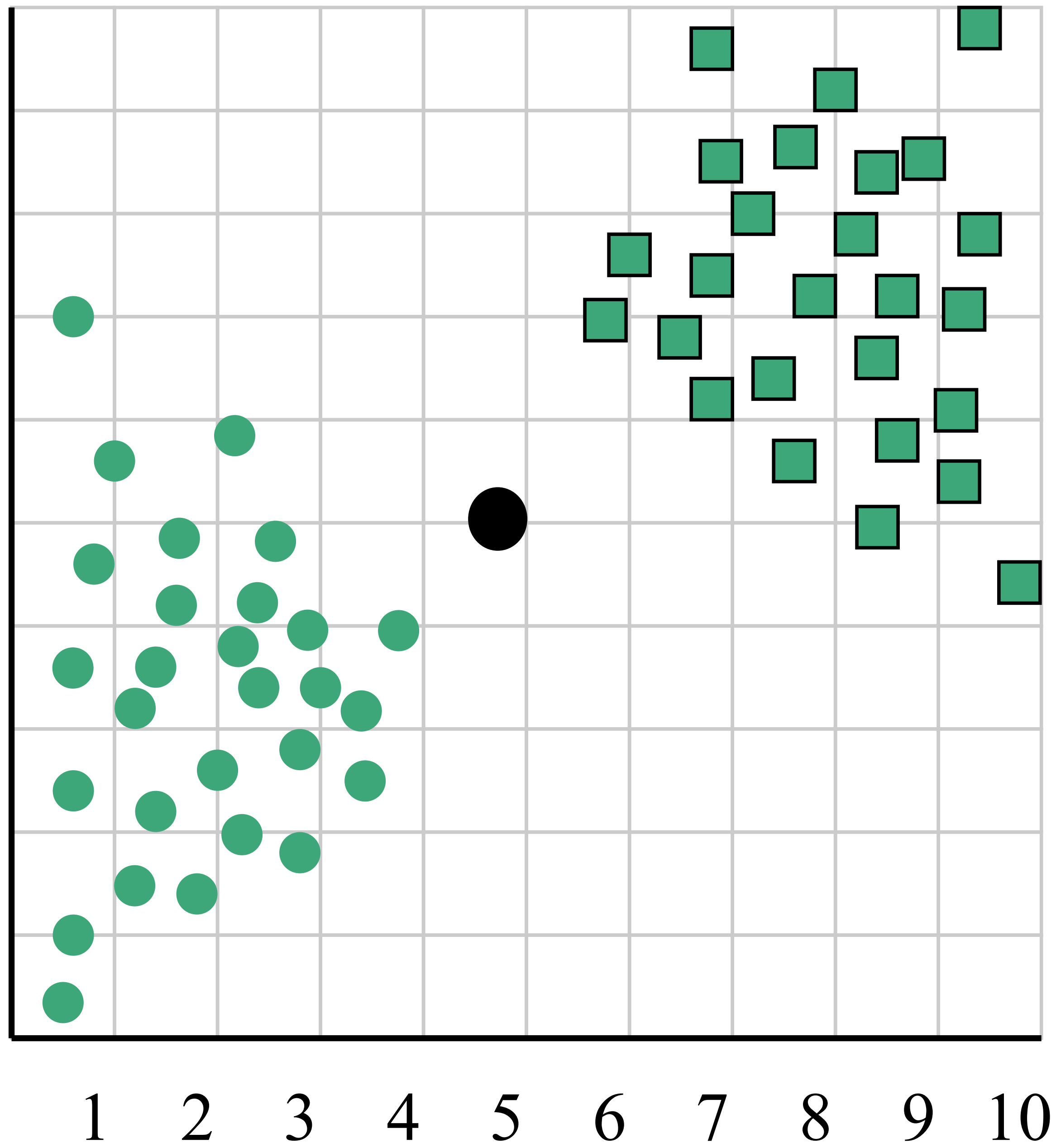
5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise goto 3.

Comments on the K-Means Method

- Strength
  - Relatively efficient: O(tkn), where n is # objects, k is # clusters, and t is # iterations. Normally, k, t << n.
  - Often terminates at a local optimum. The global optimum may be found using techniques such as: deterministic annealing and genetic algorithms
- Weakness
  - Need to specify k in advance
  - Unable to handle noisy data and outliers
  - Not suitable for clusters with non-convex shapes

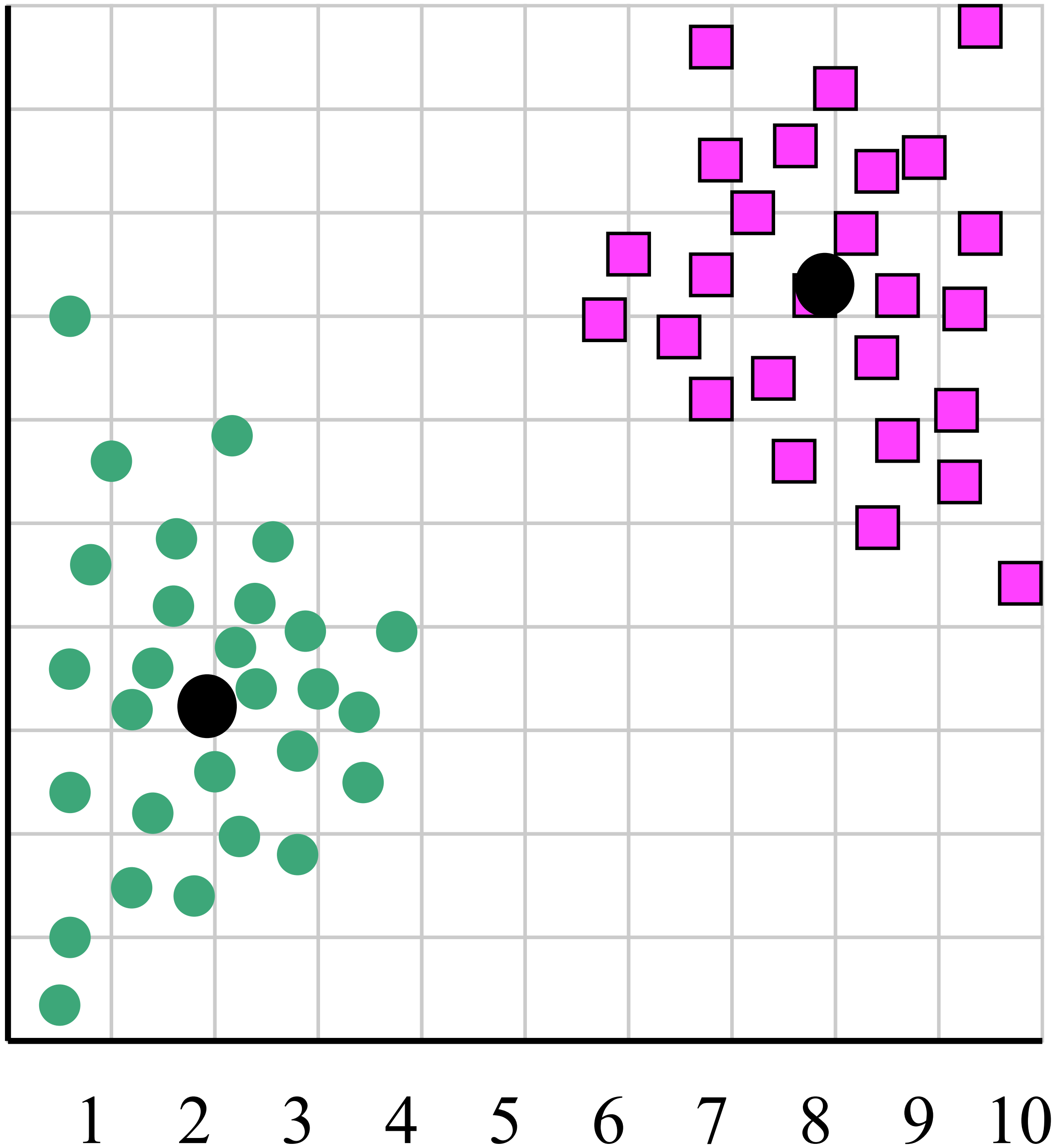
Objective function
"Need to specify k in advance"

When k = 1, the objective function is 873.0



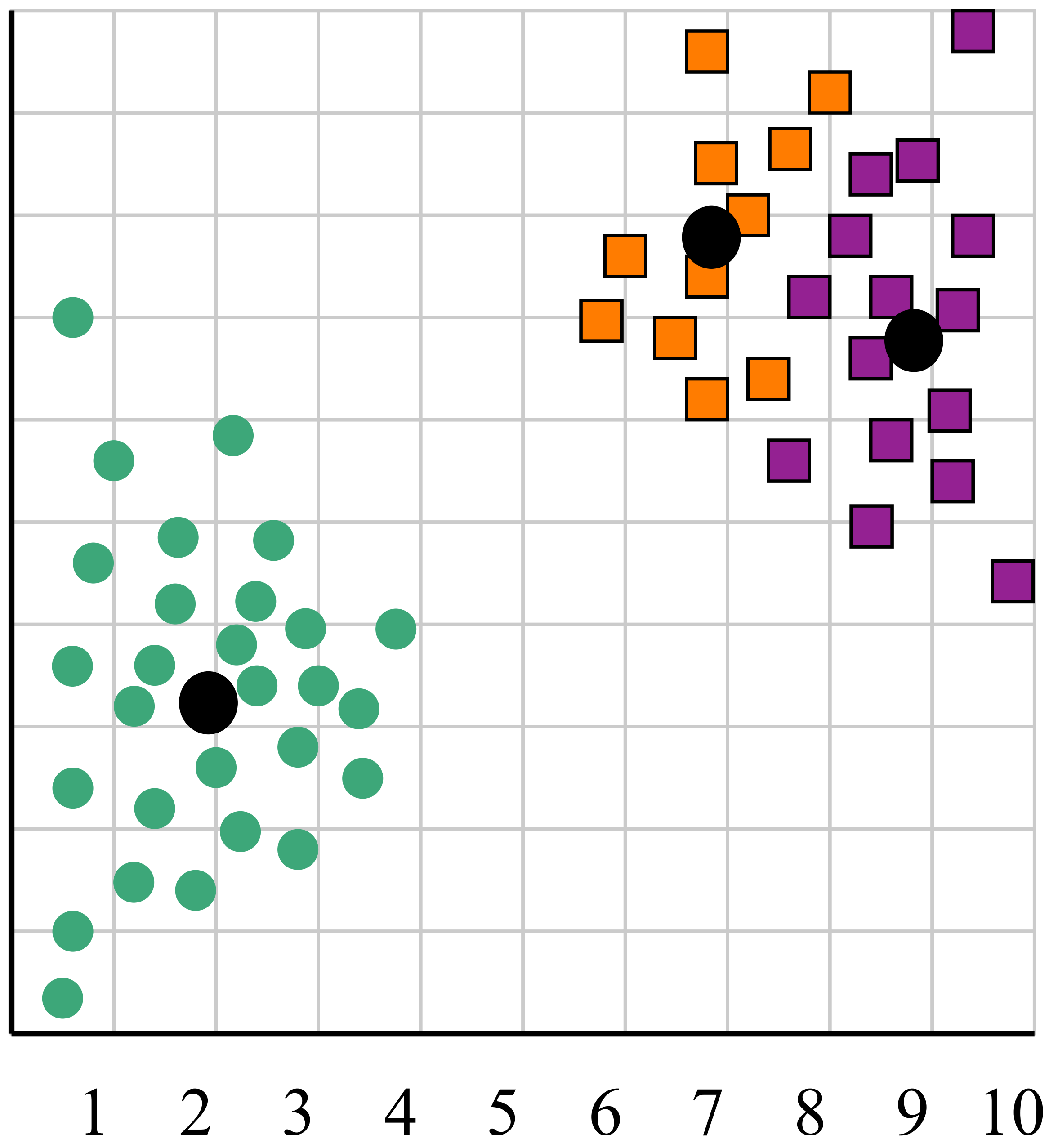
Objective function
"Need to specify k in advance"

When k = 2, the objective function is 173.1



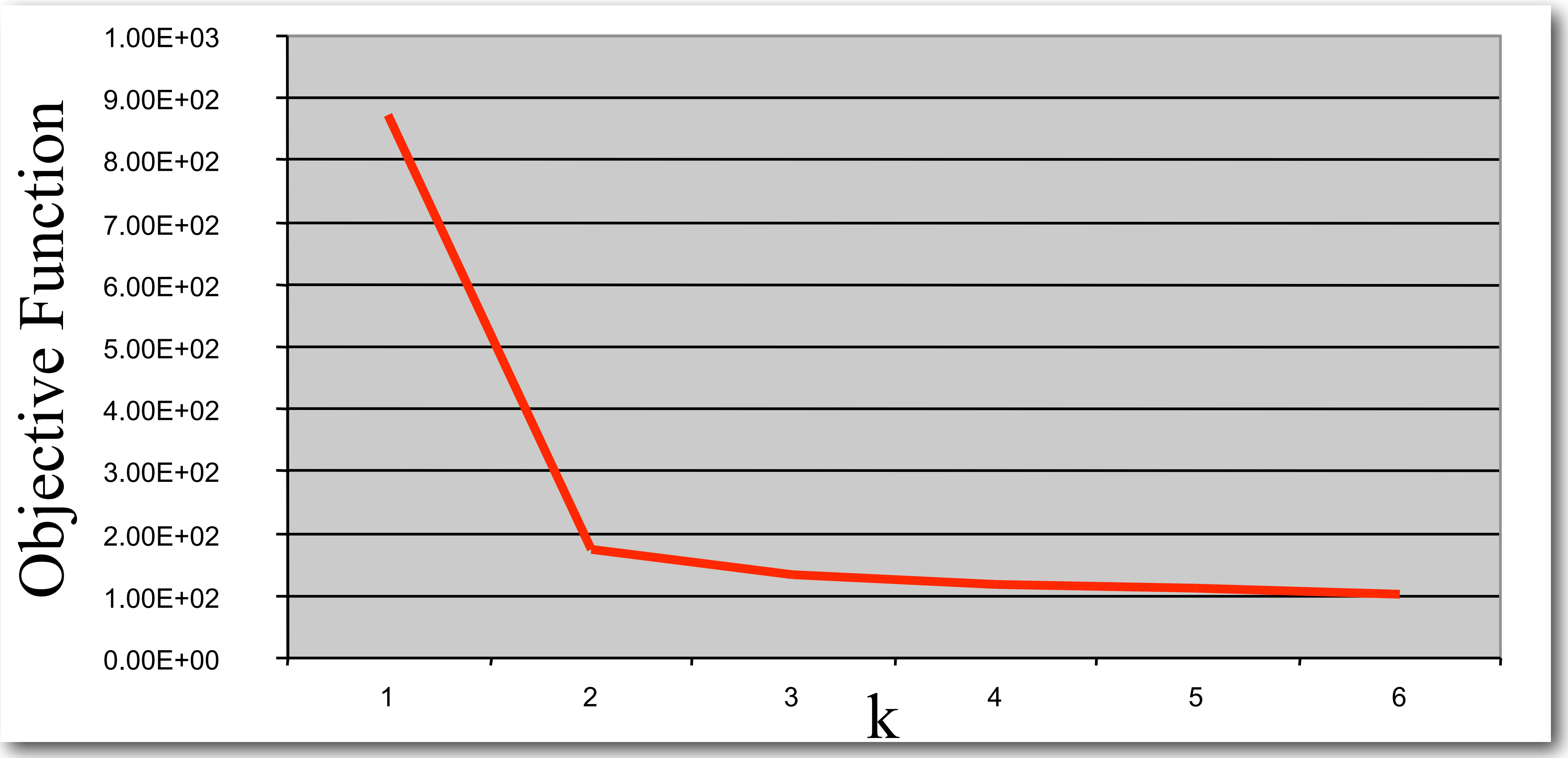
Objective function
"Need to specify k in advance"

When k = 3, the objective function is 133.6



Objective function
"Need to specify k in advance"

The abrupt change at k = 2, is highly suggestive of two clusters in the data. This technique for determining the number of clusters is colloquially known as "knee finding" or "elbow finding"



FUZZY C-MEANS

Fuzzy Set Theory Basics

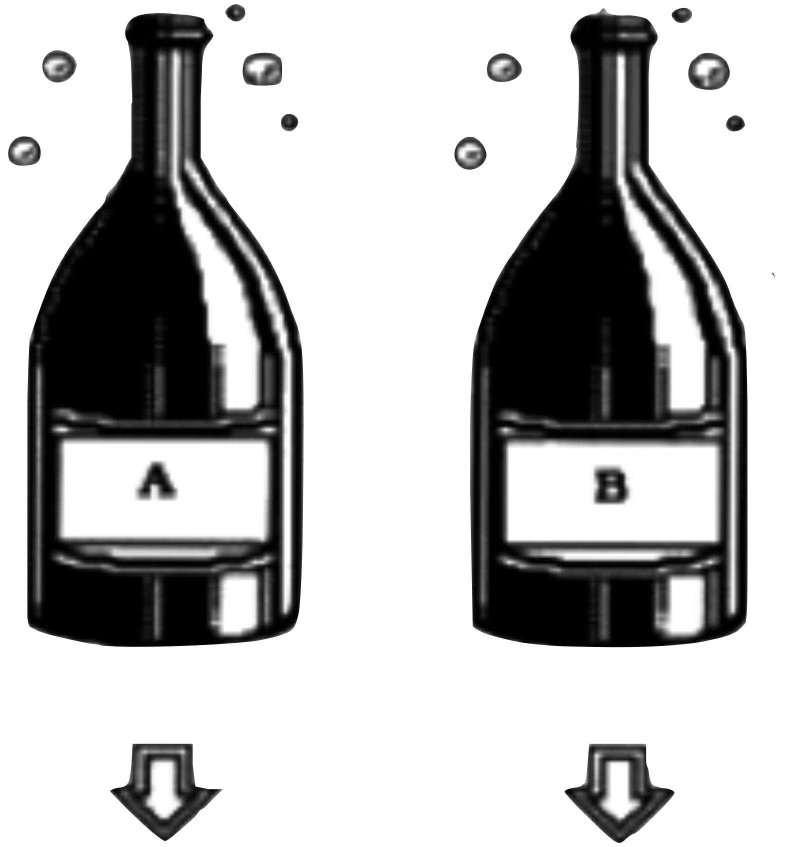
- Conventional set theory
  - Derives from symbolic, two-valued (T/F) logic
  - Depends upon binary decisions to determine set membership
- Fuzzy set theory
  - Permits continuous valued grading of set membership
  - Allows for reasoning subject to imprecision

Fuzzy Set Theory Basics

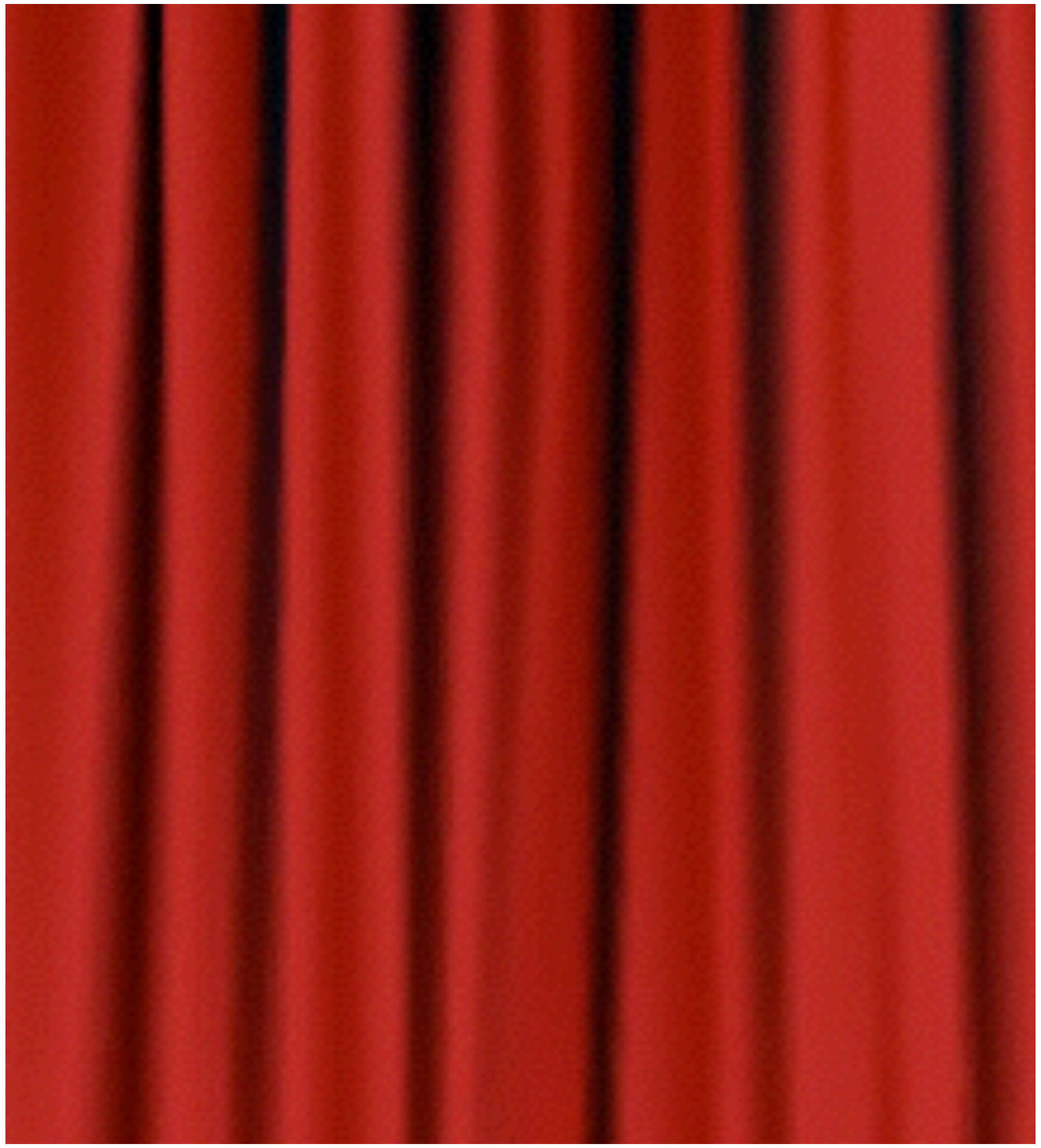
- Assume:
  - A domain of discourse X with entities generically denoted by x
- A fuzzy set A is a set of ordered pairs
  - $A = \{ (x, m_A(x)) \mid x \varepsilon X \}$
  - Where $m_A(x)$ is the fuzzy membership function for A and

$$0 \leq m_A(x) \leq 1$$

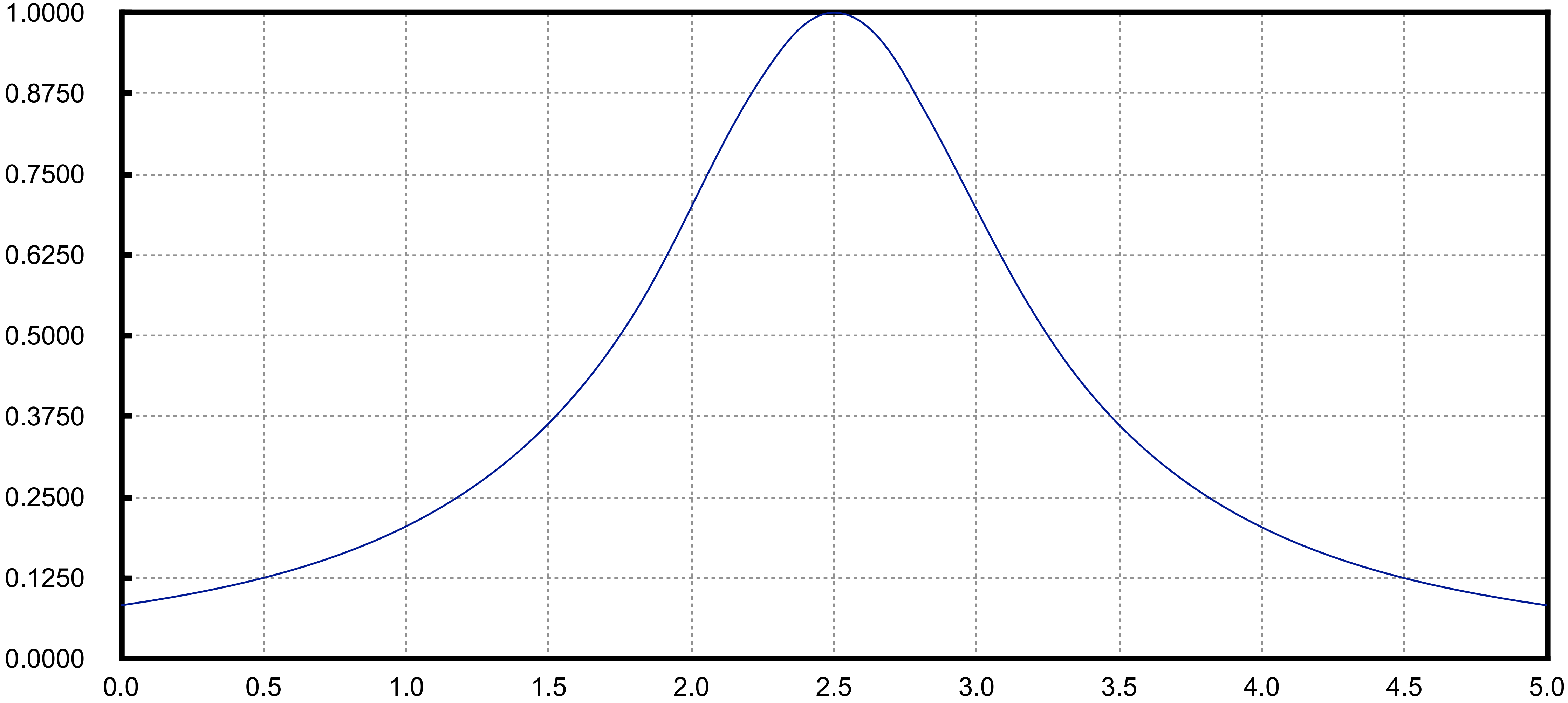
Fuzzy Set Theory Basics



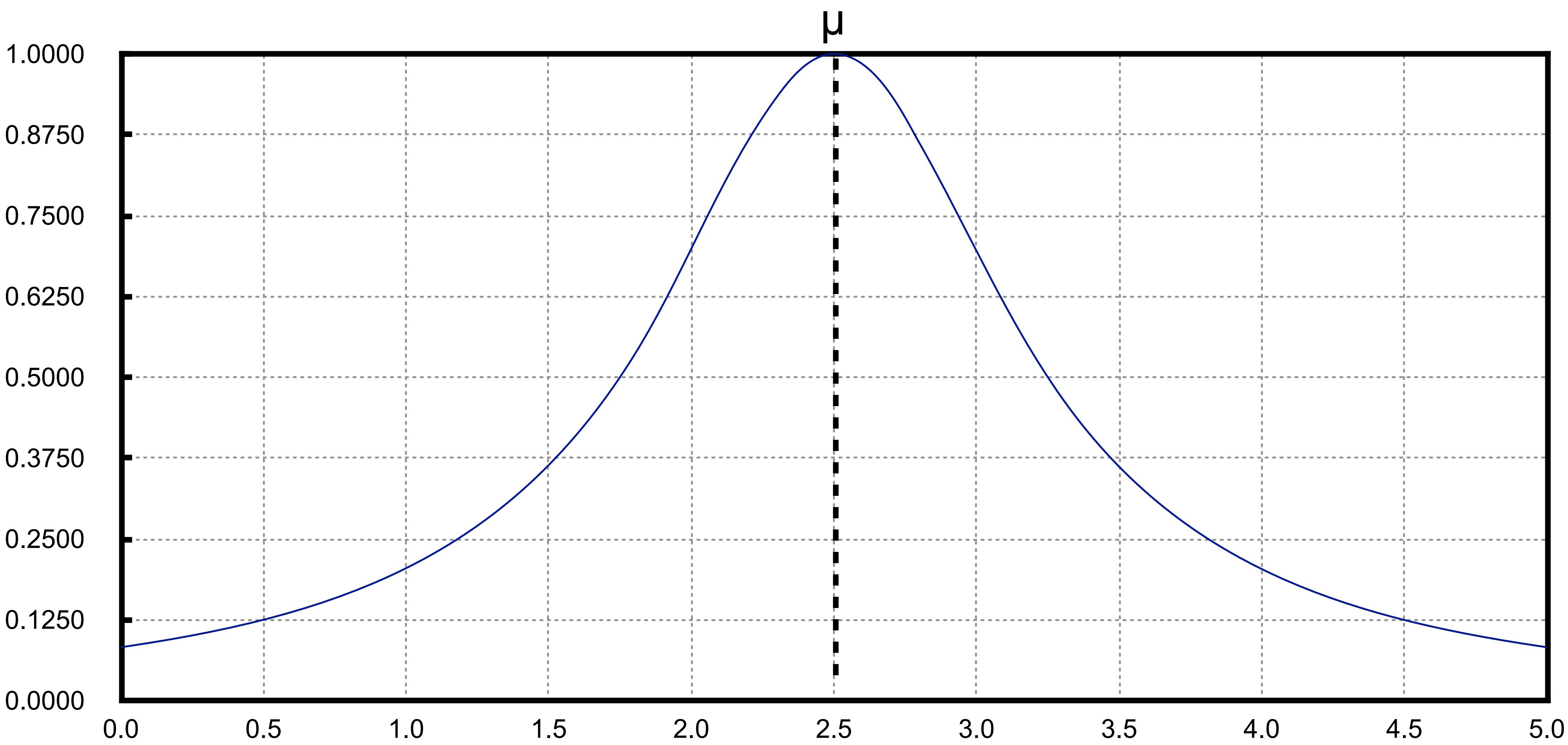
$memb(A \epsilon P)=0.9$ $prob(A \epsilon P)=0.9$



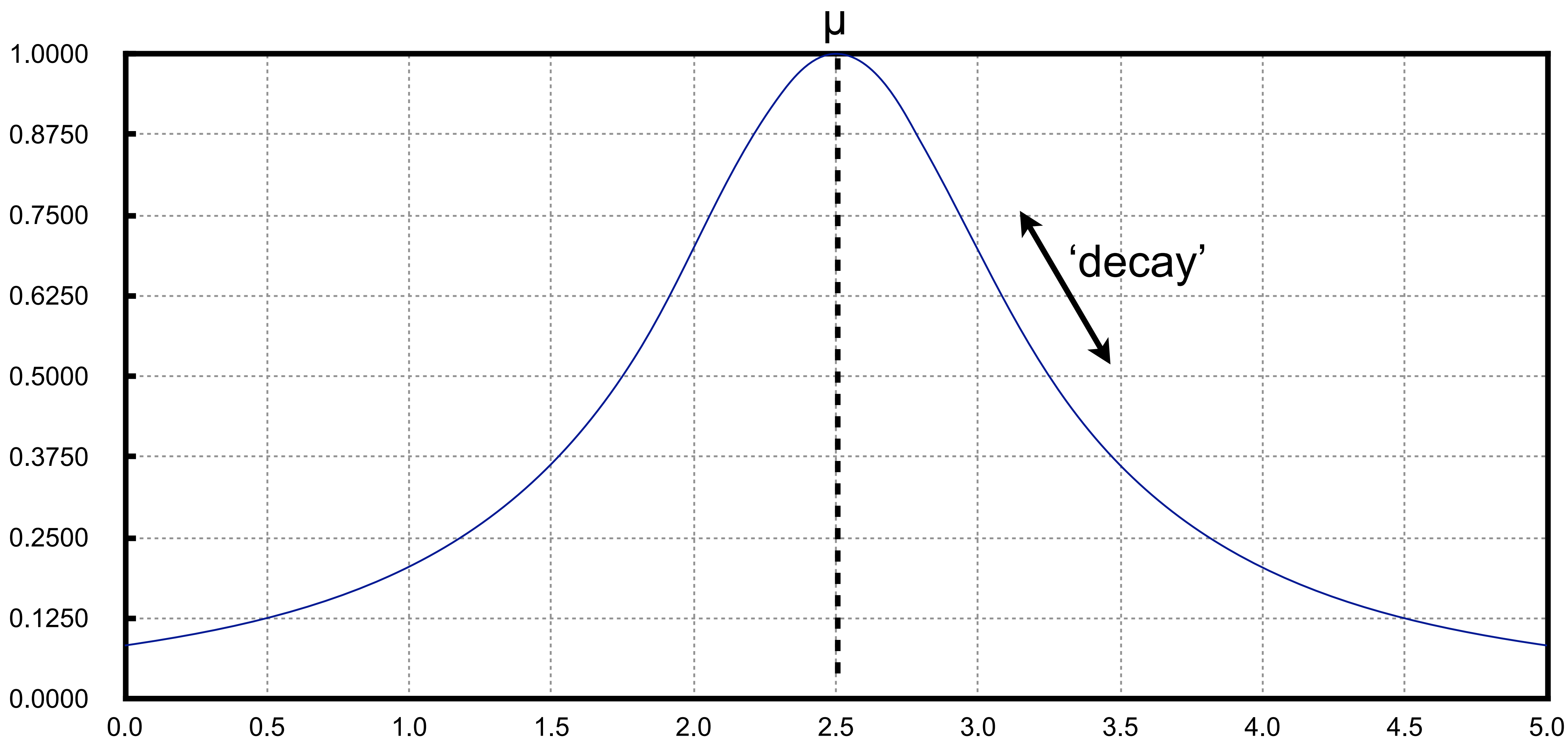
Fuzzy C-Means (FCM) Algorithm



Fuzzy C-Means (FCM) Algorithm



Fuzzy C-Means (FCM) Algorithm



Fuzzy C-Means (FCM) Algorithm

$$cost(U, \mu_1, \mu_2, ..., \mu_n) = \sum_{r=1}^{k} \sum_{c} u(r,c)^m d(r,c)^2$$

where

- $0 \leq u(r,c) \leq 1$ and $\sum_{r=1}^{k} u(r,c) = 1$
- μ_r is the cluster centre of group r
- $d(e,r)=|| \mu_r - x_c ||$ is the Euclidean distance

  from the point x_c *to* cluster centre μ_r
- $m \in (1,\infty)$ is a weighting exponent

$$\mu_r = \frac{\sum_{c=1}^{n} u(r,c)^m x_c}{\sum_{c=1}^{n} u(r,c)^m}$$

$$u(r,c) = \frac{1}{\sum_{j=1}^{k} \left(\frac{d(r,c)}{d(j,c)}\right)^{\frac{2}{m-1}}}$$

Fuzzy C-Means (FCM) Algorithm

1. Initialise the cluster assignment matrix

2. Calculate k fuzzy cluster centres

$$\mu_r = \frac{\sum_{c=1}^{n} u(r,c)^m x_c}{\sum_{c=1}^{n} u(r,c)^m}$$

3. Compute the cost function using

$$cost(U, \mu_1, \mu_2, ..., \mu_n) = \sum_{r=1}^{k} \sum_{c} u(r,c)^m d(r,c)^2$$

4. Compute a new cluster assignment matrix

$$u(r,c) = \frac{1}{\sum_{j=1}^{k} \left(\frac{d(r,c)}{d(j,c)}\right)^{\frac{2}{m-1}}}$$

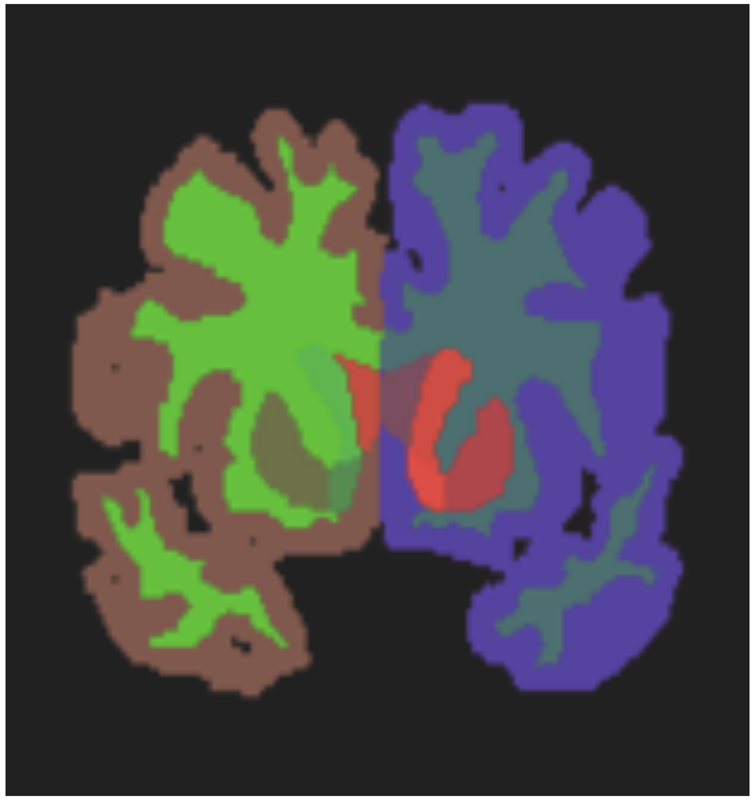
5. While (cost is decreasing “significantly”)

EM BASED SEGMENTATION

EXPECTATION - MAXIMISATION

Probabilistic Model of how the data is generated

Lable (L)



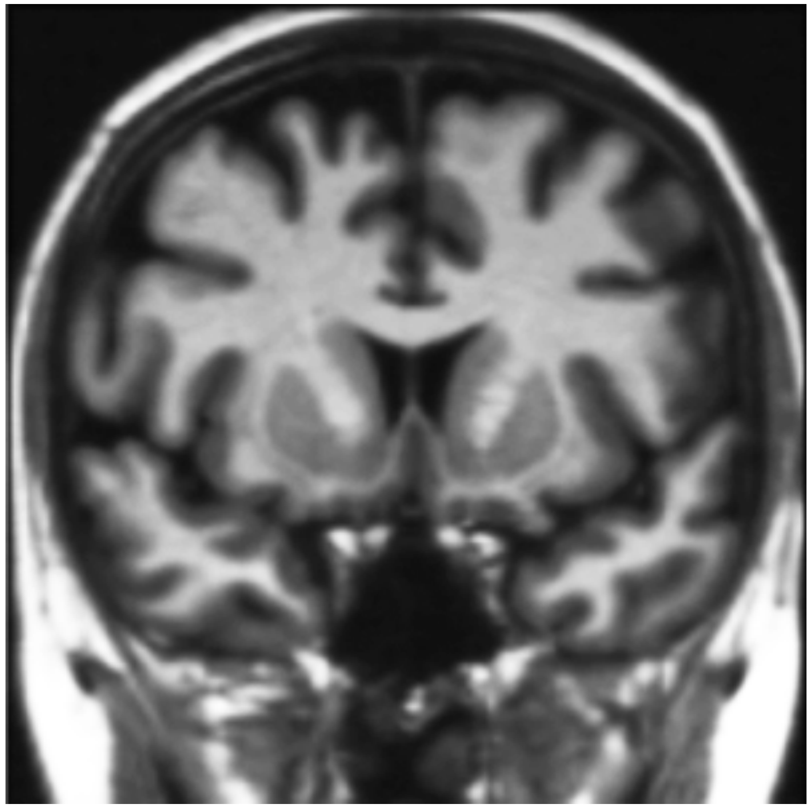
$$P(Y|Z, \Phi_y)$$

$$\longrightarrow$$

$$\longleftarrow$$

$$P(Z|Y; \Phi_y)$$

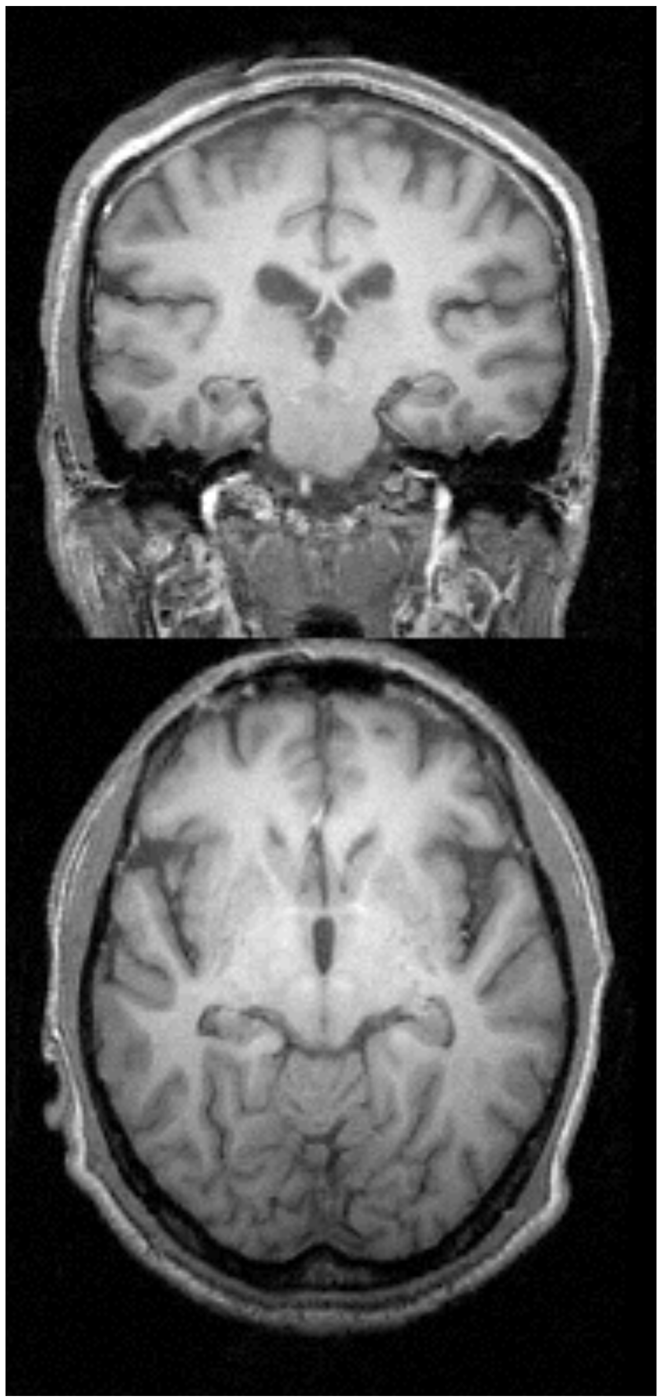
MRI image (Y)

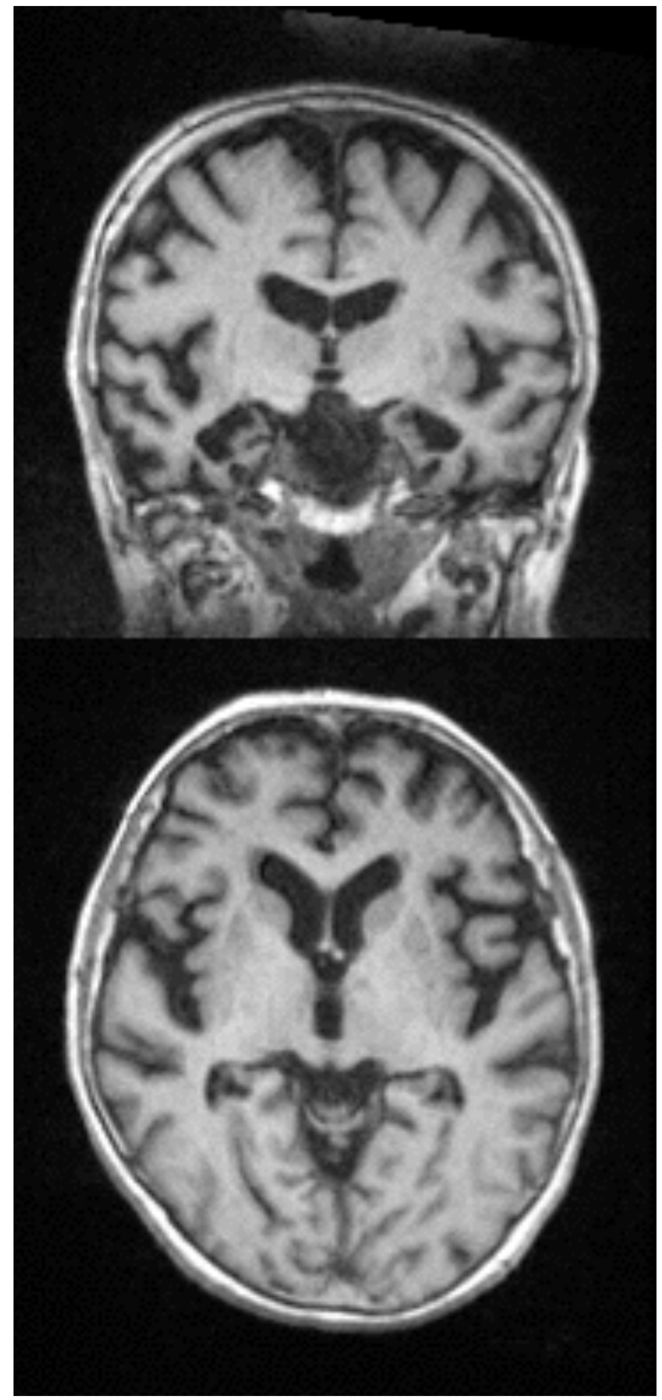


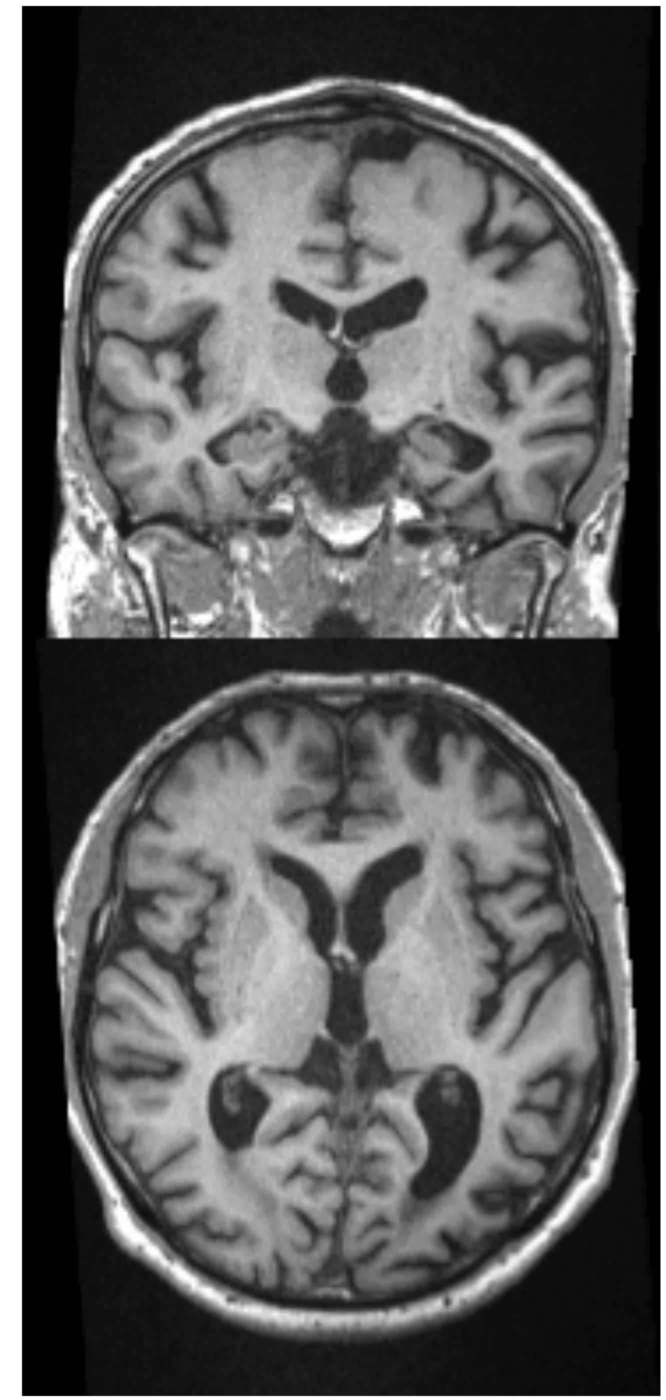
How is the data generated?

- If we exclude the brain, there are 3 tissue classes
  - WM, GM, CSF
- Each tissue has a characteristic intensity
  - Intensity inhomogeneity free image
  - Simplification of human anatomy
- The image acquisition process has an inherent level of noise
  - In theory: Rician noise
  - Normally modelled as Gaussian

Image data: MRI

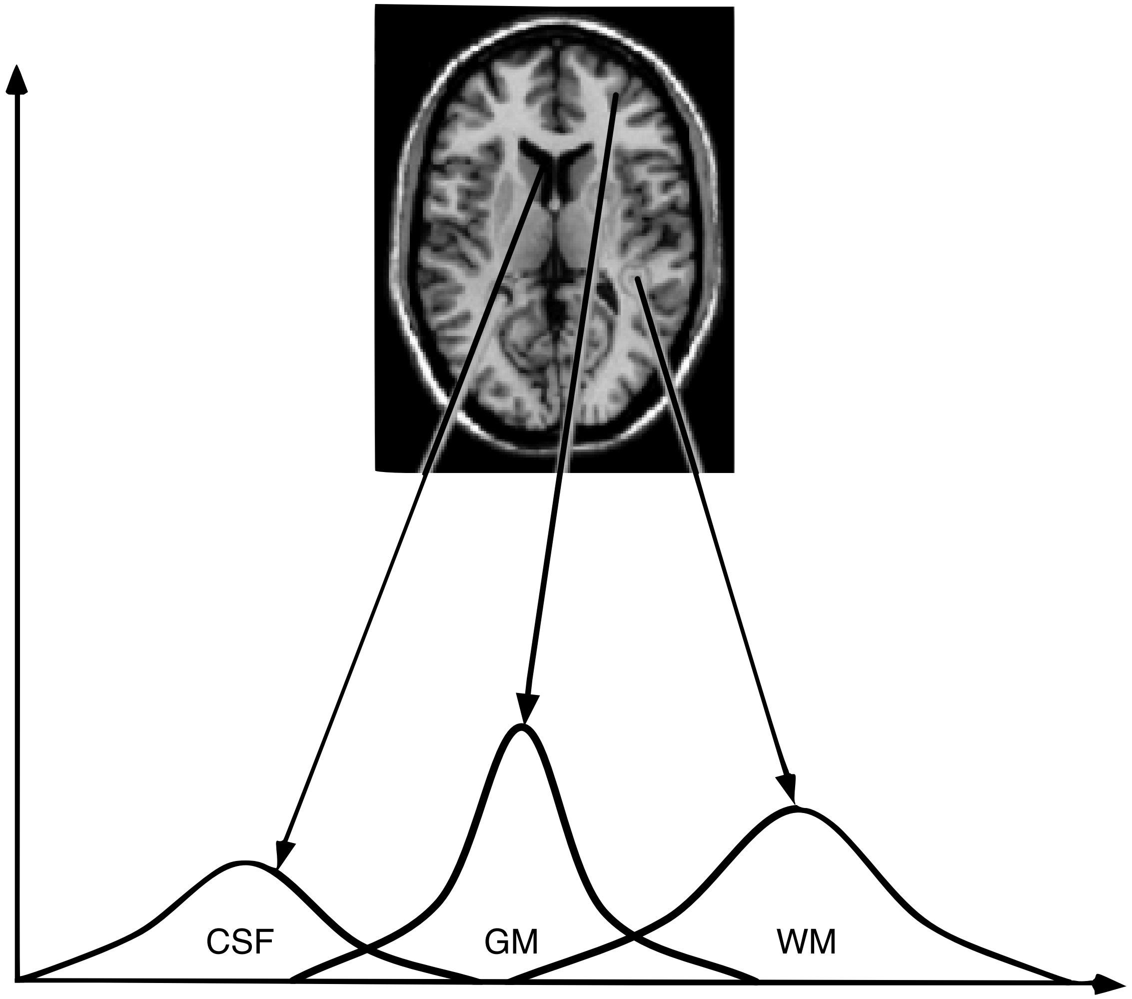

NC


MCI


AD

Mixture Models

Let $y_1, y_2, \ldots, y_n$, denote a random sample of size n, where y_j, is a p-dimensional random vector with probability density function $f(y_j)$ on $\mathbb{R}^p$.



$$f(y_i \mid \Phi_y) = \sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)$$

where

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\left(y_i - \mu_k\right)$$

with $G_{\sigma_k}(\ )$ denoting a zero-mean normal distribution with variance σ_k^2

Expectation Maximisation (EM)

- Finds the maximum likelihood estimates of parameters in a probabilistic setting
- Alternates between performing an expectation (E) step and a maximisation (M) step
- Used for data clustering:
  - image segmentation,
  - machine learning and
  - computer vision or for example in medical image reconstruction

Expectation Maximisation (EM)

By considering statistical independence between voxels, the overall probability density for the full image can be given by

$$f(y \mid \Phi_y) = \prod_i f(y_i \mid \Phi_y)$$

The EM algorithm assumes both the segmentation and the model parameters can be estimated simultaneously by interleaving the segmentation with estimation of the model parameters. This is a maximum likelihood (ML) problem, where

$$\hat{\Phi} = arg\ max_{\Phi}\ log\ f(y|\Phi)$$

Expectation Maximisation (EM)

- Why is it called EM?
  - Because it is a 2 step process

E Step: Find the function

$$Q(\Phi|\Phi^{(m)}) = E[\log f(q|\Phi)|y, \Phi^{(m)}]. \qquad\qquad q = (y, z)$$

M Step: Find

$$\Phi^{(m+1)} = \arg\max_{\Phi} Q(\Phi|\Phi^{(m)})$$

Maximisation Step

- Consequently, the ML estimates of the average and std will be given by the explicit maximisation of Q

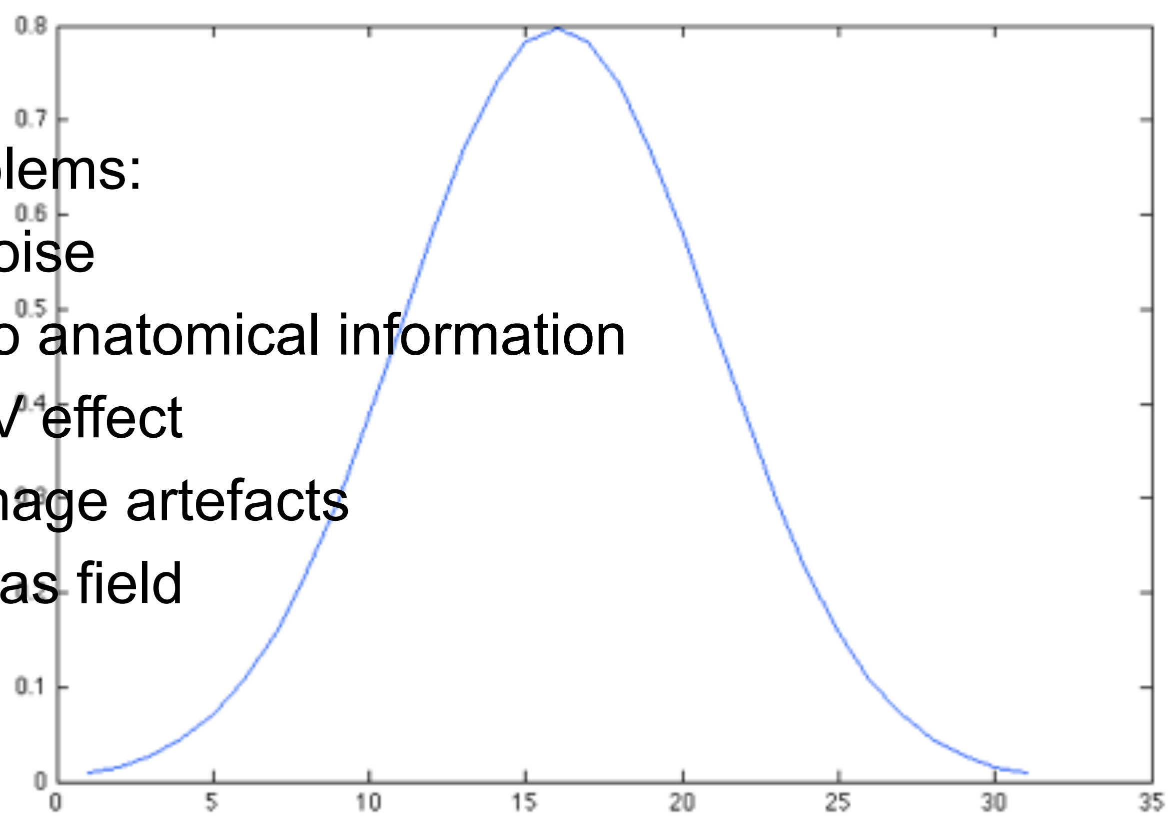
$$\frac{\partial}{\partial \mu_k}\left(E[\log f(q|\Phi)|y,\Phi^{(m)}]\right)=0$$

$$\frac{\partial}{\partial \sigma_k^2}\left(E[\log f(q|\Phi)|y,\Phi^{(m)}]\right)=0$$

Expectation Step

- Gaussian PDF

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$



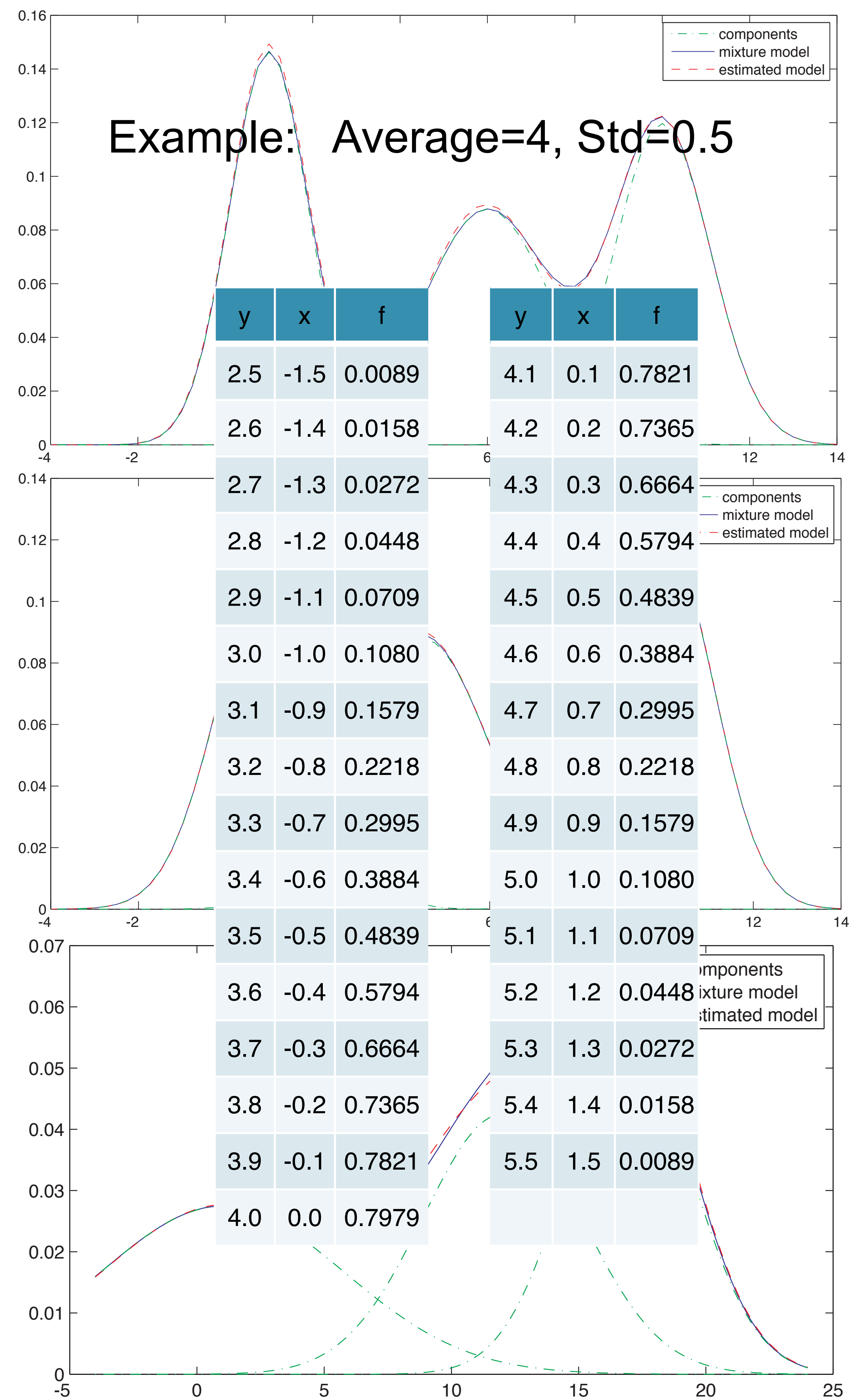
Problems:

- Noise
- No anatomical information
- PV effect
- Image artefacts
- Bias field

Example: Average=4, Std=0.5

y	x	f
2.5	-1.5	0.0089
2.6	-1.4	0.0158
2.7	-1.3	0.0272
2.8	-1.2	0.0448
2.9	-1.1	0.0709
3.0	-1.0	0.1080
3.1	-0.9	0.1579
3.2	-0.8	0.2218
3.3	-0.7	0.2995
3.4	-0.6	0.3884
3.5	-0.5	0.4839
3.6	-0.4	0.5794
3.7	-0.3	0.6664
3.8	-0.2	0.7365
3.9	-0.1	0.7821
4.0	0.0	0.7979

y	x	f
4.1	0.1	0.7821
4.2	0.2	0.7365
4.3	0.3	0.6664
4.4	0.4	0.5794
4.5	0.5	0.4839
4.6	0.6	0.3884
4.7	0.7	0.2995
4.8	0.8	0.2218
4.9	0.9	0.1579
5.0	1.0	0.1080
5.1	1.1	0.0709
5.2	1.2	0.0448
5.3	1.3	0.0272
5.4	1.4	0.0158
5.5	1.5	0.0089



Maximisation Step

- Instead, one can calculate a new and improved average and covariance matrix

- Using Bayes' theorem and differentiating Q, the ML estimates of the model parameters are given by the following update equations

$$p_{ik}^{(m+1)} = \frac{f\left(y_i \mid z_i = e_k, \Phi_y^{(m)}\right) f\left(z_i = e_k\right)}{\sum_{j=1}^{K} f\left(y_i \mid z_i = e_j, \Phi_y^{(m)}\right) f\left(z_i = e_j\right)}$$

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} y_i}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

$$\left(\sigma_k^{(m+1)}\right)^2 = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} \left(y_i - \mu_k^{(m+1)}\right)^2}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

Convergence Criteria

$$\frac{\partial \log L(\Phi)}{\partial \Phi} = 0$$

- The final p_{ik} value contains information about the probability of each voxel for each class.

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

Step by step:

Calculate *p* *and log L*

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

$$p_{ik}^{(m+1)} = \frac{f\Big(y_i \mid z_i = e_k, \Phi_y^{(m)}\Big)\, f\Big(z_i = e_k\Big)}{\sum_{j=1}^{K} f\Big(y_i \mid z_i = e_j, \Phi_y^{(m)}\Big)\, f\Big(z_i = e_j\Big)}$$

For all y and for all classes

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\Big(y_i - \mu_k\Big)$$

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

Update parameters

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} y_i}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

$$\Big(\sigma_k^{(m+1)}\Big)^2 = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} \Big(y_i - \mu_k^{(m+1)}\Big)^2}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

Compare current log L with old log L

Converged?

No

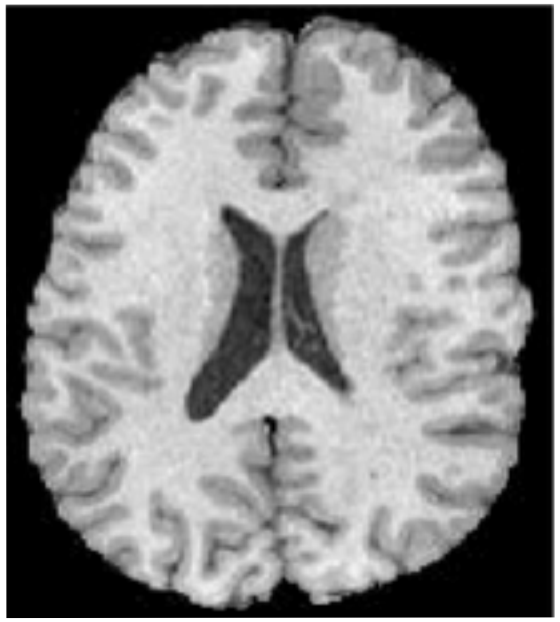
Yes

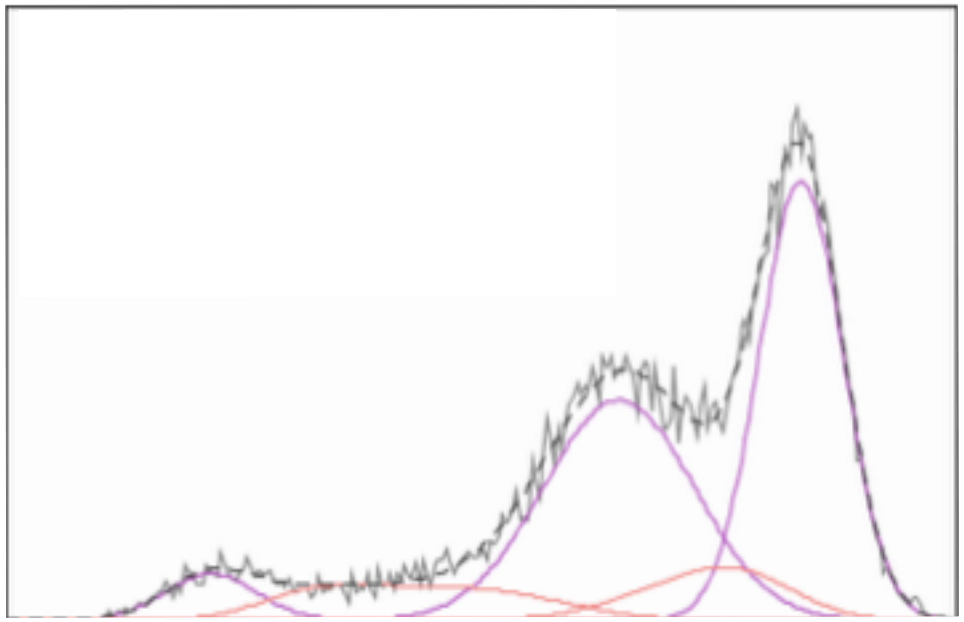
Finish

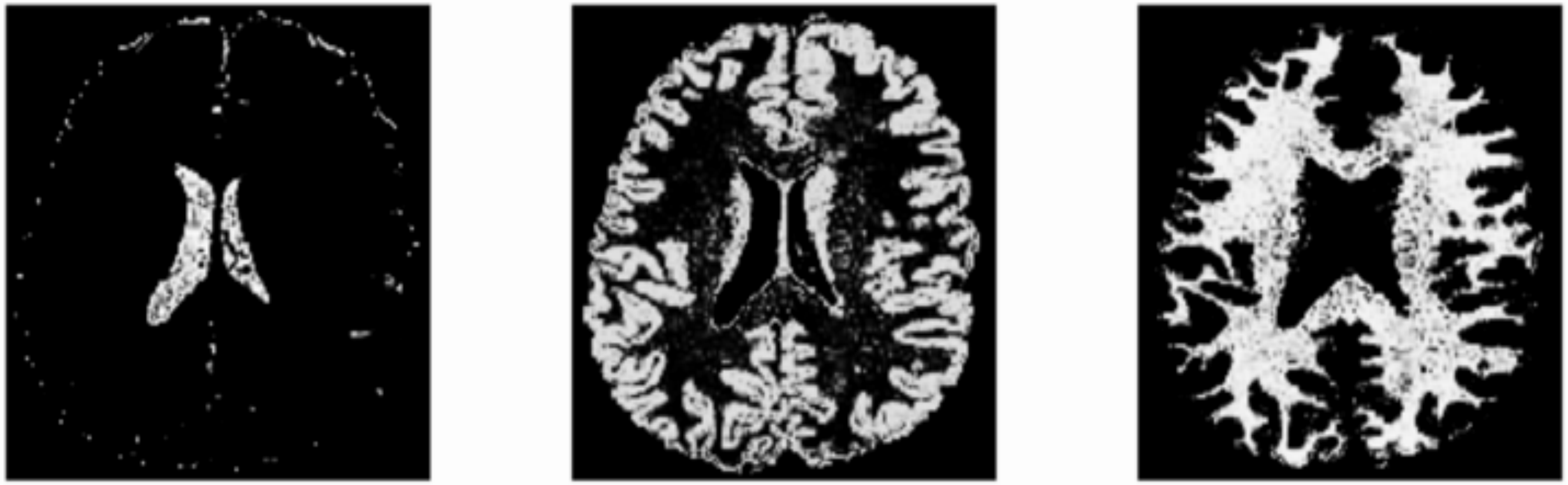
However...

- Noise
- Anatomical Priors
- Bias Field
- Multi-modal segmentation

Noise



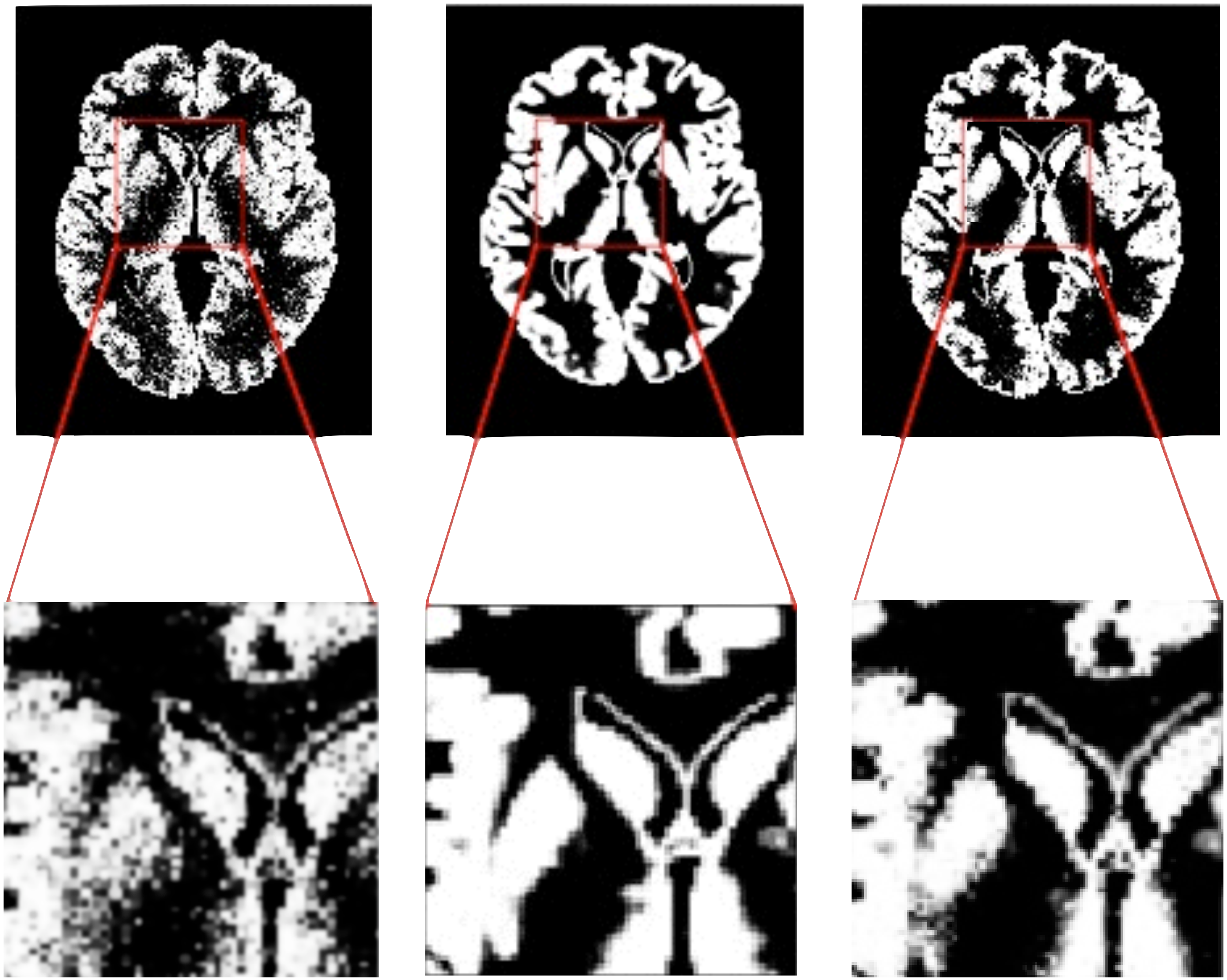




MRF

Full Brain
- GM -

Zoomed



9% noise, no MRF | 0% noise, no MRF | 9% noise, with MRF

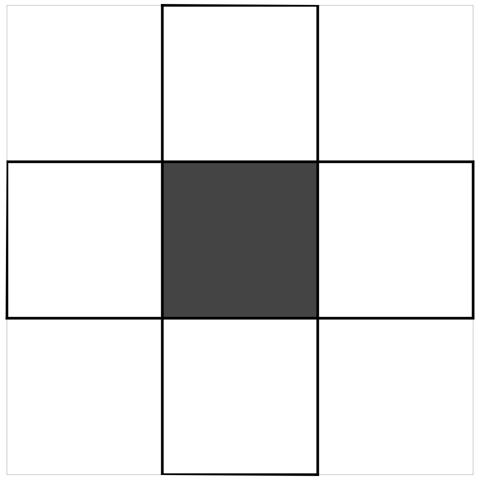
How to “solve” it? Markov Random Field (MRF)

- MRF uses an extension of the Potts model to *k* classes and 3 dimensions on the 6 nearest neighbours (N,S,E,W,T,B). Acts as de-noising and at the same time, statistically forces the natural layering of the brain

WM → WM/GM$_{pv}$ → GM → GM/CSF$_{pv}$ → CSF

$$f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{\pi_{ik}\, e^{-\beta U_{\mathrm{MRF}}(e_k | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^{K} \pi_{ij}\, e^{-\beta U_{\mathrm{MRF}}(e_j | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}$$

$$U_{MRF}(e_k | p_{N\,i}, \phi_z) = \sum_{j=1}^{K} \Big(\sum_{i=N_i^G} p_{ij} G_{kj} \Big)$$

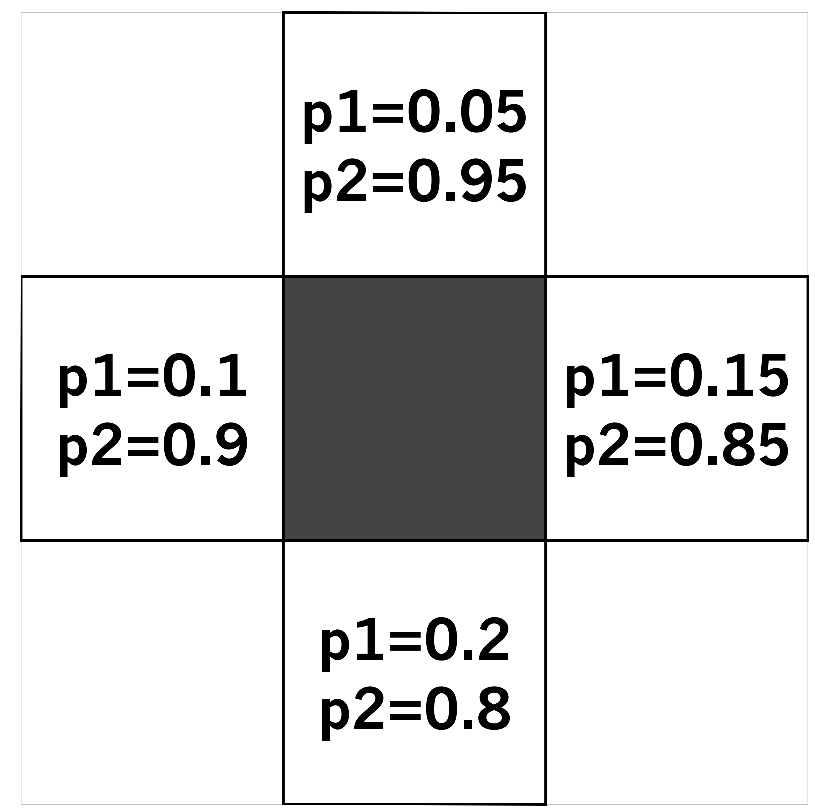


How does the MRF works

- For k=1
  - For j=1
    - U_{MRF}=(0.05+0.15+0.2+0.1)*0=0
  - For j=2
    - U_{MRF}=(0.95+0.85+0.8+0.9)*1=3.5
  - Total U_{MRF}=3.5
- For k=2
  - For j=1
    - U_{MRF}=(0.05+0.15+0.2+0.1)*1=0.5
  - For j=2
    - U_{MRF}=(0.95+0.85+0.8+0.9)*0=0
  - Total U_{MRF}=0.5

$$G=\begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix}$$

$$U_{MRF}(e_k|p_{Ni},\phi_z)=\sum_{j=1}^{K}\Big(\sum_{i=N_i^G} p_{ij}G_{kj}\Big)$$



How does the MRF works

- For k=1
  - Total U_{MRF}=3.5
- For k=2
  - Total U_{MRF}=0.5

$$f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{\pi_{ik}\, e^{-\beta U_{\mathrm{MRF}}(e_k | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^{K} \pi_{ij}\, e^{-\beta U_{\mathrm{MRF}}(e_j | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}$$

	p1=0.05 **p2=0.95**	
p1=0.1 **p2=0.9**		**p1=0.15** **p2=0.85**
	p1=0.2 **p2=0.8**	

- If we consider π_{ik}=1 and β=1
- For k=1
  - $f=e^{-3.5}/(e^{-3.5}+e^{-0.5})=0.0474$
- For k=2
  - $f=e^{-0.5}/(e^{-3.5}+e^{-0.5})=0.9526$

How does the MRF works

- For k=1
  - $f=e^{-3.5}/(e^{-3.5}+e^{-0.5})=0.0474$
- For k=2
  - $f=e^{-0.5}/(e^{-3.5}+e^{-0.5})=0.9526$

$$f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{\pi_{ik}\, e^{-\beta U_{\mathrm{MRF}}(e_k | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^{K} \pi_{ij}\, e^{-\beta U_{\mathrm{MRF}}(e_j | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}$$

$$p_{ik}^{(m+1)} = \frac{f\left(y_i \mid z_i = e_k, \Phi_y^{(m)}\right) f\left(z_i = e_k\right)}{\sum_{j=1}^{K} f\left(y_i \mid z_i = e_j, \Phi_y^{(m)}\right) f\left(z_i = e_j\right)}$$

Step by step:

Calculate *p* *and log L*

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

$$p_{ik}^{(m+1)} = \frac{f\left(y_i \mid z_i = e_k, \Phi_y^{(m)}\right) f\left(z_i = e_k\right)}{\sum_{j=1}^{K} f\left(y_i \mid z_i = e_j, \Phi_y^{(m)}\right) f\left(z_i = e_j\right)}$$

For all y and for all classes

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\left(y_i - \mu_k\right)$$

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

Update parameters

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} y_i}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

$$\left(\sigma_k^{(m+1)}\right)^2 = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} \left(y_i - \mu_k^{(m+1)}\right)^2}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

Compare current log L with old log L

Converged?

No

Yes

Finish

Step by step:

For all y and for all classes

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\Big(y_i - \mu_k\Big)$$

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

Calculate *p* *and log L*

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

$$p_{ik}^{(m+1)} = \frac{f\Big(y_i \mid z_i = e_k, \Phi_y^{(m)}\Big)\, f\Big(z_i = e_k\Big)}{\sum_{j=1}^{K} f\Big(y_i \mid z_i = e_j, \Phi_y^{(m)}\Big)\, f\Big(z_i = e_j\Big)}$$

Update parameters

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} y_i}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

$$f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{e^{-\beta U_{\mathrm{MRF}}(e_k \mid p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^{K} e^{-\beta U_{\mathrm{MRF}}(e_j \mid p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}$$

$$\left(\sigma_k^{(m+1)}\right)^2 = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} \Big(y_i - \mu_k^{(m+1)}\Big)^2}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

Compare current log L with old log L

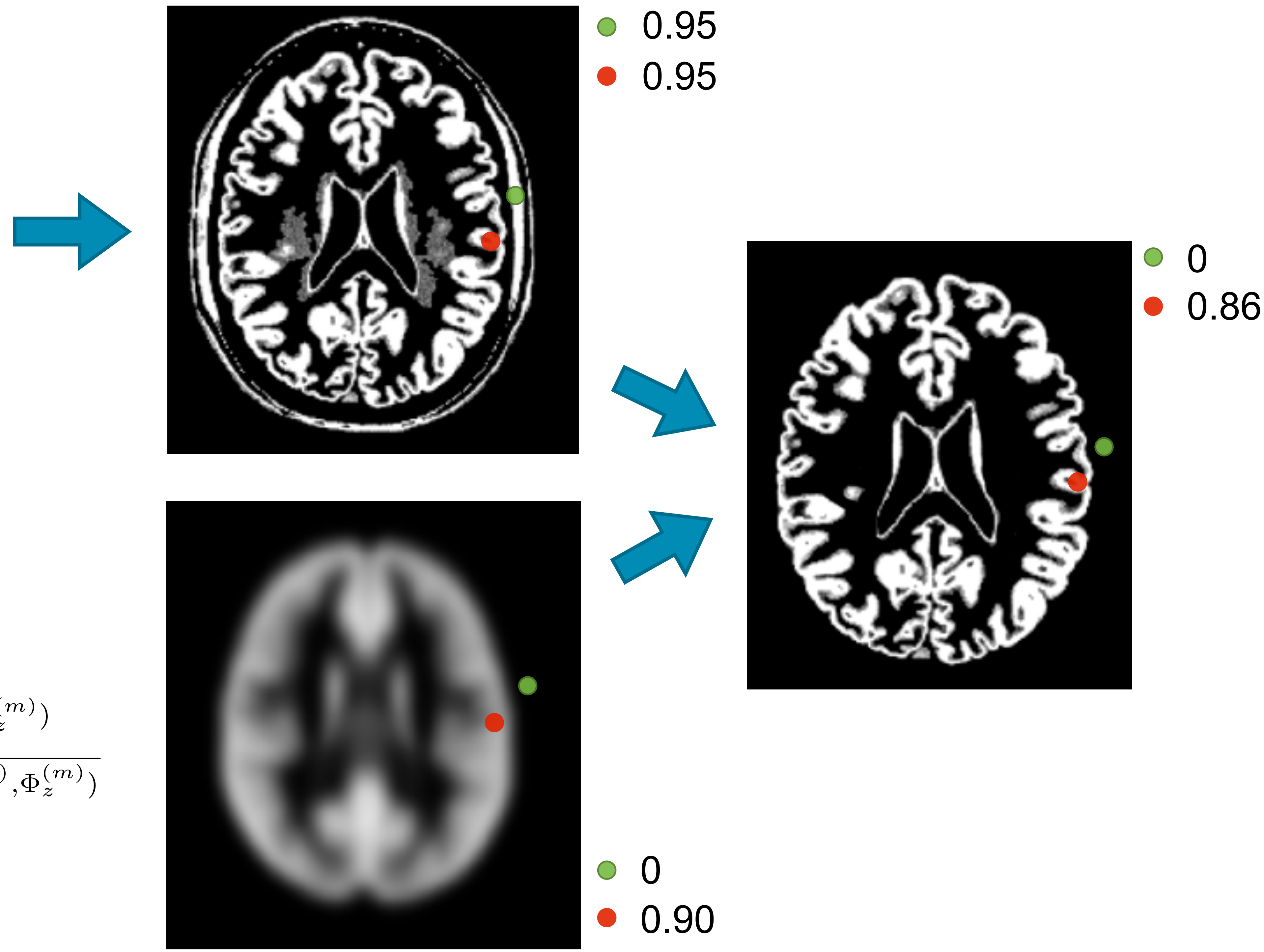
Converged?

No

Yes

Finish

Anatomical Priors



$$f(z_i = e_k \mid p^{(m)}_{\mathcal{N}_i} \Phi^{(m)}_z) =$$

$$\frac{\pi_{ik}\, e^{-\beta U_{\mathrm{MRF}}(e_k | p^{(m)}_{\mathcal{N}_i}, \Phi^{(m)}_z)}}{\sum_{j=1}^{K} \pi_{ij}\, e^{-\beta U_{\mathrm{MRF}}(e_j | p^{(m)}_{\mathcal{N}_i}, \Phi^{(m)}_z)}}$$

Step by step:

For all y and for all classes

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\Big(y_i - \mu_k\Big)$$

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

Calculate *p* *and log L*

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

$$p_{ik}^{(m+1)} = \frac{f\Big(y_i \mid z_i = e_k, \Phi_y^{(m)}\Big)\, f\Big(z_i = e_k\Big)}{\sum_{j=1}^{K} f\Big(y_i \mid z_i = e_j, \Phi_y^{(m)}\Big)\, f\Big(z_i = e_j\Big)}$$

Update parameters

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} y_i}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

$$f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{e^{-\beta U_{\mathrm{MRF}}(e_k | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^{K} e^{-\beta U_{\mathrm{MRF}}(e_j | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}$$

$$\Big(\sigma_k^{(m+1)}\Big)^2 = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} \Big(y_i - \mu_k^{(m+1)}\Big)^2}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

Compare current log L with old log L

Converged?

No

Yes

Finish

Step by step:

Calculate *p* and *log L*

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

$$p_{ik}^{(m+1)} = \frac{f\left(y_i \mid z_i = e_k, \Phi_y^{(m)}\right) f\left(z_i = e_k\right)}{\sum_{j=1}^{K} f\left(y_i \mid z_i = e_j, \Phi_y^{(m)}\right) f\left(z_i = e_j\right)}$$

For all y and for all classes

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\left(y_i - \mu_k\right)$$

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

Update parameters

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} y_i}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

$$\boxed{f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{\pi_{ik}\, e^{-\beta U_{\mathrm{MRF}}(e_k \mid p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^{K} \pi_{ij}\, e^{-\beta U_{\mathrm{MRF}}(e_j \mid p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}}$$

$$\left(\sigma_k^{(m+1)}\right)^2 = \frac{\sum_{i=1}^{n} p_{ik}^{(m+1)} \left(y_i - \mu_k^{(m+1)}\right)^2}{\sum_{i=1}^{n} p_{ik}^{(m+1)}}$$

Compare current log L with old log L

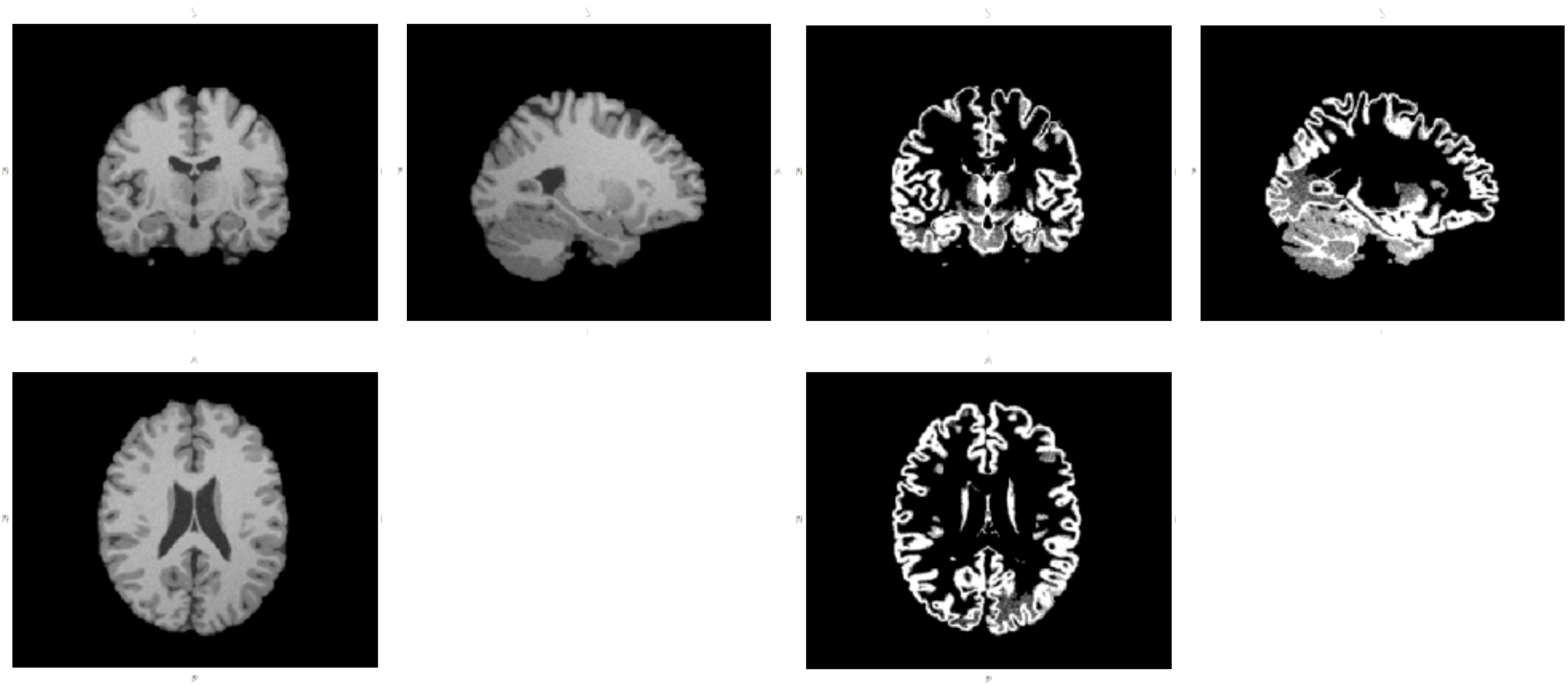
Converged?

No

Yes

Finish

Bias Field

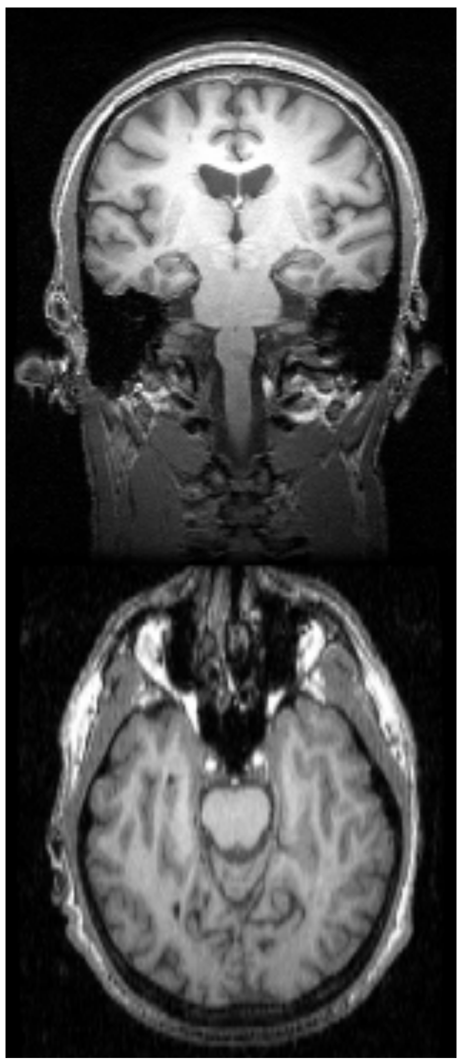


- Bias field is modelled as a linear combination of polynomial basis functions

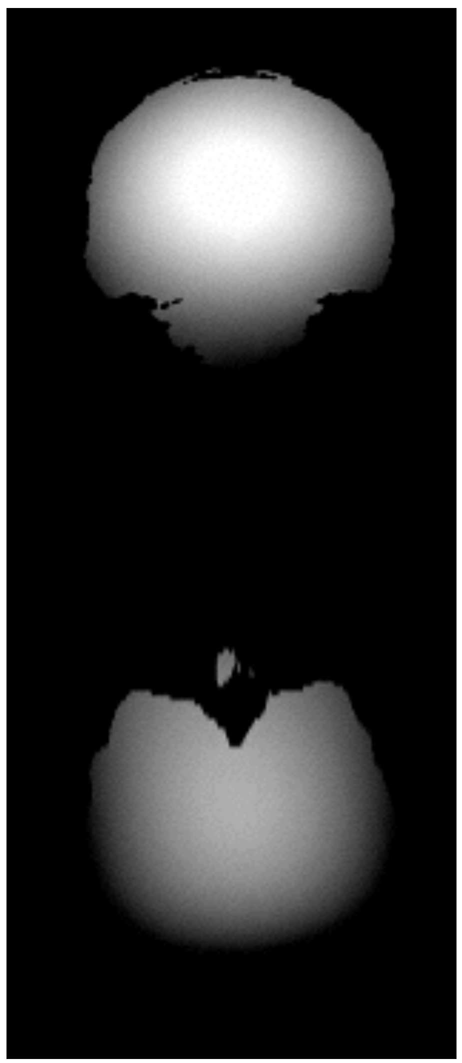
$$\mathbf{BF} = \sum_{k=1}^{K} c_k \Phi_k(p_i)$$

Bias Field

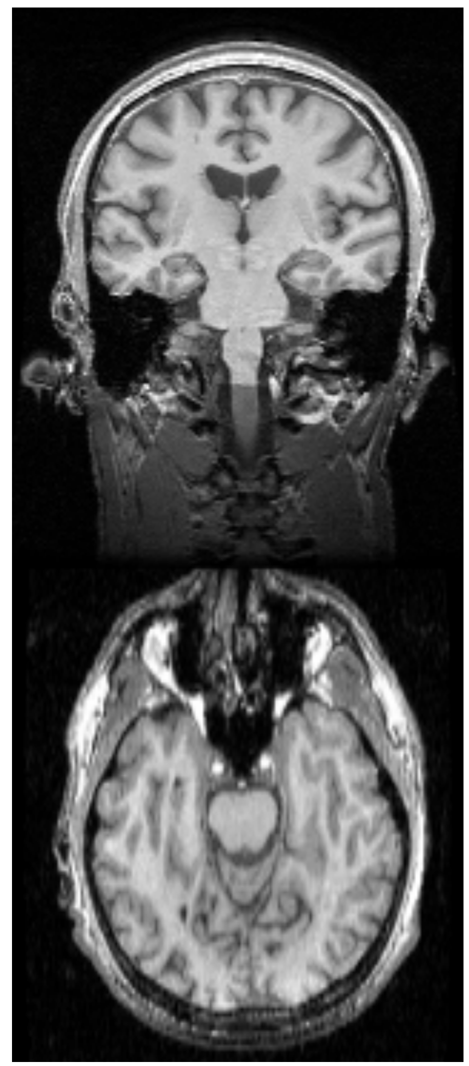
No Bias Field Correction



Estimated Bias Field



Bias Field Corrected



Bias field is modelled as a linear combination of polynomial basis functions

$$\mathbf{BF} = \sum_{k=1}^{K} c_k \Phi_k(p_i)$$

$$\begin{aligned} \Phi_k(p_i) &\equiv \text{Basis function} \\ c_k &\equiv \text{Coefficient of the basis function} \\ p_i &\equiv \text{Coordinate of pixel } i \end{aligned}$$

$$G_{\sigma_k}\left(y_i - \mu_k - \sum_j c_j \phi_j(x_i)\right)$$

Coefficient optimisation

Setting the partial derivative of the loglik to 0 we have

$$\sum_i \phi_k(x_i)\left(\sum_j \frac{p(\Gamma_i = j|y_i,\theta,C)}{\sigma_j^2} \cdot \left[y_i - \mu_j - \sum_l c_l\phi_l(x_i)\right]\right) = 0 \quad \forall k.$$

Matrix form

$$C = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \end{bmatrix} = (A^TWA)^{-1}A^TWR$$

$$A = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) & \cdots \\ \phi_1(x_2) & \phi_2(x_2) & \phi_3(x_2) & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

$$W = \mathrm{diag}(w_i), \quad w_i = \sum_j w_{ij},$$

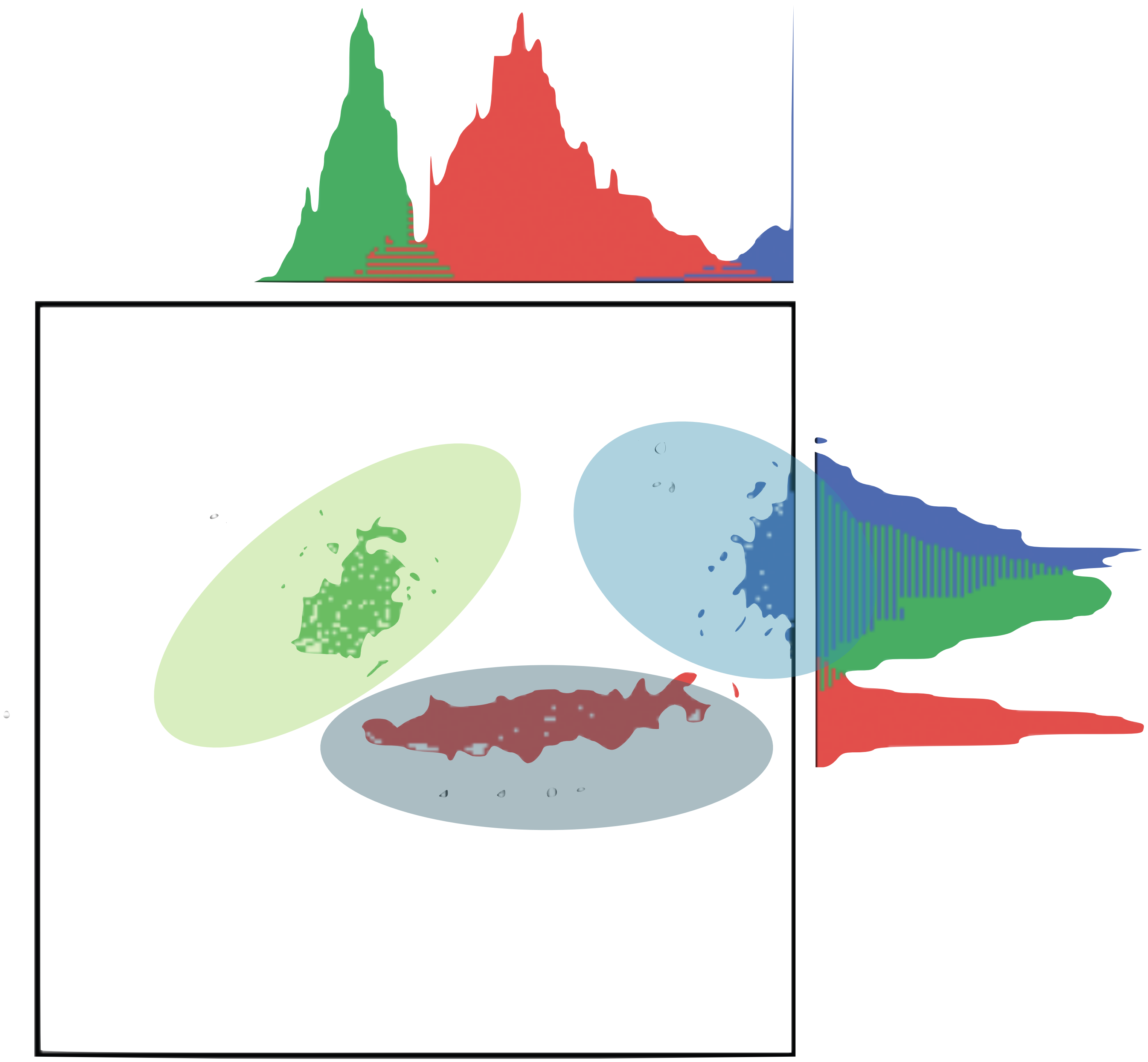
$$w_{ij} = \frac{p(\Gamma_i = j|y_i,\theta,C)}{\sigma_j^2}$$

$$R = \begin{bmatrix} y_1 - \tilde{y}_1 \\ y_2 - \tilde{y}_2 \\ \vdots \end{bmatrix}, \quad \tilde{y}_i = \frac{\sum_j w_{ij}\mu_j}{\sum_j w_{ij}}.$$

Multispectral/Multimodal Segmentation

- Use multiple images of the same scene with different physical proprieties to improve the accuracy and the results:
  - MRI - T1, T2 and PD, ...
  - CT

Multispectral/Multimodal Segmentation



What changes?

Calculate *p* and *log L*

$$log\, L(\Phi) = \sum_i log\Big(\sum_k f(y_i \mid z_i = e_k, \Phi_y) f(z_i = e_k)\Big)$$

$$p_{ik}^{(m+1)} = \frac{f\left(y_i \mid z_i = e_k, \Phi_y^{(m)}\right) f\left(z_i = e_k\right)}{\sum_{j=1}^K f\left(y_i \mid z_i = e_j, \Phi_y^{(m)}\right) f\left(z_i = e_j\right)}$$

For all y and for all classes

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\left(y_i - \mu_k\right)$$

$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

Update parameters

$$\mu_k^{(m+1)} = \frac{\sum_{i=1}^n p_{ik}^{(m+1)} y_i}{\sum_{i=1}^n p_{ik}^{(m+1)}}$$

$$\boxed{f(z_i = e_k \mid p_{\mathcal{N}_i}^{(m)} \Phi_z^{(m)}) = \frac{e^{-\beta U_{\mathrm{MRF}}(e_k | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}{\sum_{j=1}^K e^{-\beta U_{\mathrm{MRF}}(e_j | p_{\mathcal{N}_i}^{(m)}, \Phi_z^{(m)})}}}$$

$$\left(\sigma_k^{(m+1)}\right)^2 = \frac{\sum_{i=1}^n p_{ik}^{(m+1)} \left(y_i - \mu_k^{(m+1)}\right)^2}{\sum_{i=1}^n p_{ik}^{(m+1)}}$$

Compare current log L with old log L

Converged?

No

Yes

Finish

What changes?

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\sigma_k}\left(y_i - \mu_k\right)$$

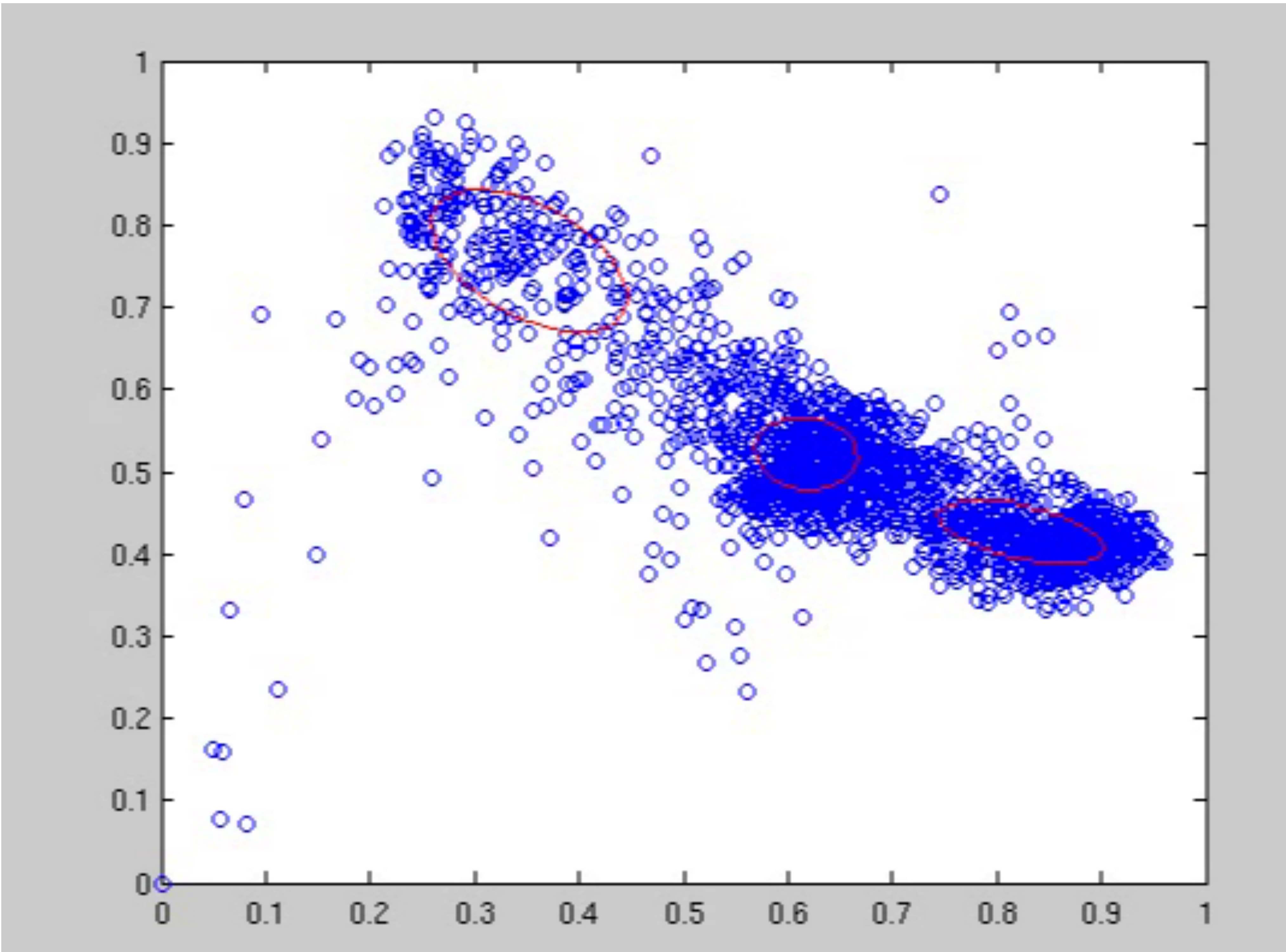
$$G_{\sigma_k}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x)^2}{2\sigma^2}}$$

$$\downarrow$$

$$f(y_i \mid z_i = e_k, \Phi_y) = G_{\Sigma_k}\left(\vec{y_i} - \vec{\mu_k}\right)$$

$$G_{\sigma_k}(\vec{x}) = \frac{1}{\sqrt{2\pi\Sigma^2}}\, e^{\frac{-(\vec{x})^2}{2\Sigma^2}}$$

Multispectral/Multimodal Segmentation

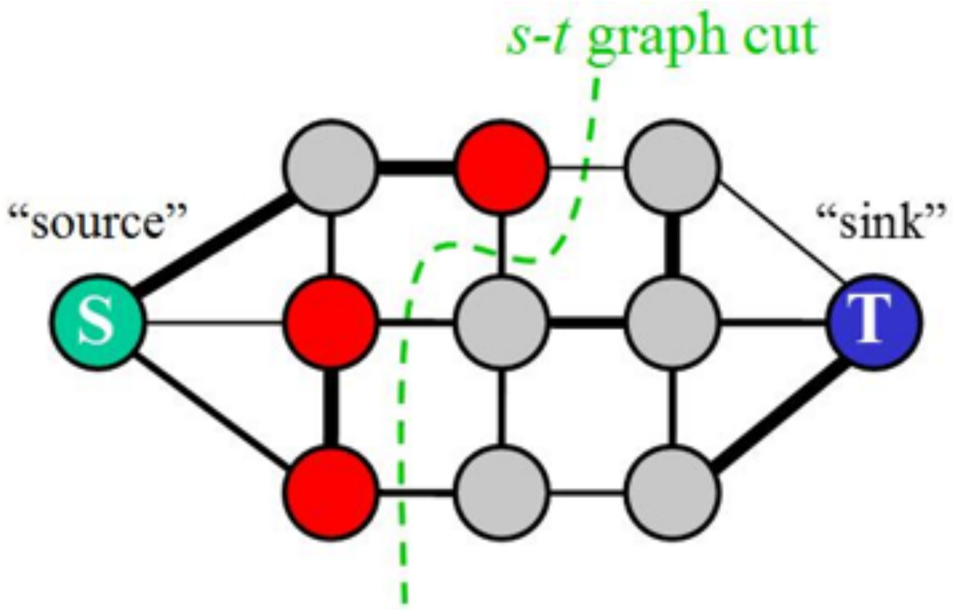


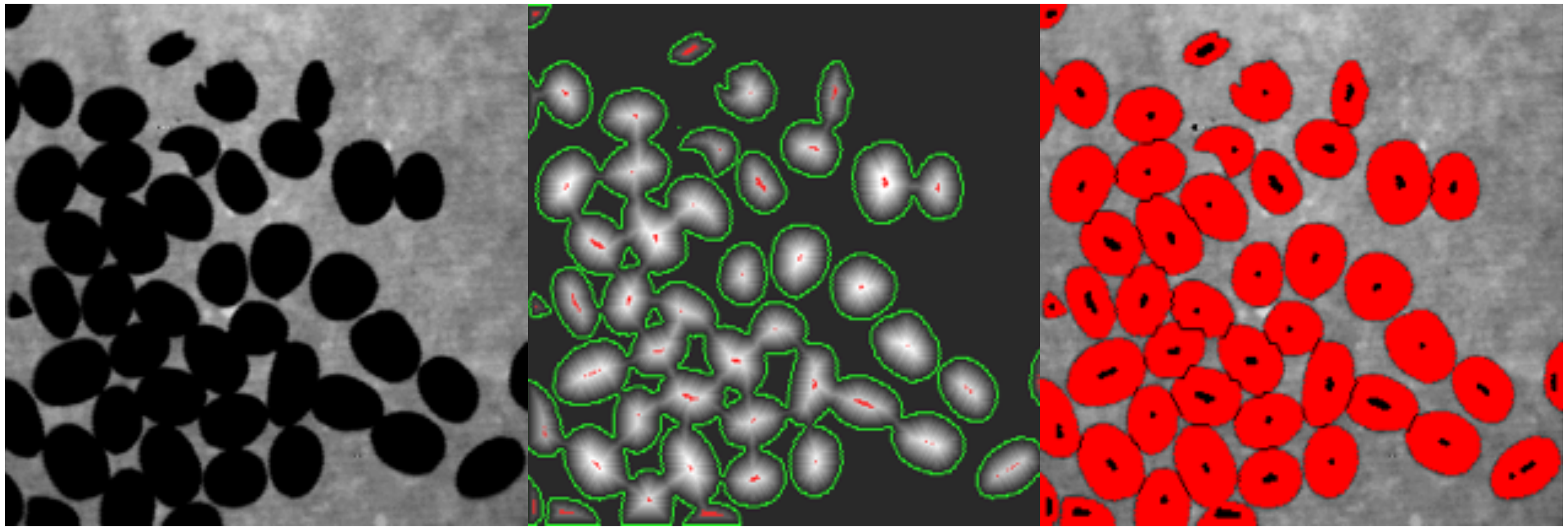
Probabilistic Frameworks

- Flexible frameworks for image segmentation
- Can become highly complex but also very powerful
- They can, when correctly optimised, model the data adequately
- Many algorithmic tricks to improve convergence
- Overall, if one can model it, one can do it

REGION GROWING / EDGE BASED

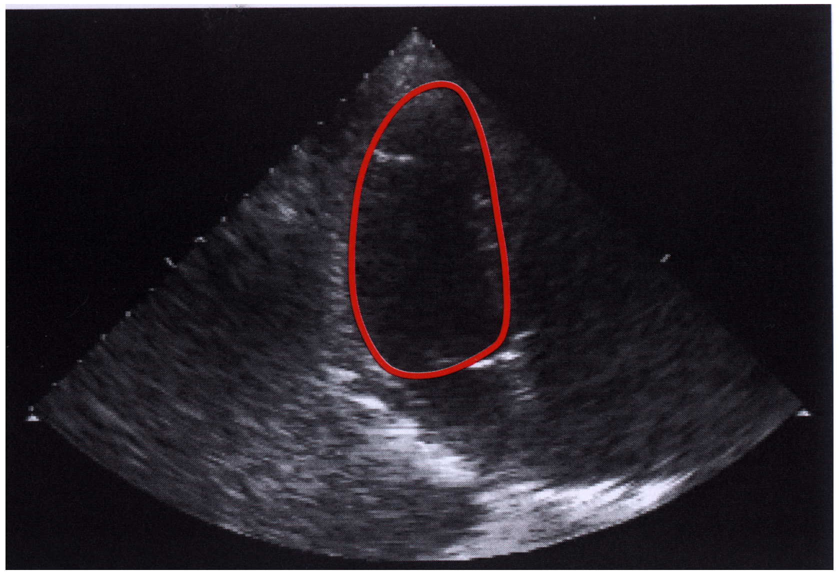
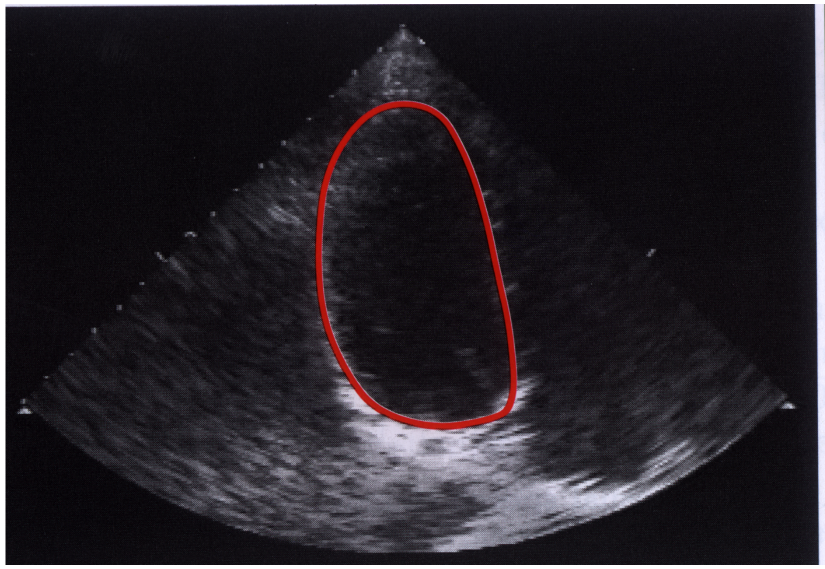
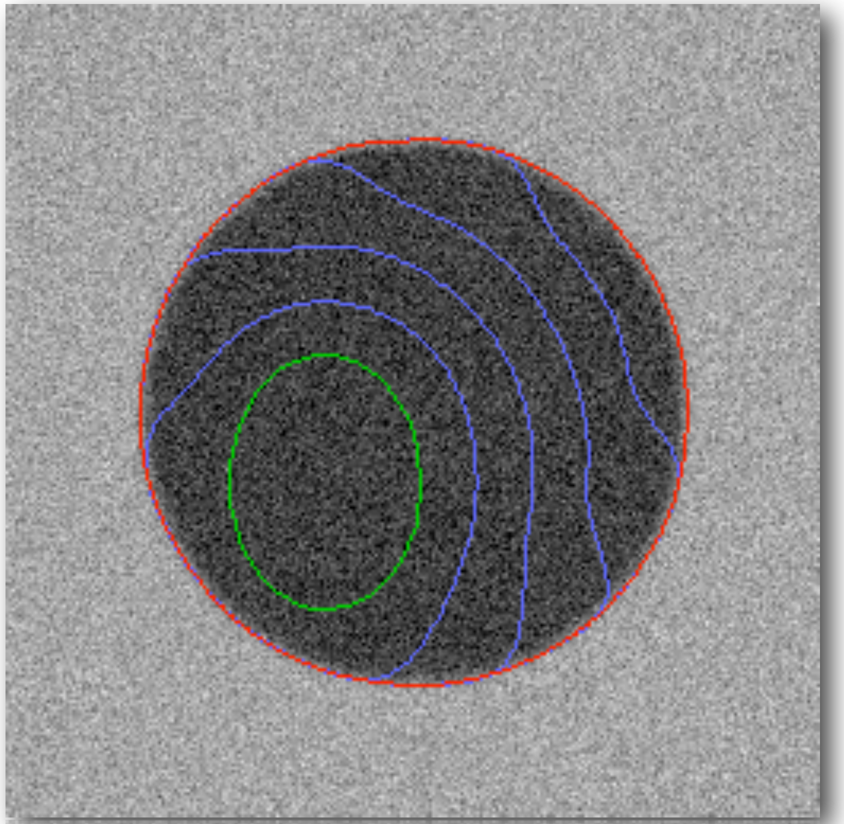
Segmentation - Region/Edges based





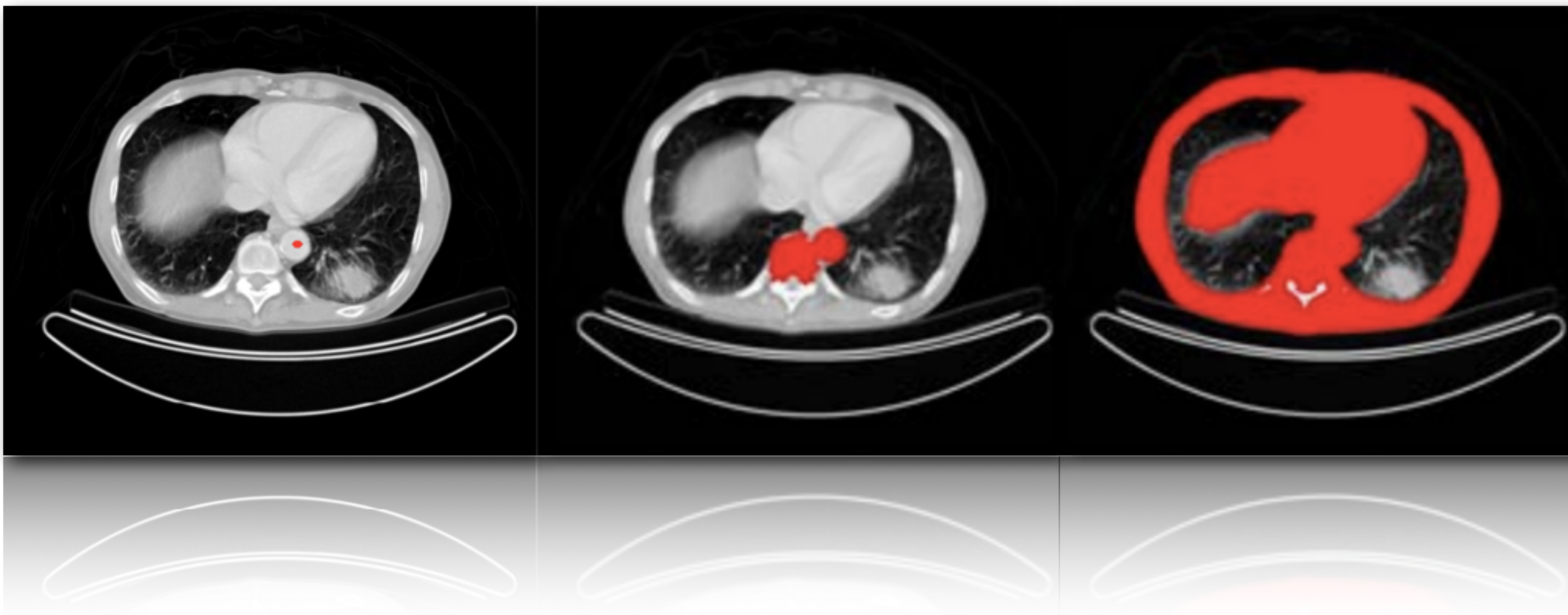
Why do we need regional information?

- It is important to include localisation information instead of only intensity values
- Segmentation of objects that have some configuration, connectivity, shape etc.

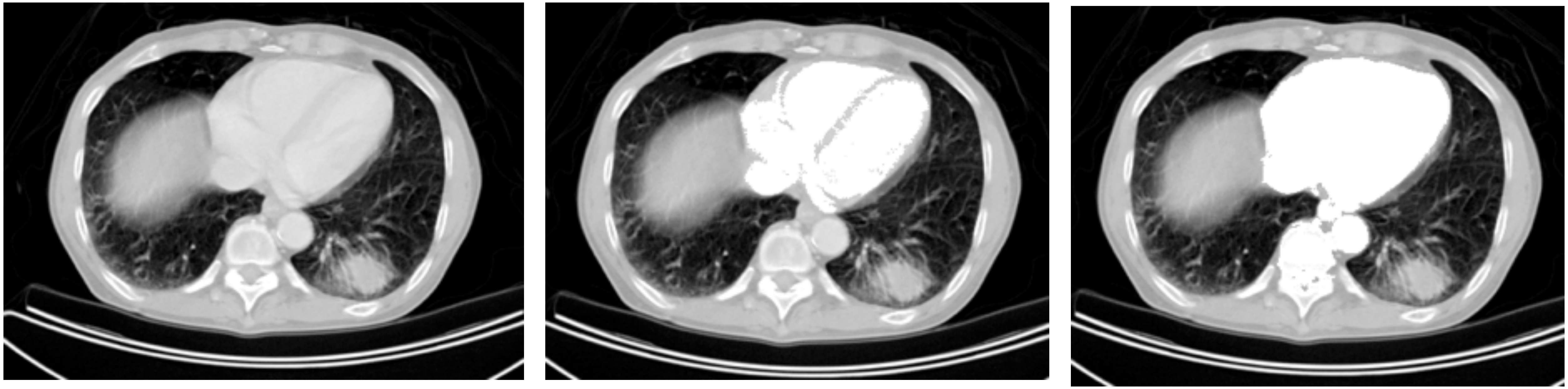
Region Growing

- The simplest region based method



Region Growing

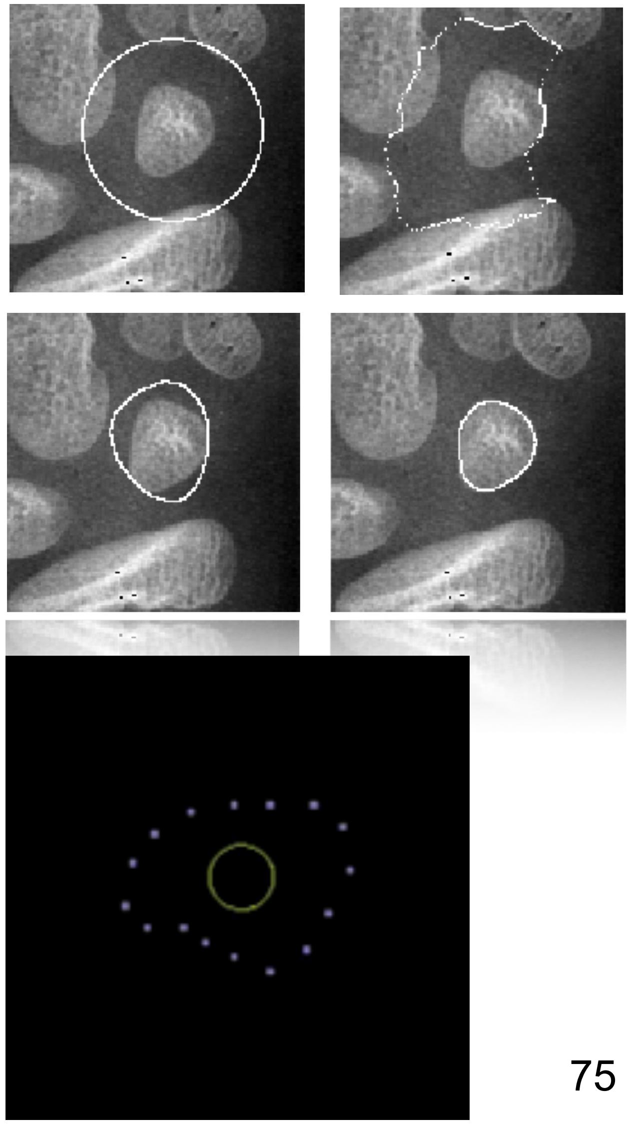
- Works quite well when regions are well-defined
- Works badly when borders are fuzzy and is not robust to noise
- Can “leak”
- Usually some user-interaction is required, but often it is minimal



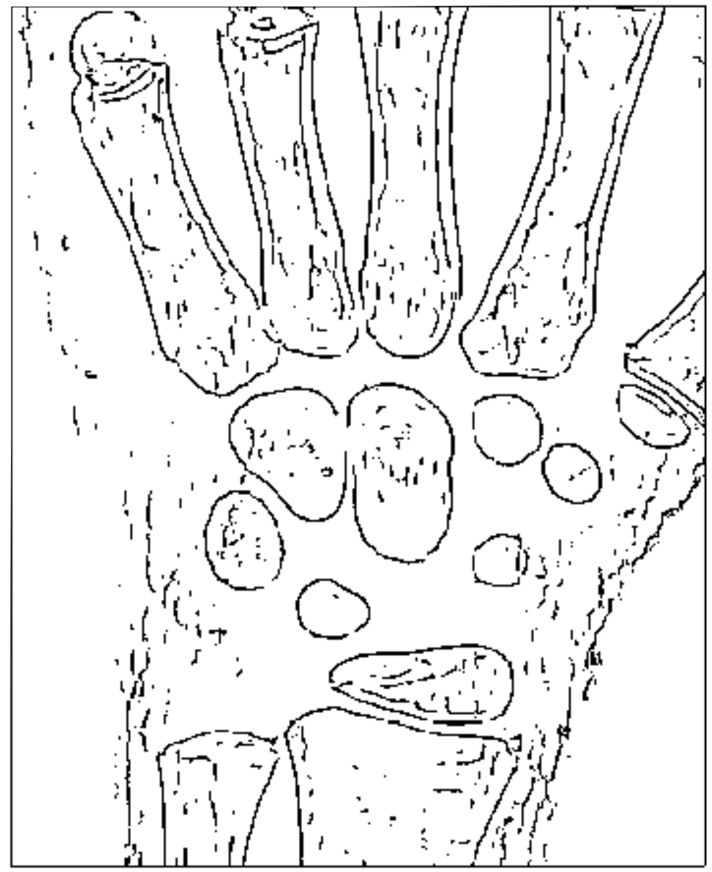
EXPLICIT PARAMETRIC MODELS

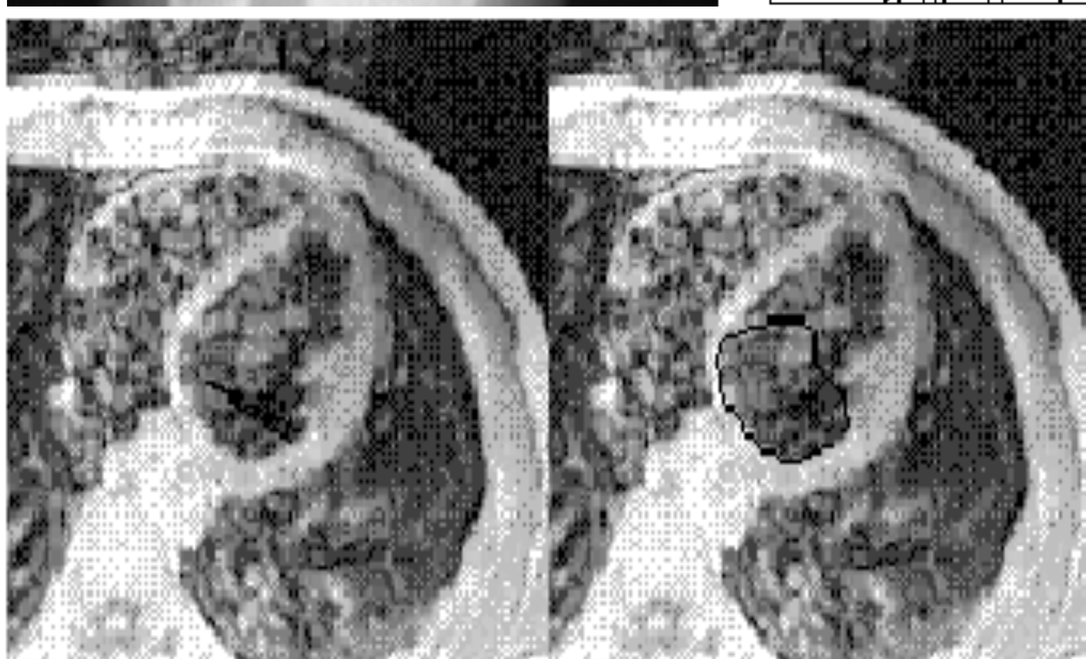
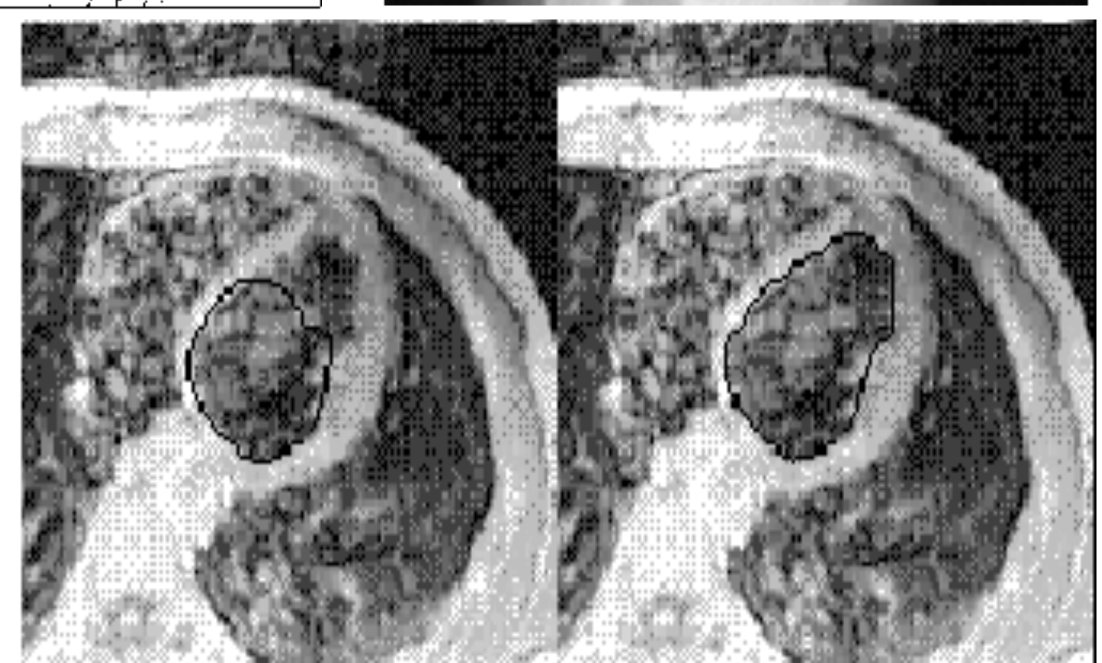
Snakes

- Is a framework where a surface/curve is implicitly parameterised, that attempts to minimise an energy associated to a parameterised contour or curve as a sum of an internal and external energy
  - The external energy is minimal when the snake is at the object boundary position.
  - The internal energy is minimal when the snake has specific shape characteristics. Normally:
    - High energy in elongated contours (elastic force)
    - High energy in bended/high curvature contours (rigid force)
    - The internal energy is minimised when the shape is as regular and smooth as possible.



Why Snakes?

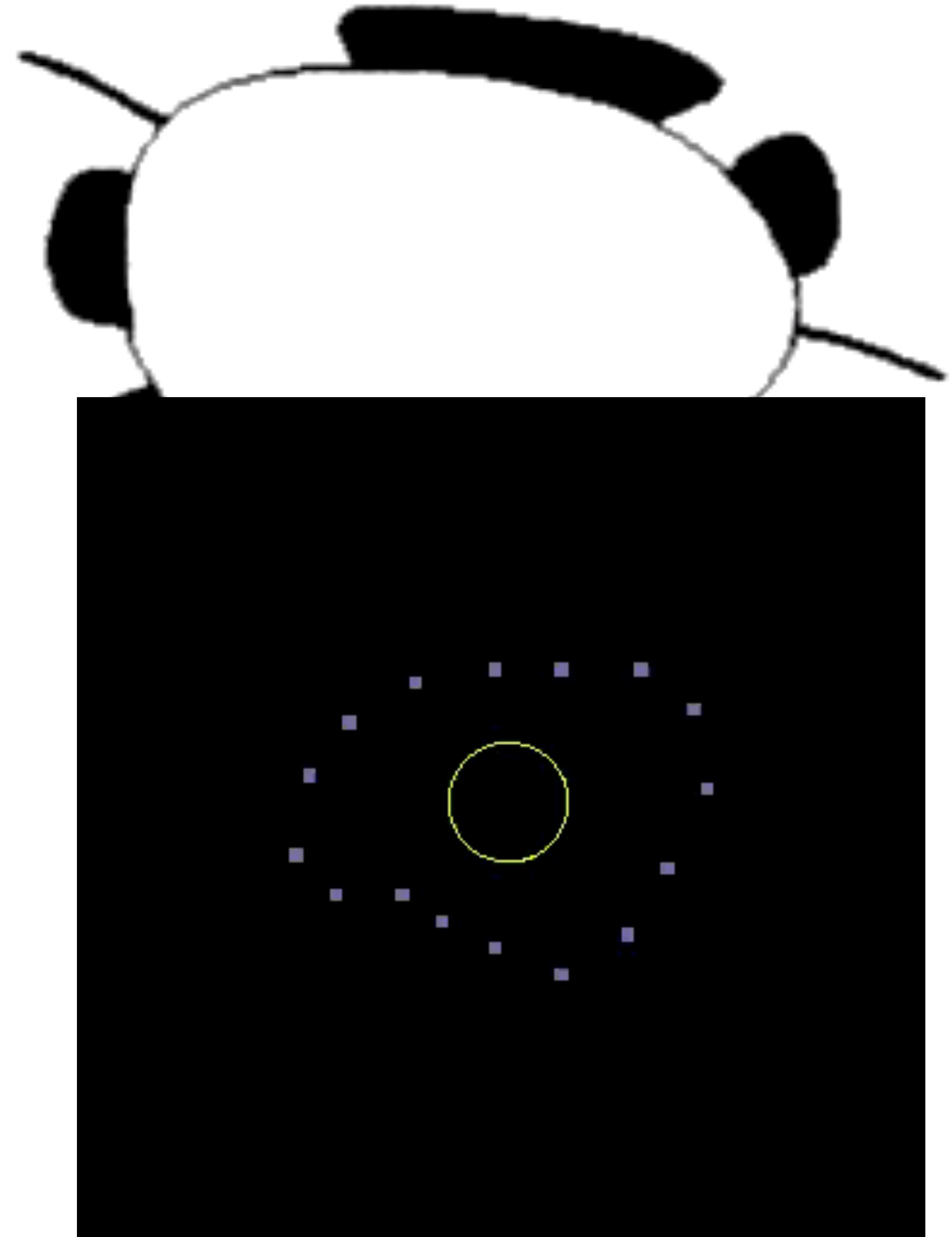



Why Snakes?

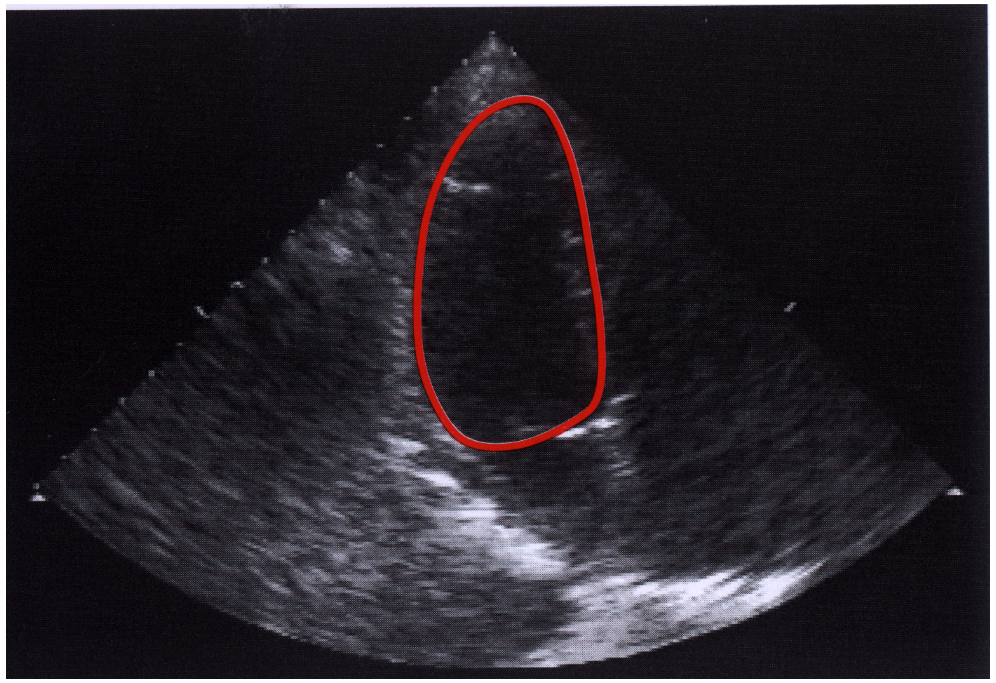
Advantages:

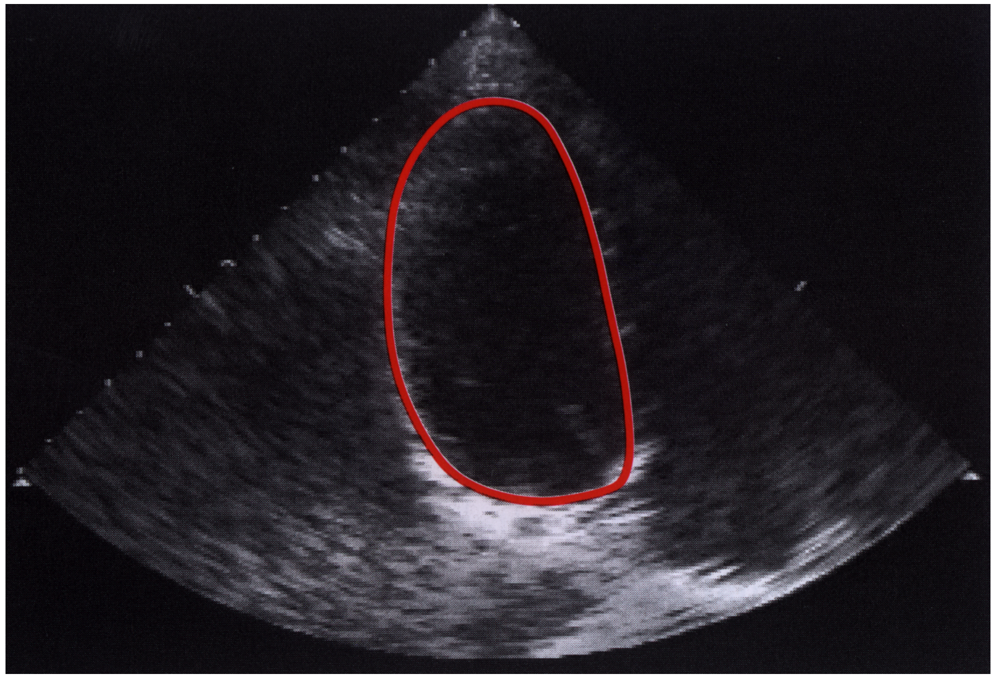
- Physics-based model;
- Soft and hard constraints;
- Interpret sparse, incomplete and redundant information;
- Generic restrictions allowed;
- Robustness to noise
- Use of geometric models
- Natural regularisation of ill-posed problems

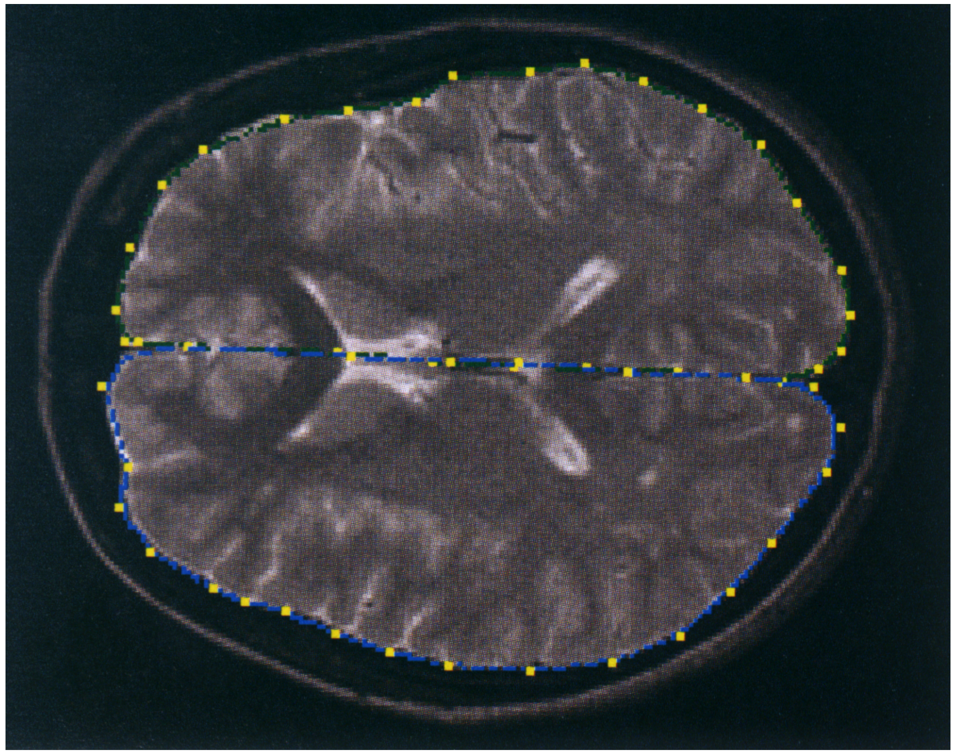
Why Snakes?

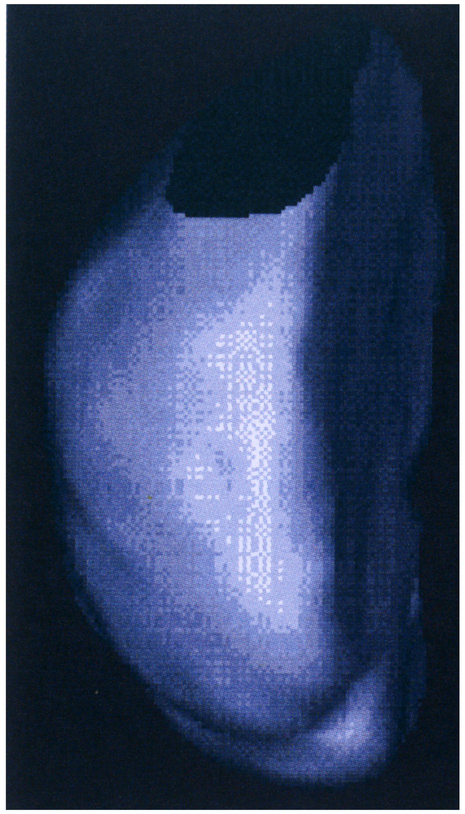


Why Snakes?









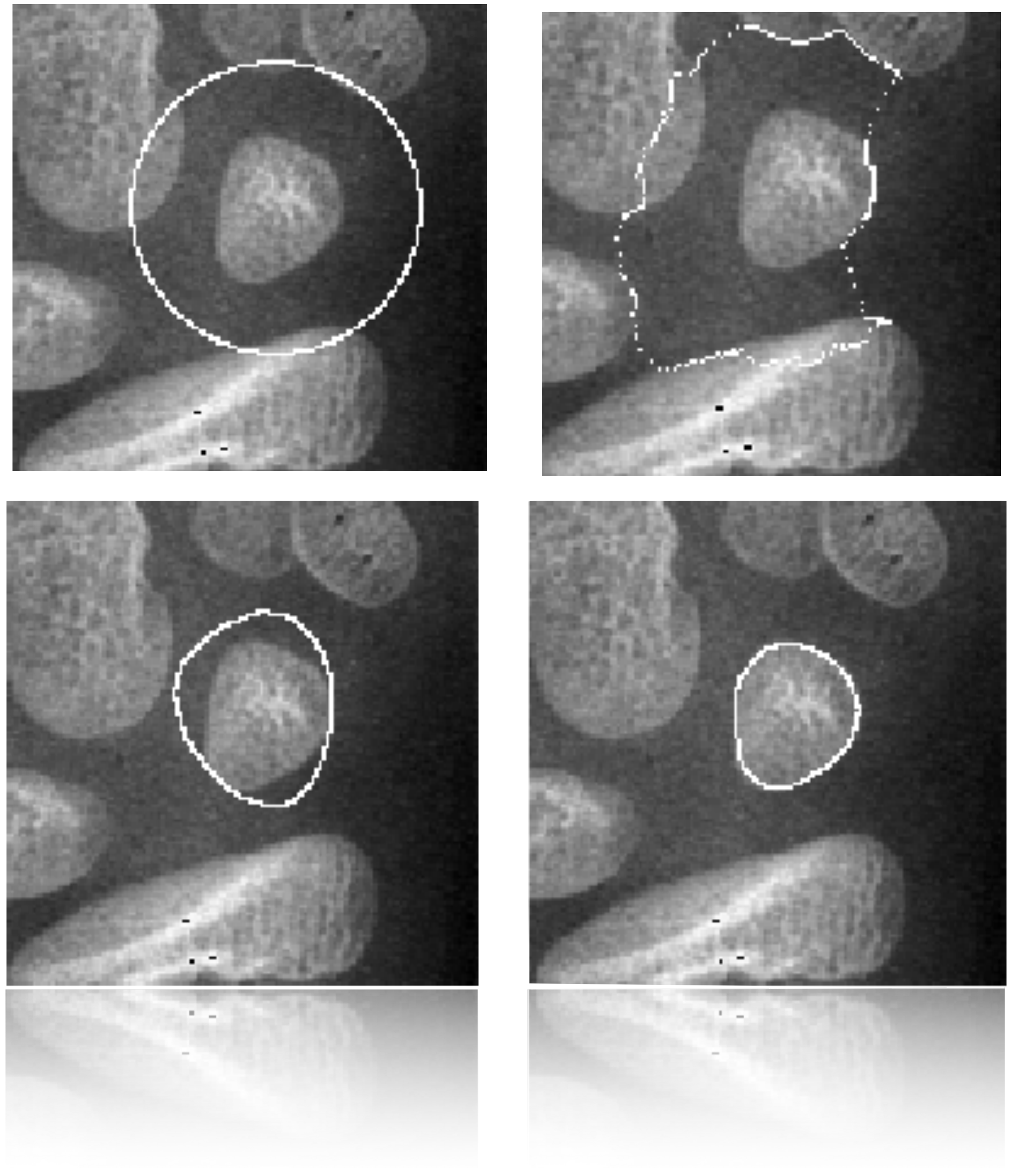
Why Snakes?

Disadvantages:

- Energy minimisation;
- Initialisation;
- Parameterisation;
- Convergence and numerical stability;
- Shape degeneration;
- Computational cost (mainly in 3D or with very complex forces)

Problems - Initialisation

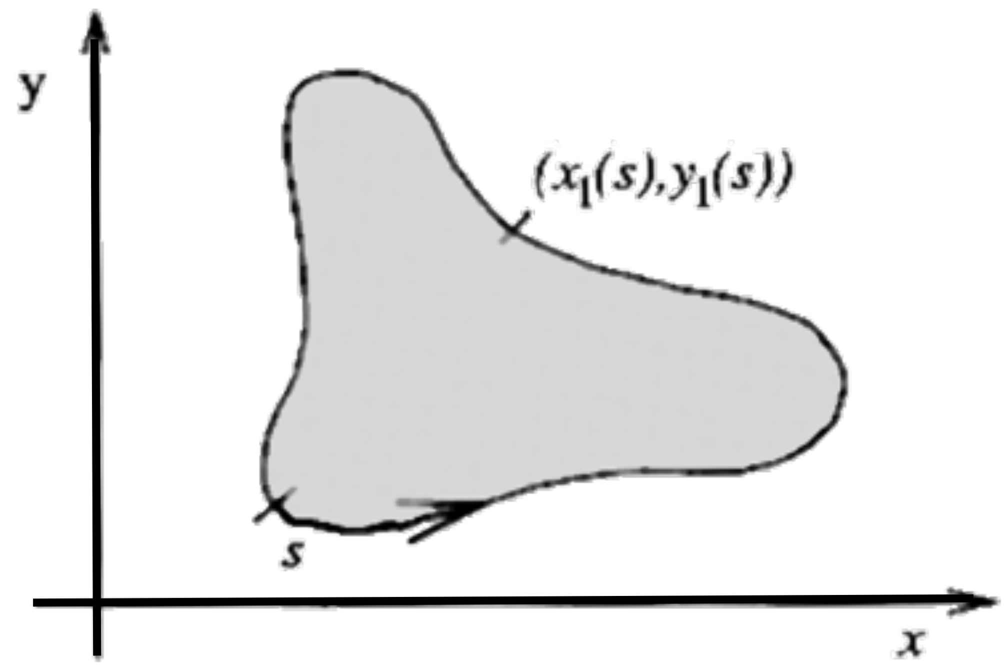
- Manual initialisation
  - by end points, snake growing
  - object localisation - balloons
- Automatic initialisation
  - Shape prior,
  - Feature detection
- Approximate model
  - Medical atlases
  - Tracking
  - Stereo matching



Curve Parameterisation

- The snake is a contour represented parametrically as $c(s) = (x(s), y(s))$

  where $x(s)$ and $y(s)$ are the coordinates along the contour and $s \in [0,1]$



Energy functional

- The energy functional used is a sum of several terms, each corresponding to some force acting on the contour.
- A suitable energy functional is the sum the following three terms:

$$E = \int (\alpha(s)E_{cont} + \beta(s)E_{curv} + \gamma(s)E_{image})ds$$

- The parameters α, β and γ control the relative influence of the corresponding energy terms and can vary along the curve.

The continuity term

Minimise the first derivative:

$$E_{cont} = \| \frac{dc}{ds} \|^2$$

In the discrete case, the contour is approximated by N points $p_1, p_2, \ldots, p_N$ and the first derivative is approximated by a finite difference:

$$E_{cont} = \|p_i - p_{i-1}\|^2 \text{ or}$$

$$E_{cont} = (x_i - x_{i-1})^2 + (y_i - y_{i-1})^2$$

This term tries to minimise the distance between the points, however, it has the effect of causing the contour to shrink.

The continuity term - Equidistance

- A better form for E_{cont} is the following:

$$E_{cont} = (\bar{d} - \|p_i - p_{i-1}\|)^2$$

where d is the average distance between the points of the snake.

The new E_{cont} attempts to keep the points at equal distances (i.e, spread them equally along the snake).

The curvature term

Minimise the second derivative:

$$E_{curv} = \| \frac{d^2 c}{ds^2} \|^2$$

In the discrete case the second derivative is approximated by a finite difference:

$$E_{curv} = \| p_{i-1} - 2p_i + p_{i+1} \|^2 \text{ or}$$

$$E_{curv} = (x_{i-1} - 2x_i + x_{i+1})^2 + (y_{i-1} - 2y_i + y_{i+1})^2$$

This term tries to minimise the bending of the curve points.

Image term (edge attraction)

- The purpose of this term is to attract the contour toward the target contour.
- This can be achieved by the following function

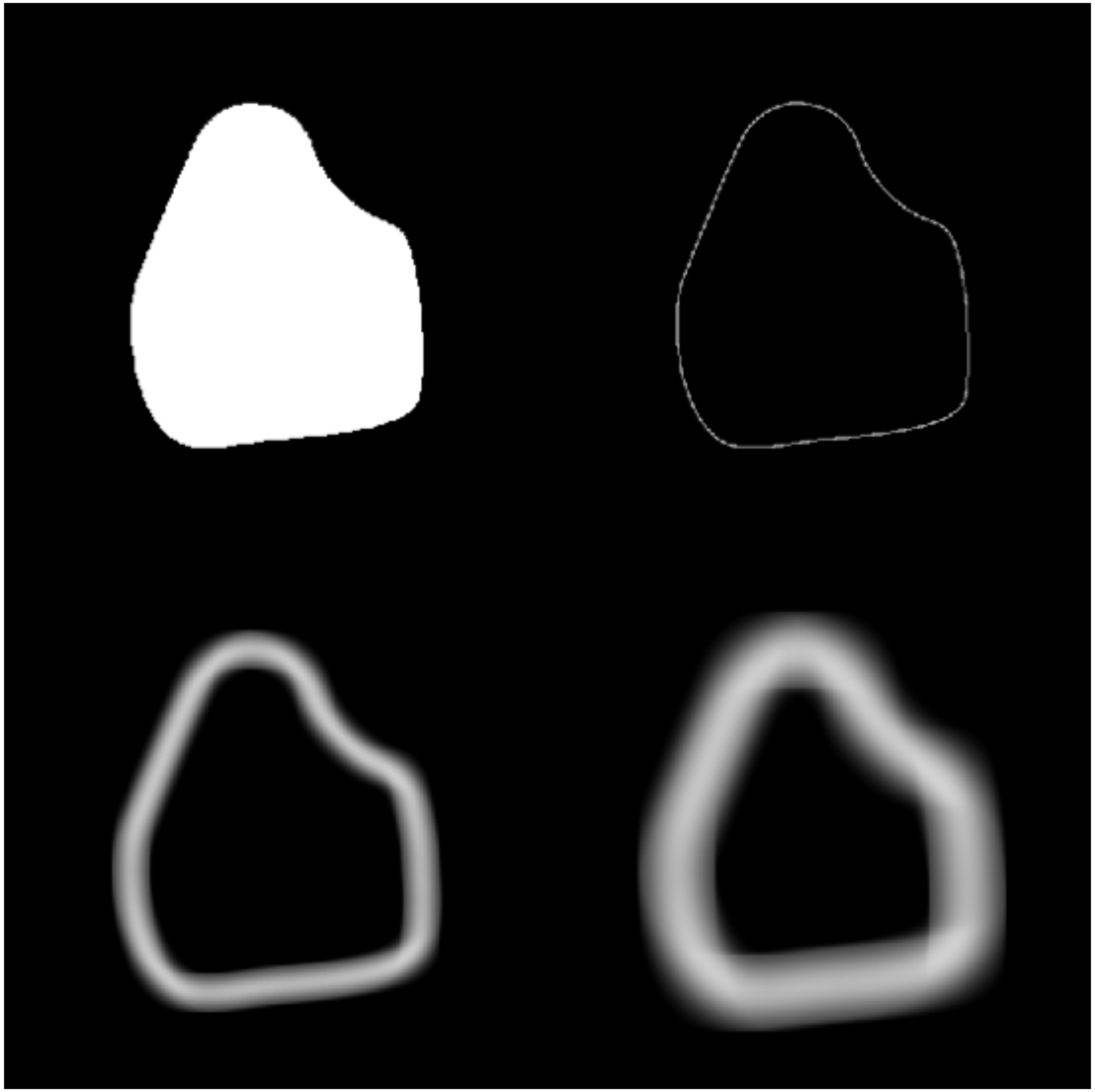
$$E_{image} = -\|\nabla I\|$$

where ∇I is the gradient of the intensity computed at each snake point.

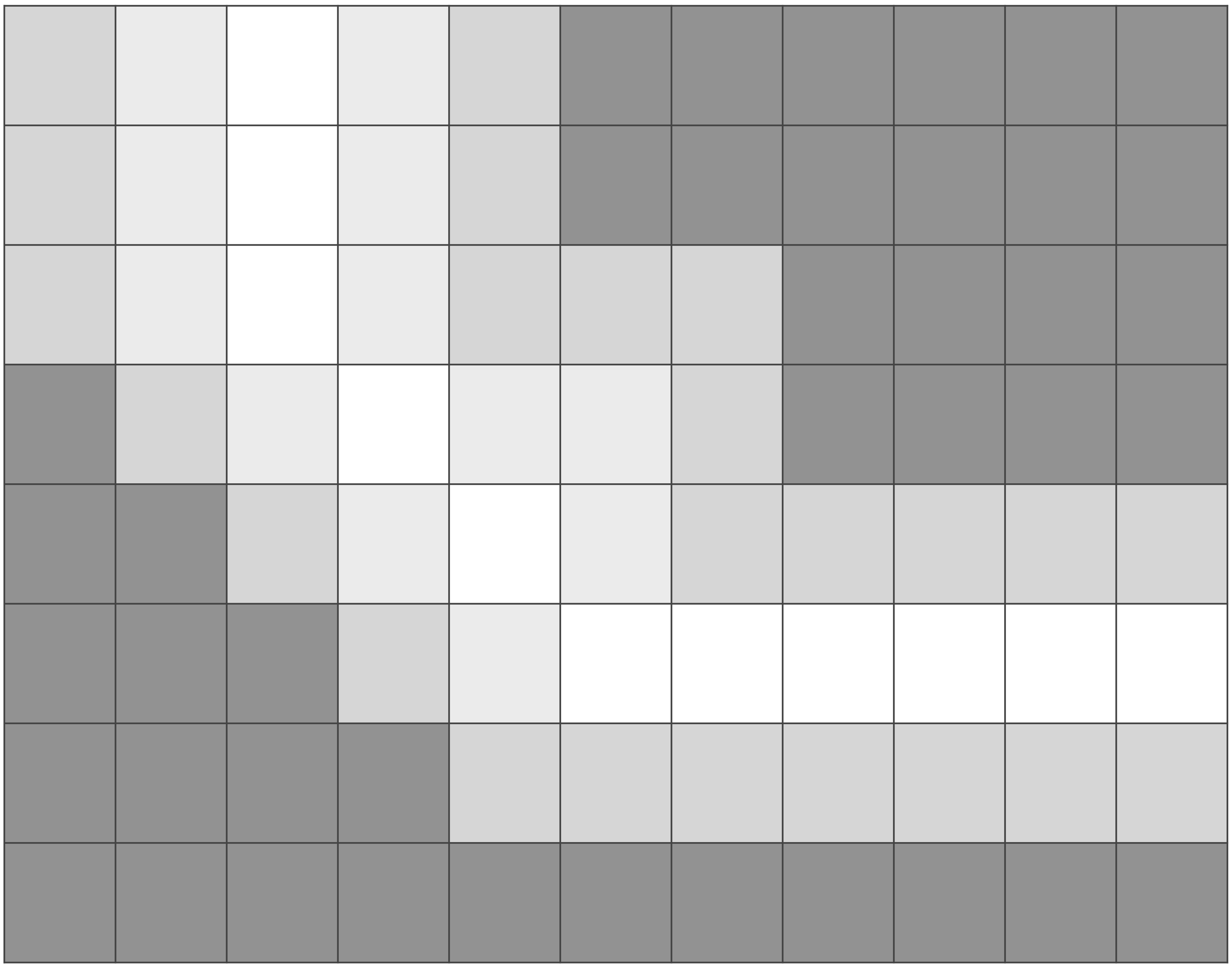
- Note that E_{image} becomes very small when the snake points get close to an edge

Gradient based forces

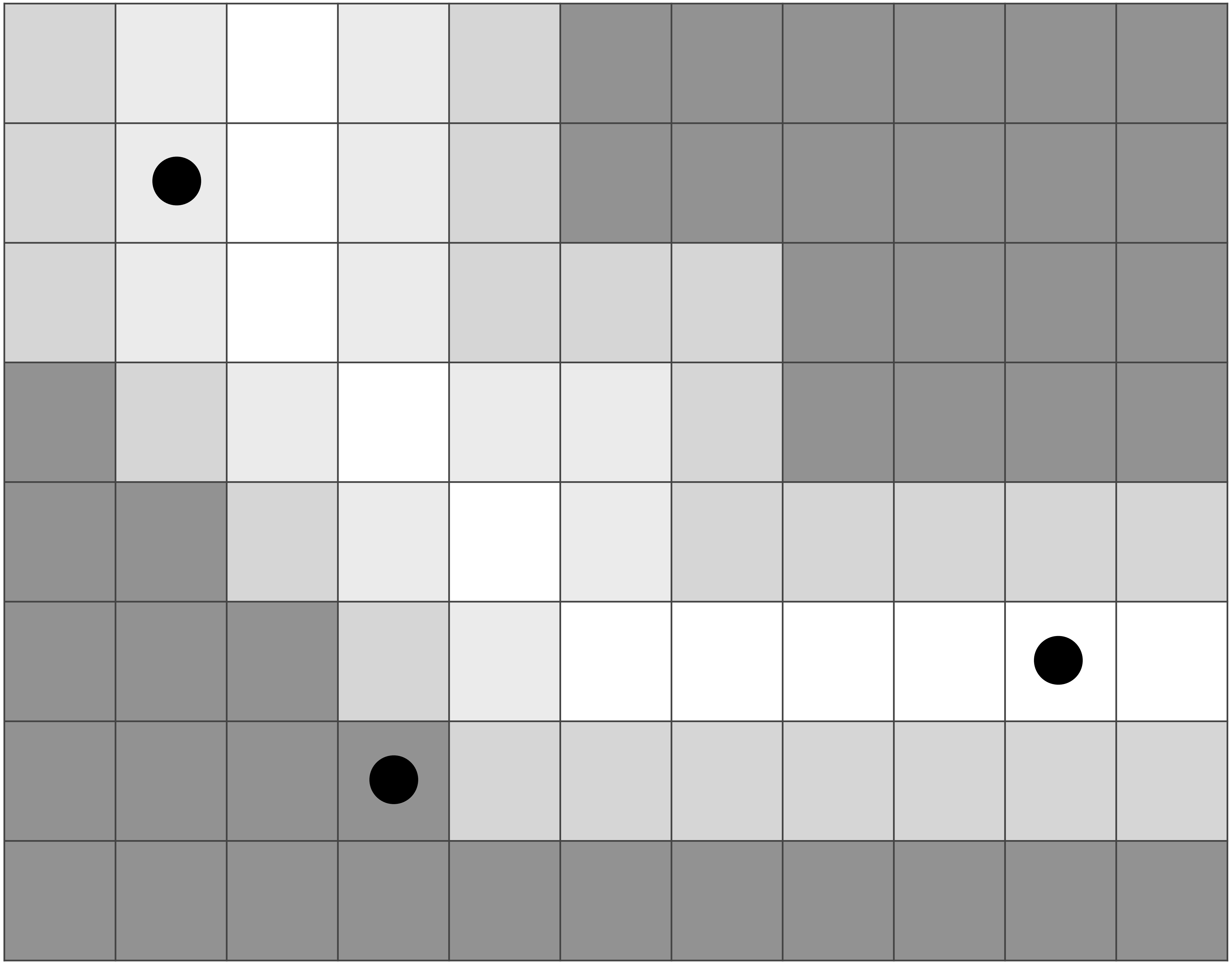
- The gradient itself only gives us the edge location
- Gaussian filtering of the gradient edge increases the capture range but reduces the precision



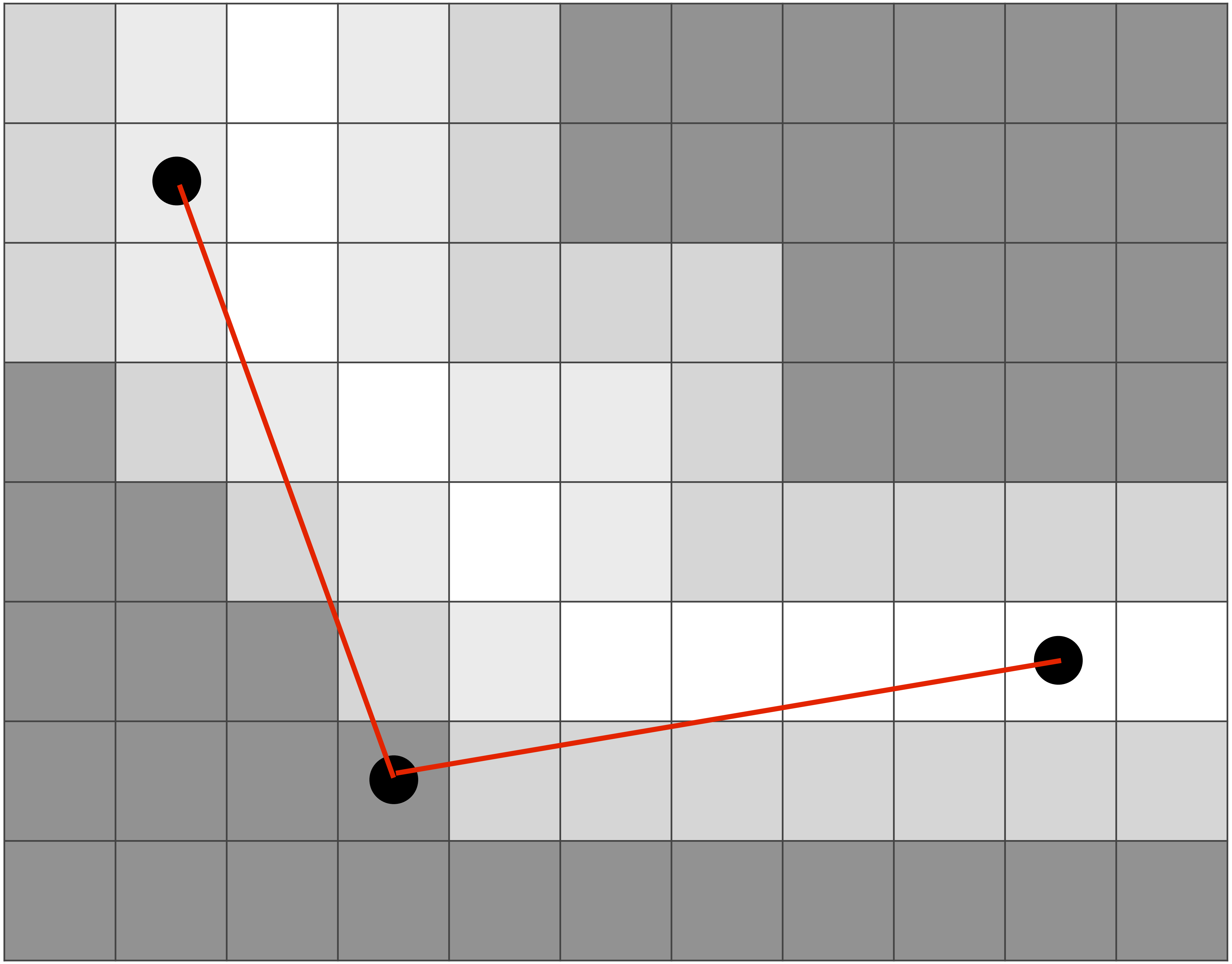
Snakes - implementation



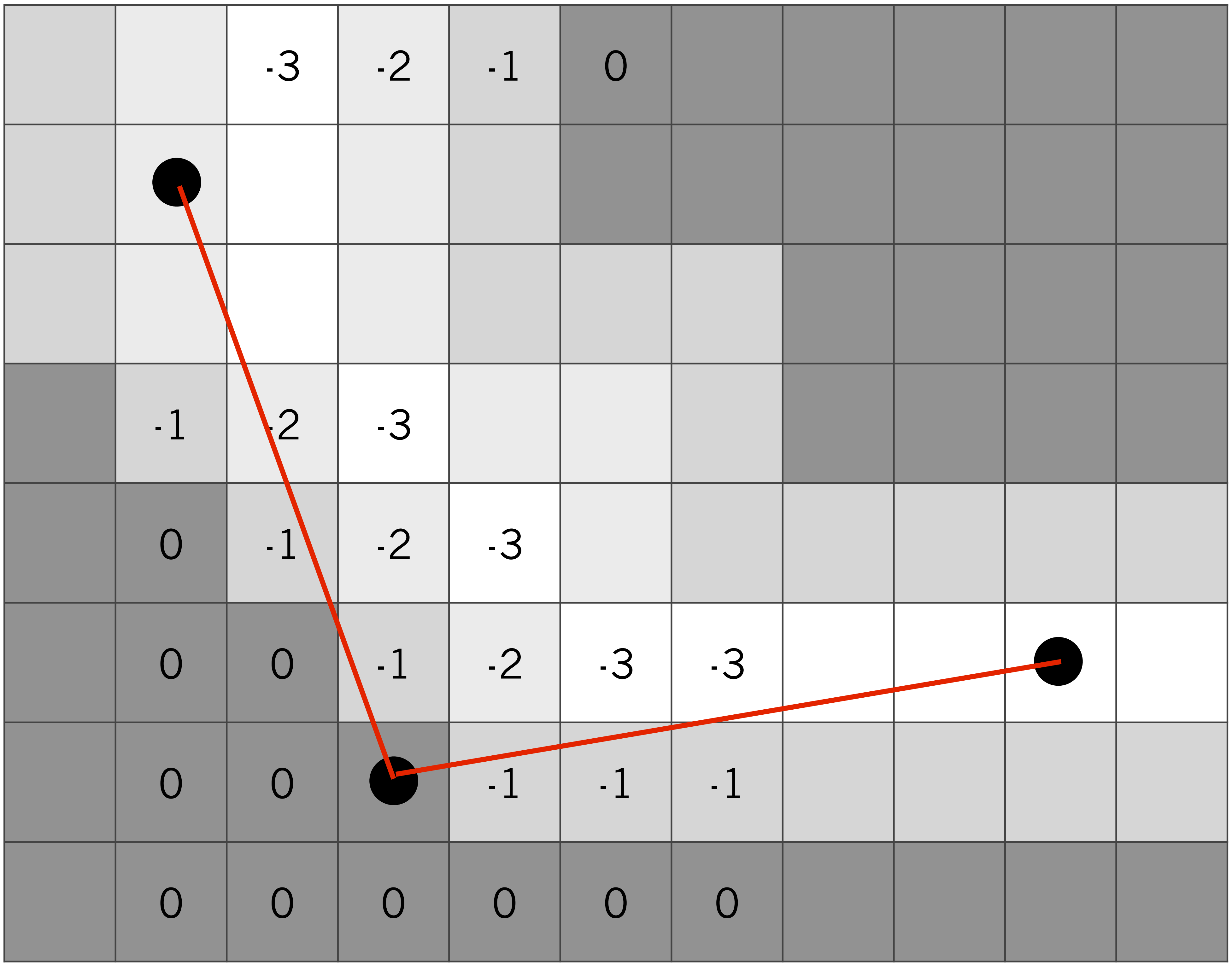
Snakes - implementation



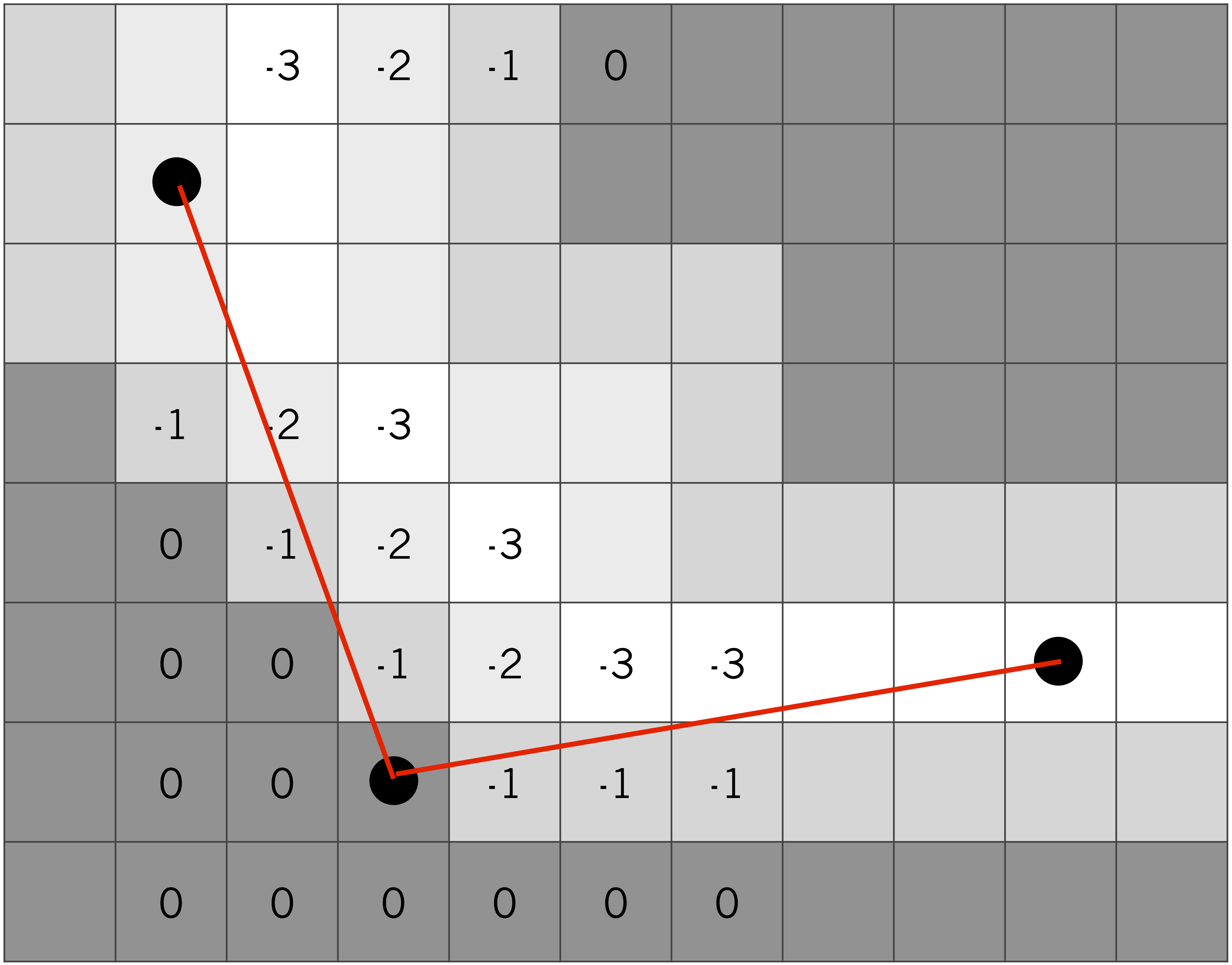
Snakes - implementation



Snakes - implementation



Snakes - implementation



$\alpha=0.1$

$\beta=0.05$

$\gamma=1$

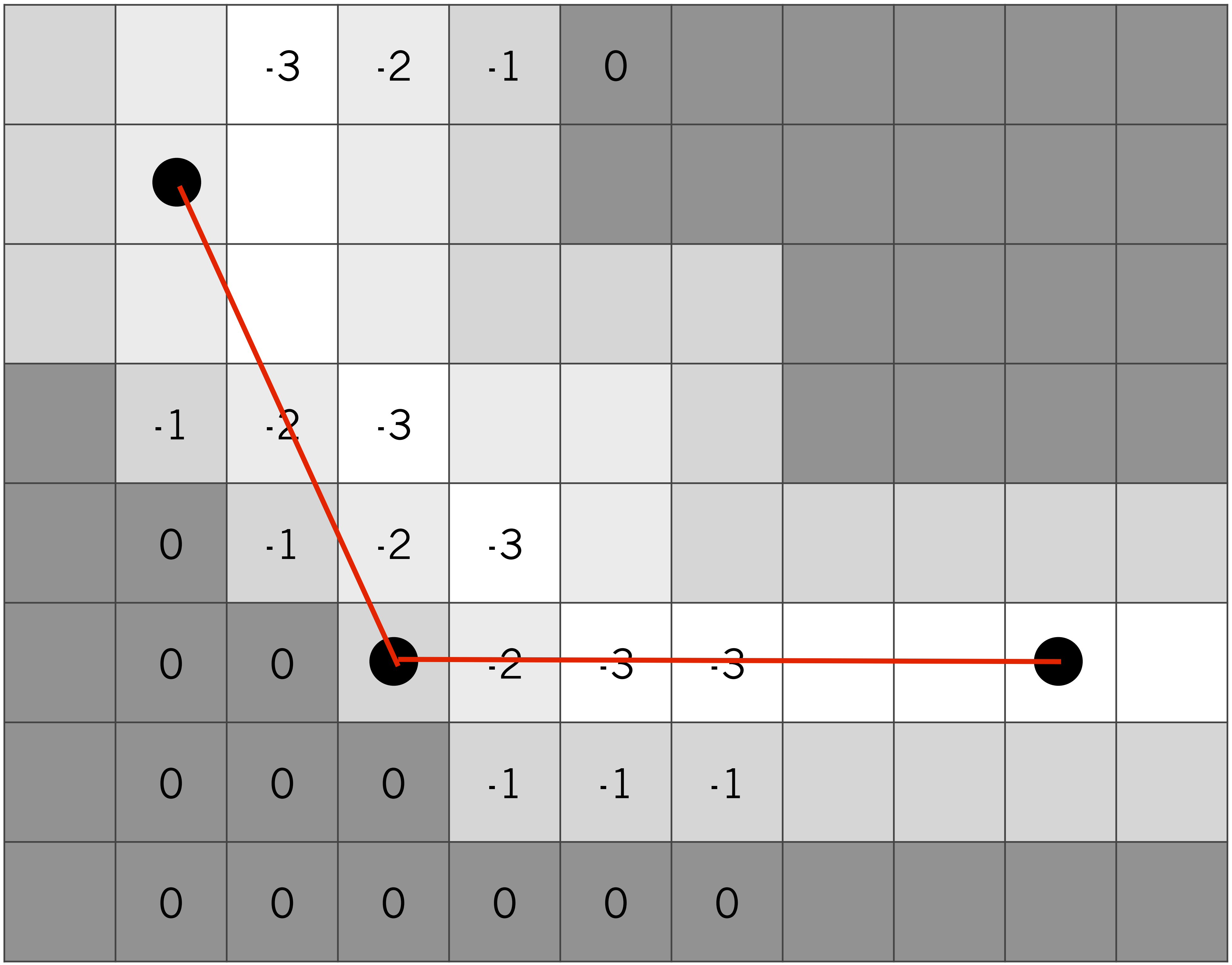
$E_{img} = 0$

$E_{cont}=(4-2)^2+(7-2)^2$
$=29$

$E_{cont}=(2-(2*4)+10)^2$
$+(2-(2*7)+6)^2$
$=52$

$E=\alpha*29+ \beta*52+ \gamma*0$
$= 5.5$

Snakes - implementation



$\alpha=0.1$

$\beta=0.05$

$\gamma=1$

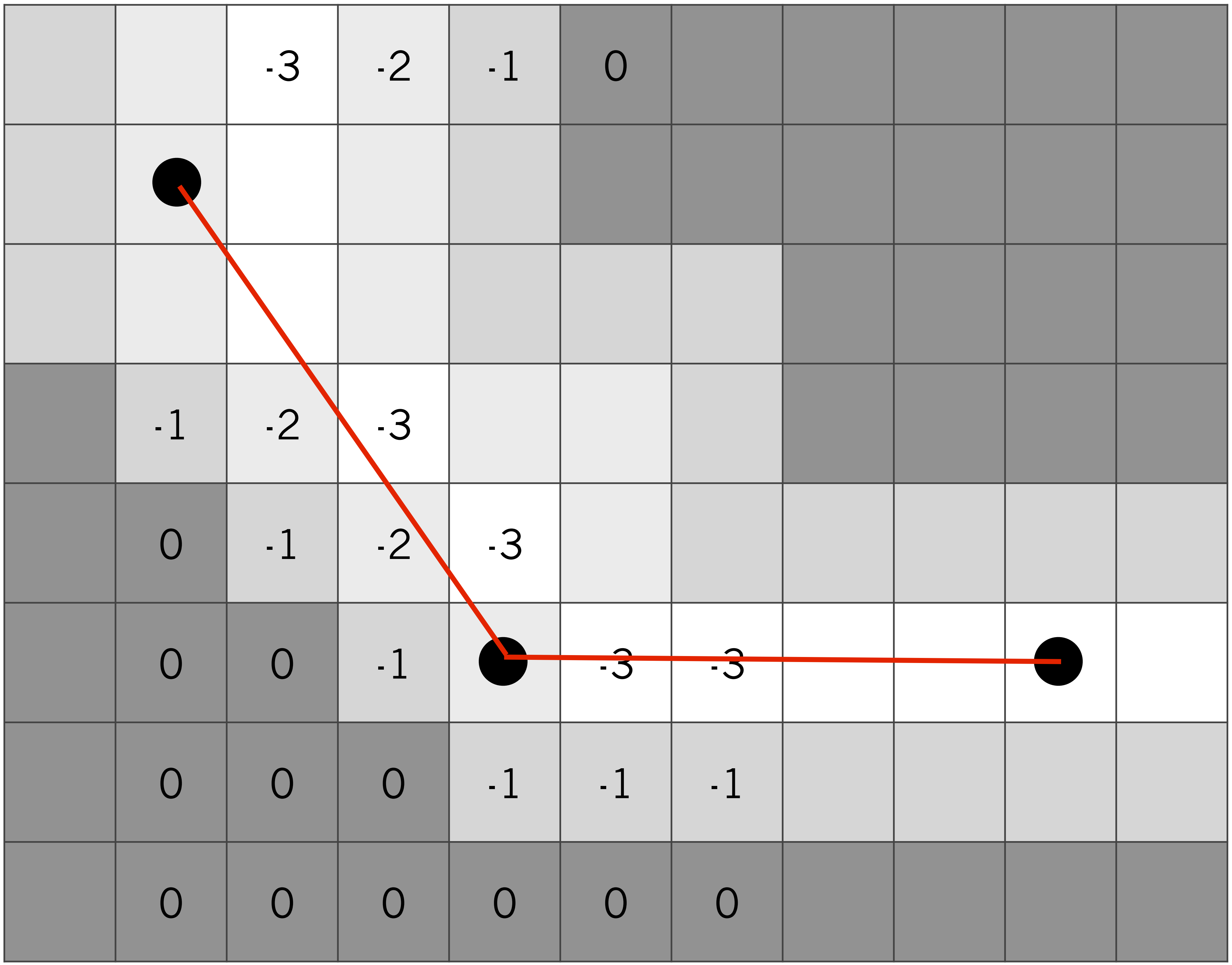
$E_{img} = -1$

$E_{cont}=(4-2)^2+(6-2)^2$
$=20$

$E_{cont}=(2-(2*4)+10)^2$
$+(2-(2*6)+6)^2$
$=32$

$E=\alpha*20+ \beta*32+ \gamma*-1$
$= 2.6$

Snakes - implementation



$\alpha=0.1$
$\beta=0.05$
$\gamma=1$

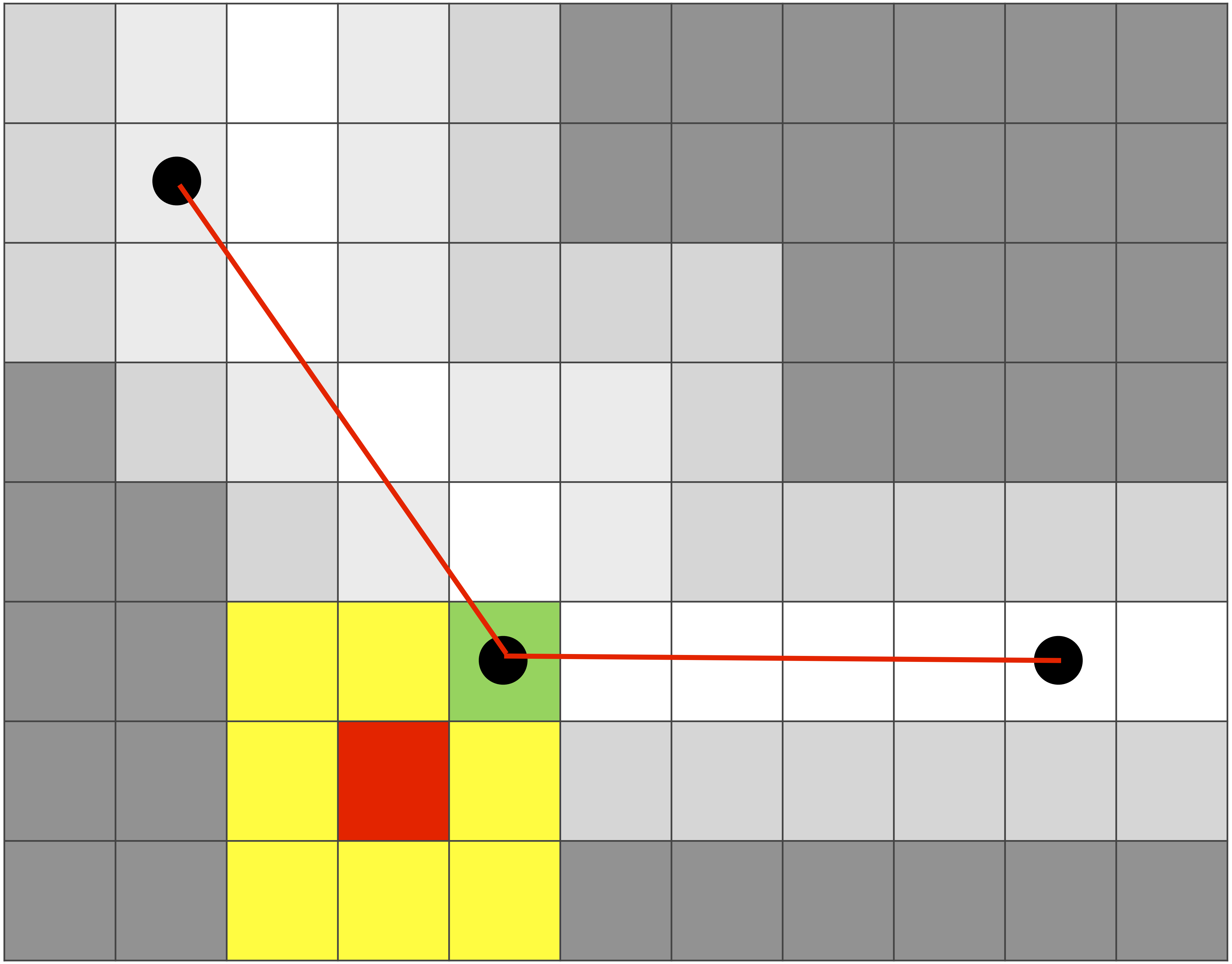
$E_{img} = -2$

$E_{cont}=(5-2)^2+(6-2)^2$
$=25$

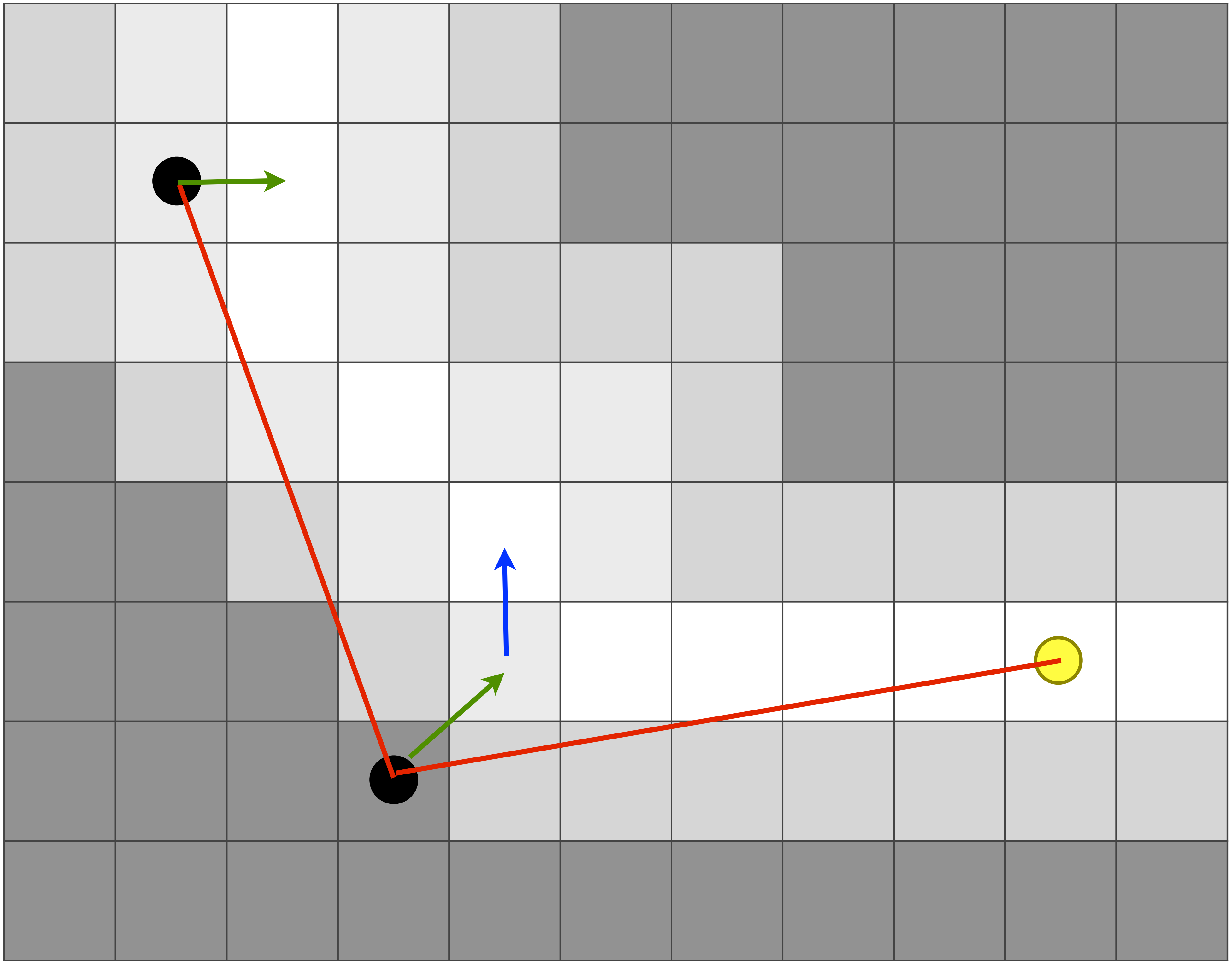
$E_{cont}=(2-(2*5)+10)^2$
$+(2-(2*6)+6)^2$
$=20$

$E=\alpha*25+ \beta*20+ \gamma*-2$
$= 1.5$

Snakes - implementation



Snakes - implementation



Snakes - implementation

- The energies should be normalised independently to reduce dependance on the parameters

$$E_{new}=(E_{old}-min)/(max-min)$$

- When normalising, verify if min!=max
  - change α_i and β_i to 0 of that happens
  - otherwise you will get ∞

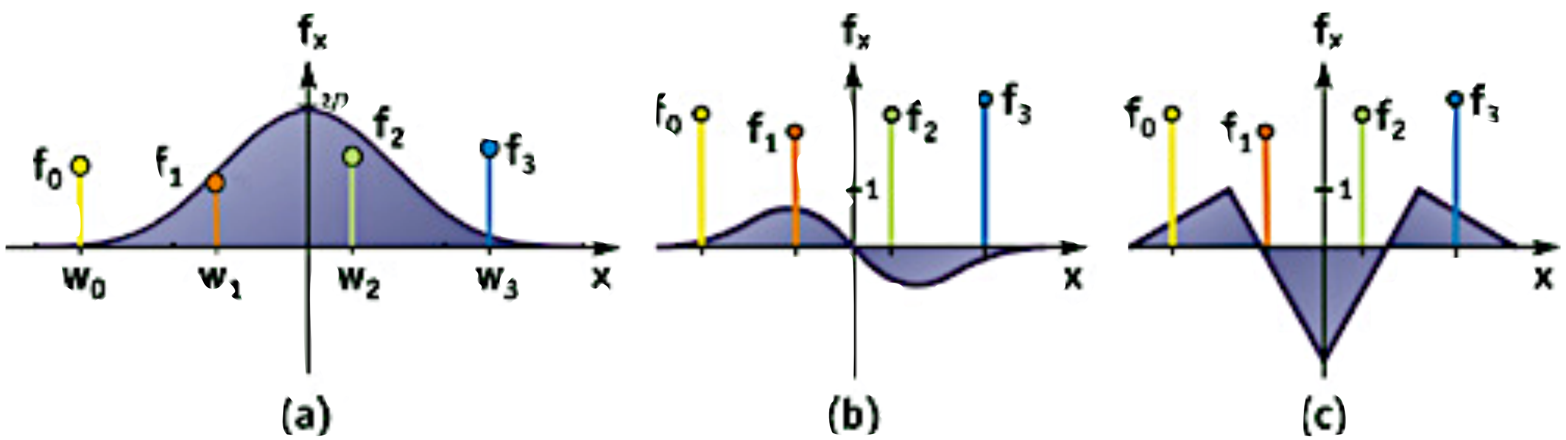
Parameterisation

- The points can "float" on the image
  - They don't need to be in the centre of the voxel

$$E = \int_0^1 \frac{1}{2} (\alpha |\boldsymbol{x}'(s)|^2 + \beta |\boldsymbol{x}''(s)|^2 + E_{\text{ext}}(\boldsymbol{x}(s)) \, \mathrm{d}s,$$

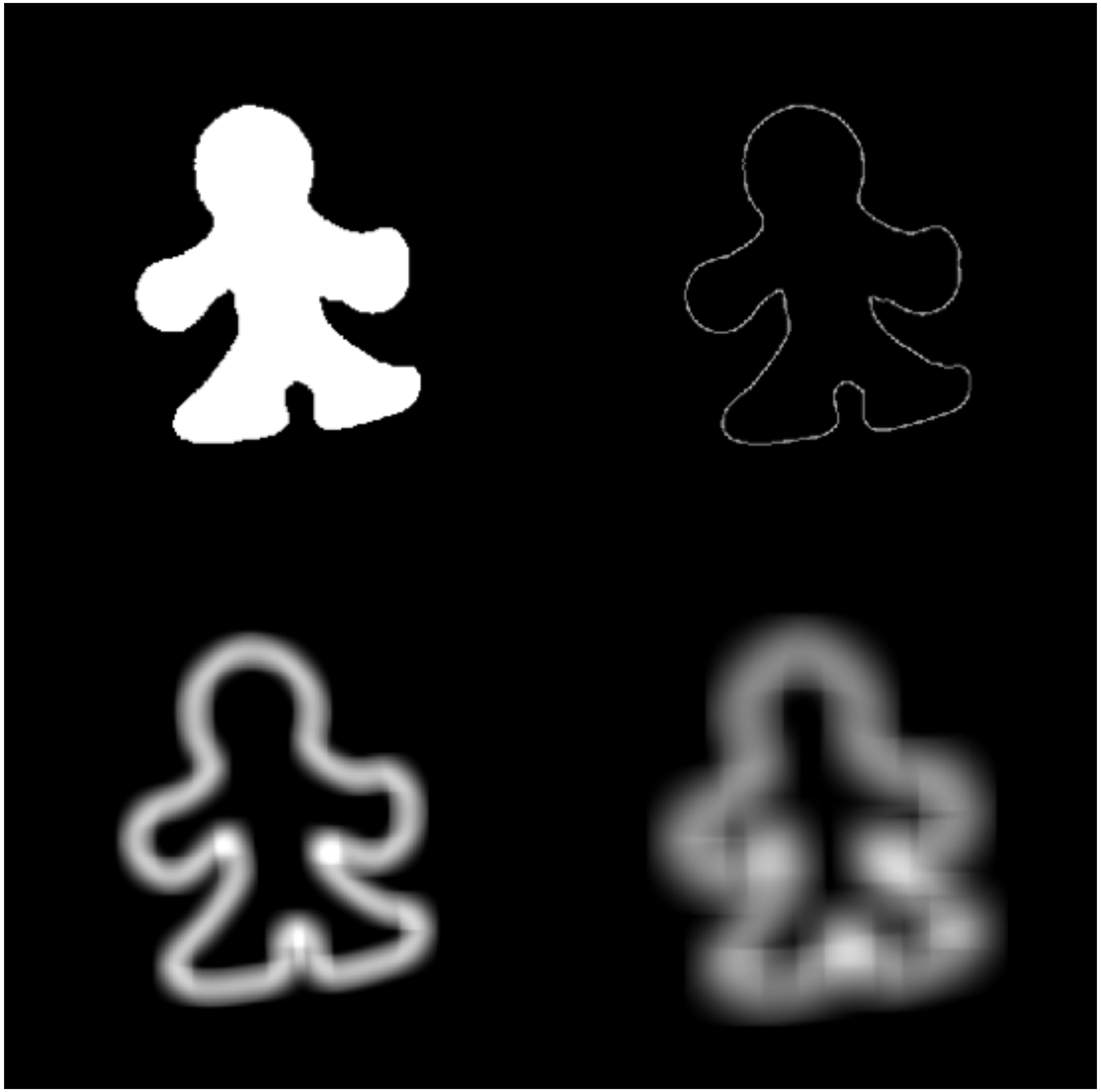
$$\boldsymbol{x}_t(s, t) = \alpha \, \boldsymbol{x}''(s, t) - \beta \, \boldsymbol{x}''''(s, t) - \nabla E_{\text{ext}}.$$

- Other types of parameterisation (B-splines)
  - Implicit first and second derivative



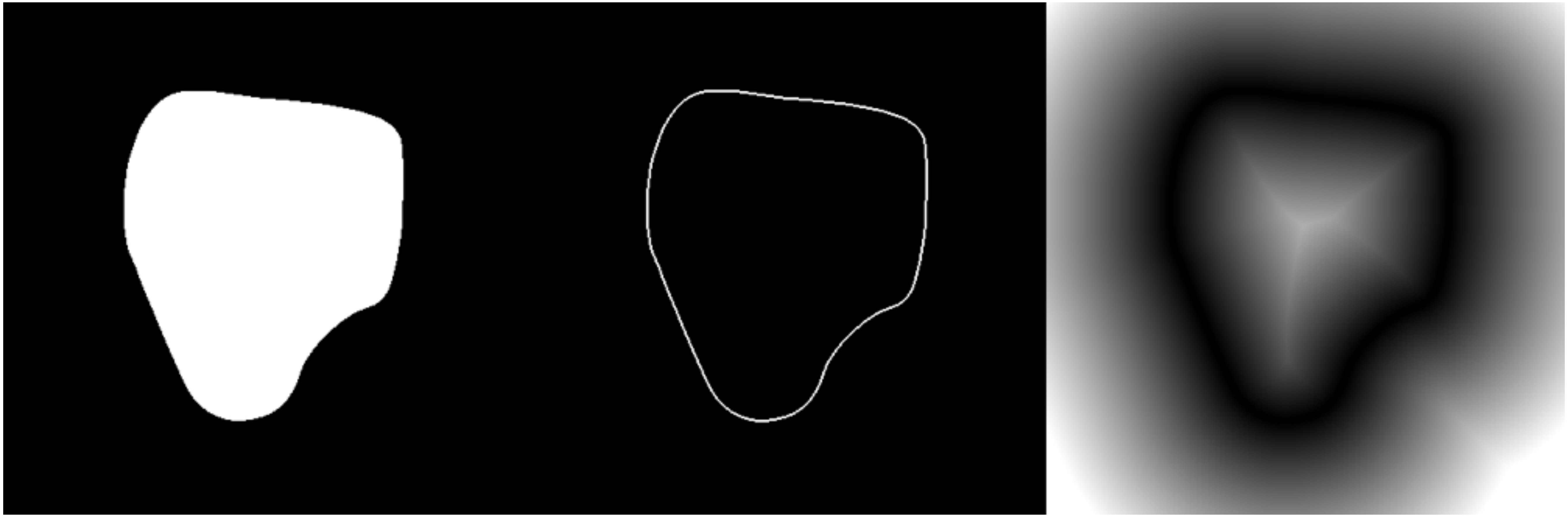
Gradient based forces

- Problem with cavities
- Point density dependant
- Initialisation dependant
- Unable to move into concavities

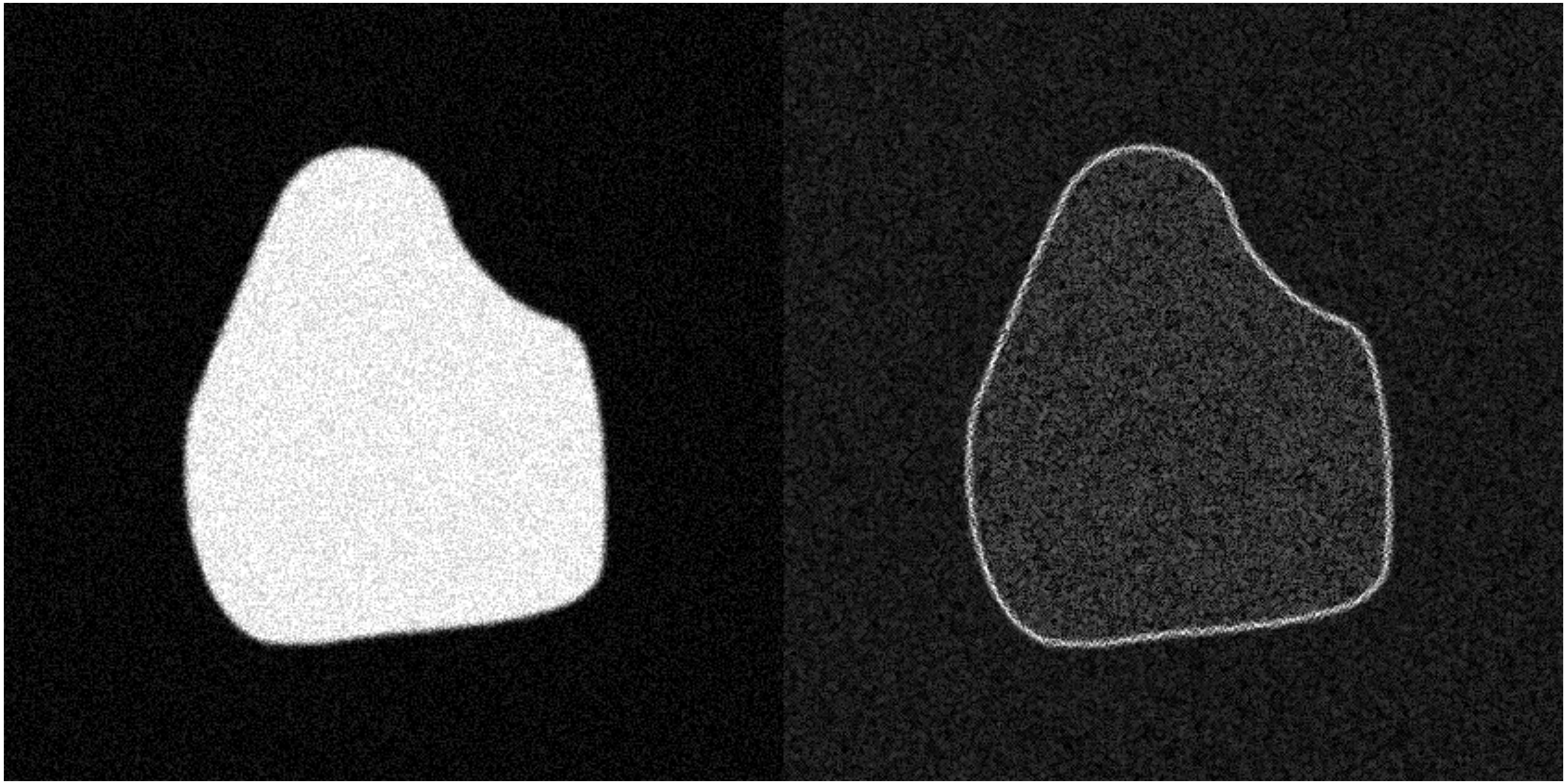


Distance potential forces

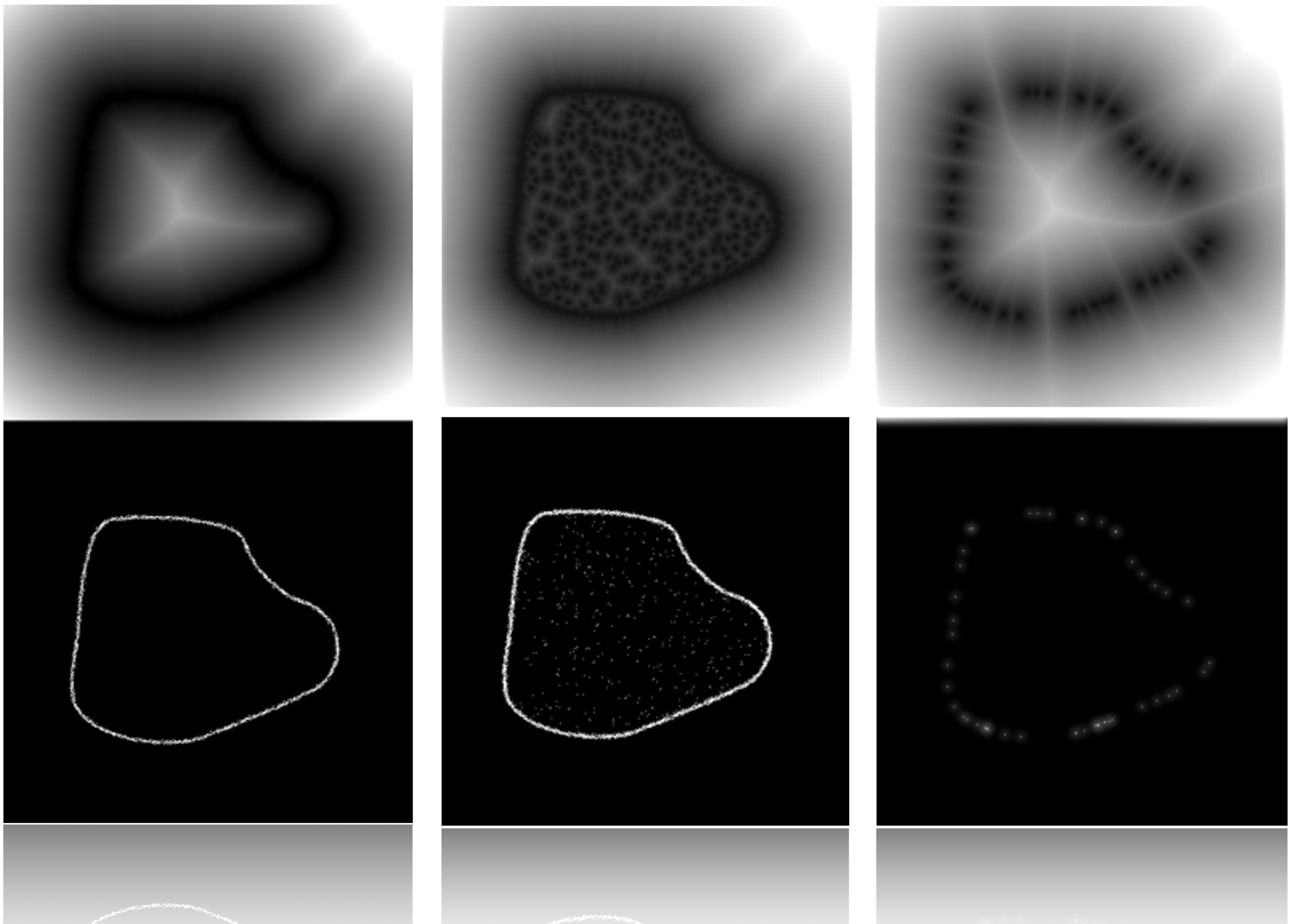
- Euclidean Distance Transform



Problem With Noise

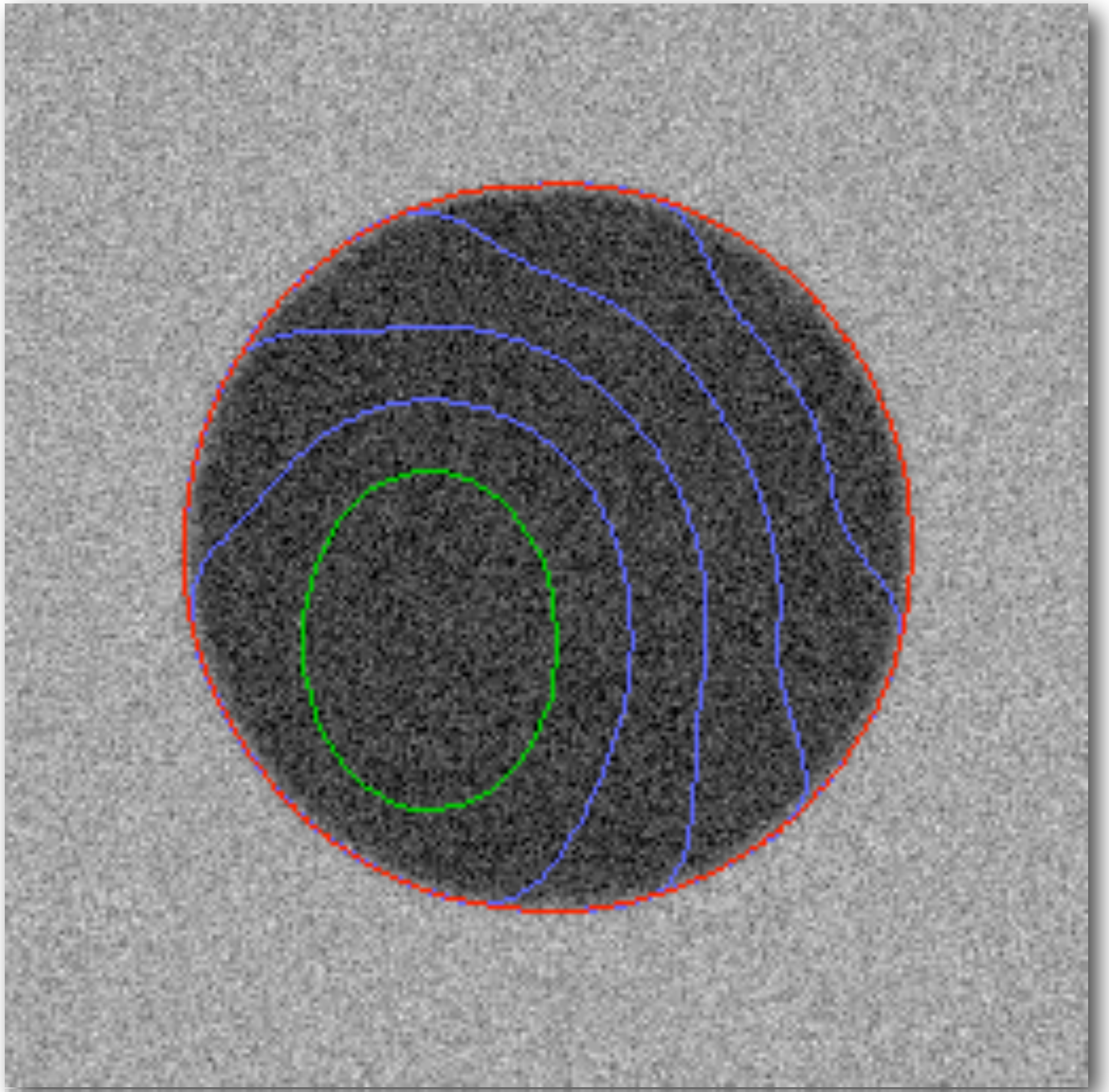


Problem With Noise



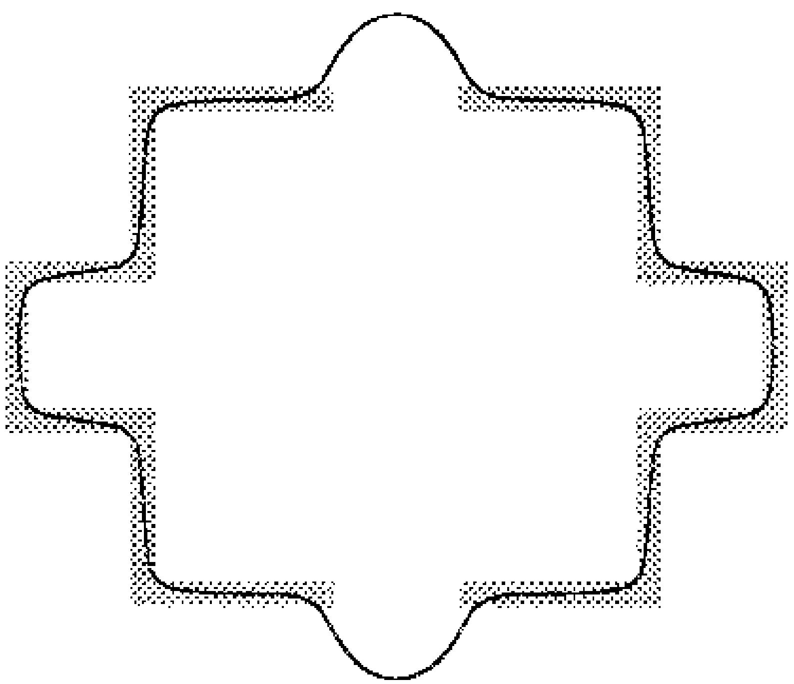
Balloon Forces

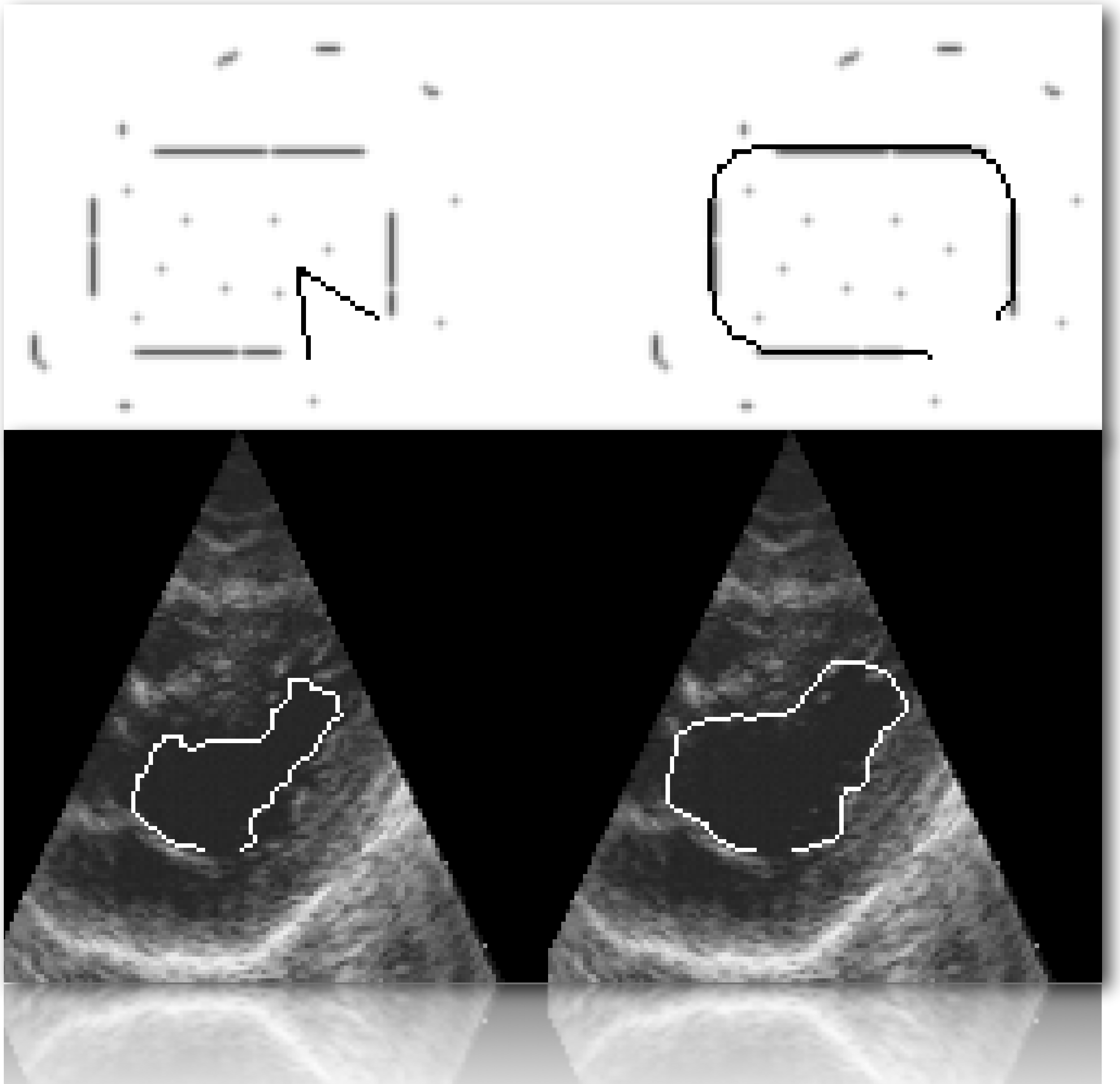
- Pressure forces in the normal direction
- Dynamic force: changes with the snake evolution



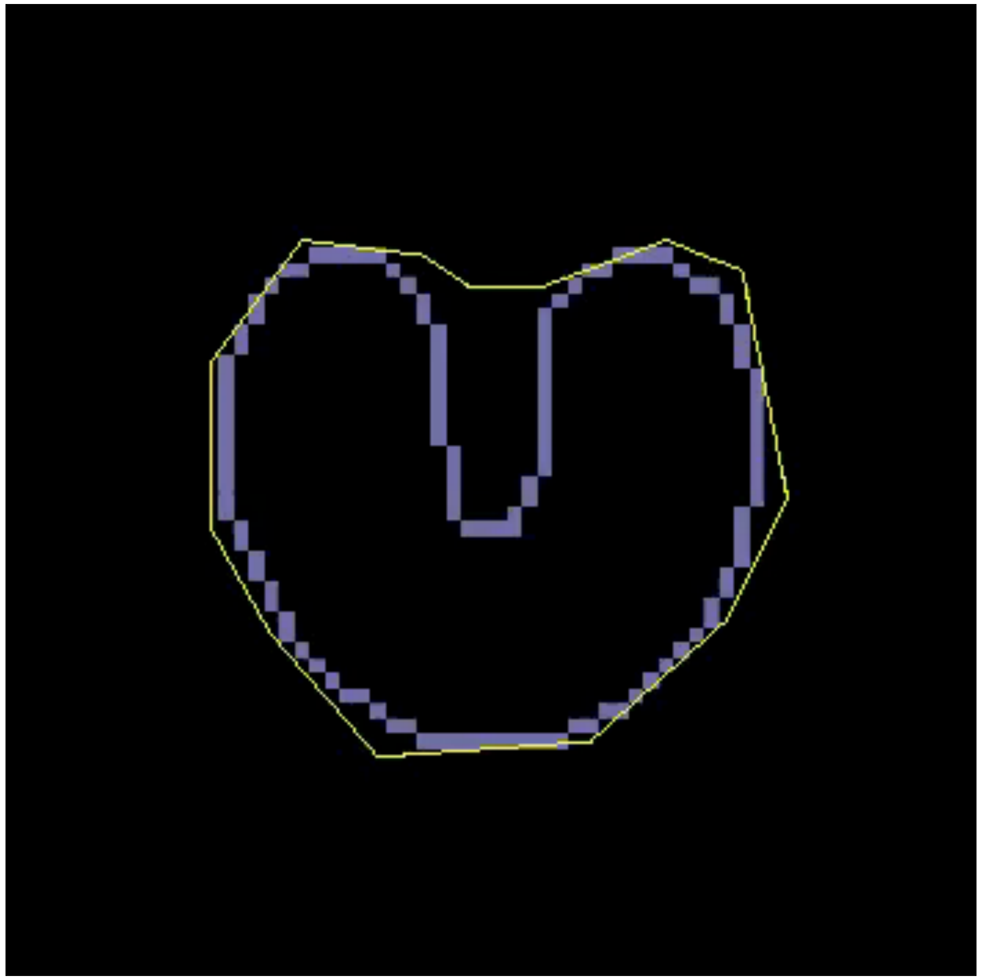
Opened Balloon Forces

- Active contour or with 2 fixed points
- Has to distinguish inside from outside
- Can handle cavities



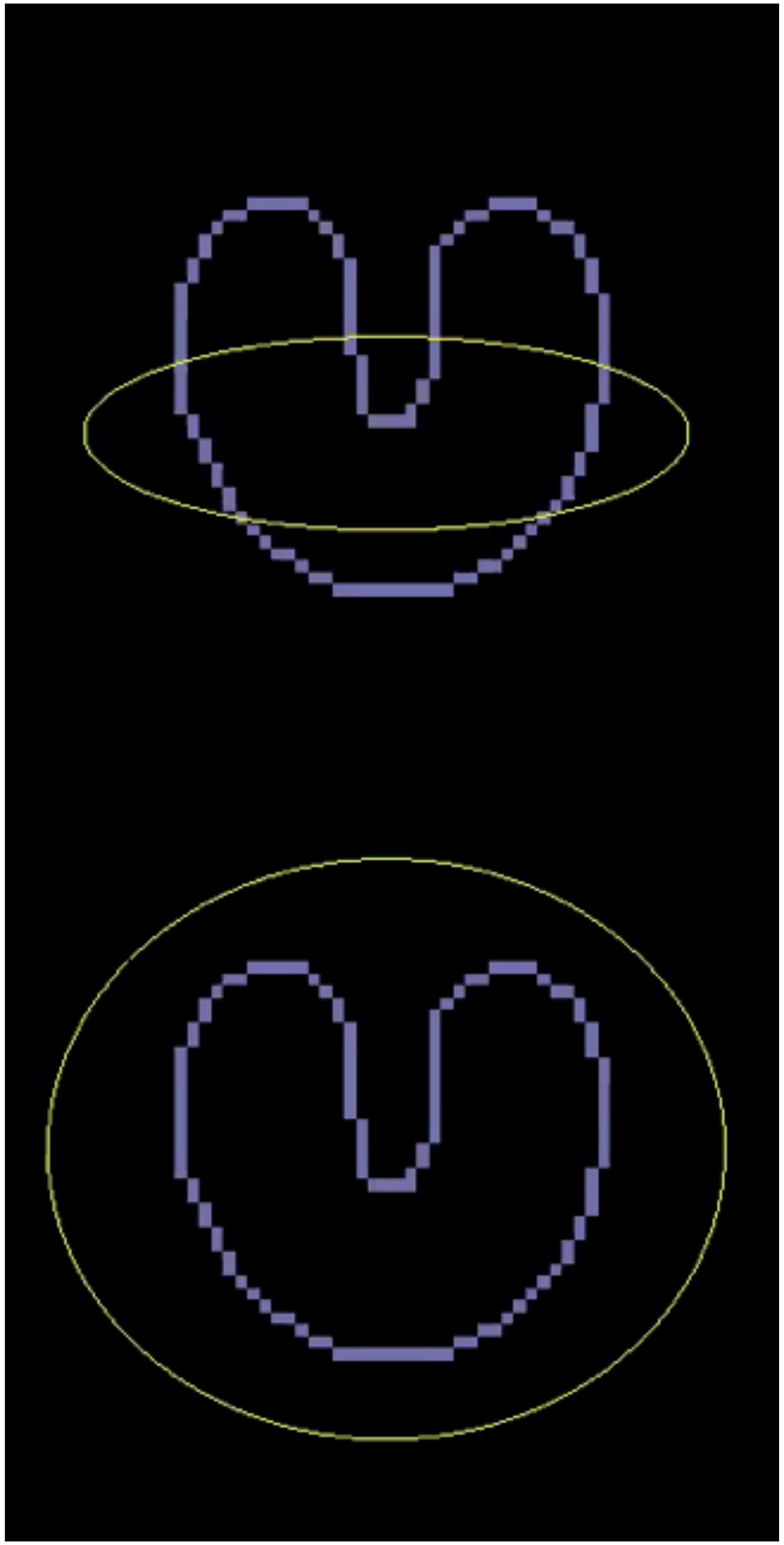


Problems with Cavities



Try different metrics...

- Multi-resolution methods,
- Pressure forces,
- Distance potential forces,
- Number of control points
- GVF (Gradient Vector Flow)



GVF

- Improved performance and capture range
- Almost no problems with cavities
- Based in a diffusion like process
- Find the vector field is the one that minimises

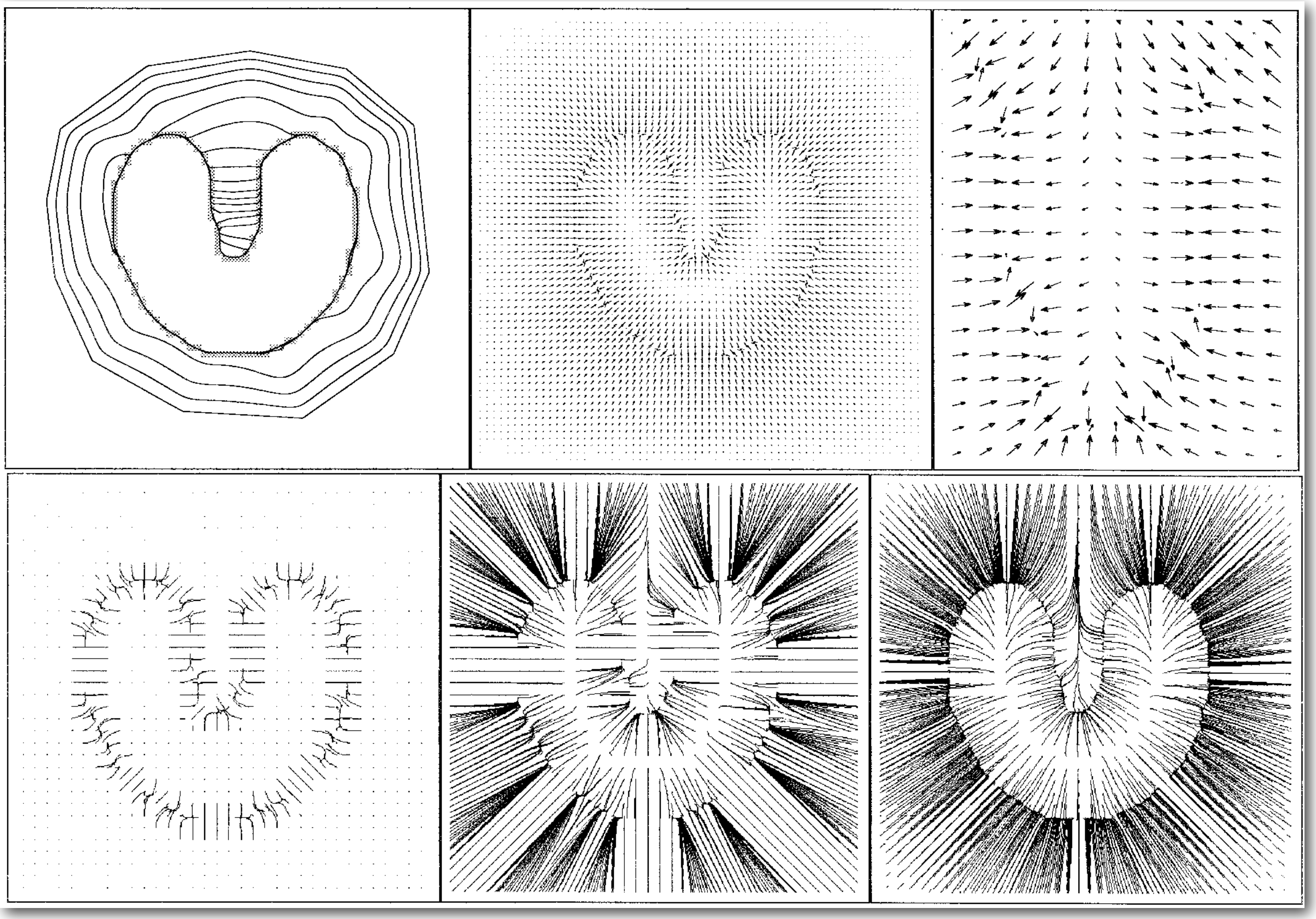
$$\mathcal{E} = \iint \mu({u_x}^2 + {u_y}^2 + {v_x}^2 + {v_y}^2) + |\nabla f|^2 |\mathbf{v} - \nabla f|^2 \, dx dy.$$

with $\mathbf{v}(x, y) = [u(x, y), v(x, y)]$, $f(x, y) = -E^{(i)}_{\text{ext}}(x, y)$ and μ as a weighting parameter (increase μ with noise)

GVF - The idea

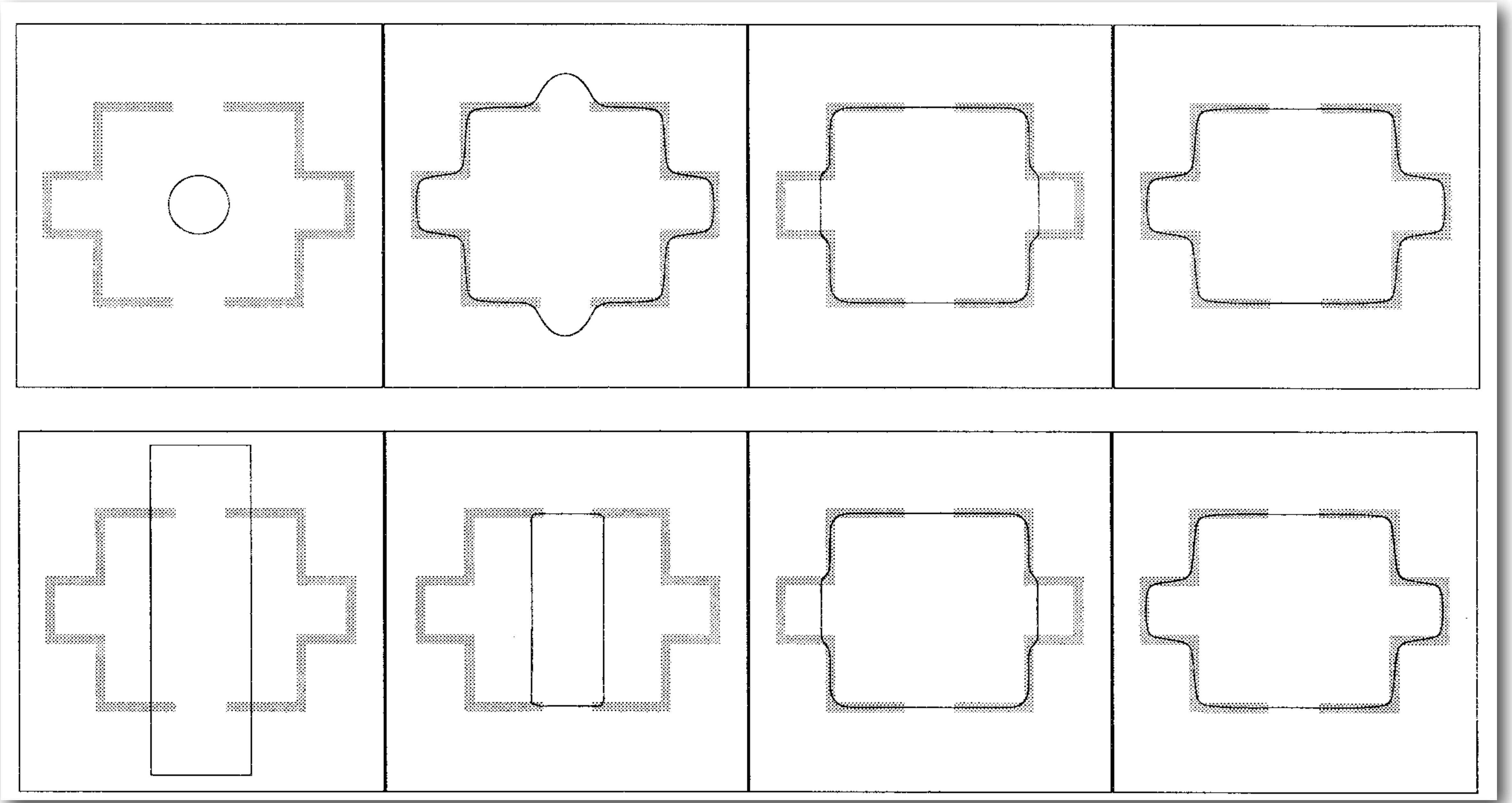
- Maintain the nice properties of gradient based metrics
  - Gradient of an edge map has vectors pointing toward the edges
  - This gradient is normal to the edges at the edges (if ∇f is large, then the energy will be minimal if $v = \nabla f$)
- Remove the bad properties
  - these vectors generally have large magnitudes only in the immediate vicinity of the edges
  - in homogeneous regions, where the intensity is nearly constant, the vector field is nearly zero

$$\mathcal{E} = \iint \mu({u_x}^2 + {u_y}^2 + {v_x}^2 + {v_y}^2) + |\nabla f|^2 |\mathbf{v} - \nabla f|^2 \, dx dy.$$



Edge Gradient Distance Transform Gradient GVF

GVF



Questions??