Level sets and graph-cuts (Deformable models)

Centro de Visión por Computador,
Departament de Matemàtica Aplicada i Anàlisi,
Universitat de Barcelona

The problems of Medical image analysis vs. Computer Vision

Segmentation

Object recognition

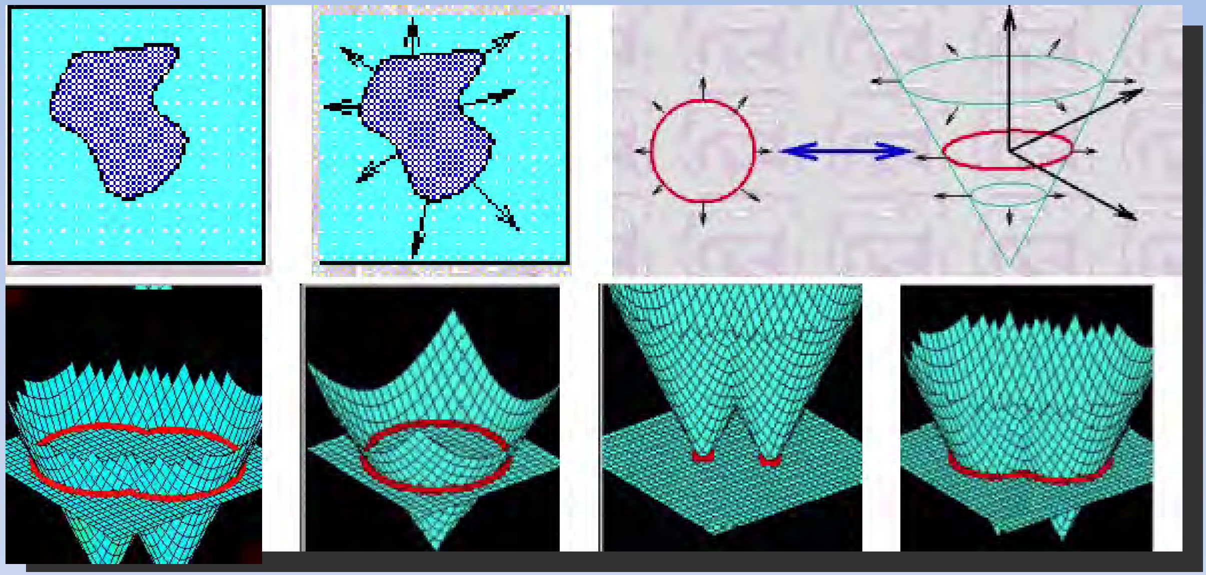
Atlas matching

Registration

3D reconstruction

Deformation/Motion analysis

Geodesic Active Contours (Level Sets)



Numerical techniques to track interface evolution between different regions

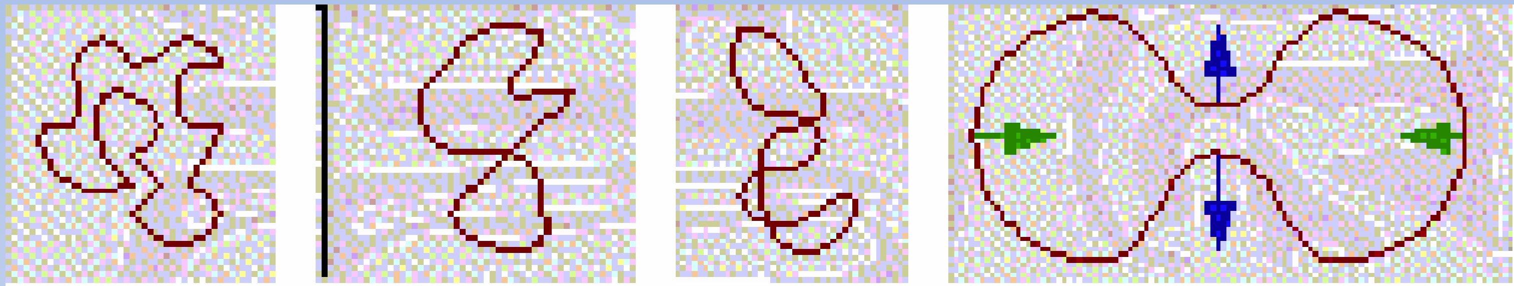
General time-dependent level set method

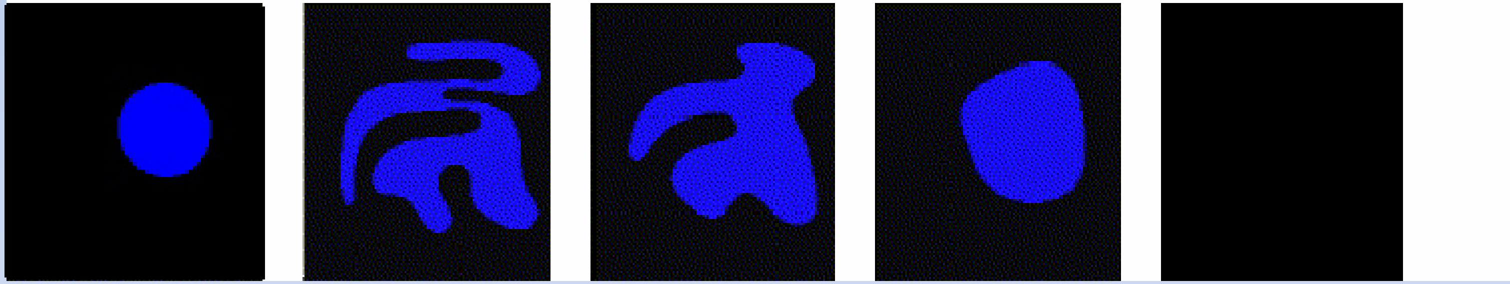
Tracking the moving boundary by the level set approach (reprint from J.A.Sethian)

- The motion of the interface is matched with the zero level set of a level set function
  - The resulting initial value partial differential equation for the evolution of the level set function resembles a Hamilton-Jacobi equation.

From J.A.Sethian

Motion under Curvature


Simple and not simple closed curves (reprint from J.A.Sethian)



Curve collapsing under its curvature (reprint from J.A.Sethian)

Theorem in differential geometry:

- Any simple closed curve moving under its curvature collapses nicely to a circle and then dissapears.

Geodesic Active Contours

The energy function:

$$E(Q) = \alpha \int_0^1 |Q'(u)|^2 du + \int_0^1 g(|\bigtriangledown I(Q(u))|^2) du$$

can be represented as:

$$L_R = \int_0^{L(Q)} g(|\bigtriangledown I(Q(u))|) ds$$

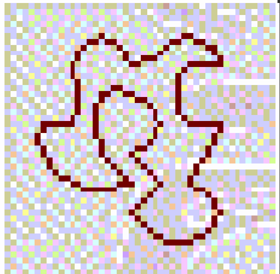
where s is the Euclidean arc-length parameter and g is a function of the image.

We are looking for **a path (curve) with minimal length definition weighted by an image component.**

Evolution equation: applying Euler-Lagrange of L_R to deform $Q(0) = Q_0$ towards a minimum of L_R:

$$\frac{\delta Q}{\delta t} = g(I)k\vec{n} - (\bigtriangledown g, \vec{n})\vec{n}$$

where k is the curvature and $\vec{n}$ is the unit inward normal.



The Level Sets Geodesic Flow

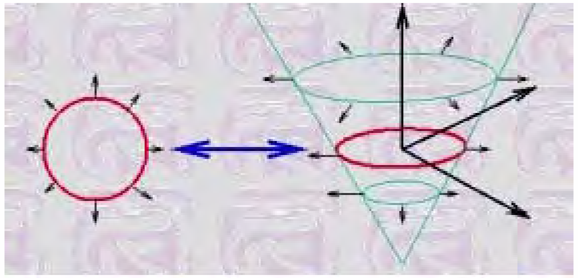
$$\frac{\delta Q}{\delta t} = g(I)k\vec{n} - (\nabla g, \vec{n})\vec{n}$$

$$\frac{\delta \phi}{\delta t} = \quad |\nabla \phi| div\left(g(I)\frac{\nabla \phi}{|\nabla \phi|}\right) + cg(I)\,|\nabla \phi| =$$
$$g|\nabla \phi| div\left(\frac{\nabla \phi}{|\nabla \phi|}\right) + (\nabla g, \nabla \phi) + cg(I)|\nabla \phi|$$

i.e. the level sets move according to: $Q_t = g(I)(c+k)\vec{n} - (\nabla g, \vec{n})\vec{n}$

Stopping criteria:

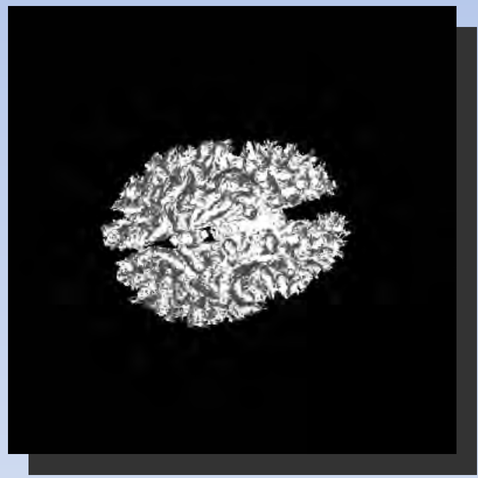
$$g = \frac{1}{1 + |G * \nabla I|^p}$$

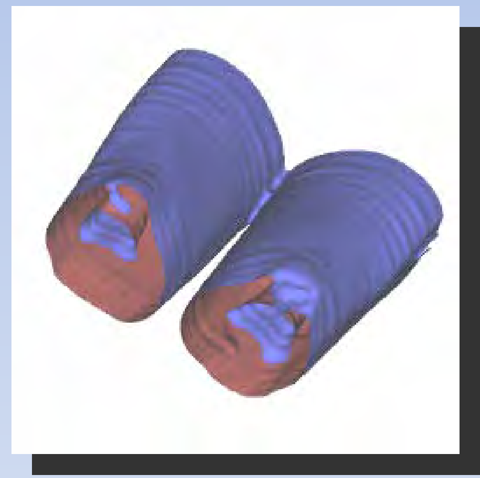


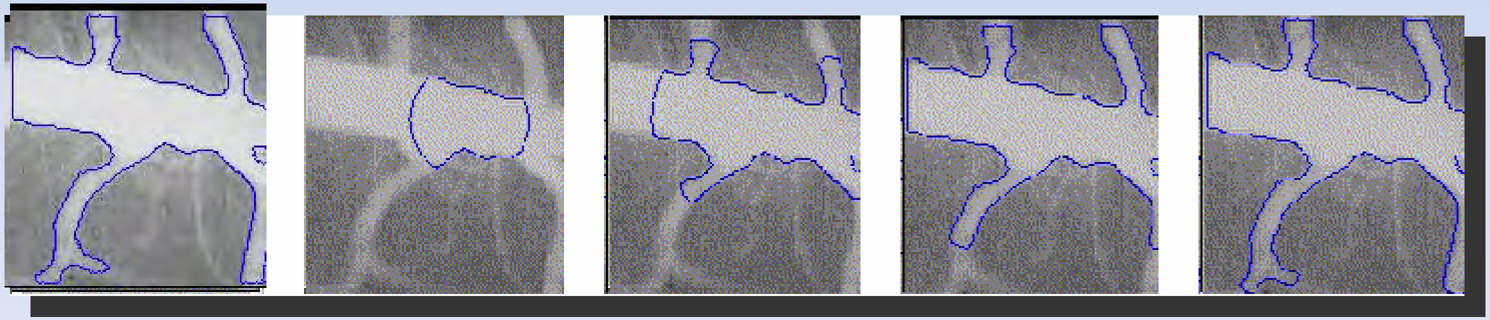
Level Set Methods for Shape Recovery

The key idea:

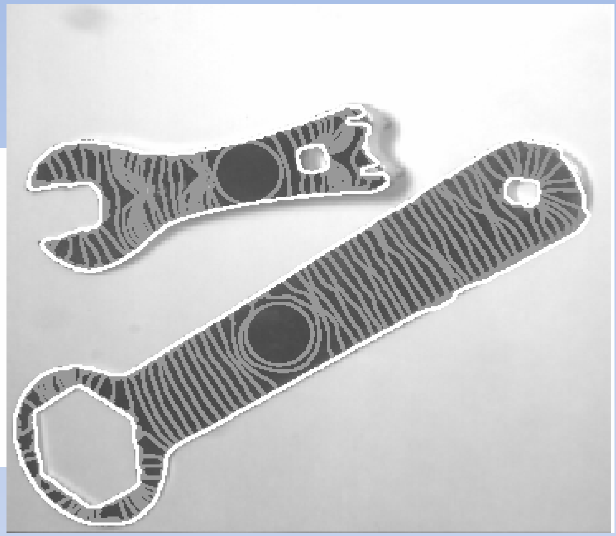
- to evolve the curve outwards with a speed depending of the curvature and the image
  - quickly expand when passing over places with small image gradient
  - slow down when crossing large image gradient places







Segmentation by Geodesic Snakes



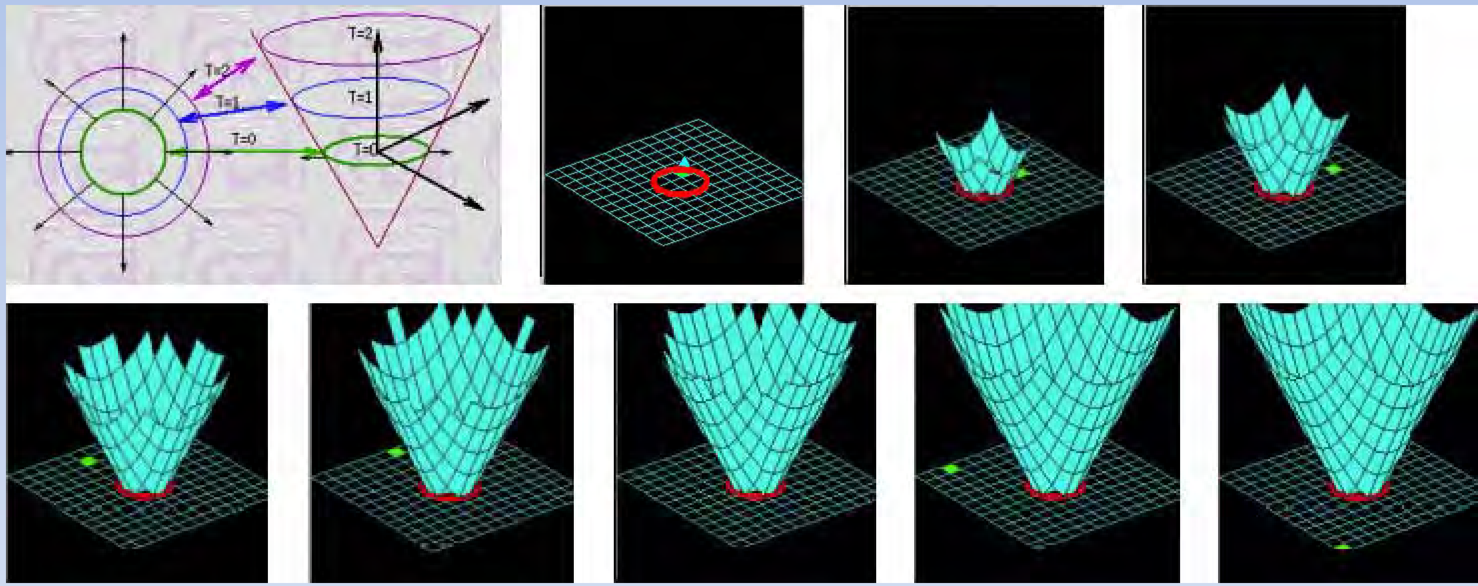
Outward motion to detect close objects. The initial contour is given by the image frame

Advantages

- The level set approach allows the evolving front to change topology, break and merge.
- Almost no change in case of surface extraction.
- Existance, uniqueness, stability and convergence of the solution of evolution equation are proved.

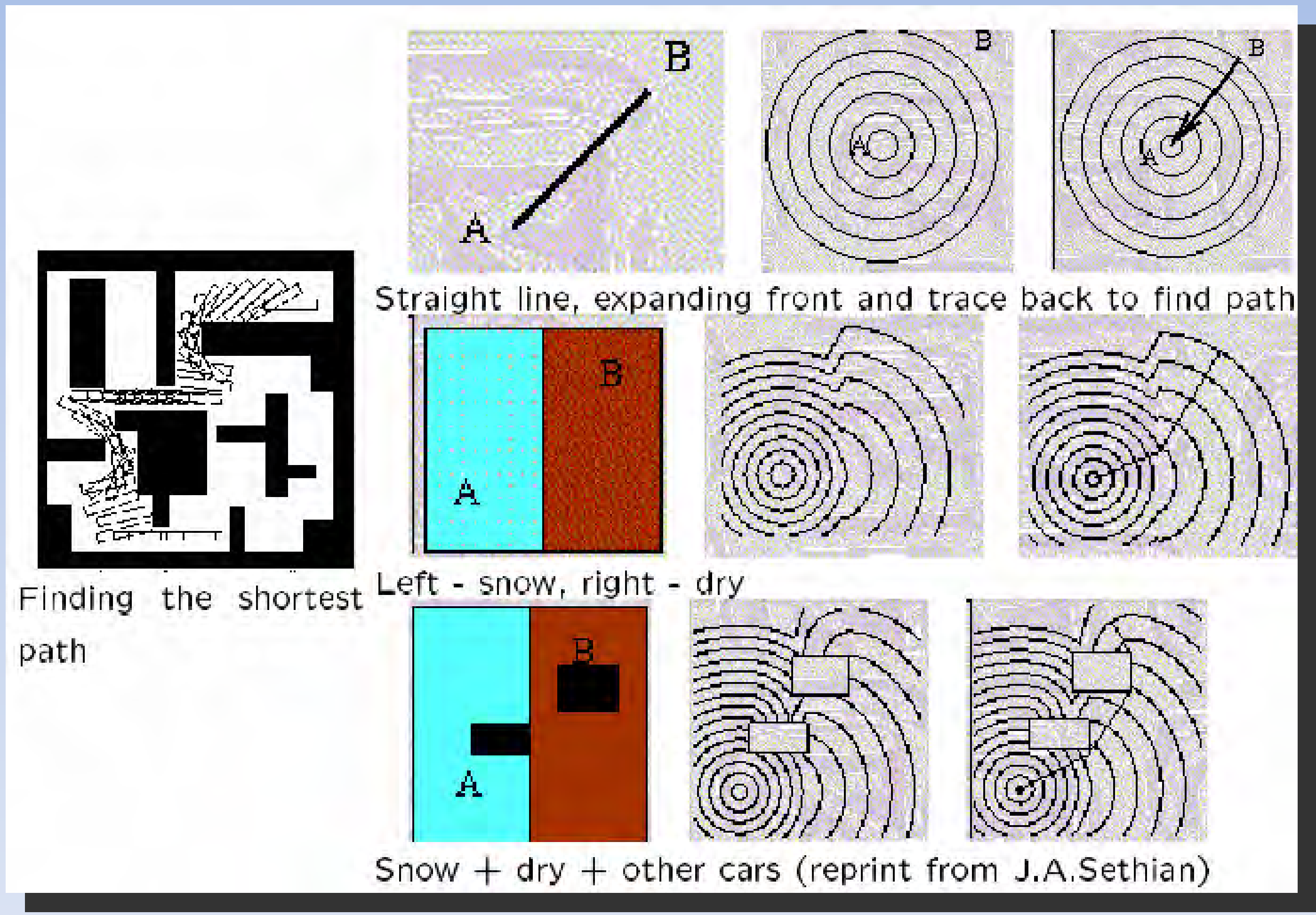
From V. Casselles

The Fast Marching Method



- Construction of stationary level set solution (reprint from J.A.Sethian)
  - The fast marching level set method solves the general static Hamilton-Jacobi equation applied to a convex non-negative speed function.

Robotic Navigation with Constraints



The Fast Marching for the Global Minimum of Active Contours

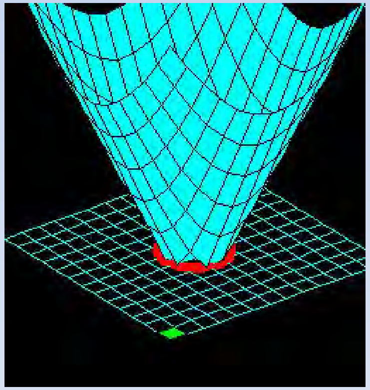
Paths of minimal action

$$U(p) = inf_{Q_{L\_p}}\{\int_Q P(Q) + \alpha|\frac{\delta Q}{\delta s}|^2 ds\}$$

define a *surface of minimal action* U starting at $p_0 = Q(0)$. The value of U is the minimal energy integrated along a path starting at p_0 and ending at p.

Left - snow, right - dry

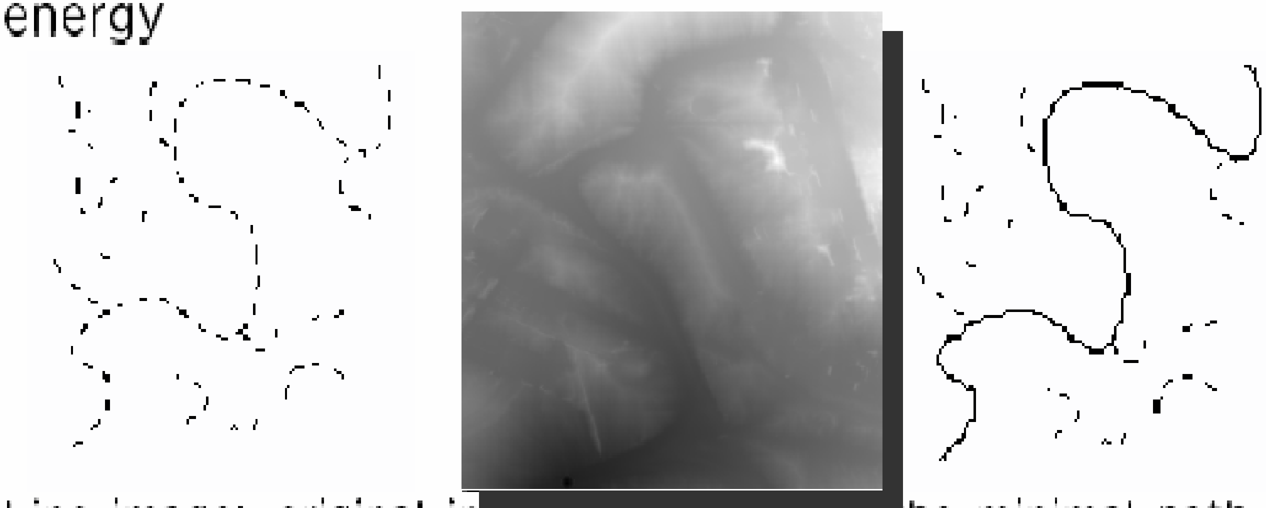
Snow + dry + other cars (reprint from J.A.Sethian)



The Fast Marching for the Global Minimum of Active Contours

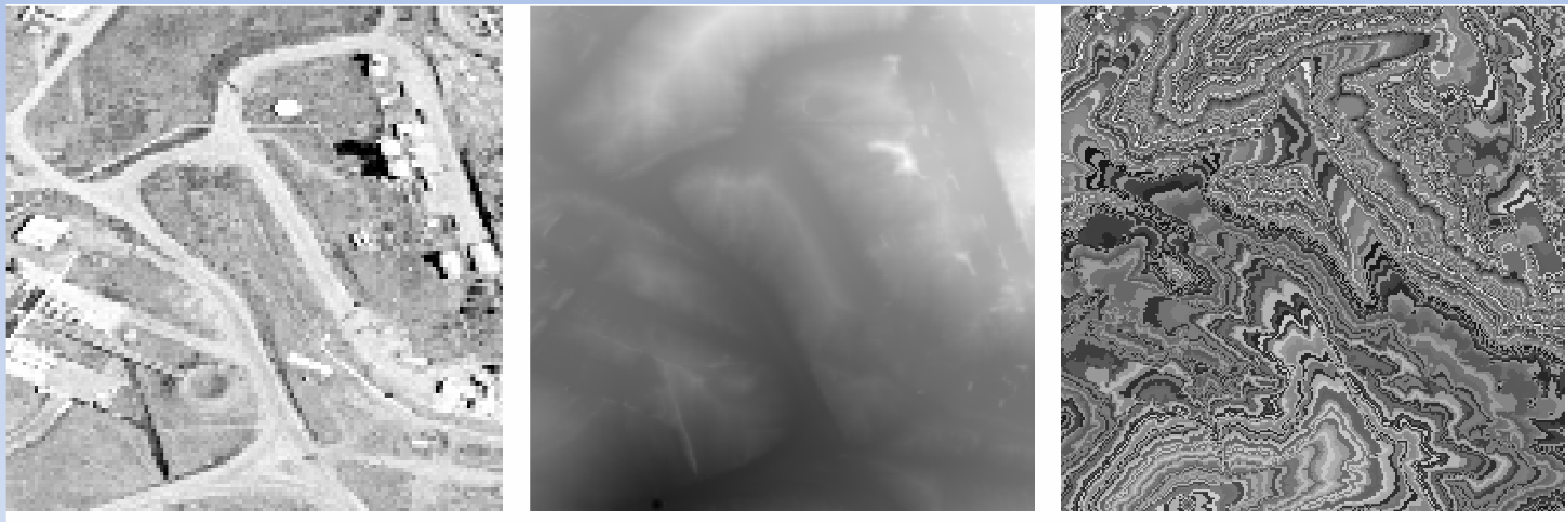
The minimal action level sets evolution: $\frac{\delta L(r,t)}{\delta t} = \frac{1}{P(Q)+\alpha|Q'|}\vec{n}(r,t)$

- describes the set of equal energy contours L in 'time' where t is the value of the energy: $\{L(r,t), r \in I\} = \{p \in R^2 | U_0(p) = t\}$, t - value of the energy



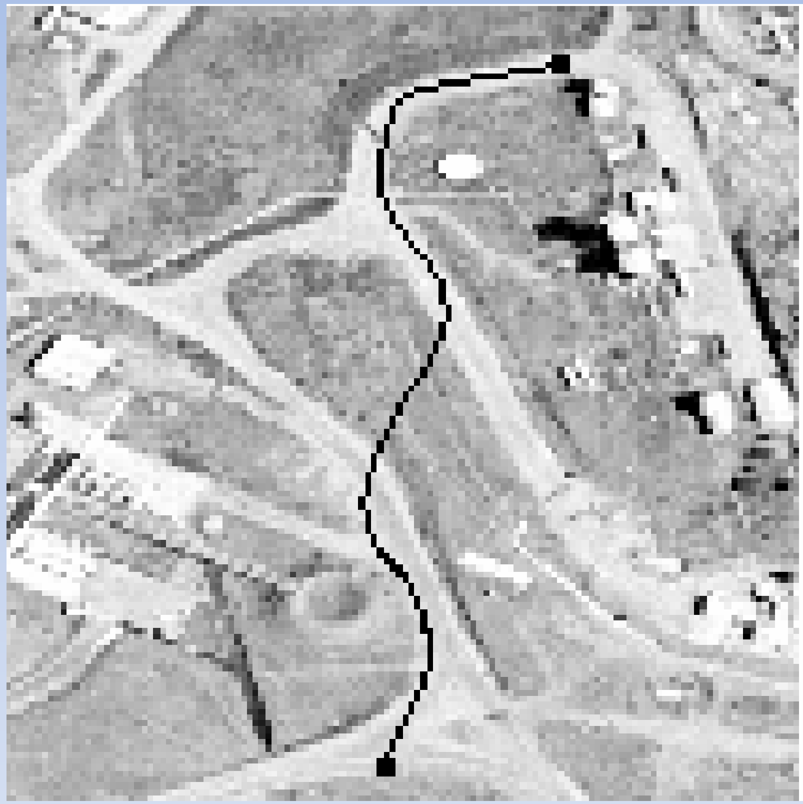
Line image: original image potential and the minimal path (reprint from L. Cohen, 1996)

Geodesic snakes



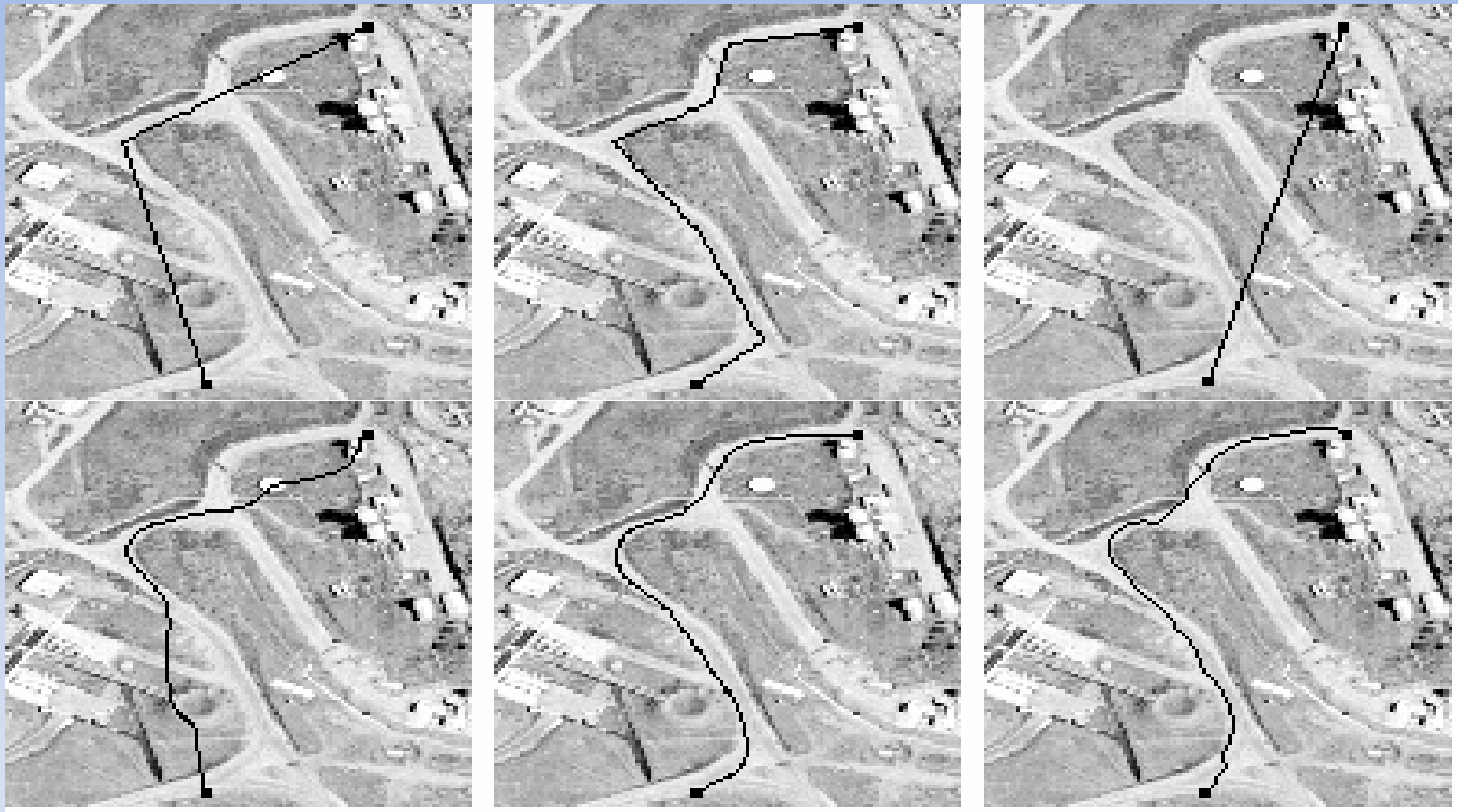
Original image and minimal action surface (in grey levels and rendered level sets)

Classical snakes



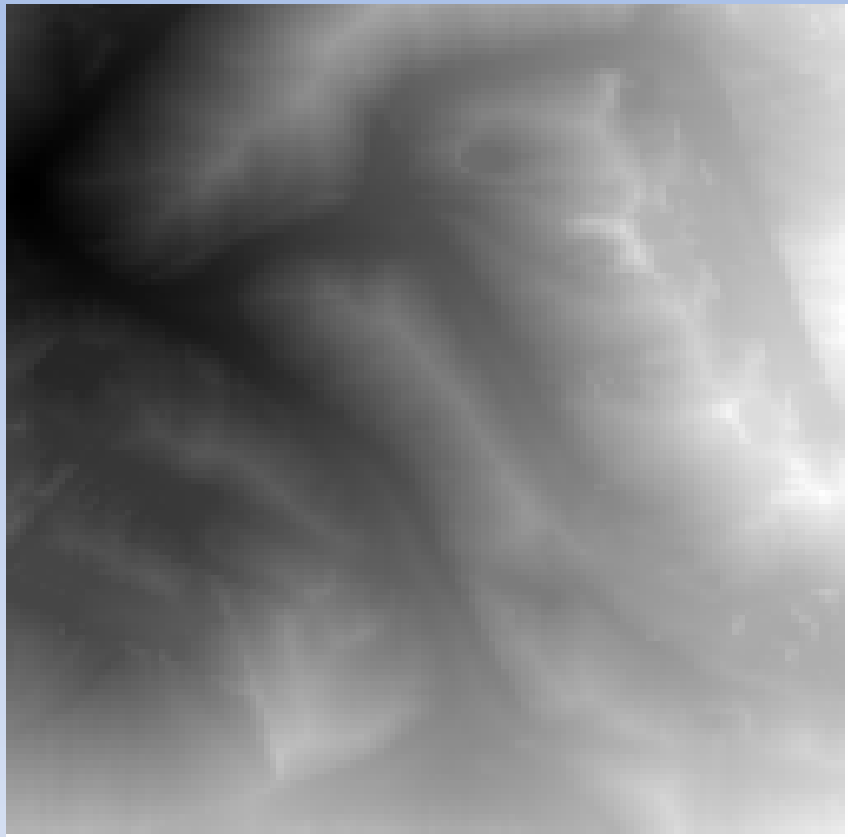
Initial snake and segmentation result by classic snakes

Geodesic snakes



Initial snake and segmentation result by classic and geodesic snakes

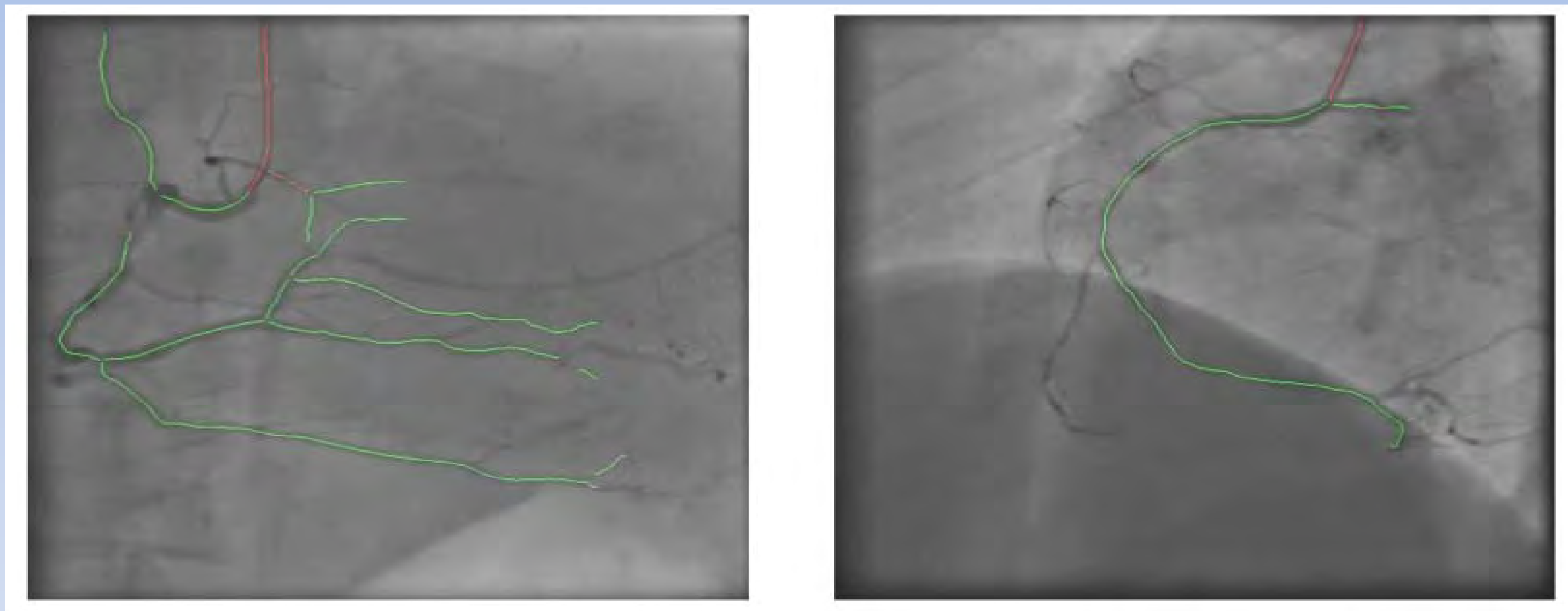
Multiple solutions of segmentation by geodesic snakes



Minimal path between multiple points (reprint from L. Cohen, 1996)

Application of Fast Marching for Artery detection in Angiograms

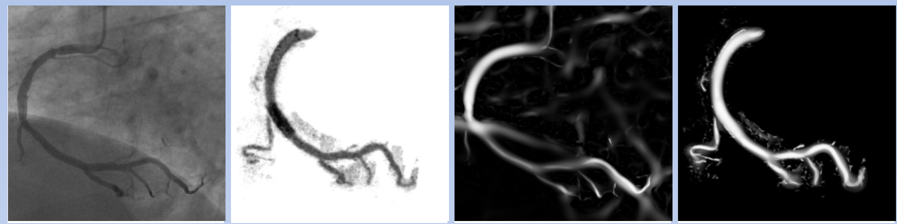
Accurate segmentation of 2D fluoroscopy and catheter guide detection



Pixels detected as catheter are shown in red. Green lines represent the vessel's centerlines as delineated automatically by the proposed method.

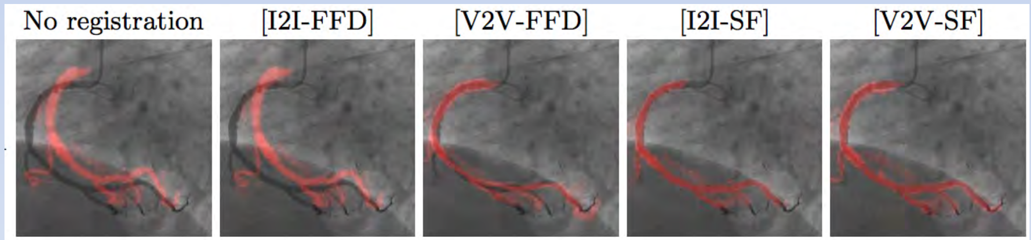
Application of Fast Marching for Artery detection in Angiograms

<u>Development of physics-based registration techniques for 3D to 2D registration based on projective geometry</u>



(a) (b) (c) (d)

An X-Ray image (a), a simulated X-Ray image using the segmented 3D CTA coronary artery (b), and the respective Vesselness maps (c-d).

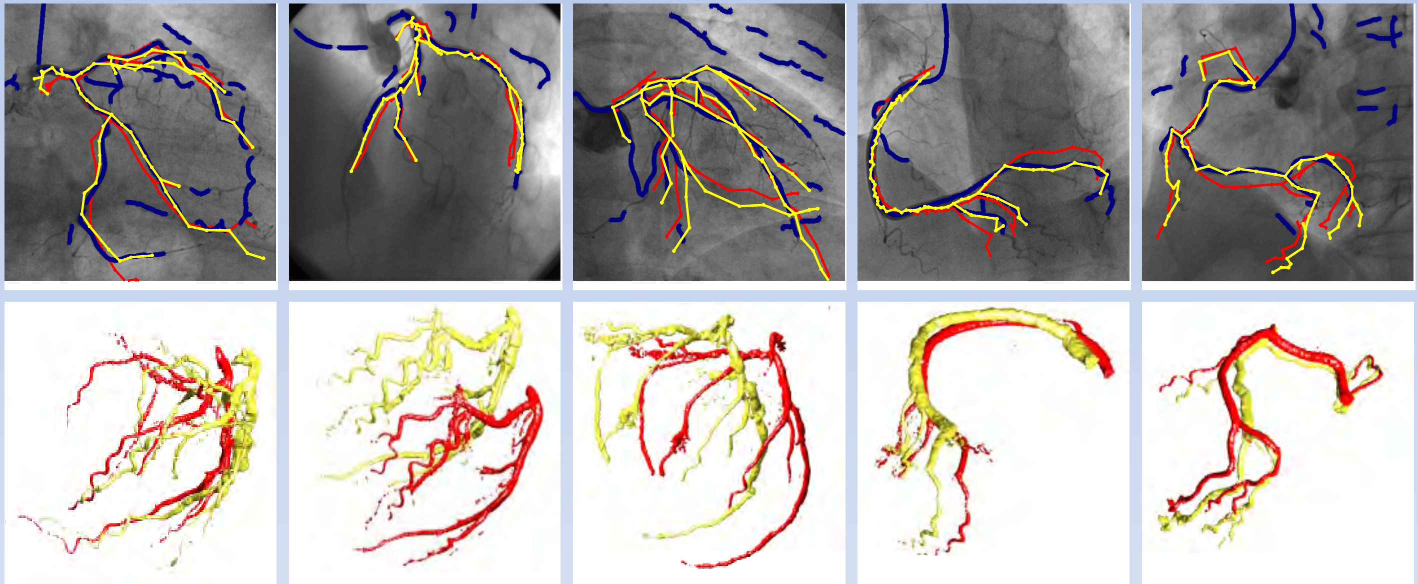


An example of successful non-rigid registration using several different pre-processing steps on the fluoroscopic image prior to registration. The red artery is the result of projecting the segmented 3D CT data.

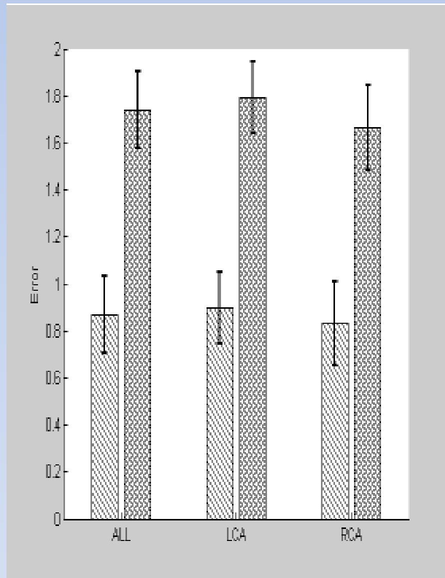
Application of Fast Marching for Artery detection in Angiograms

Development of physics-based registration techniques for 3D to 2D registration based on projective geometry

Left Coronary Artery **Right Coronary Artery**



2D Reprojection Error



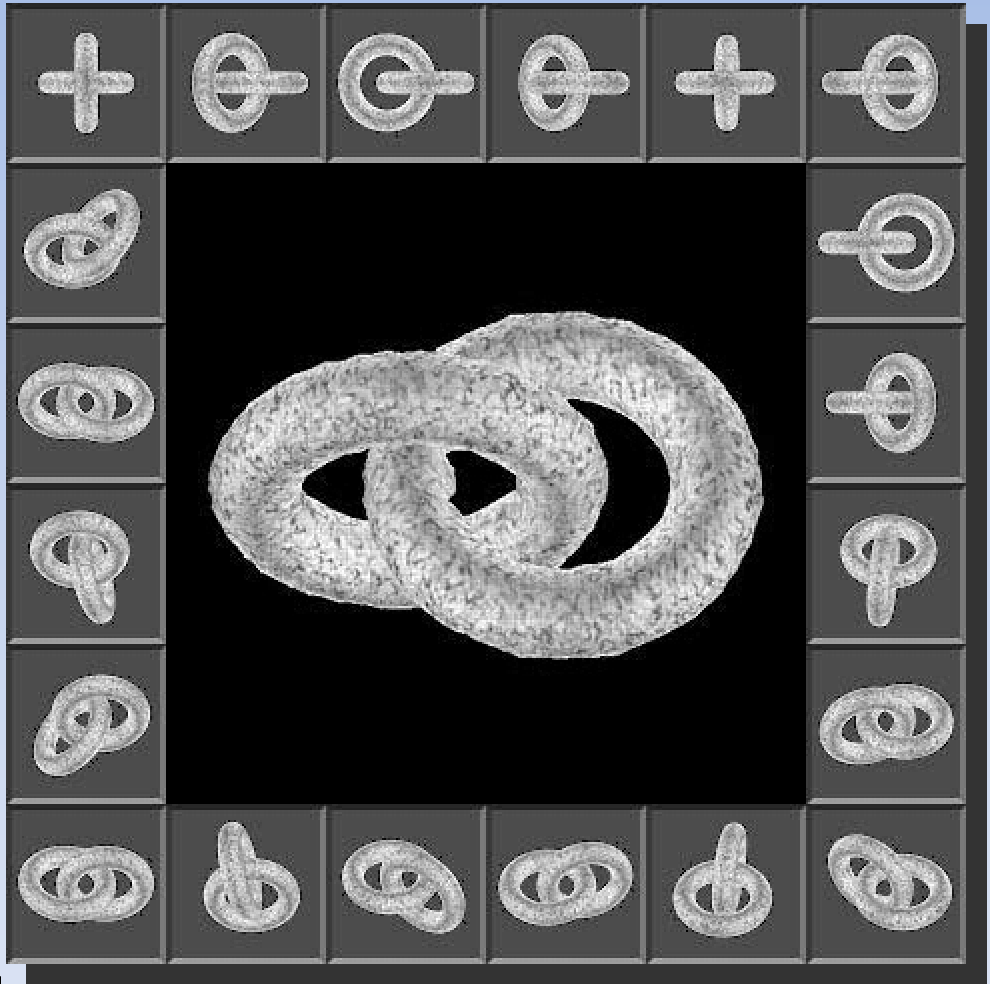
Reconstruction results on real vessel structures

Advantages of level sets

+ Topologically invariant segmentation

+ Invariant to the parameterization

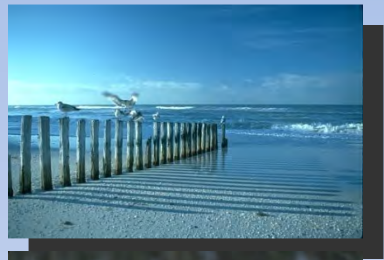
- Need for good stopping criterion

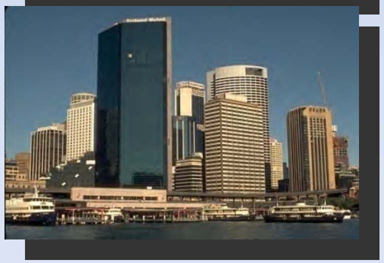


+ Fast marching assures global energy minimum
- Needs initial and final point

How to introduce high-level knowledge to regularize the segmentation problem?

- Similar pixels properties
- General high-level constraints
  - Snakes: contour smoothness (snakes, level sets).
  - Level sets: contour smoothness, topologically invariant.
  - **Graph cut:** connectivity and compactness of edge pixels, location and number of objects in images



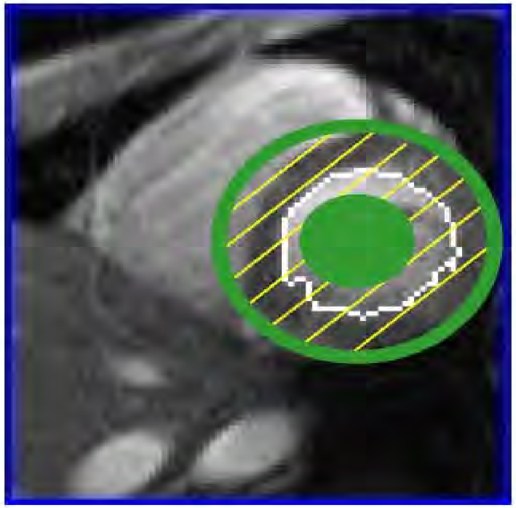



- Model-guided segmentation and recognition

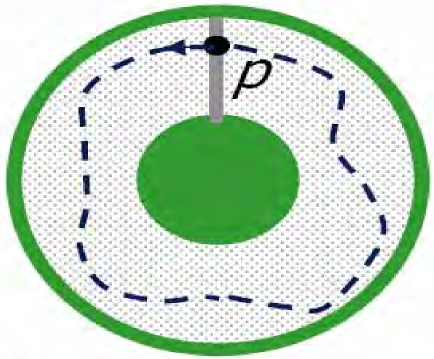
Segmentation by Graph Cuts

1D Graph cut $\Leftrightarrow$ *shortest path on a graph*

Example:
find the shortest closed contour in a given domain of a graph

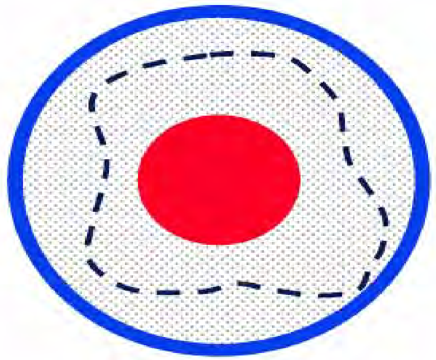


Shortest paths approach



Compute the *shortest path* $p \rightarrow p$ for a point p. Repeat for all points on the gray line. Then choose the optimal contour.

Graph Cuts approach



Compute the *minimum cut* that separates red region from blue region

From Boykov and Kolmogorov

Segmentation by Graph Cuts

E(X)

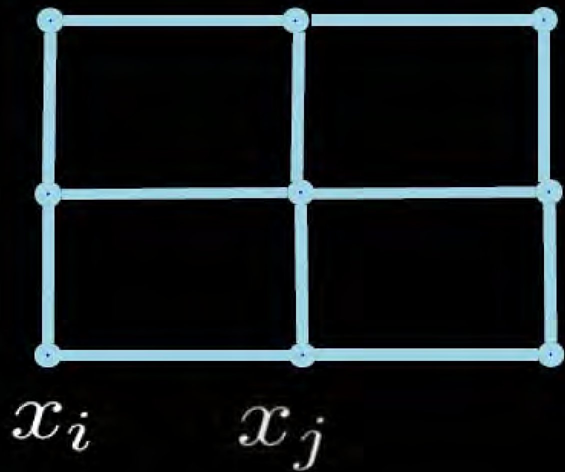
E: $\{0,1\}^n \rightarrow R$

$0 \rightarrow fg$

$1 \rightarrow bg$

n = number of pixels

Image (D)



[Boykov and Jolly ' 01] [Blake et al. '04] [Rother, Kolmogorov and Blake `04]

Segmentation by Graph Cuts

$$E(X) = \boxed{\sum c_i x_i}$$

Pixel Colour

$E: \{0,1\}^n \rightarrow R$

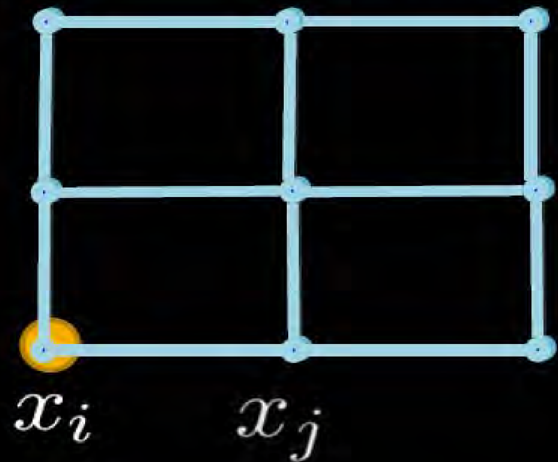
$0 \rightarrow fg$

$1 \rightarrow bg$

n = number of pixels

Unary Cost (c_i)

Dark (negative) Bright (positive)



[Boykov and Jolly ' 01] [Blake et al. '04] [Rother, Kolmogorov and Blake `04]

Segmentation by Graph Cuts

$$E(X) = \boxed{\sum c_i x_i}$$

Pixel Colour

$E: \{0,1\}^n \rightarrow R$

$0 \rightarrow fg$

$1 \rightarrow bg$

n = number of pixels

Unary Cost (c_i)

Dark (negative) Bright (positive)

$x^* = \arg\min E(x)$

[Boykov and Jolly ' 01] [Blake et al. '04] [Rother, Kolmogorov and Blake `04]

Segmentation by Graph Cuts

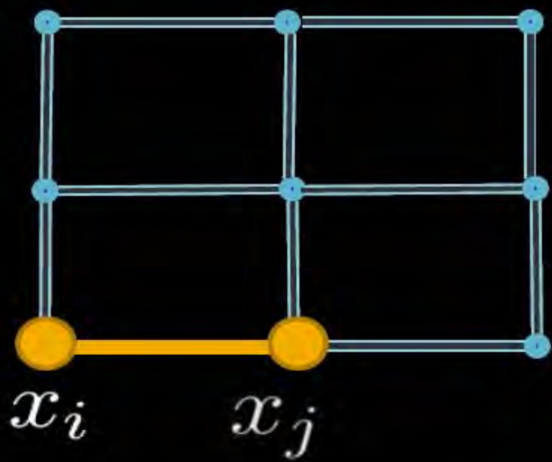
$$E(X) = \boxed{\sum c_i x_i} + \boxed{\sum d_{ij} |x_i - x_j|}$$

Pixel Colour — Smoothness Prior

$E: \{0,1\}^n \rightarrow R$
$0 \rightarrow fg$
$1 \rightarrow bg$

n = number of pixels

Discontinuity Cost (d_{ij})



[Boykov and Jolly ' 01] [Blake et al. '04] [Rother, Kolmogorov and Blake `04]

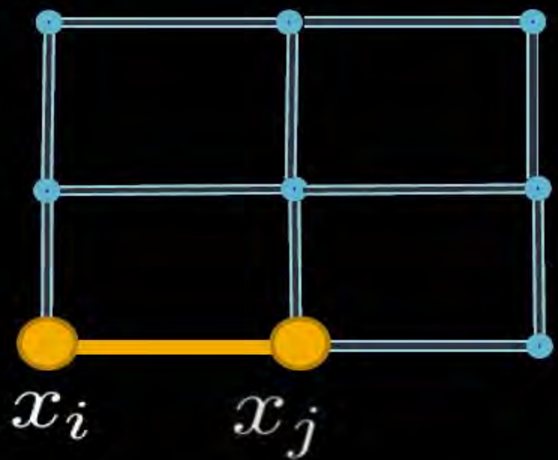
Segmentation by Graph Cuts

$$E(X) = \underbrace{\boxed{\sum c_i x_i}}_{\text{Pixel Colour}} + \underbrace{\boxed{\sum d_{ij} x_i (1-x_j) + d_{ij} x_j (1-x_i)}}_{\text{Smoothness Prior}}$$

$E: \{0,1\}^n \rightarrow R$
$0 \rightarrow fg$
$1 \rightarrow bg$

n = number of pixels

Discontinuity Cost (d_{ij})



[Boykov and Jolly ' 01] [Blake et al. '04] [Rother, Kolmogorov and Blake `04]

Segmentation by Graph Cuts

$$E(X) = \boxed{\sum c_i x_i} + \boxed{\begin{array}{l}\sum d_{ij}\, x_i\,(1-x_j) \\ +\; d_{ij}\, x_j\,(1-x_i)\end{array}}$$

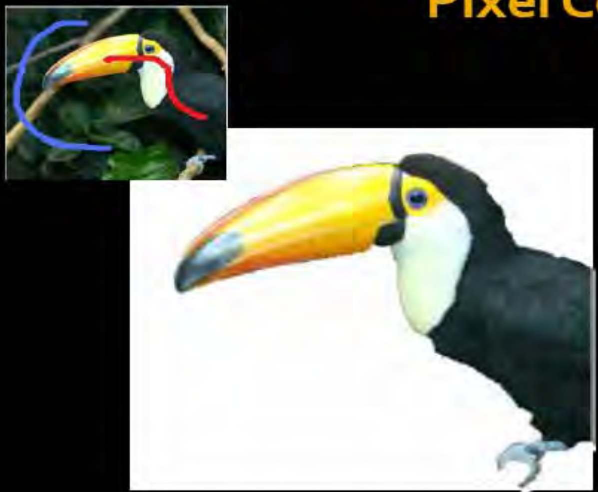
Pixel Colour　　Smoothness Prior

$E: \{0,1\}^n \rightarrow R$

$0 \rightarrow fg$

$1 \rightarrow bg$

n = number of pixels



$x^* = \arg\min E(x)$

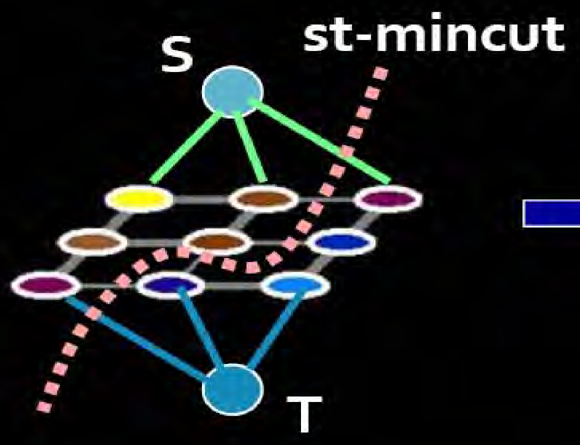
[Boykov and Jolly ' 01] [Blake et al. '04] [Rother, Kolmogorov and Blake `04]

So how does it work?

Construct a graph such that:

1. Any st-cut corresponds to an assignment of x
2. The cost of the cut is equal to the energy of x : E(x)

E(x) →



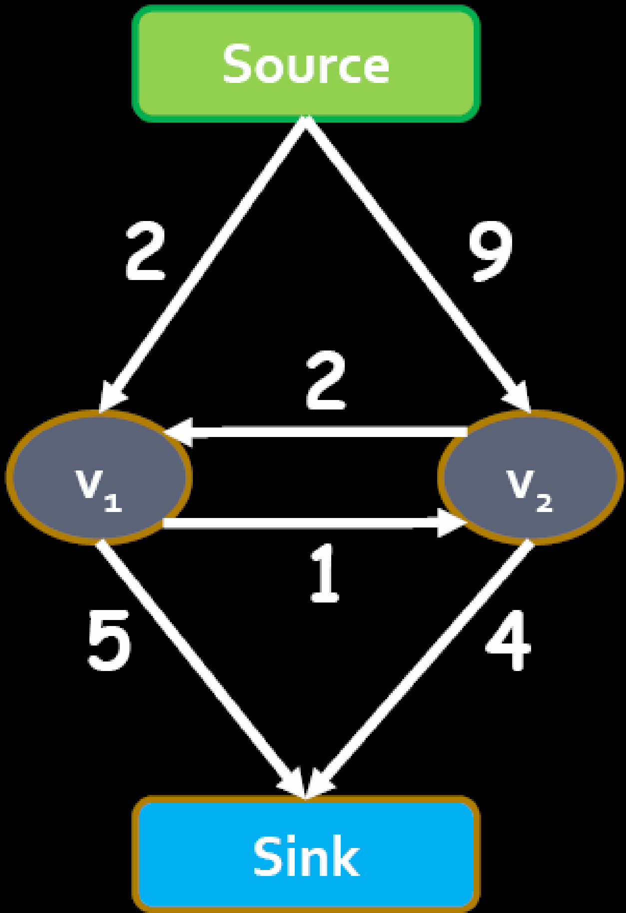
→

Solution

$$E(X) = \underbrace{\boxed{\sum c_i x_i}}_{\text{Pixel Colour}} + \underbrace{\boxed{\sum d_{ij} x_i (1-x_j) + d_{ij} x_j (1-x_i)}}_{\text{Smoothness Prior}}$$

[Hammer, 1965] [Kolmogorov and Zabih, 2002

The st-Mincut Problem



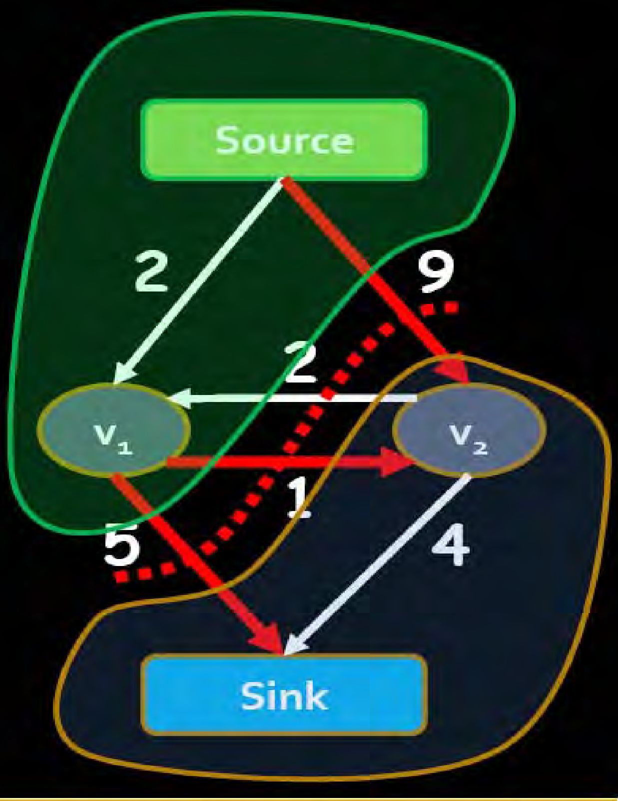
Graph (V, E, C)

Vertices $V = \{v_1, v_2 \dots v_n\}$

Edges $E = \{(v_1, v_2) \dots\}$

Costs $C = \{c_{(1,2)} \dots\}$

The st-Mincut Problem



5 + 1 + 9 = 15

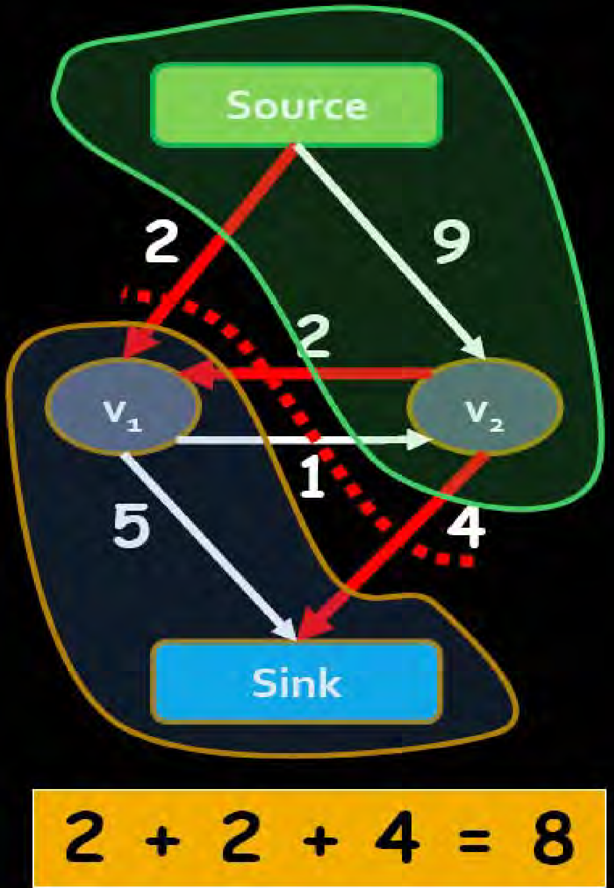
What is a st-cut?

An st-cut (**S**,**T**) divides the nodes between source and sink.

What is the cost of a st-cut?

Sum of cost of all edges going from S to T

The st-Mincut Problem



An st-cut (**S**,**T**) divides the nodes between source and sink.

What is the cost of a st-cut?

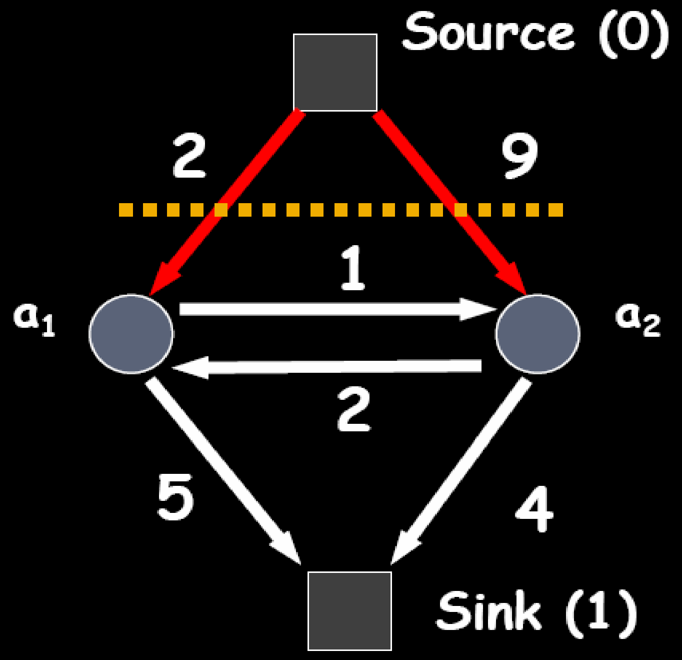
Sum of cost of all edges going from S to T

What is the st-mincut?

st-cut with the minimum cost

Graph construction

$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



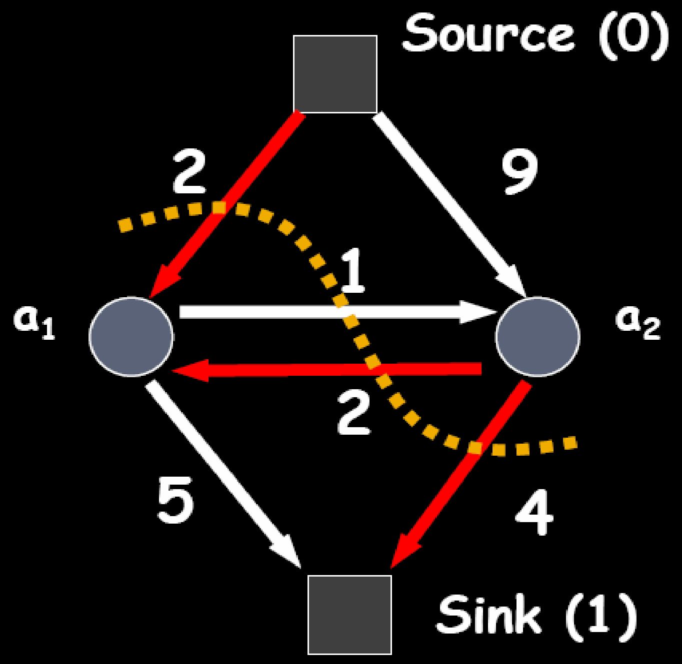
Cost of cut = 11

$a_1 = 1 \quad a_2 = 1$

$E(1,1) = 11$

Graph construction

$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

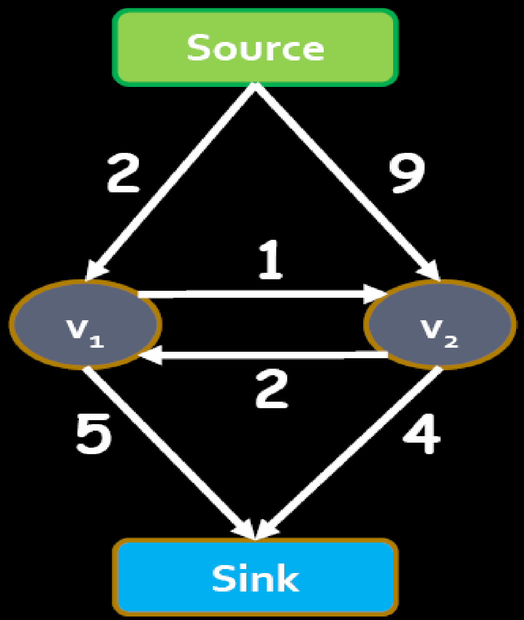


st-mincut cost = 8

$a_1 = 1 \quad a_2 = 0$

$E(1,0) = 8$

How to compute the st-Mincut?



Solve the dual **maximum flow** problem

Compute the maximum flow between Source and Sink s.t.

Edges: Flow < Capacity

Nodes: Flow in = Flow out

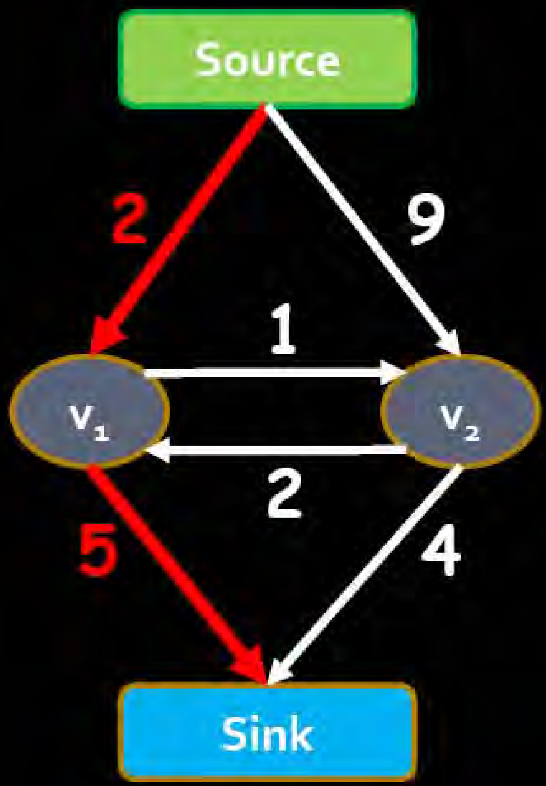
Min-cut\Max-flow Theorem

In every network, the maximum flow equals the cost of the st-mincut

Assuming non-negative capacity

Maxflow algorithm

Flow = 0

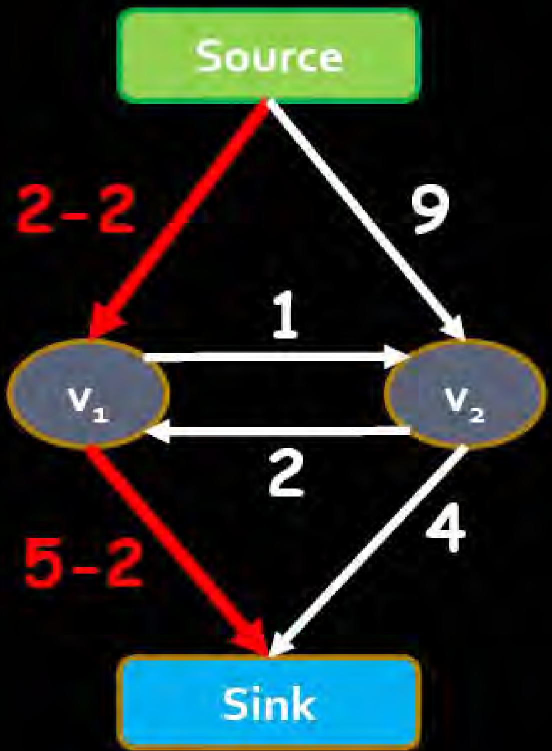


Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity

Maxflow algorithm

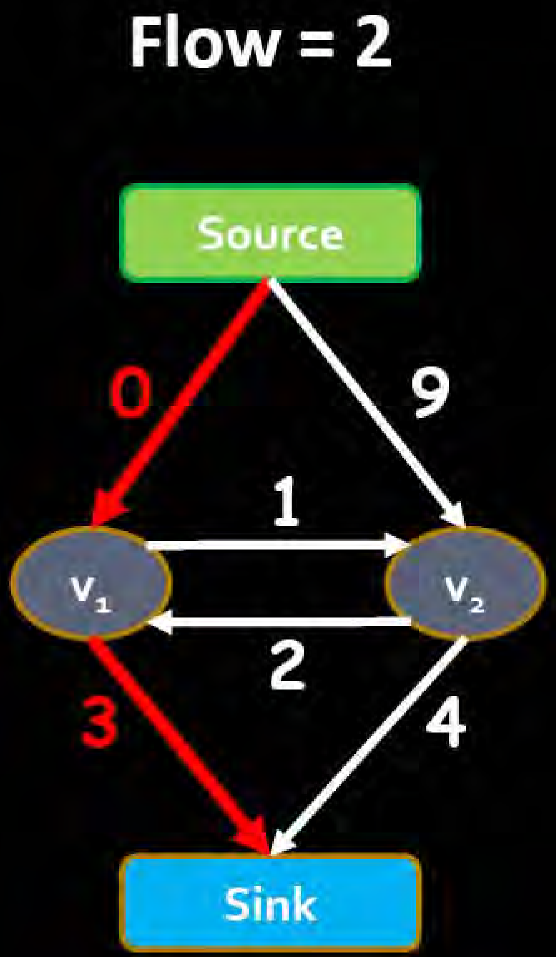
Flow = 0 + 2



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path

Maxflow algorithm

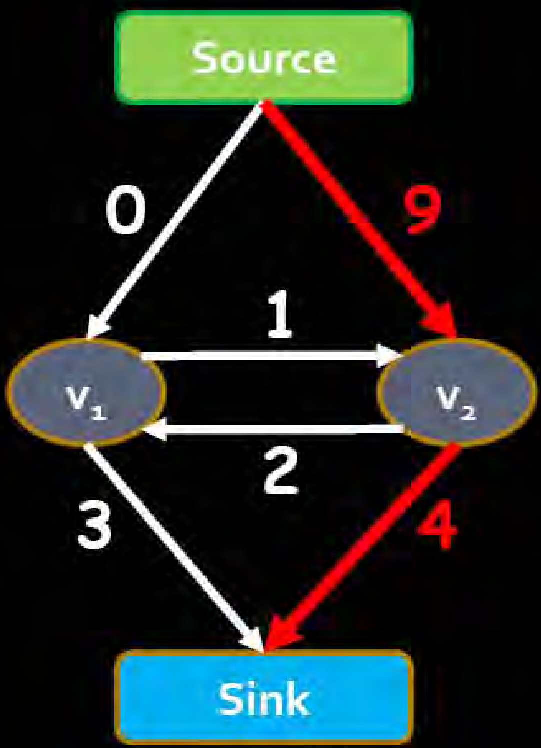


Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path

Maxflow algorithm

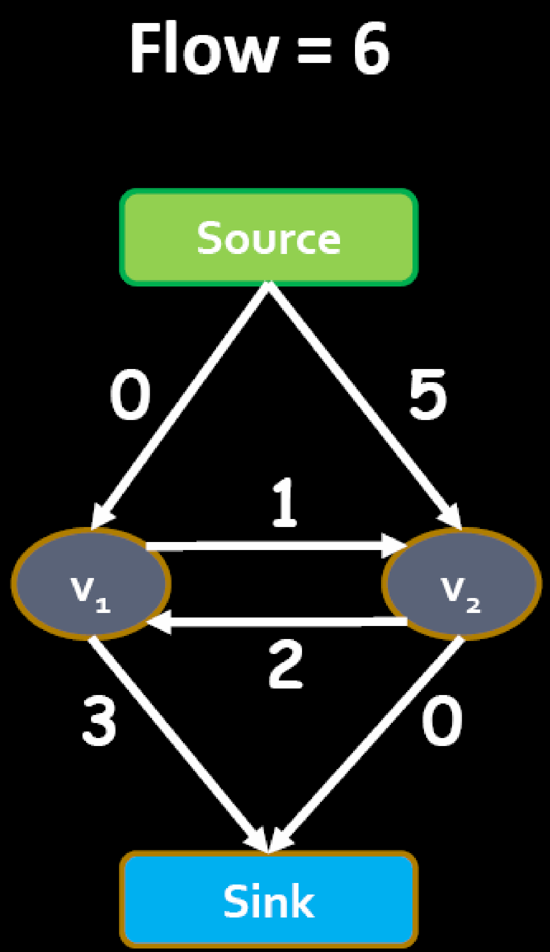
Flow = 2



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Maxflow algorithm

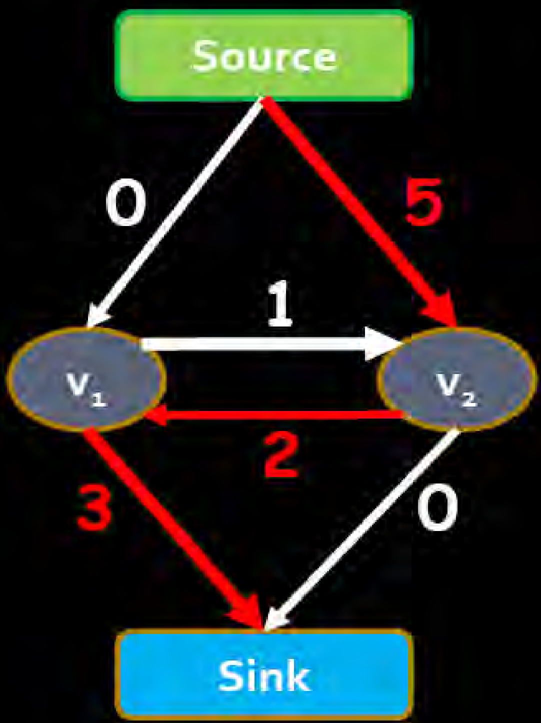


Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Maxflow algorithm

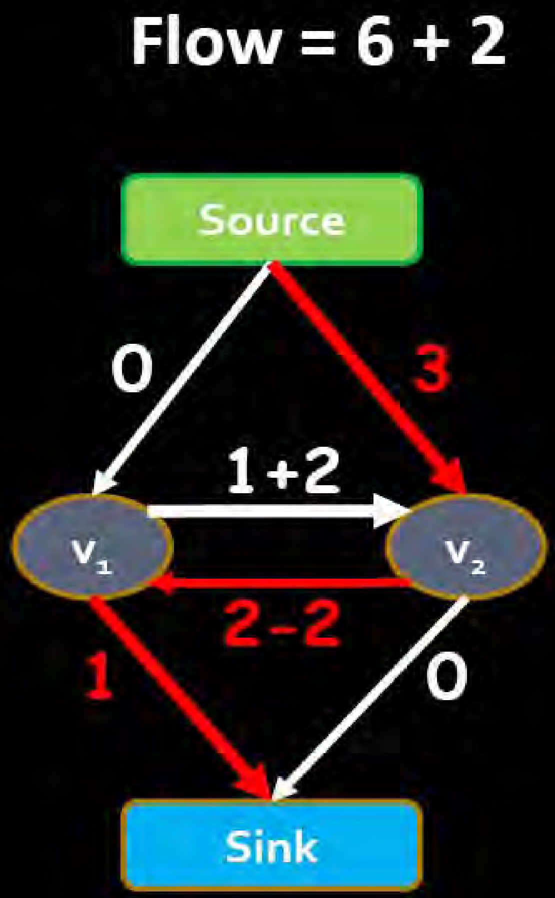
Flow = 6



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

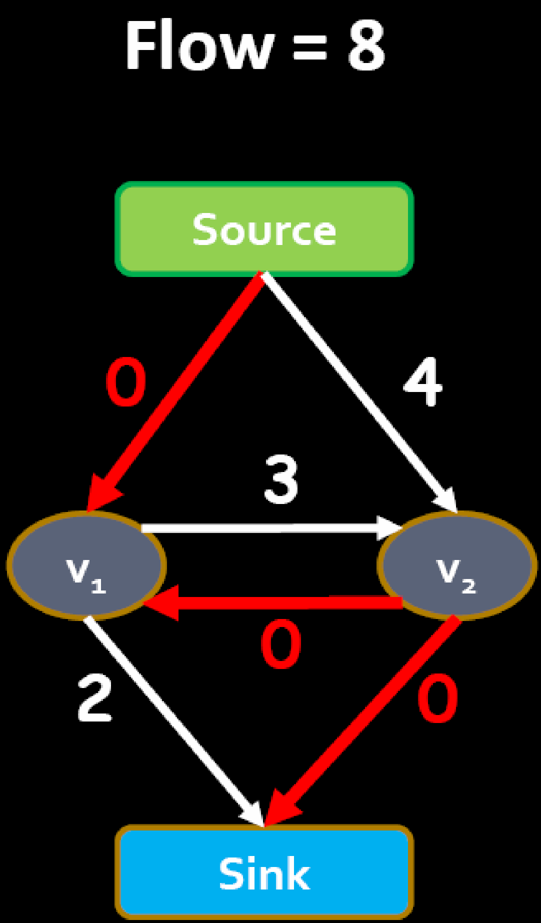
Maxflow algorithm



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

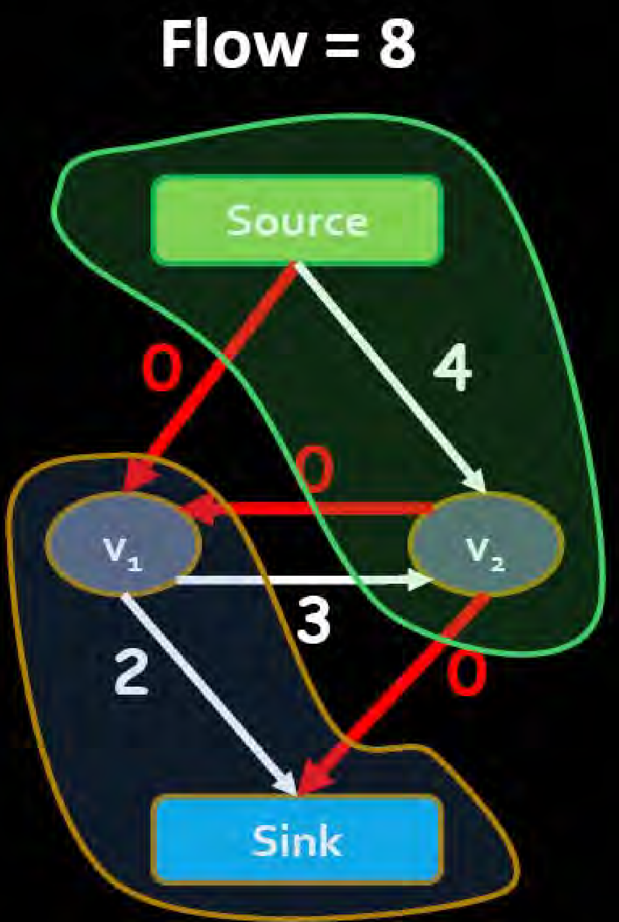
Maxflow algorithm



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Maxflow algorithm



Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Grab-cut

- Semi-automatic segmentation

Ferrari, V. , Marin-Jimenez, M. and Zisserman, A.

- Energy minimization scheme

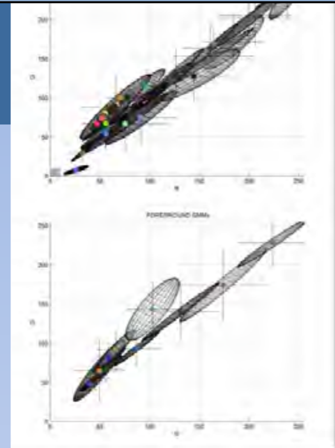
$$E(L) = \underbrace{\sum_{p \in \mathcal{P}} D_p(L_p)}_{\text{Likelihood based on image cues}} + \underbrace{\sum_{(p,q) \in \mathcal{N}} V_{p,q}(L_p, L_q)}_{\text{Spatial coherence based on pixel relationships}},$$

Likelihood based on image cues

Spatial coherence based on pixel relationships

45

Grab-cut



- Energy function

$$\mathbf{E}(\underline{\alpha},\mathbf{k},\underline{\theta},\mathbf{z}) = \boxed{U(\underline{\alpha},\mathbf{k},\underline{\theta},\mathbf{z})} + \boxed{V(\underline{\alpha},\mathbf{z})}$$

Pixel differences based on RGB Euclidean distance

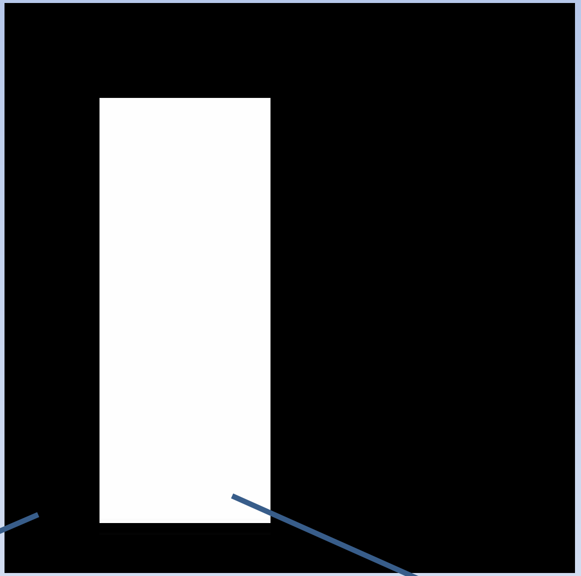
$$V(\underline{\alpha},\mathbf{z}) = \gamma \sum_{(m,n)\in C} [\alpha_n \neq \alpha_m] \exp -\beta \|z_m - z_n\|^2$$

RGB

GMM modelling → Assign pixels to GMM components → Segmentation (min-cut)

[3] C Rother, V Kolmogorov, A Blake. "**Grabcut: Interactive foreground extraction using iterated graph cuts**" , *ACM Transactions on Graphics, 2004.*

GrabCut example



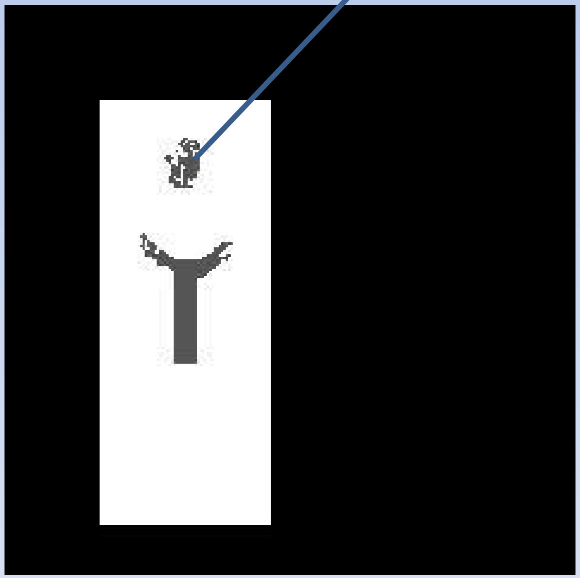
Background seeds

Unknown region

47

GrabCut example

Foreground seeds

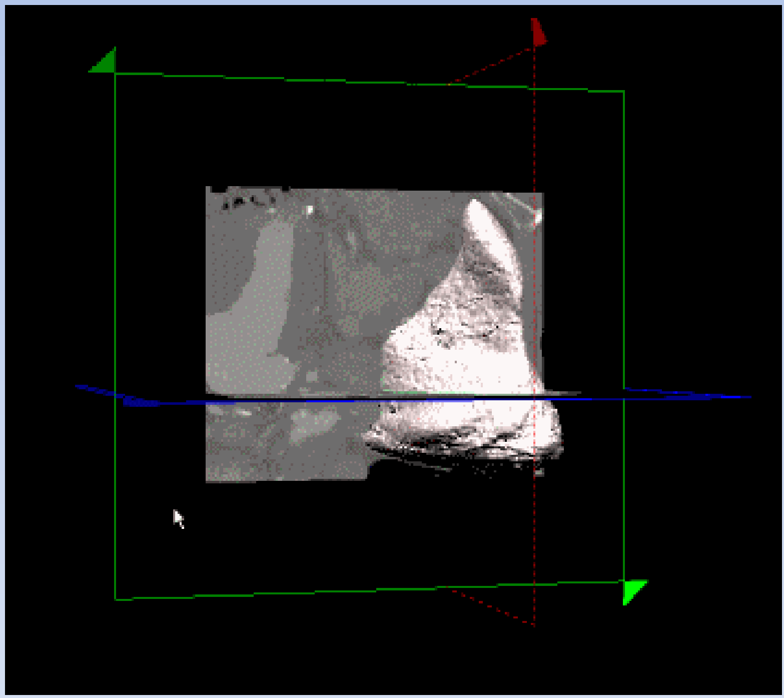


Medical image graph-cut segmentation

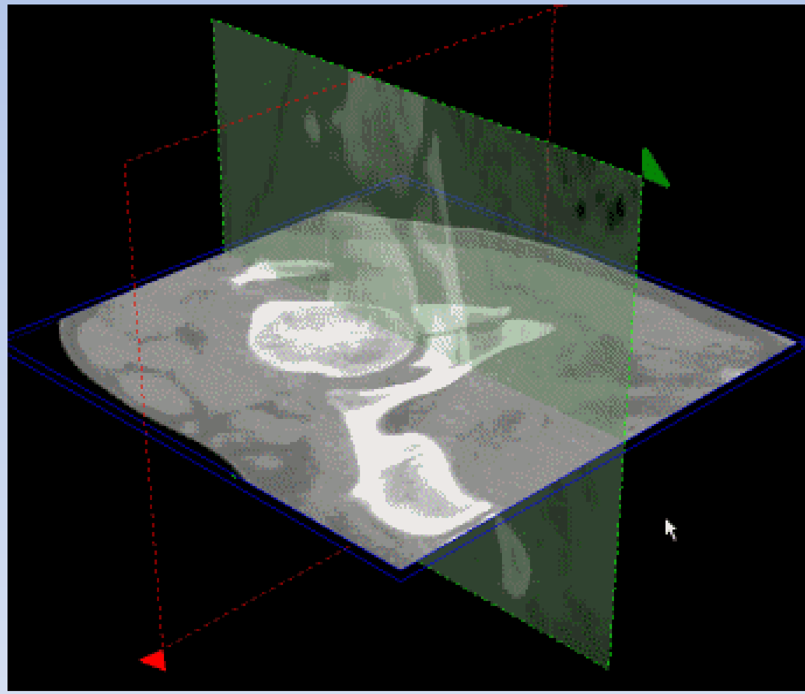
Abdomen segmentation

Baby segmentation

Examples



Liver segmentation



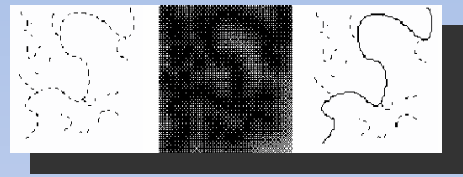
Bone segmentation

Conclusions

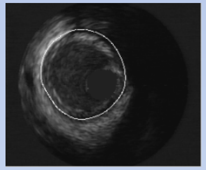
- **Fast and elegant region** segmentation method
- Assures the **global minimum** of the energy
- Needs **good cost** definition
- Grab cut is a natural extension to color segmentation of **multicolor** objects by GMM
- **Weak inicialization** can help significantly to the segmentation

Identify and compare the 4 deformable models

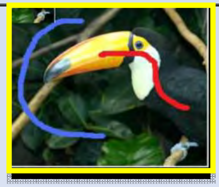
$$U(p) = inf_{Q_{L\_p}}\{\int_Q P(Q) + \alpha|\frac{\delta Q}{\delta s}|^2 ds\}$$



$$E_{snake} = \int_0^1 E_{int}(u(s)) + E_{ext}(u(s))\, ds.$$



$$Q_t = g(I)(c+k)\vec{n} - (\nabla g, \vec{n})\vec{n}$$

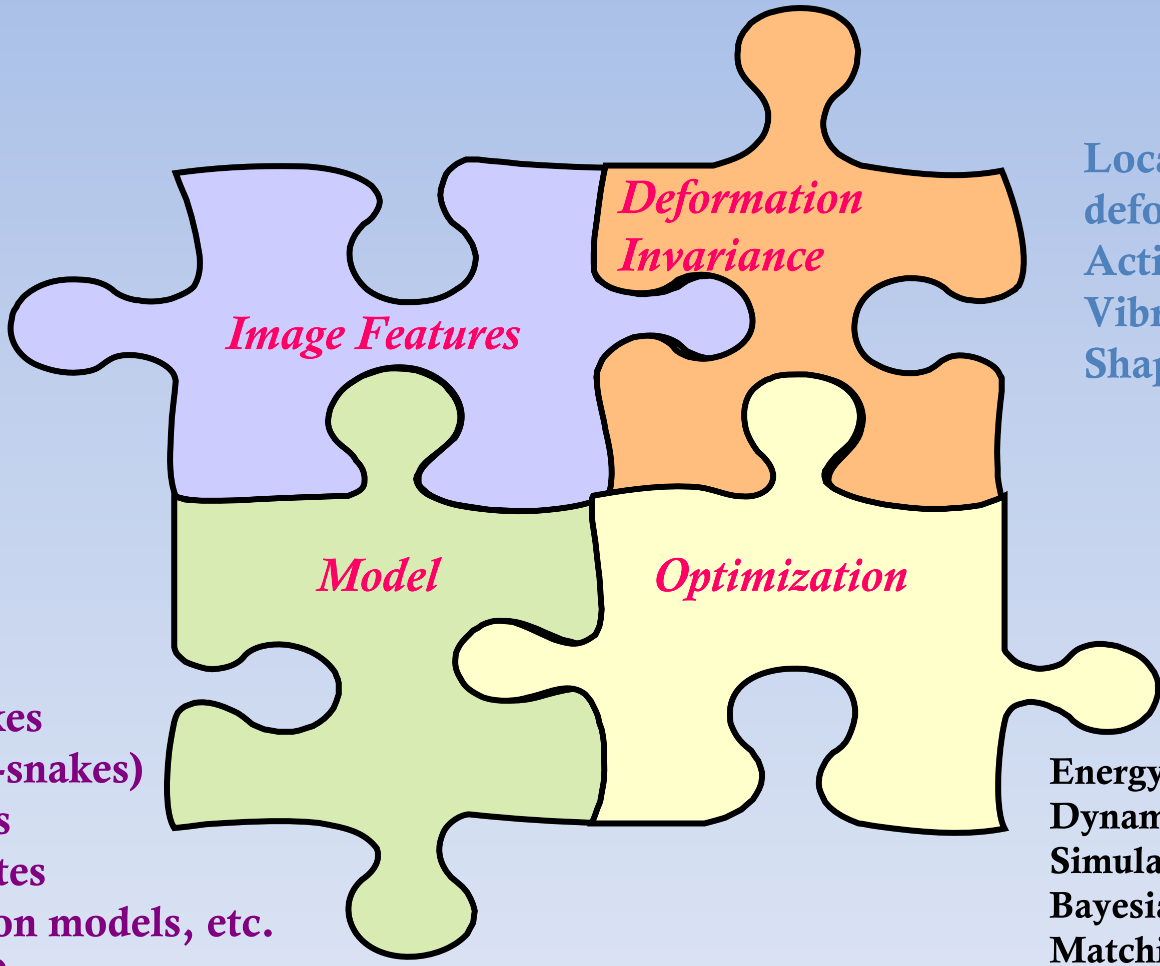


$$E(L) = \sum_{p\in\mathcal{P}} D_p(L_p) + \sum_{(p,q)\in\mathcal{N}} V_{p,q}(L_p, L_q),$$

Deformable Models "Unpuzzled"

Contour-based
Region-based
Texture-based
Color-based
Motion-based
Stereo, etc.

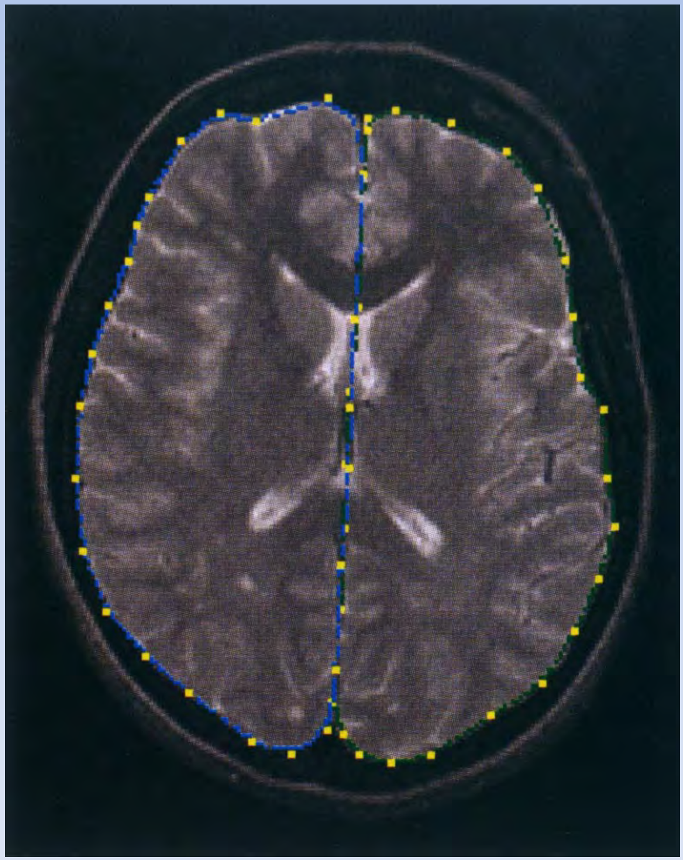
Local continuity deformations
Active shape models
Vibration modes
Shape spaces

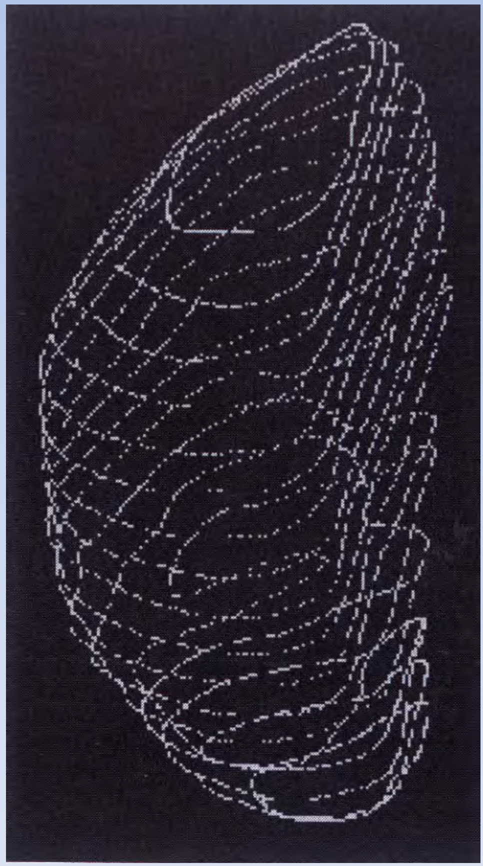


Parametric snakes
(snakes, B-snakes)
Geodesic snakes
Flexible templates
Point distribution models, etc.
1D, 2D, 3D, 4D

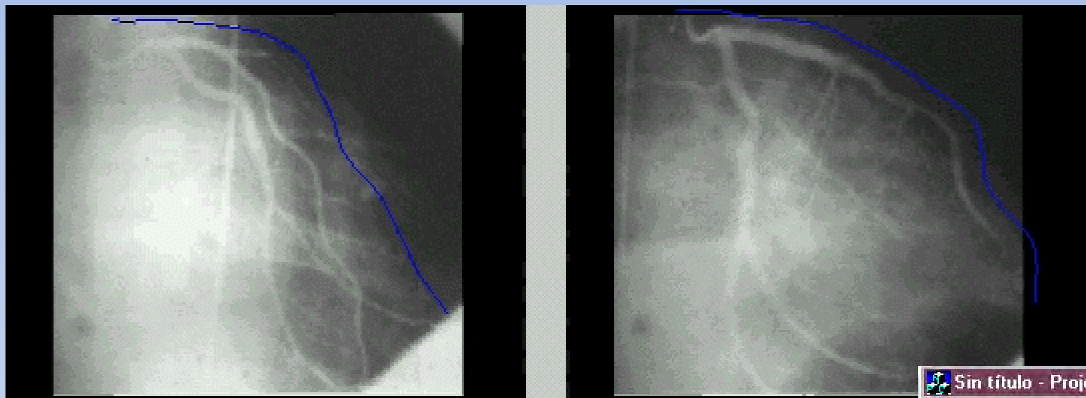
Energy minimization
Dynamic programming
Simulated annealing
Bayesian approach
Matching
Classification, etc.

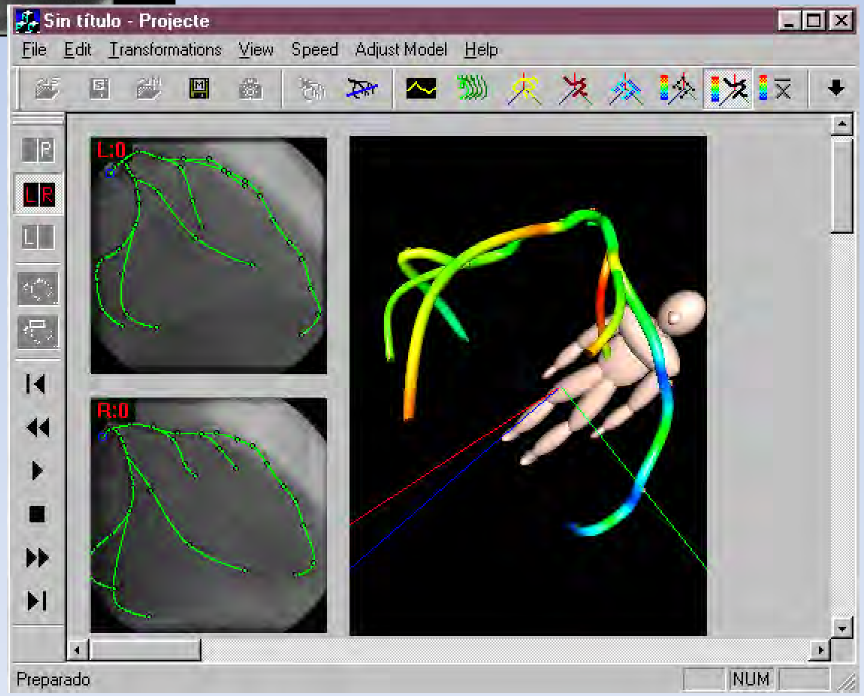
Applications of deformable shapes



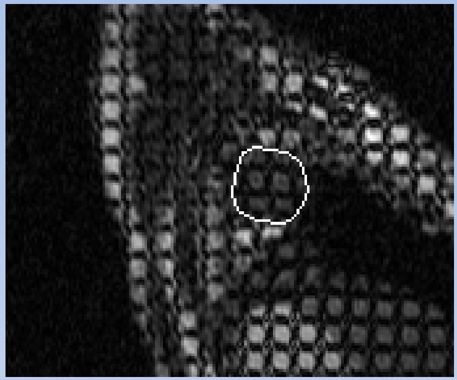
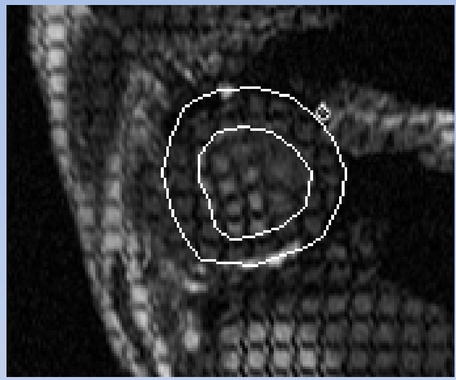


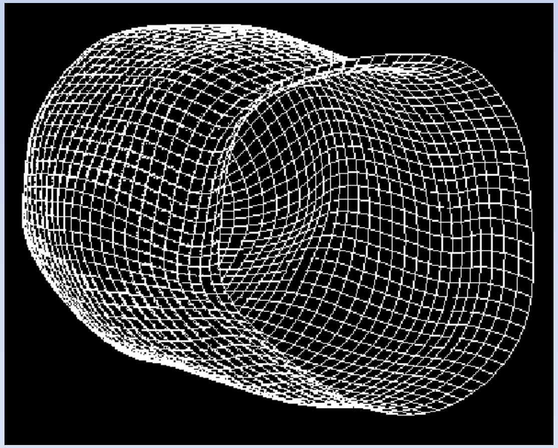
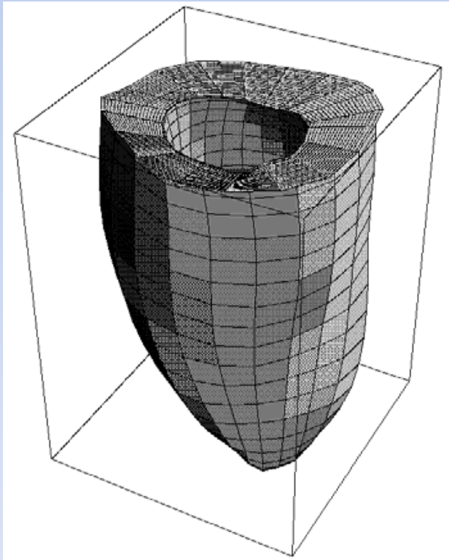
Applications of deformable shapes

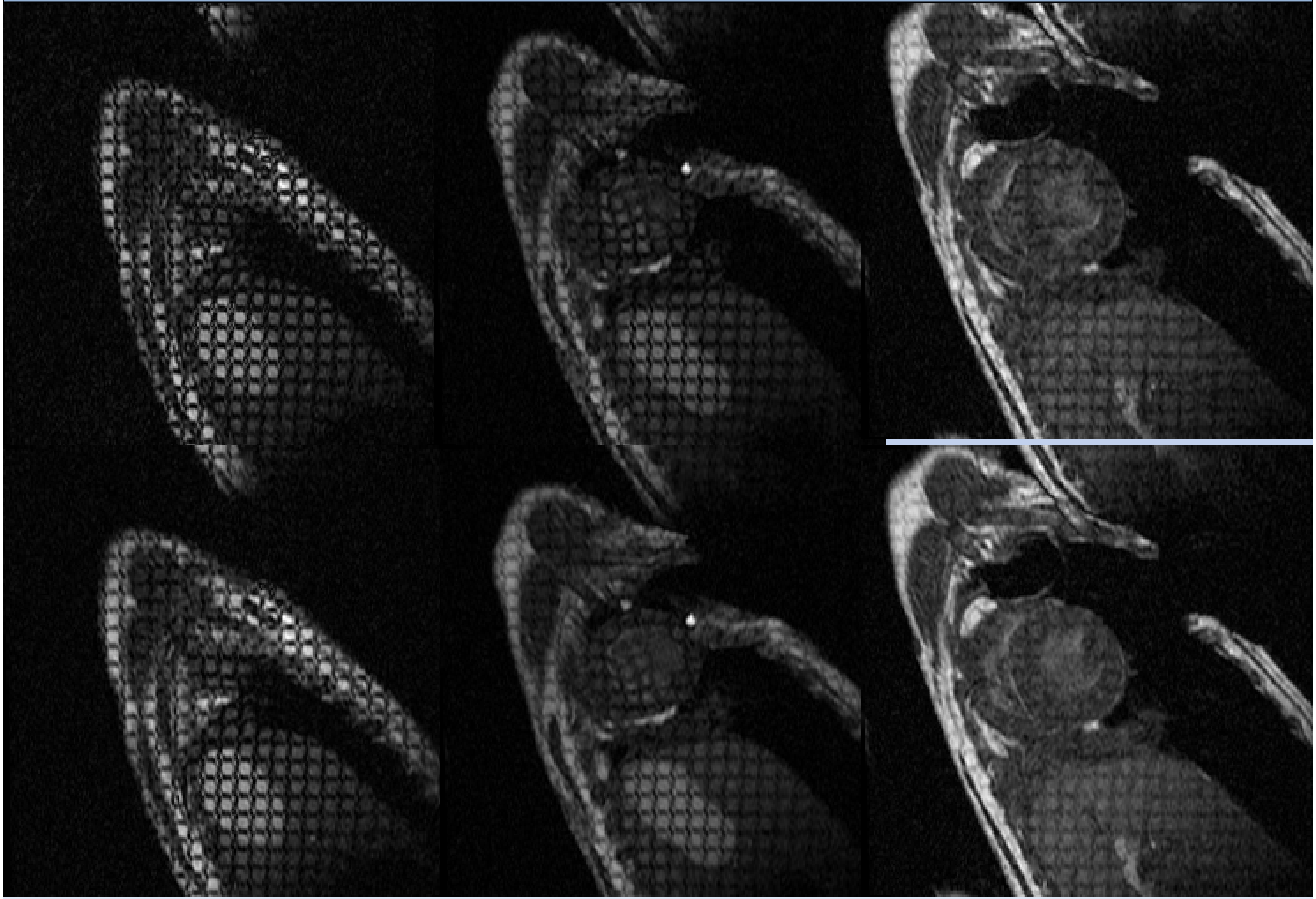




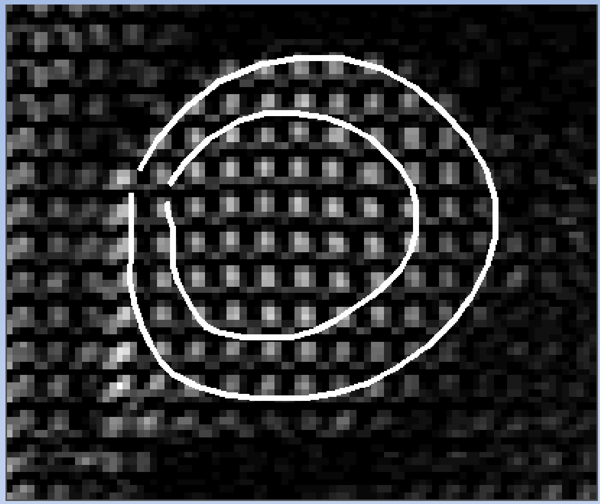
Applications of deformable shapes

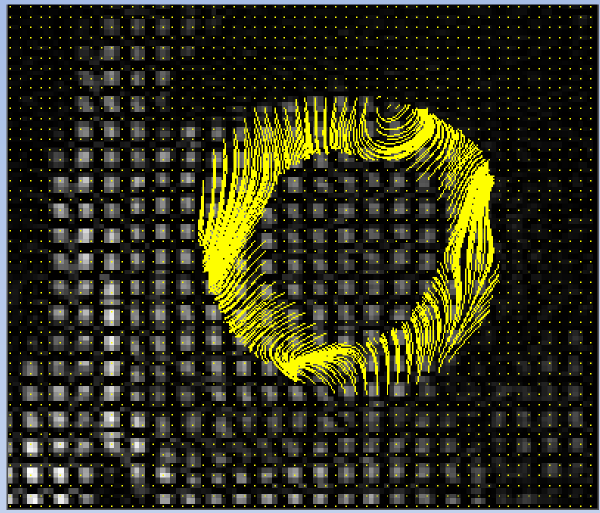



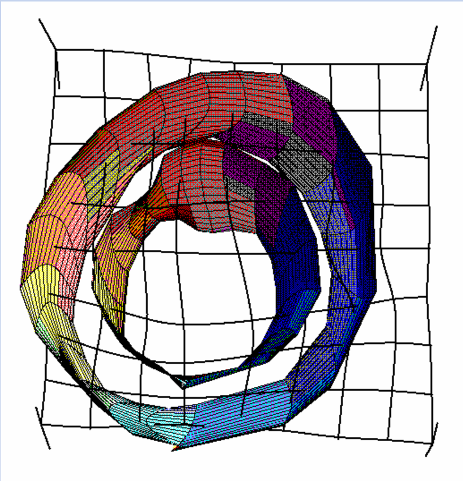



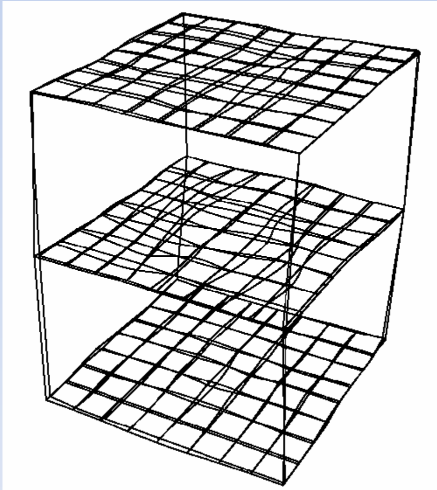


Applications of deformable shapes

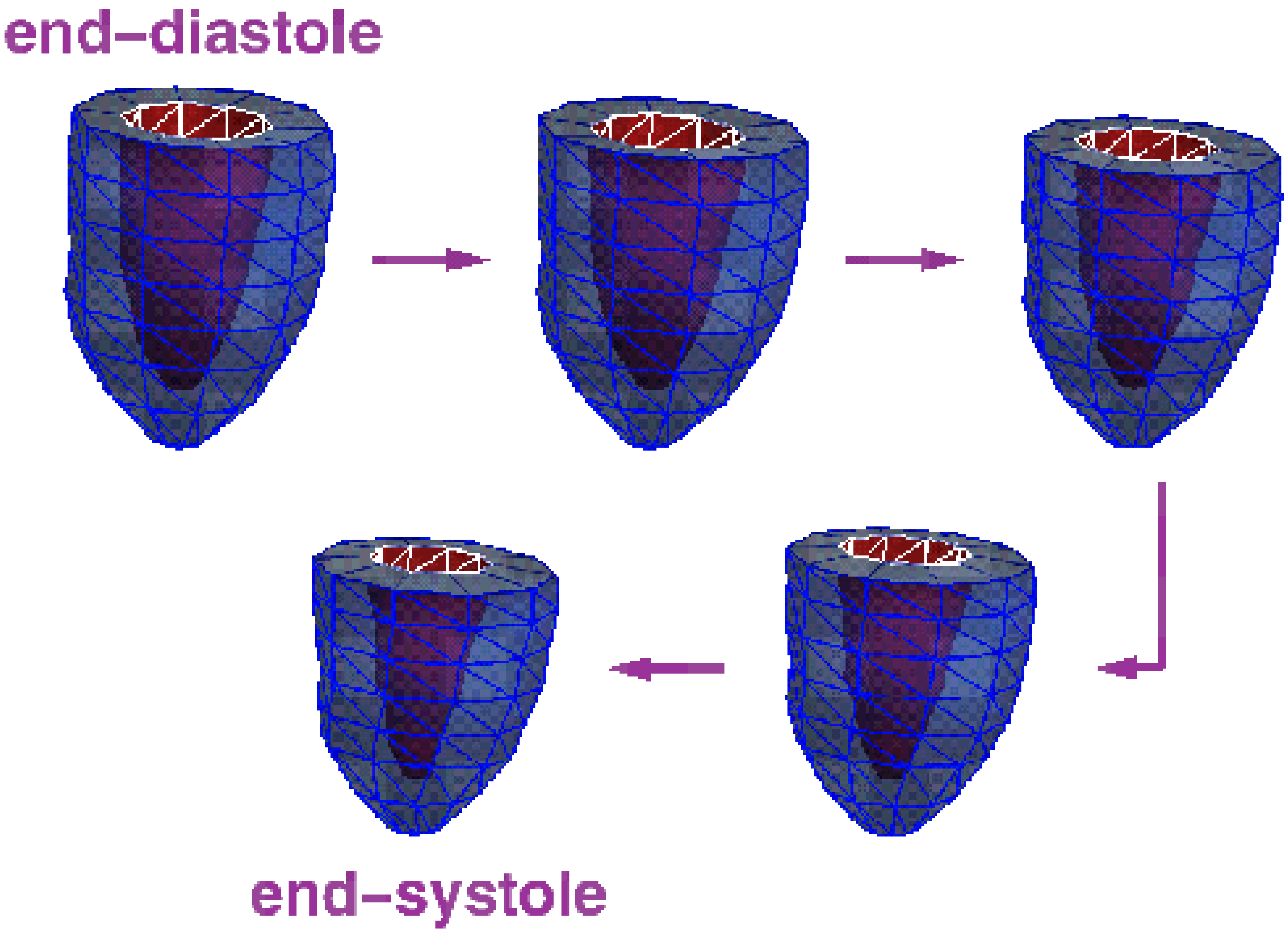




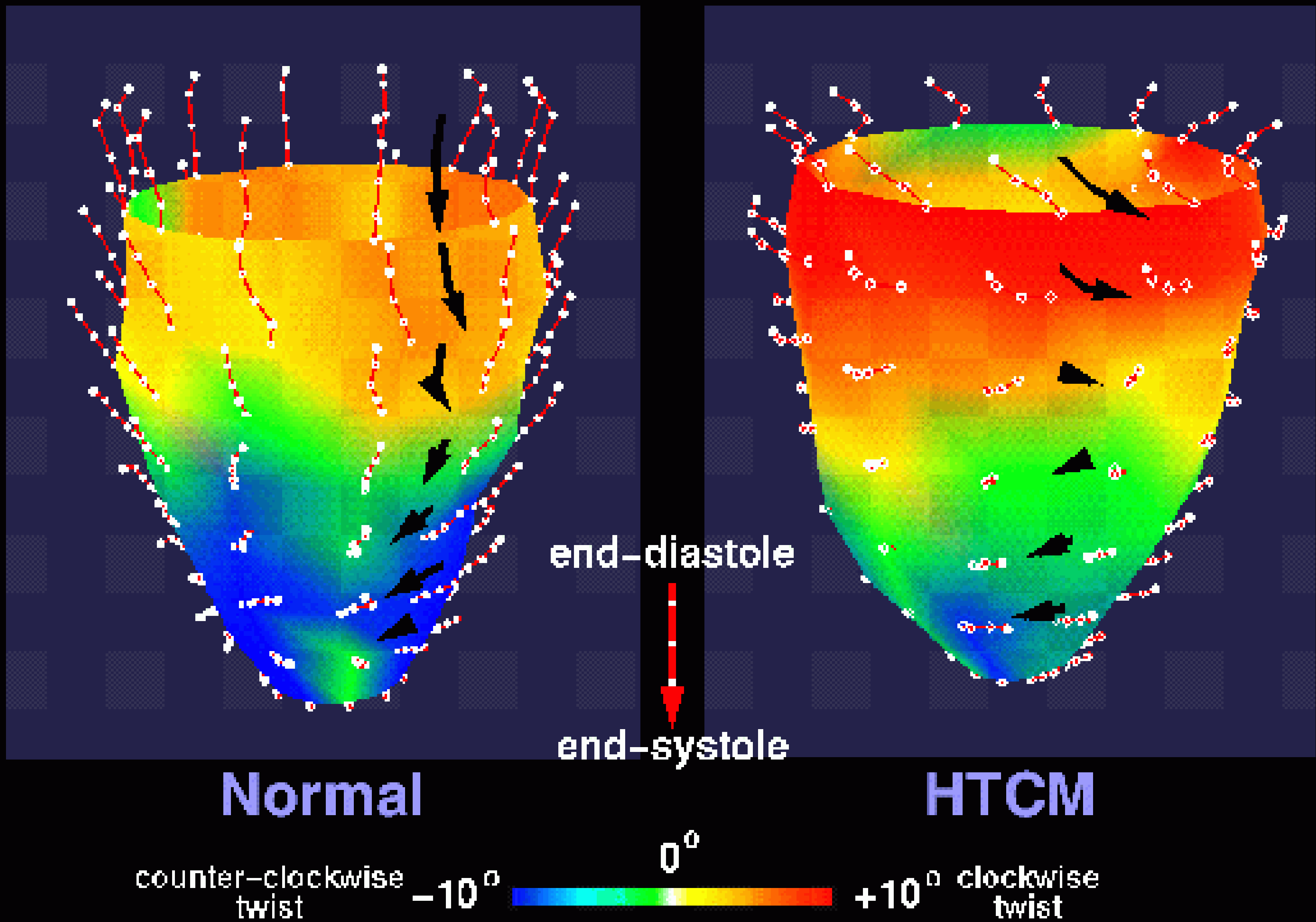




Left Ventricle in Systole



Twisting Motion of Left Ventricles during Systole



Thank you! ☺