

# Level Set Methods and Fast Marching Methods

Seminar Scientific Computing  
Group

June 5, 2002

*Bratislav Tasić*

# Outline

- Overview of some theoretical issues
- Introduction to applications
- Geometry problems
- Image enhancement and noise removal
- Shape detection and recognition
- Robotic navigation with constraints
- Summary

# Overview of some theoretical issues

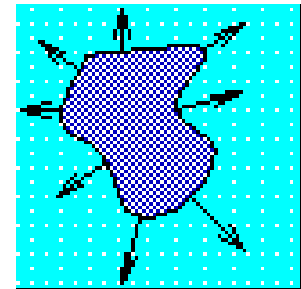
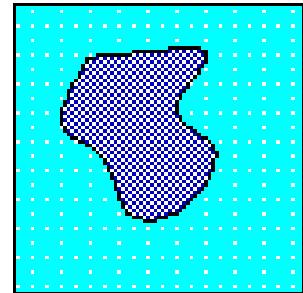
Moving interface problem

Speed function:  $F = F(L, G, I)$

$L$  – Local properties

$G$  – Global properties

$I$  – Independent properties



# Overview of some theoretical issues

## Boundary value formulation

$$|\nabla T| F = 1$$

$$\Gamma(t) = \{(x, y) : T(x, y) = t\}$$

$$F > 0$$

## Initial value formulation

$$\phi_t + F |\nabla \phi| = 0$$

$$\Gamma(t) = \{(x, y) : \phi(x, y, t) = 0\}$$

$$F \text{ arbitrary}$$



# Introduction to Applications

**Applications** in all areas where free boundaries are present. Examples:

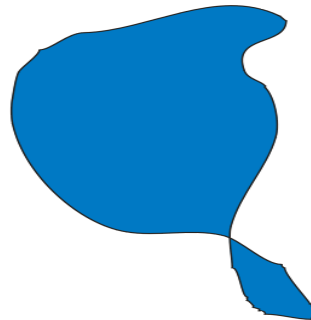
- Geometry problems
- Image enhancement and noise removal
- Shape detection and recognition
- Combustion, solidification, fluids, electromigration
- Computer-aided design
- Etching and deposition in microchip fabrication
- Optimal path planning, etc.

# Geometry Problems

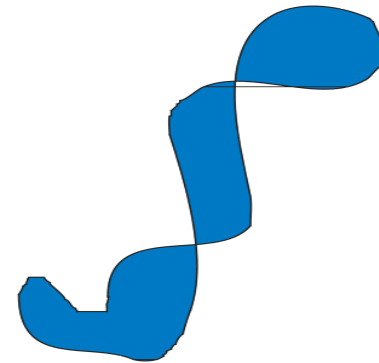
Imagine a closed curve lying on this page, and don't let the curve cross over itself (a simple, closed curve). For example, take a piece of rope, glue the two ends together, and drop it on the ground (the rope doesn't cross over itself).



Simple Closed  
Curve



Not Simple  
Curve

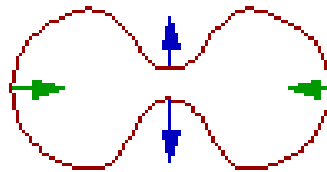


Even Less Simple  
Curve

# Geometry Problems

Curvature – measure of how fast a curve bends at any spot

$$\kappa = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \left( \nabla \cdot \frac{\nabla T}{|\nabla T|} \right) = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}}$$



→ - curvature is negative

→ - curvature is positive

# Geometry Problems

**A Famous Theorem** in differential geometry (proved less than 10 years ago) says that any simple closed curve moving under its curvature collapses nicely to a circle and then disappears. That is, no matter how wildly twisting a curve is, as long as it is simple, it will "relax" to a circle and then disappear.

## **Applications:**

- Surface tension of an interface (soap bubble)
- Dynamics of ink in an ink jet plotter
- Evolution of boundaries between fluids
- Removing noise from an image, etc.

# Geometry Problems

Motion by curvature – velocity as a function of  $\kappa$

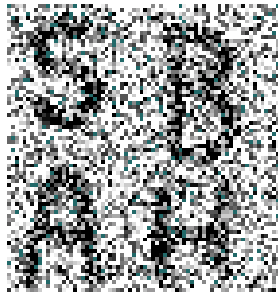
$$F := F(\kappa) = -\kappa$$



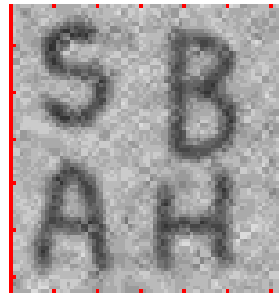
Weird Simple Curve Collapsing under Its Curvature

# Image Enhancement and Noise Removal

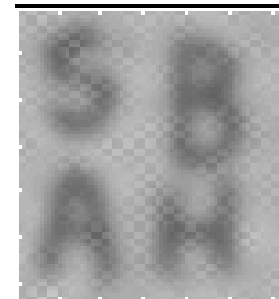
Imagine an image with noise (60% of the pixels are thrown away) – Gaussian filter



A "noisy" image



Gaussian Smoothing



Continued smoothing

**Positive** – much of the spotty noise has been muted out  
**Negative** – the sharp boundaries have been smeared due to the averaging

# Image Enhancement and Noise Removal

## The level set approach

- Pixel values as a topographic map
- Intensity  $I$  (somewhere between white and black) at each pixel is the height of the surface at that point
- Let each contour undergo motion by curvature

Intensity under the flow  
(noise will be removed)

$$I_t = \nabla^2 I$$

$$I_t = F |\nabla I|, \quad F = \frac{\nabla \cdot \nabla I}{|\nabla I|} = \kappa$$

But, everything will disappear! We need a “stopping criterion”

# Image Enhancement and Noise Removal

The Min/Max switch – imagine a disk with a distance function  $\phi(x,y)$  positive outside and negative inside of the disk

$$\bar{F}_{\min/\max}^{\text{Stencil}=a} = \begin{cases} \min(\kappa, 0), & \text{Ave}_{\phi(x,y)}^{R=ah} < 0 \\ \max(\kappa, 0), & \text{Ave}_{\phi(x,y)}^{R=ah} \geq 0 \end{cases}$$

$$\bar{F} = -F$$

$\text{Ave}_{\phi(x,y)}^{R=ah}$  – Average value of  $\phi$  in a disk of radius  $R$  around  $(x,y)$

$h$  – grid size,  $a = 0, 1, \dots$  – stencil radius

$\bar{F} = \min(\kappa, 0)$  – preserves the shape

$\bar{F} = \max(\kappa, 0)$  – diffuse away all information

# Image Enhancement and Noise Removal

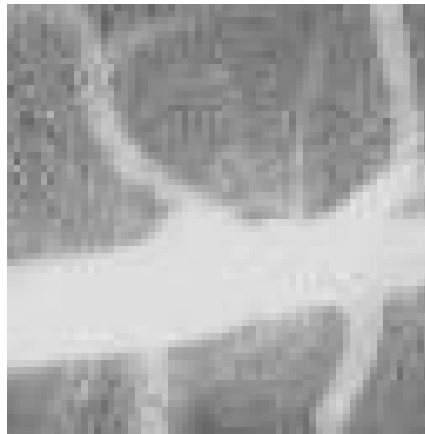
The level set approach – exploiting the fact that curves moving under their curvature smooth out and disappear



Noise removal under the level set approach

# Shape Detection and Recognition

Problem of extracting important features within the image, e.g. medical (MRI or CT) scans

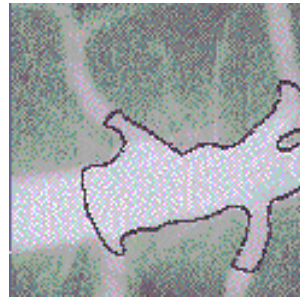
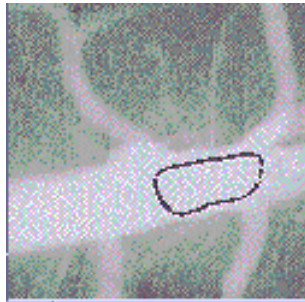


The outline of an artery

One idea: Detection of big jumps in intensity – various problems (threshold, noise)

# Shape Detection and Recognition

Another idea: Initialization of a small circle inside and allowing it to grow outwards until boundary



**The algorithm:** Fast marching method for propagation the initial seed point outward and level set method for fine tuning

**The key idea:** evolve the curve outwards with a speed that depends on the image itself

# Shape Detection and Recognition

The velocity function from the image data that acts as a stopping criterion

$$F = (\pm 1 - \varepsilon \kappa) g_I(x, y)$$

$$g_I(x, y) = \frac{1}{1 + |\nabla(G_\sigma * I(x, y))|}$$

$G_\sigma * I$  – Image convolved with Gaussian smoothing filter with characteristic width  $\sigma$  (the gradient is essentially zero except where the image gradient changes rapidly)

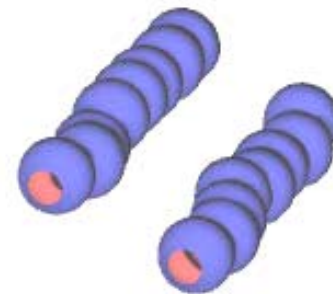
$F$  – becoming small near the boundaries

# Shape Detection and Recognition

## *Examples*



Arterial Tree



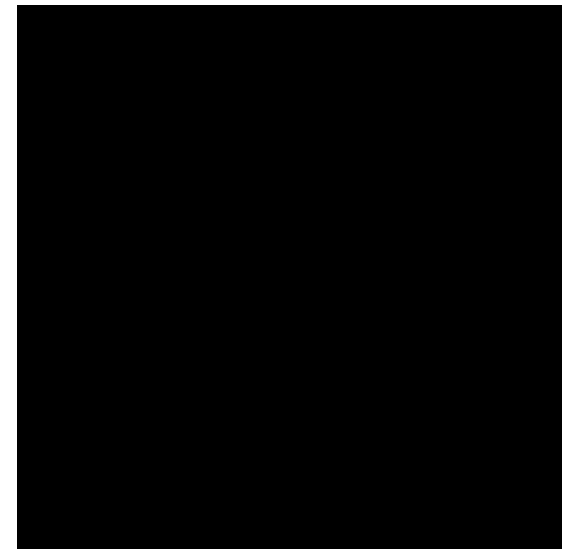
Femurs and surrounding soft tissue

# Shape Detection and Recognition

## *Examples*



Beating Heart



Reconstruction of  
Brain

# Robotic Navigation with Constraints

*Suppose you live in an apartment with lots of corridors and long skinny halls. And, you've finally scraped up the money to buy a piano, which you need to get from the front door to the back room. The first question is, is it actually possible to twist and turn the piano in such a way as to get it back there? And second, if it is, what is the shortest path?*

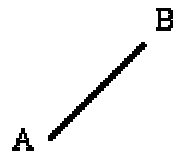
**Optimal path planning – finding the shortest possible path from the initial state to the final state in a domain which may contain obstacles**

# Robotic Navigation with Constraints

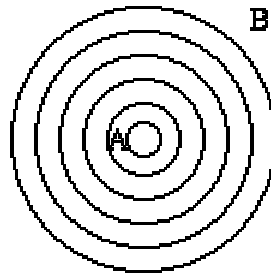
*Example* – A parking lot, a man and his car

- Easy version: A parking lot is empty – the shortest path is the straight line

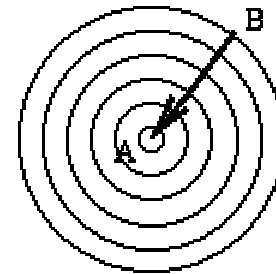
Fast marching method approach – let a front expand in all directions with speed 1; after reaching a target go back perpendicular to the front



Man and car



Expanding front

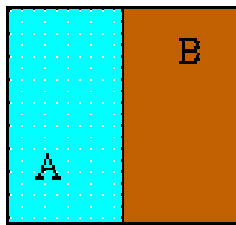


Trace back

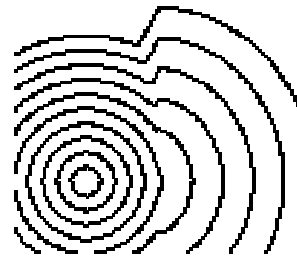
# Robotic Navigation with Constraints

- Slightly harder version: One half of a parking lot is covered in snow (harder to walk) – the “shortest” path is the line which takes the least time

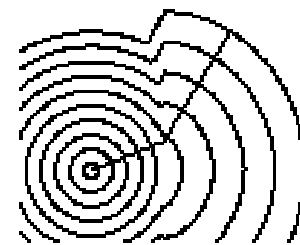
The front expands faster on dry pavement than on the snow



Snow/dry



Expanding front

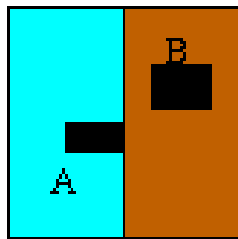


Trace back

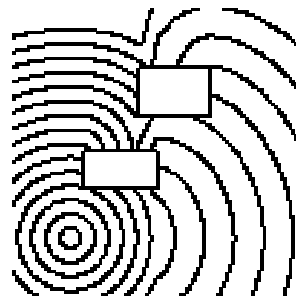
# Robotic Navigation with Constraints

- Even harder version: There are some cars on the parking lot (together with snow)

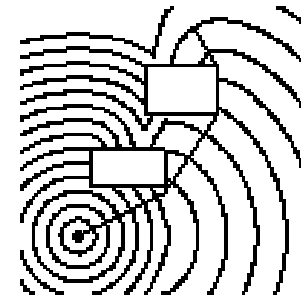
The cars are replaced by places where the “cost” of travelling is infinity (takes forever to walk through the car)



Snow/dry  
+ other cars



Expanding front



Trace back

# Robotic Navigation with Constraints

Statement of problem:

- Given cost function:  $F(x_1, x_2, \dots, x_n)$
- Starting point:  $A$
- The unknown path:  $\gamma(\tau) : [0, \infty) \rightarrow \mathbb{R}^n$   
 $\tau$  – arc length parameterisation of  $\gamma$  ( $|\dot{\gamma}_\tau| = 1$ )

Minimal cost:

$$T(x, y) = \min_{\gamma} \int_A^{(x,y)} F(\gamma(\tau)) d\tau$$

The level set  $T(x, y) = C$  – all points which can be reached with minimal cost (e.g. same time)

# Robotic Navigation with Constraints

Problem reduced to Eikonal equation

$$|\nabla T| = F(x, y)$$

Fast marching method:  $T(x, y), \quad \forall x, y \in \mathbb{R}^2$

Given ending point:  $B$

The shortest path from  $B$  to  $A$  is the solution of ODE

$$X_t = -\nabla T, \quad X(0) = B$$

For objects:  $F = F(x, y, \theta)$

$\theta \in [0, 2\pi]$  – rotating angle of object centre

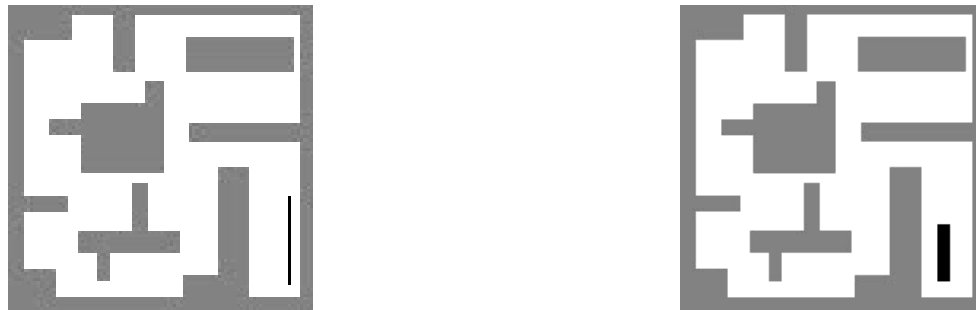


# Robotic Navigation with Constraints

Implementation (back to the piano problem):

- Fast marching method:  $|\nabla T| = F(x, y, \theta)$
- Heun's method (2<sup>nd</sup> order):  $X_t = -\nabla T, \quad X(0) = B$
- Domain (3D) discretised in  $x, y$  and  $\theta$

Results:



Movies of moving a piano in an apartment

# Summary

- LSM and FMM can be used in all areas where evolving interface is present
- The velocity on the interface (from underlying physics/chemistry) can be obtained using either a finite difference or finite element grid
- **Negative:** The implementation is not simple
- **Positive:** The algorithm is robust, accurate and versatile for complex problems