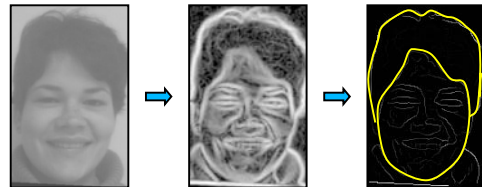


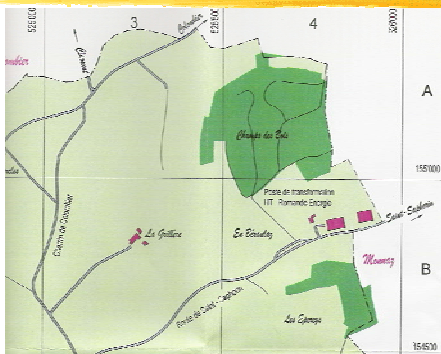
DELINEATION

Pascal Fua
CVLab
EPFL, Lausanne

FROM LOCAL EDGES TO OUTLINES



MAPS

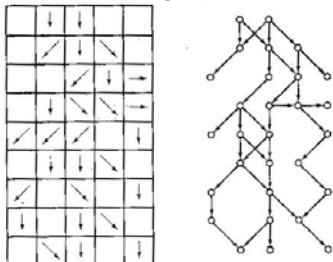


TECHNIQUES

- Graph search
- Dynamic programming
- Hough transform
- Deformable models

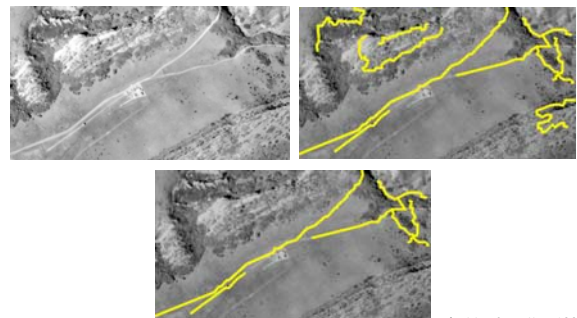
MINIMUM SPANNING TREE

Image modeled as a graph:



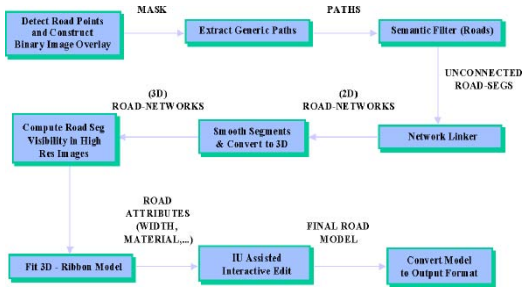
-> Generate minimal distance graph ($N \log(N)$ algorithm)

ROAD DELINEATION



Fischler & Heller, 1998.

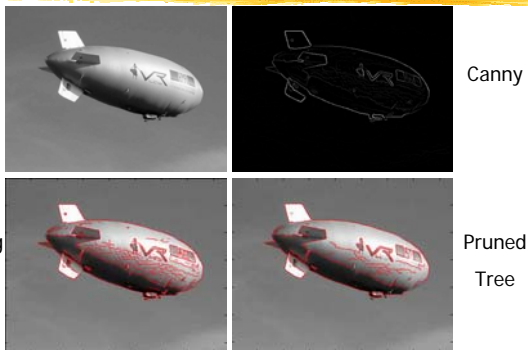
FROM IMAGE TO ROADS



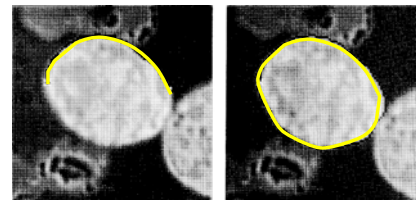
PROCESSING STEPS



BLIMP



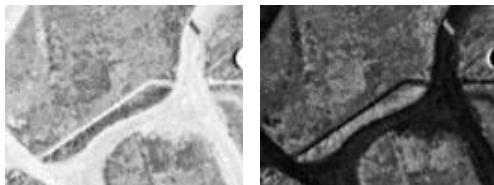
DYNAMIC PROGRAMMING



$$h(x_1, x_2, \dots, x_n) = \sum_{k=1}^n g(x_k) + \sum_{k=1}^{n-1} q(x_k, x_{k+1})$$

$$q(x_k, x_{k+1}) = \text{diff}[\phi(x_k), \phi(x_{k+1})]$$

ROAD IMAGE

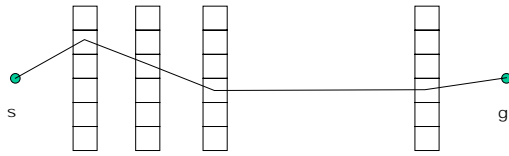


LIVE WIRE



Mortensen and Barrett, SIGGRAPH, 1995

DYNAMIC PROGRAMMING



N locations

Q quantized values

→ Global Optimum in $O(NQ^2)$

DYNAMIC PROGRAMMING IN 1-D

To find

$$\max_{x_i} h(x_1, x_2, x_3, x_4)$$

where

$$h(\) = h_1(x_1, x_2) + h_2(x_2, x_3) + h_3(x_3, x_4)$$

define

$$f_1(x_2) = \max_{x_1} h_1(x_1, x_2)$$

$$f_2(x_3) = \max_{x_2} (h_2(x_2, x_3) + f_1(x_2))$$

$$f_3(x_4) = \max_{x_3} (h_3(x_3, x_4) + f_2(x_3))$$

$$\Rightarrow \max_{x_i} h(x_1, x_2, x_3, x_4) = \max_{x_4} f_3(x_4)$$

DIJKSTRA'S ALGORITHM

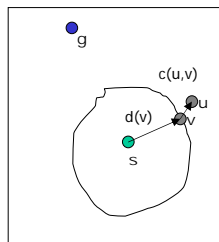
Notations:

s Start point

L List of active nodes

$c(u, v)$ Local costs for link $u \rightarrow v$

$d(v)$ Total costs from v to s



DIJKSTRA'S ALGORITHM

Algorithm:

$d(s) = 0$ and $d(u) = \infty$ for $u \neq s$

$L = \{s\}$

while L not empty do

$u = \min(L)$

for each edge $u \rightarrow v$ with v not processed do

add v to L

if $d(v) > d(u) + c(u, v)$ then

$d(v) = d(u) + c(u, v)$

fi

od

remove u from L

end

INTELLIGENT SCISSORS

- Sorting is the expensive operation. Normally $n \log(n)$, but can be reduced to $\log(n)$ if all costs are integer costs

- Local costs computed using gradient:

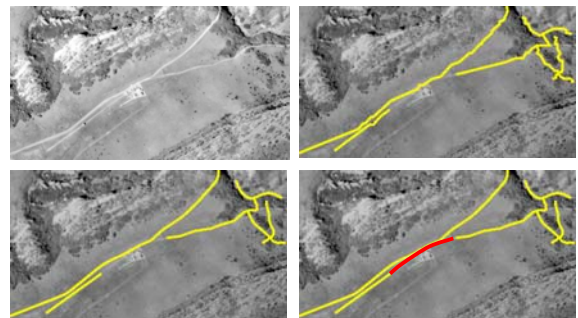
$$c(u, v) = 255 - 1/2(g(u) + g(v))$$

- Diagonal penalized by multiplying cost of non diagonal edges by:

$$\frac{1}{\sqrt{2}} = \frac{5}{7}$$

- Add a constant cost for each edge.

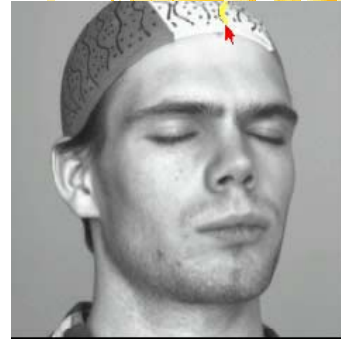
ROAD DELINEATION



FACE IMAGE



LIVE WIRE



OPEN ISSUES

- The "optimal" path is not always the "best" one.
- Difficult to impose global constraints.
- The cost grows exponentially with the dimension of the problem.

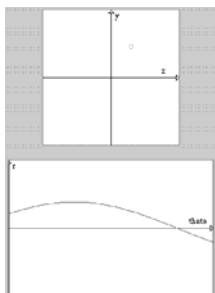
--> Must often look for local, as opposed to global, optimum using gradient descent techniques.

HOUGH TRANSFORM

Given a parametric model of a curve:

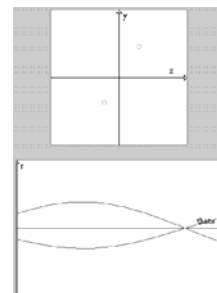
- Map each contour point onto the set of parameter values for which the curves passes through it.
- Find the intersection for all parameter sets thus mapped.

STRAIGHT LINES

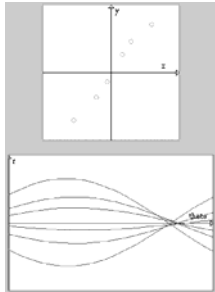


$$x \cos(\theta) + y \sin(\theta) = r, 0 \leq \theta < \pi$$

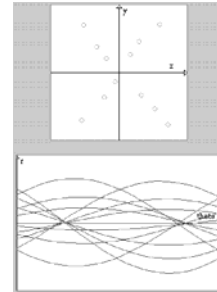
STRAIGHT LINES



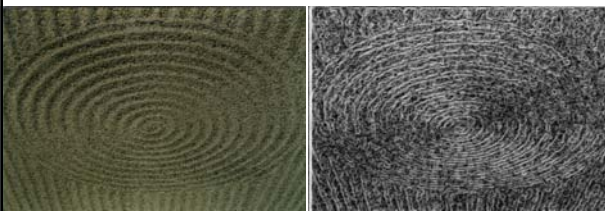
STRAIGHT LINES



STRAIGHT LINES



CIRCLES



ALGORITHM

1. Quantize parameter space (1 dimension per parameter)
2. Form an accumulator array
3. For each point in the gradient image such that the gradient strength exceeds a threshold, increment appropriate element of the accumulator
4. Find local maxima in the accumulator

ELLIPSE DETECTION

Ellipse of equation:

$$x = x_0 + a \cos(\theta)$$

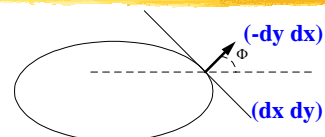
$$y = y_0 + b \sin(\theta)$$

Therefore:

$$x_0 = x - a \cos(\theta)$$

$$y_0 = y - b \sin(\theta)$$

ELLIPSE DETECTION



For each ellipse point:

$$\frac{dy}{dx} = \frac{b}{a} \cdot \frac{\cos(\theta)}{\sin(\theta)}$$

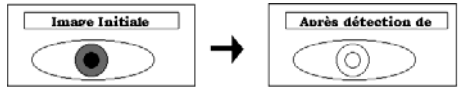
$$-\frac{dx}{dy} = \frac{a}{b} \cdot \tan(\theta)$$

The accumulator need only be incremented for:

$$\theta = \text{atan}\left(\frac{b}{a} \tan(\Phi)\right)$$

EYE DETECTION

In theory:

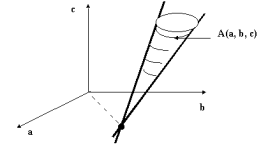


In practice:

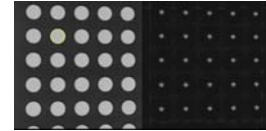


SIMPLE IMAGE

Voting scheme:

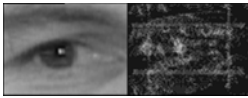


Result:

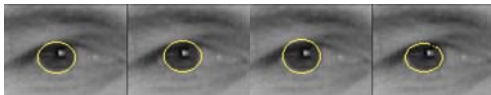


EYE IMAGES

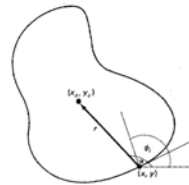
Image and accumulator:



Best four candidates:



GENERALIZED HOUGH



Angle measured from figure boundary to reference point	Set of radii $\{r^i\}$ where $r = (r, \alpha)$
ϕ_1	$r_1^1, r_1^2, \dots, r_1^m$
ϕ_2	$r_2^1, r_2^2, \dots, r_2^m$
\vdots	\vdots
ϕ_n	$r_n^1, r_n^2, \dots, r_n^m$

Code the shape in the form of an R-Table:
Set of possible positions of a reference point given boundary orientation.

--> Generalized template matching.

ALGORITHM

1. Make an R-table for the shape to be located.
2. Form an accumulator array of possible reference points initialized to zero: $A((x_{min}, x_{max}), (y_{min}, y_{max}))$
3. For each edge point,
 - Compute the possible centers, that is, for each table entry, compute

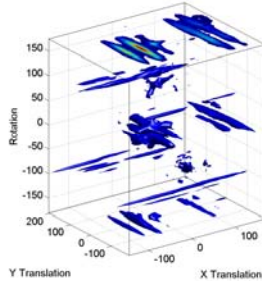
$$x = x_e + r_\phi \cos(\alpha(\phi))$$

$$y = y_e + r_\phi \sin(\alpha(\phi))$$
 - Increment the accumulator array

REAL-TIME HOUGH



ACCUMULATOR



LIMITATIONS

Computational cost grows exponentially with the number of model parameters:

- Only works for objects whose shape can be defined by a small number of parameters.
- Approach is robust but lacks flexibility.

SNAKES

Deformable Contours that

- Maximize the gradient along the curve
 - Minimize their deformation energy
- > Interactive tools for contour detection.

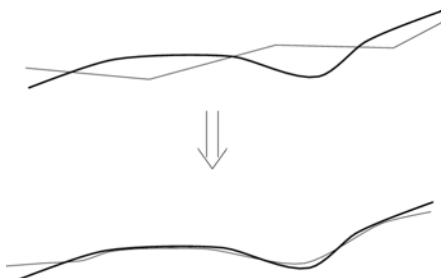
Snakes can be generalized to handle

- More sophisticated models
 - Global constraints
- > Model-Based Optimization

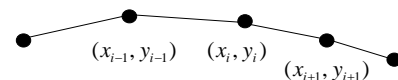
SMOOTH SNAKE



2-D SNAKE



ENERGY LANDSCAPE



$$\begin{aligned}
 E &= \frac{1}{L} \left(-\lambda \int_{s=0}^L G(x(s), y(s)) ds + \int_{s=0}^L \gamma(s)^2 ds \right) \\
 &\approx \frac{1}{L} \left(-\lambda \int_{s=0}^L G(x(s), y(s)) ds + \int_{s=0}^L \left(\left(\frac{\partial^2 x}{\partial s^2} \right)^2 + \left(\frac{\partial^2 y}{\partial s^2} \right)^2 \right) ds \right) \\
 &\approx \frac{1}{N} \left(-\lambda \sum_{i=1}^N G(x_i, y_i) + \sum_{i=2}^{N-1} \left((2x_i - x_{i-1} - x_{i+1})^2 + (2y_i - y_{i-1} - y_{i+1})^2 \right) \right)
 \end{aligned}$$

MATRIX NOTATION

$$E \approx E_G + \frac{1}{2} X' K X + \frac{1}{2} Y' K Y$$

$$X = [x_1 \cdots x_N]$$

$$Y = [y_1 \cdots y_N]$$

$$K = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & -4 & 6 & -4 & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & -4 & 6 & -4 & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & -4 & 6 & -4 & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

LOCAL OPTIMUM

$$\frac{\partial E}{\partial X} = \frac{\partial E_G}{\partial X} + KX = 0$$

$$\frac{\partial E}{\partial Y} = \frac{\partial E_G}{\partial Y} + KY = 0$$

But K is *not* invertible!

DYNAMICS

Embed curve in a viscous medium and solve at each step:

$$\frac{\partial E}{\partial X} + \alpha \frac{dX}{dt} = 0$$

$$\frac{\partial E}{\partial Y} + \alpha \frac{dY}{dt} = 0$$

ITERATING

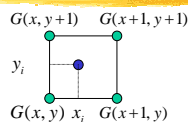
At every step:

$$KX_t + \alpha(X_t - X_{t-1}) = -\frac{\partial E_G}{\partial X} \Rightarrow (K + \alpha I)X_t = \alpha X_{t-1} - \frac{\partial E_G}{\partial X}$$

$$KY_t + \alpha(Y_t - Y_{t-1}) = -\frac{\partial E_G}{\partial Y} \Rightarrow (K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\partial E_G}{\partial Y}$$

→ Solve two linear equations at each iteration.

DERIVATIVES OF THE GRADIENT

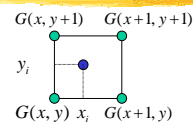


$$E_D = -\frac{\lambda}{N} \sum_{i=1}^N G(x_i, y_i)$$

$$\frac{\partial E_D}{\partial X} = \begin{bmatrix} \frac{\partial E_D}{\partial x_1} & \cdots & \frac{\partial E_D}{\partial x_N} \end{bmatrix}, \quad \frac{\partial E_D}{\partial Y} = \begin{bmatrix} \frac{\partial E_D}{\partial y_1} & \cdots & \frac{\partial E_D}{\partial y_N} \end{bmatrix}$$

$$\frac{\partial E_D}{\partial x_i} = -\frac{\lambda}{N} \frac{\partial G}{\partial x_i}(x_i, y_i), \quad \frac{\partial E_D}{\partial y_i} = -\frac{\lambda}{N} \frac{\partial G}{\partial y_i}(x_i, y_i)$$

BILINEAR INTERPOLATION



$$p = x - x_i$$

$$q = y - y_i$$

$$G(x_i, y_i) = (1-p)(1-q)G(x, y) + (1-p)qG(x, y+1) + p(1-q)G(x+1, y) + pqG(x+1, y+1)$$

$$\frac{\partial G}{\partial x_i} = (1-q)(G(x+1, y) - G(x, y)) + q(G(x+1, y+1) - G(x, y+1))$$

$$\frac{\partial G}{\partial y_i} = (1-p)(G(x, y+1) - G(x, y)) + p(G(x+1, y+1) - G(x+1, y))$$

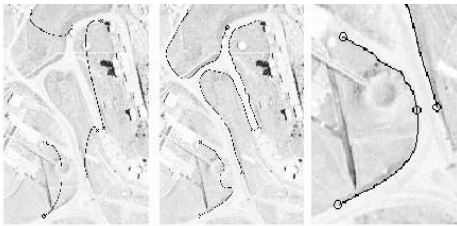
SMOOTH SNAKE



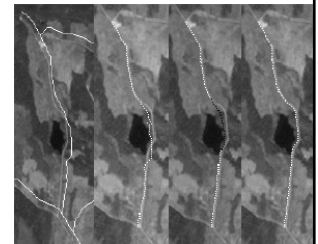
AERIAL IMAGE



ZIPLOCK SNAKES



UPDATING AN OLD MAP



Original and Corrected

Ziplock optimization

RIBBON SNAKES



$$E \approx E_G + \frac{1}{2} X' K X + \frac{1}{2} Y' K Y + \frac{1}{2} W' K_w W$$

$$W = [w_1 \dots w_N]$$

$$K = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & -1 & 2 & -1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & -1 & 2 & -1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

DYNAMICS EQUATIONS

$$(K + \alpha I) X_t = \alpha X_{t-1} - \frac{\partial E_G}{\partial X}$$

$$(K + \alpha I) Y_t = \alpha Y_{t-1} - \frac{\partial E_G}{\partial Y}$$

$$(K_w + \alpha I) W_t = \alpha W_{t-1} - \frac{\partial E_G}{\partial W}$$

→ Solve three linear equations at each iteration.

DELINEATING ROADS



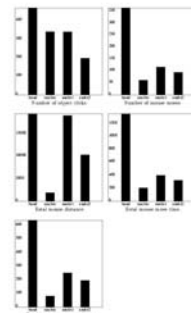
DELINEATING ROADS



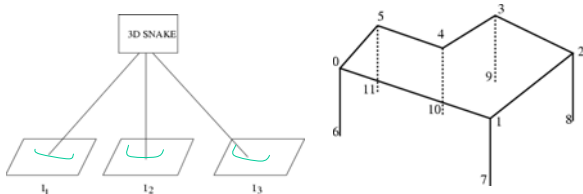
DELINEATING ROADS



EVALUATION



3—D SNAKES



Smooth 3—D snake

Rectilinear 3—D snake

DYNAMICS EQUATIONS

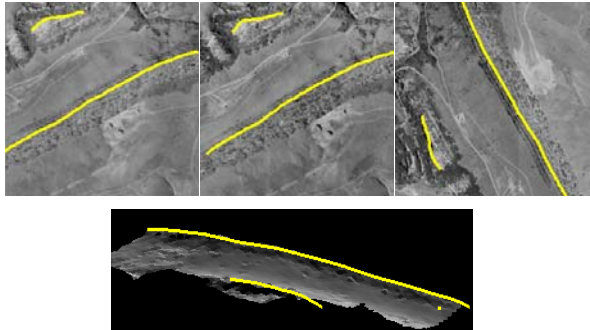
$$(K + \alpha I)X_t = \alpha X_{t-1} - \frac{\partial E_G}{\partial X}$$

$$(K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\partial E_G}{\partial Y}$$

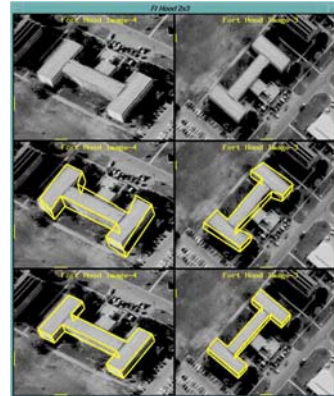
$$(K + \alpha I)Z_t = \alpha Z_{t-1} - \frac{\partial E_G}{\partial Z}$$

→ Solve three linear equations at each iteration.

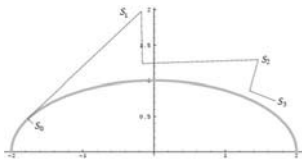
RIDGE LINES



BUILDINGS

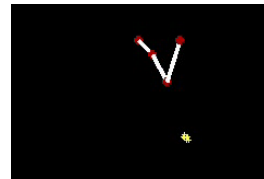


CONSTRAINED OPTIMIZATION

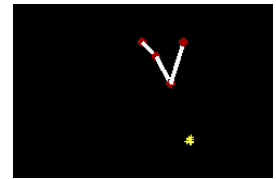


Minimize $F(S)$ subject to $C(S) = 0$

CONSTRAINED OPTIMIZATION



Unconstrained



Constrained

SITE MODELING



AUGMENTED REALITY



LEVEL SETS



CURVE EVOLUTION

Generic formulation: $\frac{\partial C}{\partial t} = \alpha(s, t)\vec{T} + \beta(s, t)\vec{N}$

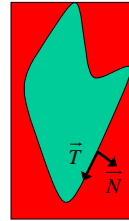
Reparametrization: $\frac{\partial C}{\partial t} = \beta(s, t)\vec{N}$

Special cases:

- prairie fire model: $\beta = \pm 1$
- reaction and diffusion model: $\beta = \beta_0 - \beta_1 \kappa$

Problems:

1. Involves curve sampling and resampling.
2. Curvature estimates are numerically unstable.



LEVEL SETS

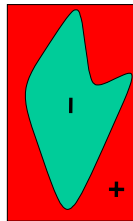
Consider the curve as the zero level set of the surface.

$$z = \Phi(x, y, t)$$

with evolution equation $\Phi_t + \beta(\kappa)|\nabla\Phi| = 0$

where

$$\kappa = \frac{(\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2)}{(\Phi_x^2 + \Phi_y^2)}$$



→ much better numerical stability.

LEVEL SET SMOOTHING

Smoothing occurs when $\beta(\kappa) = -\kappa$

Desirable properties:

- Converges towards circles.
- Total curvature decreases.
- Number of curvature extrema and zeros of curvature decreases

Relationship with Gaussian smoothing:

- Analogous to Gaussian smoothing of boundary over the short run, but does not cause self-intersections or overemphasize elongated parts.
- Can be implemented by Gaussian smoothing the characteristic function of a region.

SHAPE RECOVERY

Evolution equation: $\Phi_t + \beta(\kappa)|\nabla\Phi| = 0$

$$\beta(\kappa) = k_f(1 - \epsilon\kappa)$$

$$k_f = \frac{1}{1 + \sqrt{I}}$$

→ Expansion stops at the boundaries.

LEVEL SETS



LEVEL SETS



IN SHORT

- Edge information is noisy.
 - Models are required to make sense of it.
- Semi-automated approaches are one good way to do this.