SNAKES

- Dynamic, elastic model of curve shape
- Aka Active Contours, Deformable Contours
- Associate with each possible contour position and shape an energy functional defining the potential energy of the contour
- Find minimum of energy functional (usually local min only)

Initial State — Initial contour configuration (may be user-defined or initial estimate by another process)

↓

Final State — steady state or equilibrium state

Modeling Contour Shape

- Define energy functional in terms of internal constraints (i.e., contour shape) and external constraints (i.e., image features)
- Feature Extraction and contour constraints integrated into a single process
- Given contour $C = \mathbf{v}(s) = (x(s), y(s))$, where $0 \le s \le 1$ is normalized arc length define energy functional:

$$E = \int_0^1 \Big[\underbrace{E_{int}(v(s))}_{\text{internal constraints}} + \underbrace{E_{image}(v(s)) + E_{con}(v(s))}_{\text{external constraints}}\Big]\,ds$$

- GOAL: Minimize E

Internal Energy

- Characterize desired shape in terms of:

  * Continuity
  Degree of rigidity / stretching

  $$E_{continuity} = \left\| \frac{dv}{ds} \right\|^2$$

  Magnitude of 1^{st} deriv. of v

  * Smoothness
  Degree of bending / oscillating
  ⇒ penalize high curvatures

  $$E_{smoothness} = \left\| \frac{d^2v}{ds^2} \right\|^2$$

  Magnitude of 2^{nd} deriv of v

- Combining, we get

  $$E_{int} = \left(\alpha(s) E_{continuity} + \beta(s) E_{smoothness} \right)$$

  α, β control relative influence of terms as a function of position

Internal Energy (cont.)

- E_{int} minimized for straight line

  $$\left\| \frac{dv}{ds} \right\|^2 = 0, \quad \left\| \frac{d^2v}{ds^2} \right\|^2 = 0$$

- $\beta(s_0) = 0$ ⇒ ignore smoothness term at s_0
  ⇒ tangent discontinuity ok at s_0
  ⇒  ok

- $\alpha(s_0) = 0$ ⇒ ignore continuity term at s_0
  ⇒ position discontinuity ok at s_0
  ⇒  ok

Discretization

- In practice, C represented by a discrete set of ordered points called snaxels, $p_1, \ldots, p_n$, with contour defined by these points
- $\left\| \frac{dv}{ds} \right\|^2 \approx \| p_i - p_{i-1} \|^2$
- To prevent "bunching" of snaxels, instead use:

  $$\left(\bar{d} - \| p_i - p_{i-1} \| \right)^2$$ ← use in HW

  where $\bar{d} = \frac{1}{n} \sum \| p_i - p_{i-1} \|$

  $\Rightarrow$ If $\| p_i - p_{i-1} \| \gg \bar{d}$ then

  $$E_{continuity} \approx \| p_i - p_{i-1} \|^2$$

  * Minimum when all points equally spaced

Discretization (cont.)

- $E_{smoothness} \approx \| p_{i-1} - 2p_i + p_{i+1} \|^2$

  (approximates curvature well if enough points, since points are nearly evenly spaced)

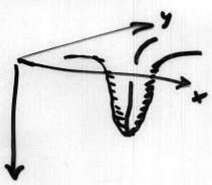
- $E_{int} = \sum_{i=0}^{n-1} \alpha(i) E_{continuity} + \beta(i) E_{smoothness}$

External Energy

- Forces due to image
- Image features used to define smooth gravitational potential energy

$z = H(x,y)$

height z defines potential energy at (x,y)



Contour C constrained to lie on 3D surface H and C moves "down" under gravitational pull

$E_{ext}(v) = g \cdot z(v)$

Defining E_{image}

- E_{image} depends only on contour, C, not on its derivatives wrt s.
- E_{image} small when C near "good" images features, and large when far from good features
  $\Rightarrow$ attraction to features of interest
- What image features?
  E.g. brightness, edges, lines, endpoints

$$E_{image} = w_1 E_{intensity} + w_2 E_{edge} + w_3 E_{line} + w_4 E_{endpt}$$

- $E_{intensity}(i) = I(x_i, y_i)$ — attraction to dark ~~points~~
- $E_{edge}(i) = -\|\nabla I(x_i, y_i)\|^2$ — attraction to strongest edge pts (where gradient is large)
- Want smooth E_{image}

Other External Forces, E_{con}

- Springs

  Tension (attraction) force between a snaxel, p_i, and an image point

  $$E_{spring} = -K_{spring}(p_{image} - p_{snake})^2$$

  $\Rightarrow$ apply for each (snaxel, spring) pair

- Volcanoes

  Local repulsion force by deforming H

  $$E_{volcano} = -K_{volcano}/r$$

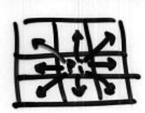
  * Add conic surface to image
  * Prevents snake from getting stuck in local min or valley

Solving for Local Min of E

- Given initial snaxels, $p_1, \ldots, p_n$, weights α, β, etc. and image defining E_{image}, find the contour defined by $p_1', \ldots, p_n'$ that minimizes the energy functional

- A Greedy Algorithm

  At each iteration, consider a small neighborhood of each snaxel, p_i, and find location where energy is min

  9 possible moves of p_i

- Stop when number of pts moved < threshold

- Update $\bar{d}$ at end of each iteration
- Parallel vs. Serial snaxel update
  - Update all snaxels at end of iteration
  - Update snaxels in fixed order
- O(nm) time where n = # snaxels and m = # neighbors
- May oscillate & never converge (even to a local min)
- Need to normalize each energy term

Ex. $E_{continuity}$: divide by largest value in neighborhood of snaxel p_i

$E_{smoothness}$: divide by largest value in neighborhood

E_{dge} : $\dfrac{\|\nabla I\| - m}{|M - m|}$

where M = max, m = min value in neighborhood

- To allow piecewise smooth contours, detect corners:

foreach each $i = 1, \ldots, n$
  estimate curvature, k, at p_i :
  $$k = |p_{i-1} - 2p_i + p_{i+1}|$$
  detect snaxels that are local maxima of k
  set $\beta_j = 0$ for all p_j which
  1. are local maxima, and
  2. $k >$ threshold, and
  3. intensity gradient $|\nabla I|$ at $p_j >$ thres.

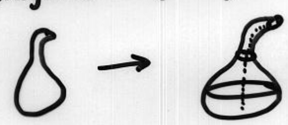
- Initial contour needs to be "close" to desired solution

⇒ coarse-to-fine search if solution is isolated in image

Other Applications

- Surface Reconstruction (Depth Map Recovery)
  Initial State = plane at depth $z=0$
  Final state = surface $z=Z(x,y)$

- 3D Object Reconstruction from 1 Image
  Building "squash models" from elongated image regions



$\Rightarrow$ determine position of "spine" + radii of circular cross-sections

- Deformable Templates
  Ex. "Eye" template (Yuille)
- Signal Matching (for Stereo or Motion)
  determine "motion field" or "disparity field"

Snakes for Tracking

- Assume: Object motion "small" between consecutive frames
- Initialize snake on object in 1st frame
- Final position of snake in previous frame is used as initial position for current frame
- If object motion is large
  - use previous motion to predict new location
  - use Kalman filtering to combine prediction and observation
- Hard to deal with
  - occlusion
  - "lost" segments of contour

Intelligent Scissors

- User <u>interactively</u>, <u>sequentially</u> moves one end of contour, and algorithm causes contour between "free endpoint" (cursor position) and starting point (seed point) to automatically "snap" to and "wrap around" object boundary
- User moves around object "snapping" successive segments to fit object
- Can "reset" seed point to any previous point to fix boundary prior to seed point
- Computes, starting at seed point, a minimum cost spanning tree of image using dynamic programming
  - spanning tree determines optimal path b/w seed pt and free pt.

Dynamic Programming for Boundary Detection

- Key Idea: At n^{th} stage, given optimal paths of length $n-1$ starting from each possible point, consider all possible extensions of length 1 and select optimal path of length n.
- Cost function for length 1 path between 2 adjacent pixels:

$$\text{cost}(p,q) = w_1 f_E(q) + w_2 f_D(p,q) + w_3 f_G(q)$$

where $f_E = 1 -$ binary edge map
$f_D =$ change in gradient direction from p to q
$f_G = -\|\nabla I\|^2$

Snakes Summary

- Advantages
  - Uses all info in image
    $\Rightarrow$ least commitment in selecting set of features
  - Contour remains connected at all times $\Rightarrow$ no gap filling
  - Can detect subjective contours

  - Good for tracking non-rigid objects
  - Info integrated along entire contour
- Disadvantages
  - Needs smooth potential surface, H
  - Needs initialization close to solution
  - Needs a priori knowledge to set parameters
  - Numerical instability