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)

\[
\begin{align*}
\text{Initial State} & \quad \text{Initial contour configuration} \\
\text{Final State} & \quad \text{steady state or equilibrium state}
\end{align*}
\]

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 \( C : v(s) = (x(s), y(s)) \)
  where \( 0 \leq s \leq 1 \) is normalized arc length, define energy functional:
  \[
  E = \int_{0}^{1} E_{\text{int}}(v(s)) + E_{\text{img}}(v(s)) + E_{\text{con}}(v(s)) ds
  \]
  \( E_{\text{int}} \) internal constraints
  \( E_{\text{img}} \) external constraints

- **GOAL:** Minimize \( E \)
Internal Energy

- Characterize desired shape in terms of:
  - Continuity
    - Degree of rigidity/stretching
    \[ E_{\text{continuity}} = \left\| \frac{d\psi}{ds} \right\|^2 \]
    - Magnitude of 1st deriv. of \( \psi \)

- Smoothness
  - Degree of bending/oscillating
  - Penalize high curvatures
  \[ E_{\text{smoothness}} = \left\| \frac{d^2\psi}{ds^2} \right\|^2 \]
  - Magnitude of 2nd deriv. of \( \psi \)

- Combining, we get
  \[ E_{\text{int}} = (\alpha(s) E_{\text{continuity}} + \beta(s) E_{\text{smoothness}}) \]
  - \( \alpha, \beta \) control relative influence of terms as a function of position

---

Internal Energy (cont.)

- \( E_{\text{int}} \) minimized for straight line
  \[ \left\| \frac{d\psi}{ds} \right\|^2 = 0, \quad \left\| \frac{d^2\psi}{ds^2} \right\|^2 = 0 \]

- \( \beta(s_0) = 0 \) \( \Rightarrow \) ignore smoothness term at \( s_0 \)
  \( \Rightarrow \) tangent discontinuity
  \( \Rightarrow \) okay at \( s_0 \)

- \( \alpha(s_0) = 0 \) \( \Rightarrow \) ignore continuity term at \( s_0 \)
  \( \Rightarrow \) position discontinuity
  \( \Rightarrow \) okay at \( s_0 \)
Discretization

- In practice, C represented by a discrete set of ordered points called Snaxals, \( P_1, \ldots, P_n \), with contour defined by these points.

\[ \left\| \frac{dv}{dx} \right\|^2 + \left\| P_i - P_{i-1} \right\|^2 \]

- To prevent "bunching" of Snaxals, instead use:

\[ (\bar{d} - \left\| P_i - P_{i-1} \right\|)^2 \]

where \( \bar{d} = \frac{1}{n} \sum_{i=1}^{n} \left\| P_i - P_{i-1} \right\| \)

\[ \Rightarrow \text{if } \left\| P_i - P_{i-1} \right\| \gg \bar{d} \text{ then} \]

\[ E_{\text{continuity}} \approx \left\| P_i - P_{i-1} \right\|^2 \]

Minimum when all points equally spaced.

Discretization (cont.)

- \( E_{\text{smoothness}} \approx \left\| P_{i-1} - 2P_i + P_{i+1} \right\|^2 \)

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

- \( E_{\text{int}} = \sum_{i=0}^{n} \alpha(i) E_{\text{continuity}} + \beta(i) E_{\text{smoothness}} \)
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 moves "down" under gravitational pull

\[ E_{\text{ext}}(v) = g \cdot z(v) \]

Defining \( E_{\text{image}} \)

- \( E_{\text{image}} \) depends only on contour \( C \), not on its derivatives \( w \) or \( s \).

- \( E_{\text{image}} \) small when \( C \) near "good" image features, and large when far from good features
  \( \Rightarrow \) attraction to features of interest

- What image features?
  - E.g., brightness, edges, lines, endpoints
  - \( E_{\text{image}} = w_1 E_{\text{intensity}} + w_2 E_{\text{edge}} + w_3 E_{\text{line}} + w_4 E_{\text{edge}} \)

- \( E_{\text{intensity}} (i) = I(x, y) \) (attraction to dark pixels)

- \( E_{\text{edge}} (i) = -\| \nabla I(x, y) \| \) (attraction to strongest edge pixels)

- Want smooth \( E_{\text{image}} \)
Other External Forces, \( E_{\text{con}} \)

- **Springs**
  - Tension (attraction) force between a snaked, \( p_i \), and an image point \( p_{i'} \): 
    \[
    E_{\text{spring}} = -k_{\text{spring}} (p_{i'} - p_i)^2
    \]
  - Apply for each (snaked, spring) pair

- **Volcanoes**
  - Local repulsion force by deforming \( H \): 
    \[
    E_{\text{volcano}} = -k_{\text{volcano}} / r
    \]

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

---

Solving for Local Min of \( E \)

- Given initial snakes, \( p_{i_1}, \ldots, p_{i_n} \), weights \( \alpha, \beta, \) etc. and image defining \( E_{\text{image}} \), find the contour defined by \( p_{i_1}, \ldots, p_{i_n} \) that minimizes the energy functional

- A Greedy Algorithm
  - At each iteration, consider a small neighborhood of each snaked, \( p_i \), and find location where energy is min

\[
\begin{array}{c|c|c|c|c|c|c|c|c|c}
  & a & b & c & d & e & f & g & h & i \\
\hline
a & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\end{array}
\]

9 possible moves

- Stop when number of pts moved < threshold
- Update $\mathcal{J}$ at end of each iteration
- Parallel vs. Serial smooth update
  - Update all smooths at end of iteration
  - Update smooths in fixed order
- $O(n^2)$ time, where $n = \# smooths$ and $n = \# neighbors$
- May oscillate & never converge (even to a local min)

Needed to normalize each energy term

Example: Continuity: divide by largest value in neighborhood of smooth $p_i$

Smoothness: divide by largest value in neighborhood

$$\text{Edge} = \frac{\|s_i\| - m}{|m - m'|}$$

where $m = \max, m = \min$ value in neighborhood

- To allow piecewise smooth contours, detect corners:
  
  for each $i = 1, \ldots, n$
  estimate curvature, $k$, at $p_i$:
  $$k = \frac{|p_{i-1} - 2p_i + p_{i+1}|}{\|p_{i-1} - 2p_i + p_{i+1}\|}$$
  detect smooths that are local maxima of $k$
  set $k_i = 0$ for all $p_j$ which share local maxima, and
  
  $\circ k > \text{threshold}$, and
  
  $\circ$ intensity gradient $|\nabla I|$ at $p_j > \text{thr.}$

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

- Course-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 = f(x, y)

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

  \[ \Rightarrow \text{determine position of "spine"} \]
  + radii of circular cross-sections

- Deformable Templates
  Ex. "Eye" template (yei) with motion (for stereo or motion)
  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 interactively, sequentially 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.

- Computer, starting at seed point, constructs minimum cost spanning tree of image using dynamic programming:
  - spanning tree determines optimal path from seed pt and free pt.

Dynamic Programming for Boundary Detection

- Key Idea: At nth 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:

\[
\text{Cost}(p,q) = w_1 f_e(q) + w_2 f_0(p,q) + w_3 f_g(q)
\]

where
- \( f_e \) = binary edge map
- \( f_0 \) = change in gradient
- \( f_g \) = distance from \( p \) to \( q \)

\[
f_g = -\| \nabla f \|_2
\]
Snakes Summary

- **Advantages**
  - Uses all info in image
  - Least commitment in selecting set of features
  - Contour remains connected at all times => no gap filling
  - Can detect subjective contours
  - Good for tracking non-rigid objects
  - Data integrated along entire contour

- **Disadvantages**
  - Needs smooth potential surface, \( H \)
  - Needs initialization close to solution
  - Needs a priori knowledge to set parameters
  - Numerical instability