CS664 Lecture #23: Curve evolution, active contour models

Some material taken from:

- **Yuri Boykov, Western Ontario**
- **Nikos Paragios, Ecole Centrale de Paris**

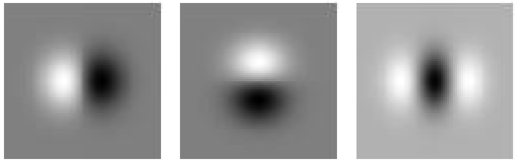
http://cermics.enpc.fr/~paragios/tutorial.ppt

Announcements

- Paper report due today (11/15)
- 1-paragraph final project description due by email on 11/23
- Final quiz will be on 11/29
- PS3 will be out soon, due Friday 12/2
- Final project will be due Thursday 12/15

Parts-based Face Detection

- Five parts: eyes, tip of nose, sides of mouth
- Each part a local image patch
  - Represented as response to oriented filters



  - 27 filters at 3 scales and 9 orientations
  - Learn coefficients from labeled examples
- Parts translate with respect to central part, tip of nose

Face Detection Results

- Runs at several frames per second
  - Compute oriented filters at 27 orientations and scales for part cost m_i
  - Distance transform m_i for each part other than central one (nose tip)
  - Find maximum of sum for detected location

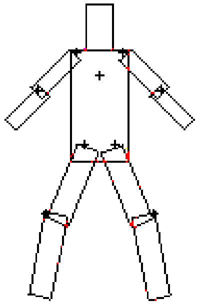
General case via DP

- Want to minimize $\sum_V m_j(l_j) + \sum_E d_{ij}(l_i,l_j)$ over (V,E)
- Can express this as a function $B_j(l_i)$
  - Cost of best location of v_j given location l_i of v_i
- Recursion in terms of children C_j of v_j
  - $B_j(l_i) = \min_{l_j}(m_j(l_j) + d_{ij}(l_i,l_j) + \sum_{C_j} B_C(l_j))$
  - For leaf node no children, so last term empty
  - For root node no parent, so second term empty

Further optimization via DT

- This recurrence can be solved in time $O(ns^2)$ for s locations, n parts
  - Still not practical since s is in the millions
- Couple with distance transform method for finding best pair-wise locations in linear time
  - Resulting $O(ns)$ method!

Example: Finding People



- Ten part 2D model
  - Rectangular regions for each part
  - Translation, rotation, scaling of parts
- Configurations may allow parts to overlap

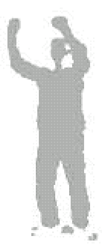
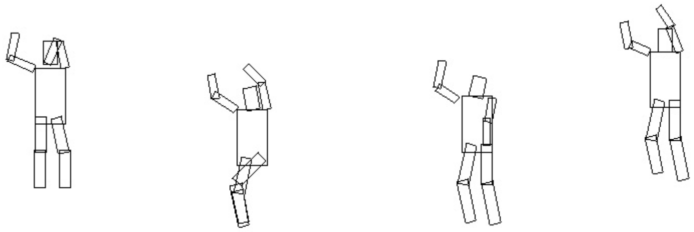
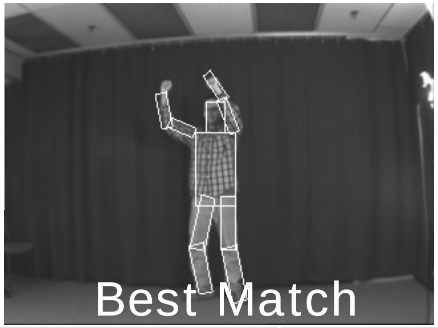


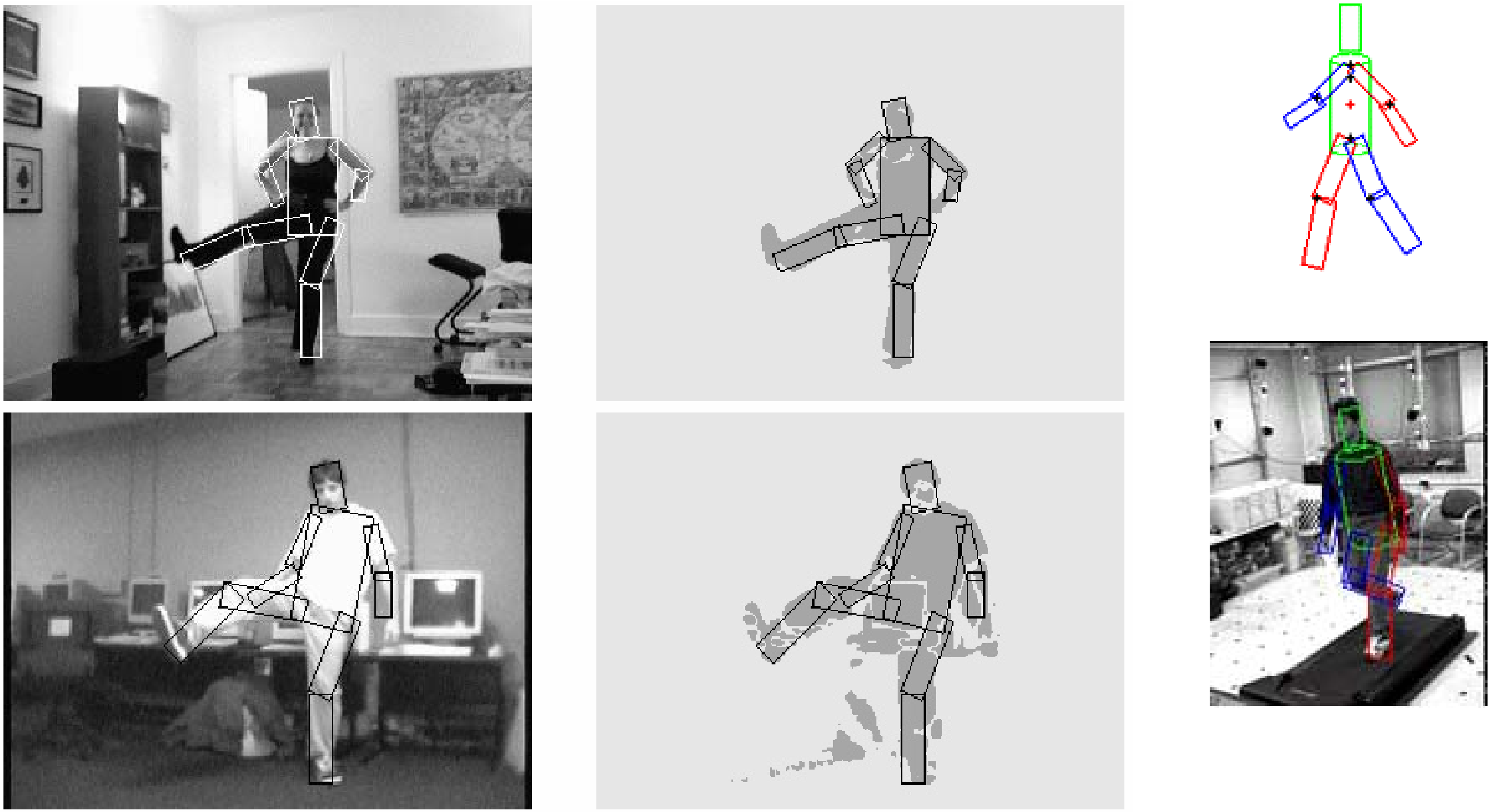
Image Data



Likely Configurations



More examples

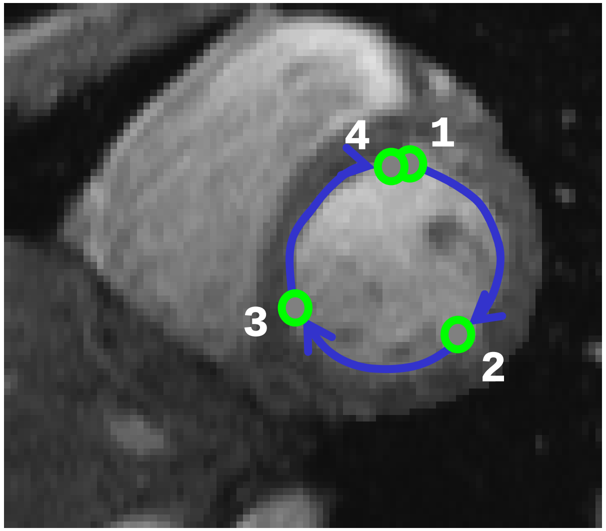


DP on curves

- For most of this course we've focused on energy minimization over an **image**
  - We can also compute the energy of a **curve**
  - Important special case: energy = length
- Curve representations
  - Control points (spline)
  - Set of edges in a graph
  - Continuous parameterizations
- DP is largely restricted to 1D objects, so it's a natural match for curves

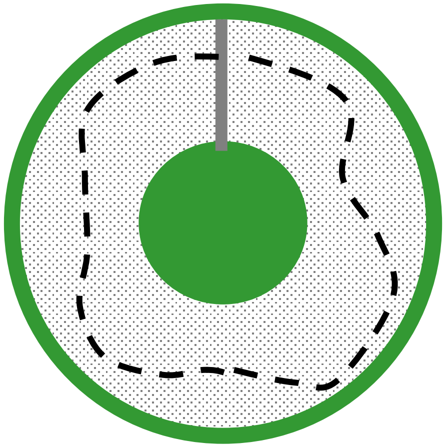
Shortest paths segmentation

- "Intelligent scissors" or "Live wire"
  - Shortest paths on image graph connect seeds, which the user places on the boundary



Minimizing user interaction

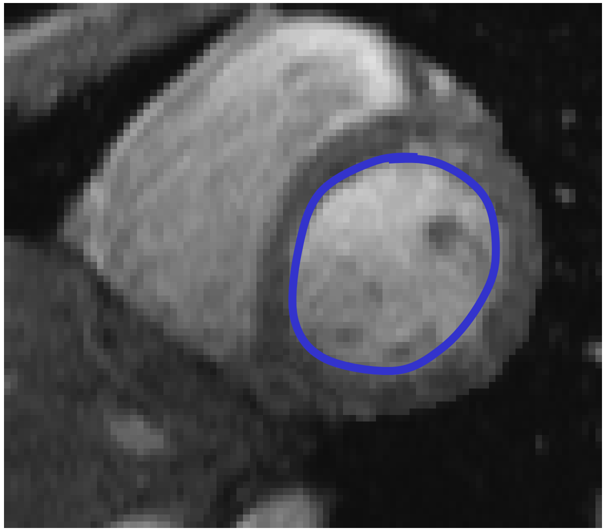
- Suppose we only know roughly where the object is. Can we do without seed points?



  – Minimize over starting points on the gray band

Active contours (snakes)

- Start with a curve near the object
  - **Evolve** the curve to fit the boundary
  - Sample application: tracking
    - Input for t+1 = output from t



Snake energy function

- Energy function on a snake has two terms
  - Does this sound familiar?

$$E_{total} = E_{in} + E_{ex}$$

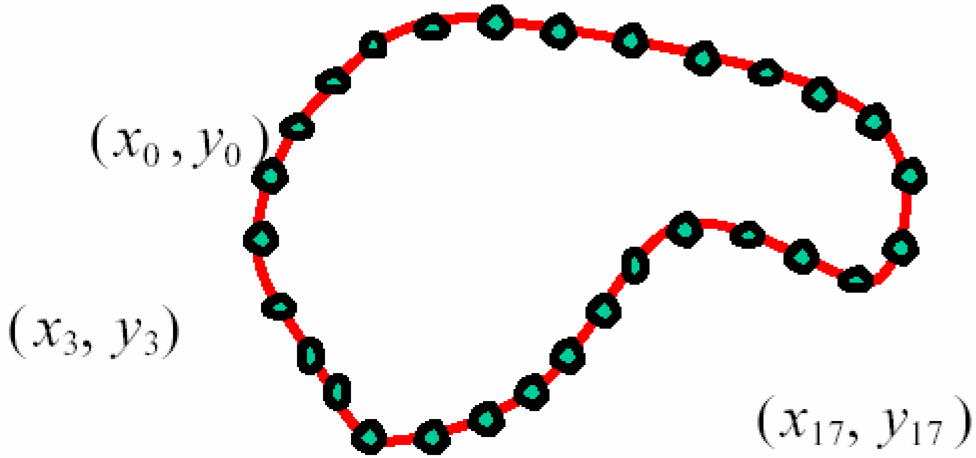
Internal energy encourages smoothness or any particular shape

Internal energy incorporates prior knowledge about object boundary allowing to extract boundary even if some image data is missing

External energy encourages curve onto image structures (e.g. image edges)

Discrete snake formulation

- Use a spline with control points $v_i = (x_i, y_i)$



Discrete external energy

- Want to attract the snake to edges

$$E_{ex} = -\sum_i |G_x(x_i, y_i)|^2 + |G_y(x_i, y_i)|^2$$

$$G_x = \frac{\partial}{\partial x} G_\sigma \otimes I$$

$$G_y = \frac{\partial}{\partial y} G_\sigma \otimes I$$

Discrete internal energy

$$\frac{dv}{ds} \approx v_{i+1} - v_i \qquad \frac{d^2v}{ds^2} \approx (v_{i+1} - v_i) - (v_i - v_{i-1}) = v_{i+1} - 2v_i + v_{i-1}$$

$$E_{in} = \sum_{i=0}^{n-1} \boxed{\alpha \left| v_{i+1} - v_i \right|^2} + \boxed{\beta \left| v_{i+1} - 2v_i + v_{i-1} \right|^2}$$

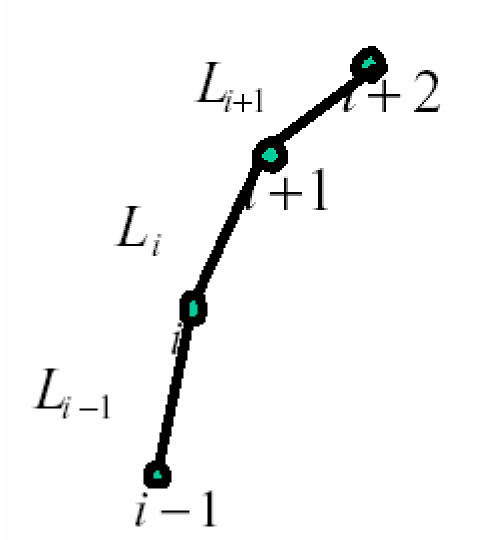
Elasticity

Stiffness

Simple elastic curve

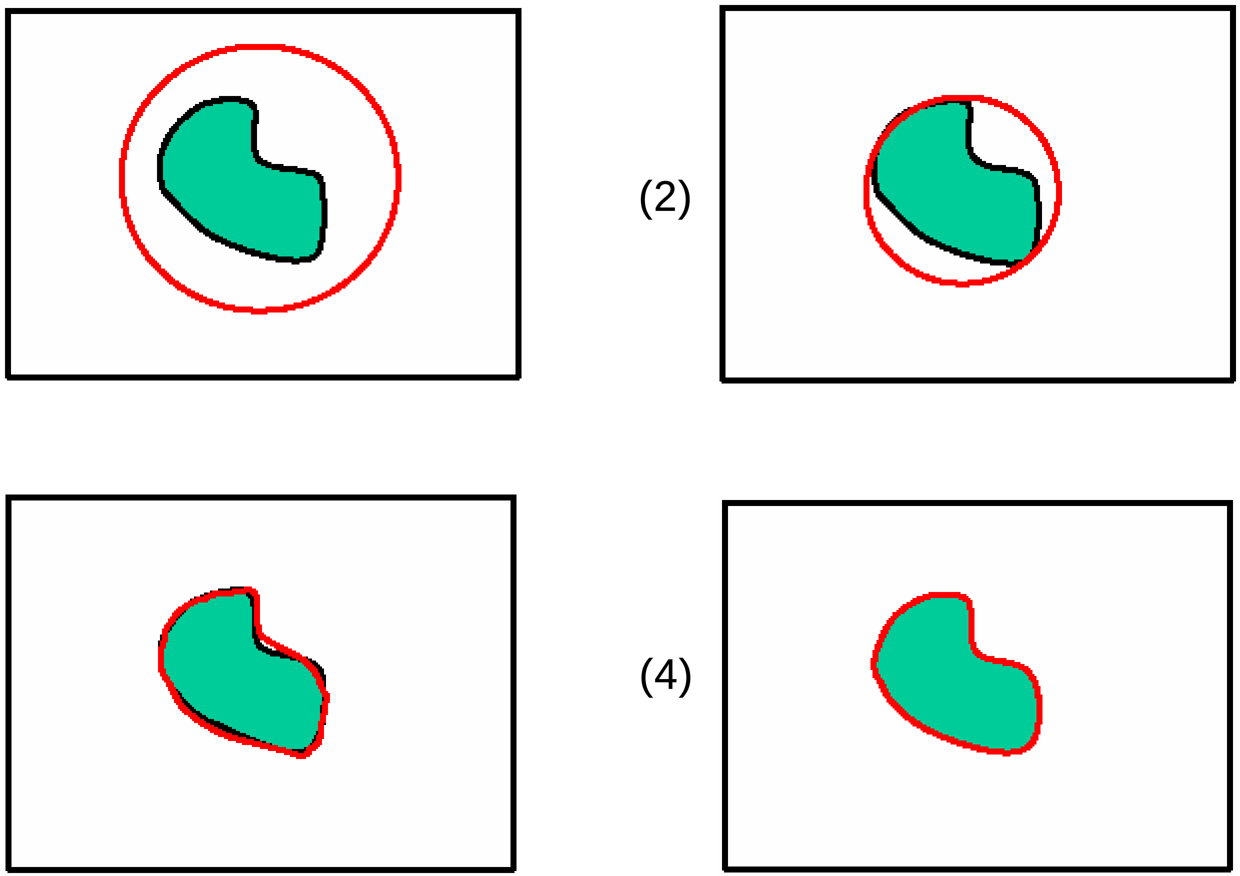
- For a curve represented as a set of points a simple elastic energy term is

$$E_{in} = \alpha \cdot \sum_{i=0}^{n-1} L_i^{\,2}$$
$$= \alpha \cdot \sum_{i=0}^{n-1} (x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2$$



This encourages the closed curve to shrink to a point (like a very small elastic band)

Synthetic example



(1) (2) (3) (4)

Snake energy

$$E_{total}(v_0,\ldots,v_{n-1}) \quad = \quad -\sum_{i=0}^{n-1} \| G(v_i) \|^2 \quad + \quad \alpha \cdot \sum_{i=0}^{n-1} \| v_{i+1} - v_i \|^2$$

$$\boxed{E_{total}(v_0,\ldots,v_{n-1}) \quad = \quad \sum_{i=0}^{n-1} E_i(v_i, v_{i+1})}$$

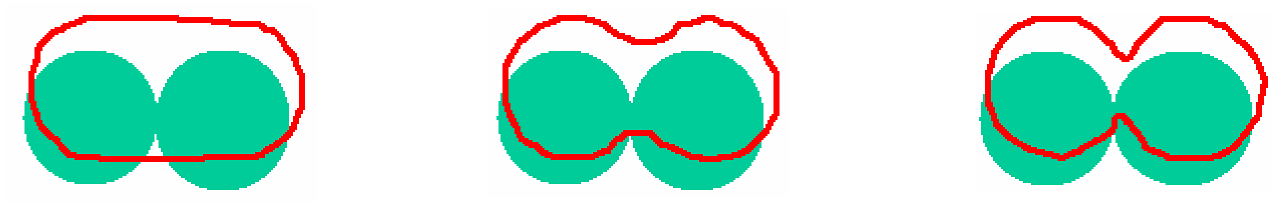
where $\quad E_i(v_i, v_{i+1}) = -\| G(v_i) \|^2 + \alpha \| v_i - v_{i+1} \|^2$

Relative weighting of terms

- Notice that the strength of the internal elastic component can be controlled by the parameter α

$$E_{in} = \alpha \cdot \sum_{i=0}^{n-1} L_i^{\,2}$$

- Increasing this increases curve stiffness

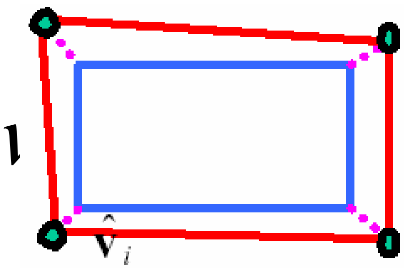


Large α Medium α Small α

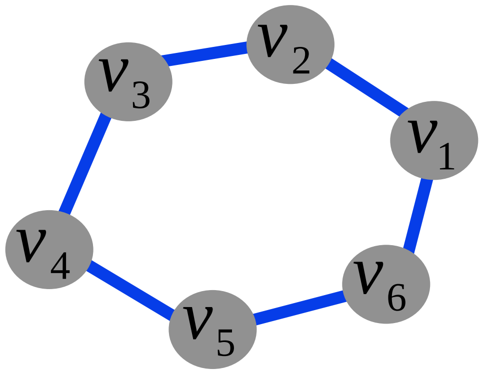
Some variants

Avoid shrinkage: $E_{in} = \alpha \cdot \sum_{i=0}^{n-1} (L_i - \hat{L}_i)^2$

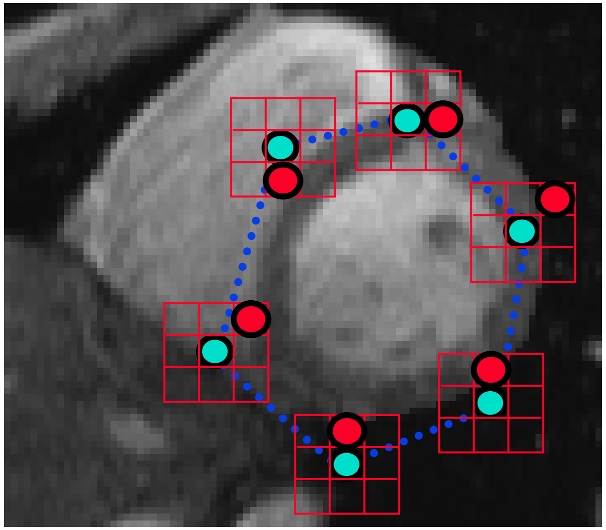
Prefer known shape: $E_{in} = \alpha \cdot \sum_{i=0}^{n-1} ($



Dynamic programming



control points

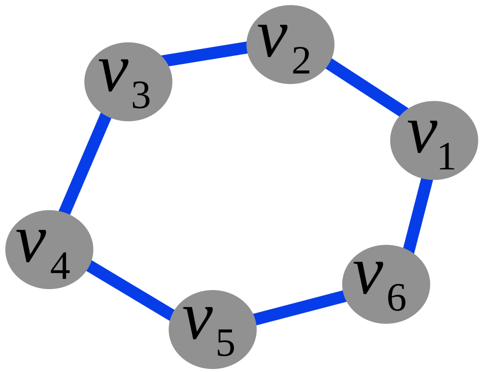


First-order interactions

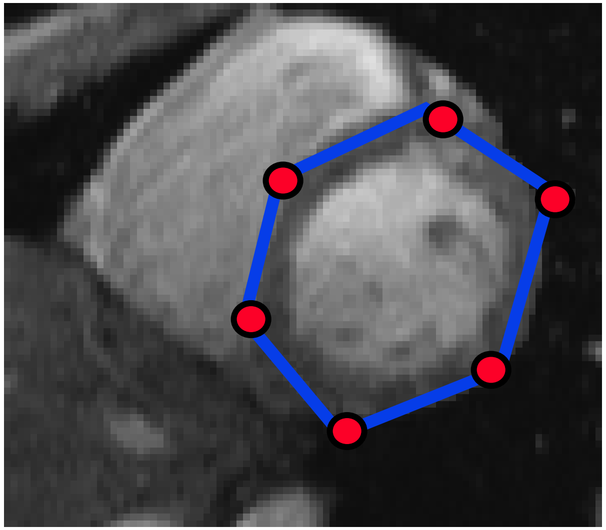
$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$

Energy E is minimized via Dynamic Programming

Dynamic programming



control points



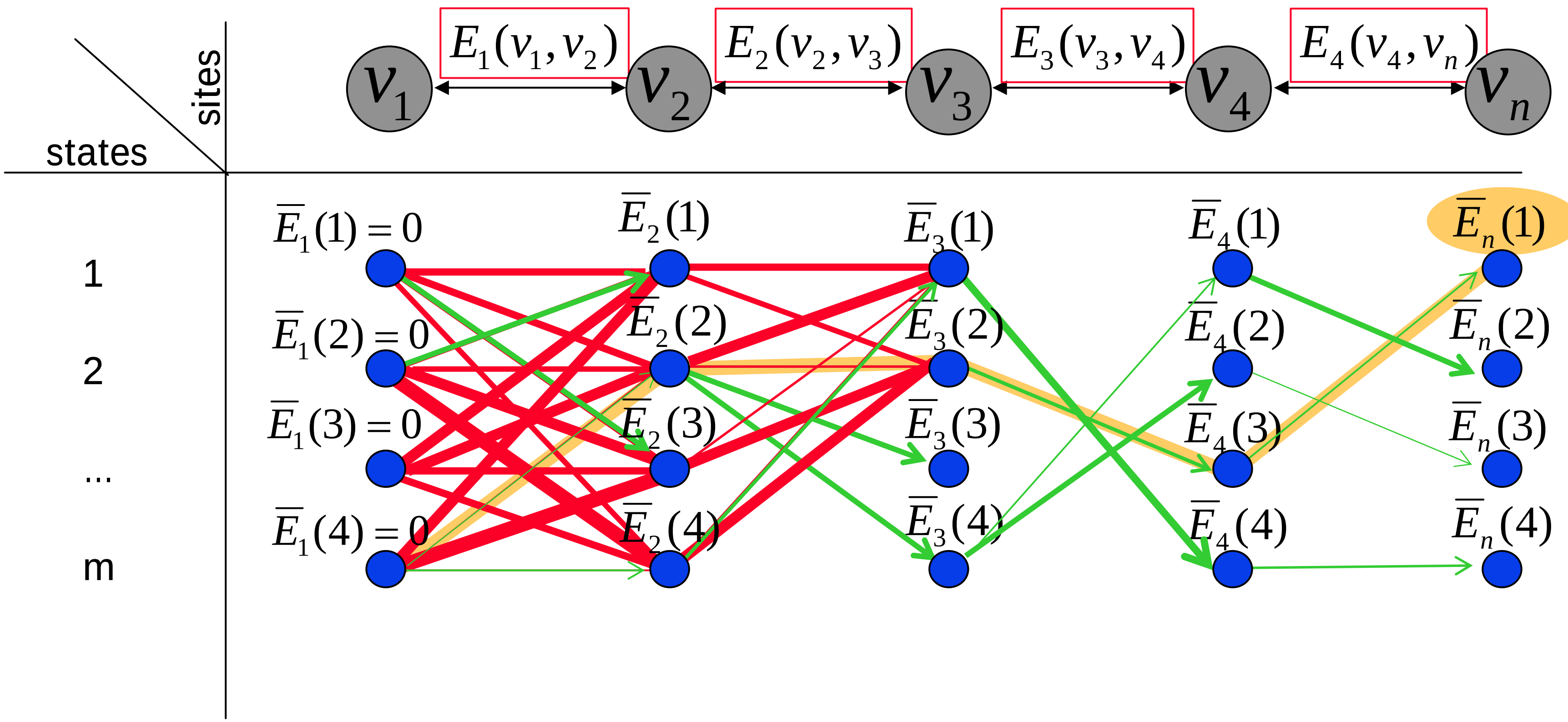
First-order interactions

$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$

Iterate until optimal position for each point is the center of the box, i.e. the snake is optimal in the local search space constrained by boxes

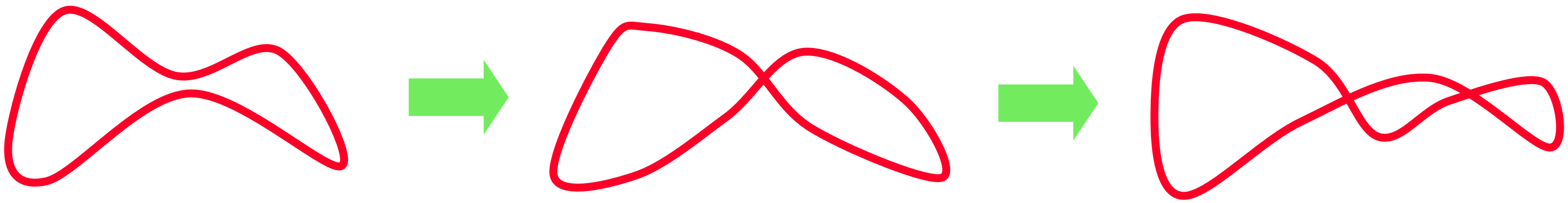
Dynamic programming

$$E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$

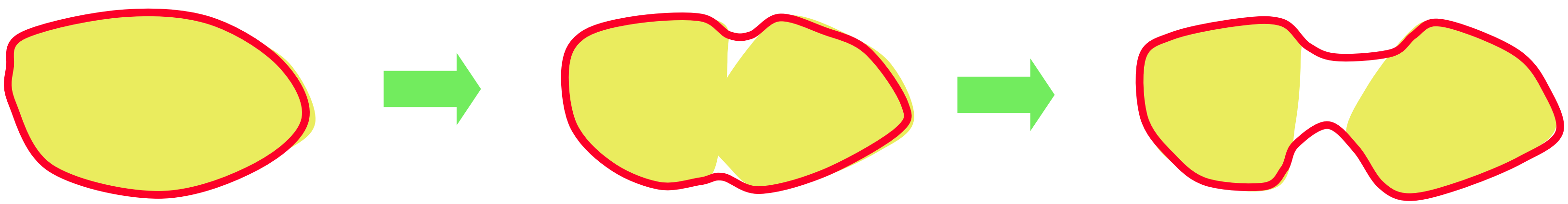


Limitations of snakes

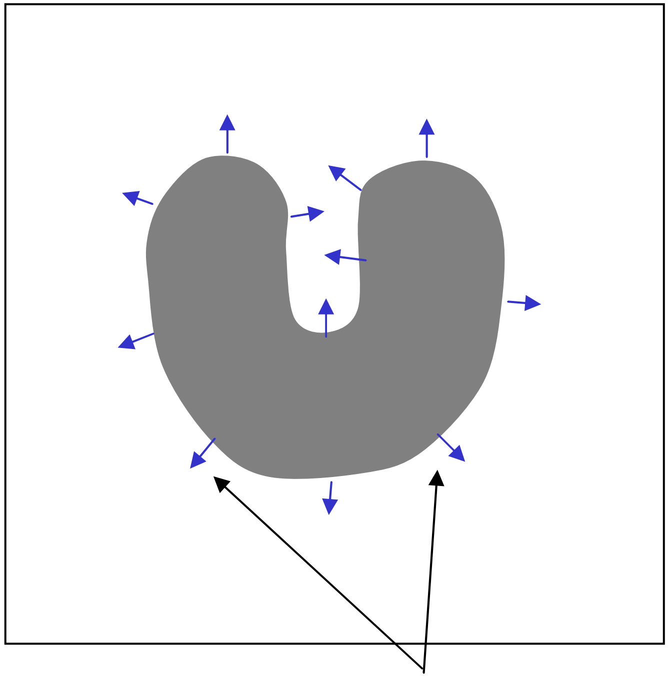
- Get stuck in local minimum
- Often miss indentations in objects
- Hard to prevent self-intersections



- Cannot follow topological changes!



Missing indentations



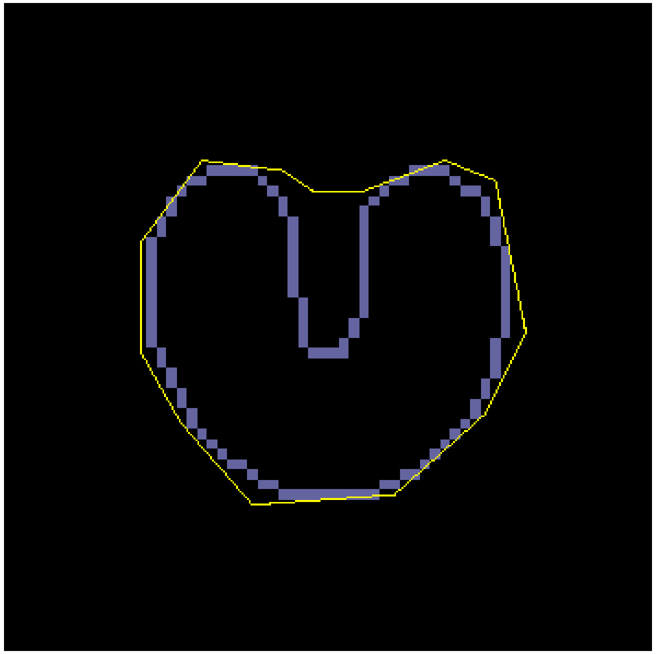
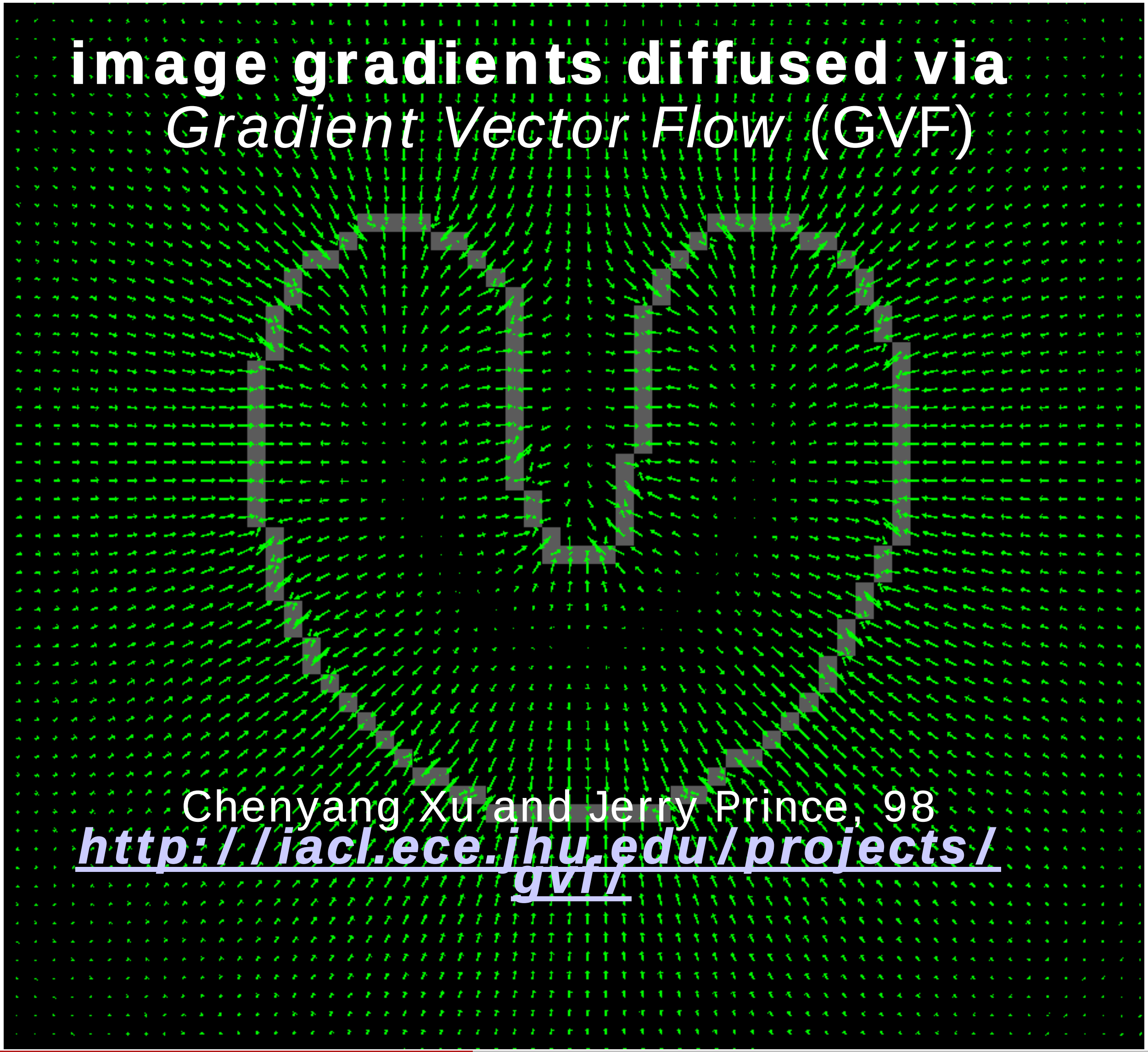
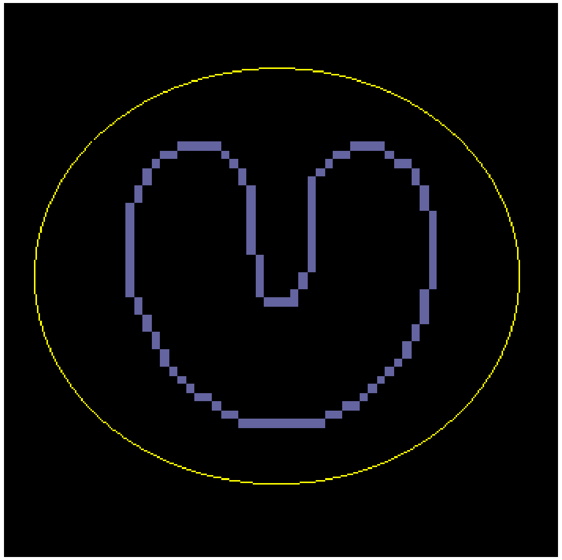
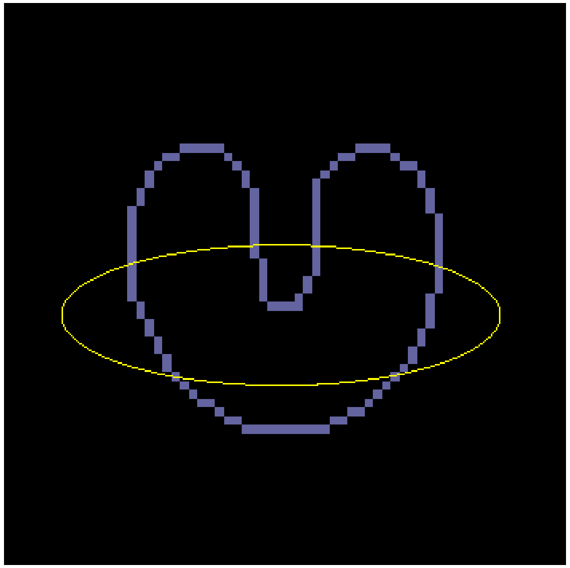


image gradients ∇I
are large only directly on the boundary

Diffusing image gradients



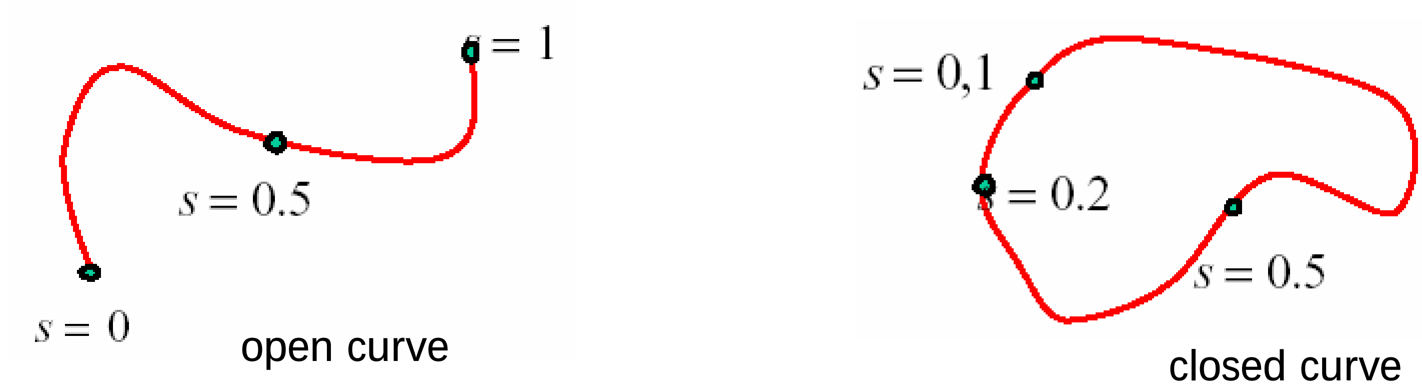




Continuous view of snakes

- Original papers, and most follow ups, take a continuous view
- A curve can be represented parametrically:

$$v(s) = (x(s), y(s)) \qquad 0 \le s \le 1$$



Continuous snake energy

$$E_{in}(v(s)) = \boxed{\alpha(s)\left|\frac{dv}{ds}\right|^2} + \boxed{\beta(s)\left|\frac{d^2v}{d^2s}\right|^2}$$

Elasticity

Stiffness

$$E_{ex}(v(s)) = -(\,|G_x(v(s))|^2 + |G_y(v(s))|^2\,)$$

$$E_{total} = \int_0^1 E_{in}(v(s))\,ds + \int_0^1 E_{ext}(v(s))\,ds$$

Curve evolution

- Basic idea: curve C evolves over time
  - Replace C(p) = (x(p),y(p)) by C(p,t)
  - Original curve is C(p,0)
  - Need a the partial derivative w.r.t. time
- Issue: curve reparameterization and intrinsic properties of curves
  - There are many different functions x(s),y(s) that give you exactly the same curve!
    - Think of driving same road at different speed

Curve evolution of snakes

- We can simplify our energy function to

$$E(C) = \int_0^1 \alpha |C_p(p)|^2 + g(C(p))dp.$$

- Calculus of variations says that at a local minimum of the energy we have

$$\alpha C_{pp}(p) + \nabla g(C(p)) = 0.$$

  – We can minimize this via gradient descent
  – Many ugly issues with this...

Arc length parameterization

- If we replaced p by ϕ, where $\phi(r)=p$, $r\in[c,d]$, first term in energy would become

$$\int_c^d |(C \circ \phi)'(r)|^2 (\phi'(r))^{-1} dr$$

  – Second term is even worse!
- Natural parameterization is in terms of arc length (distance along curve)
  – s(p) is the distance from the origin to p:

$$s(p) = \int_0^q \sqrt{x_q^2(q) + y_q^2(q)} dq$$

Reparameterization

- We can use s to reparameterize the curve

$$L = \int_0^1 \left| C_p \right| dp$$