Computer Vision II – Lecture 5

Contour based Tracking

06.05.2014

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Course Outline

- Single-Object Tracking
  - Background modeling
  - Template based tracking
  - Color based tracking
  - Contour based tracking
  - Tracking by online classification
  - Tracking-by-detection
- Bayesian Filtering
- Multi-Object Tracking
- Articulated Tracking

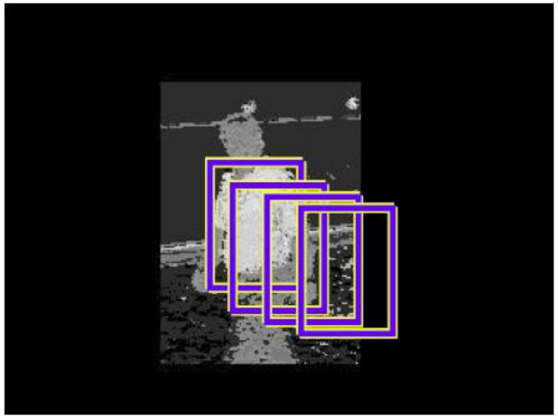
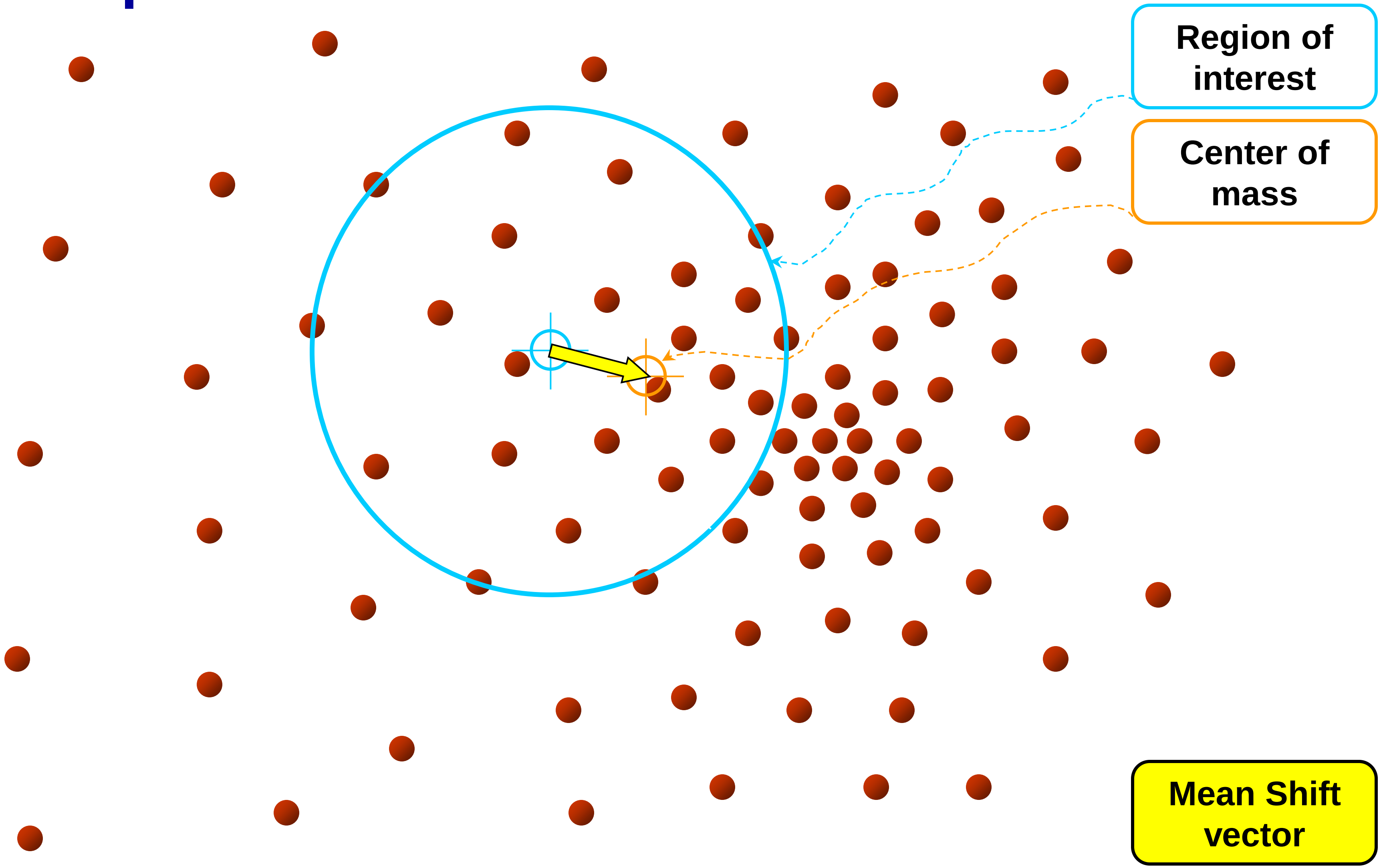


Image source: Robert Collins

Recap: Mean-Shift



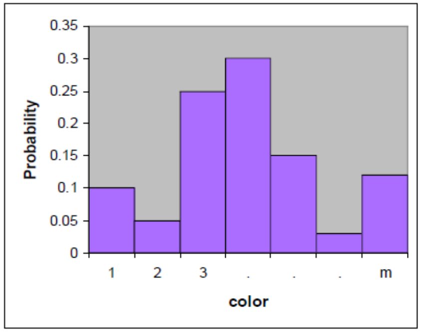
Objective: Find the densest region

Slide by Y. Ukrainitz & B. Sarel

Recap: Using Mean-Shift on Color Models

- Two main approaches
  1. Explicit weight images
     - Create a color likelihood image, with pixels weighted by the similarity to the desired color (best for unicolored objects).
     - Use mean-shift to find spatial modes of the likelihood.

  2. Implicit weight images
     - Represent color distribution by a histogram.
     - Use mean-shift to find the region that has the most similar color distribution.



Mean-Shift on Weight Images

- **Ideal case**
  - Want an indicator function that returns 1 for pixels on the tracked object and 0 for all other pixels.
- **Instead**
  - Compute likelihood maps
  - Value at a pixel is proportional to the likelihood that the pixel comes from the tracked object.

- **Likelihood can be based on**
  - Color
  - Texture
  - Shape (boundary)
  - Predicted location

Recap: Mean-Shift Tracking

- Mean-Shift finds the mode of an explicit likelihood image

Kernel weight evaluated at offset $(\mathbf{a}-\mathbf{x})$

Weight from the likelihood image at pixel $\mathbf{a}$

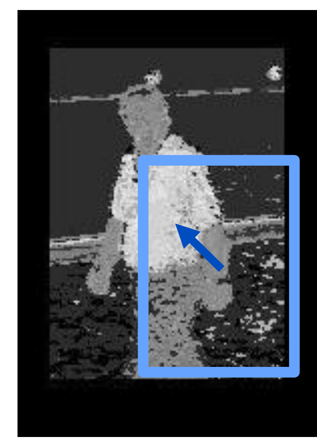
Offset of pixel $\mathbf{a}$ to kernel center $\mathbf{x}$

$$\Delta\mathbf{x} = \frac{\sum_{\mathbf{a}} K(\mathbf{a}-\mathbf{x})w(\mathbf{a})(\mathbf{a}-\mathbf{x})}{\sum_{\mathbf{a}} K(\mathbf{a}-\mathbf{x})w(\mathbf{a})}$$

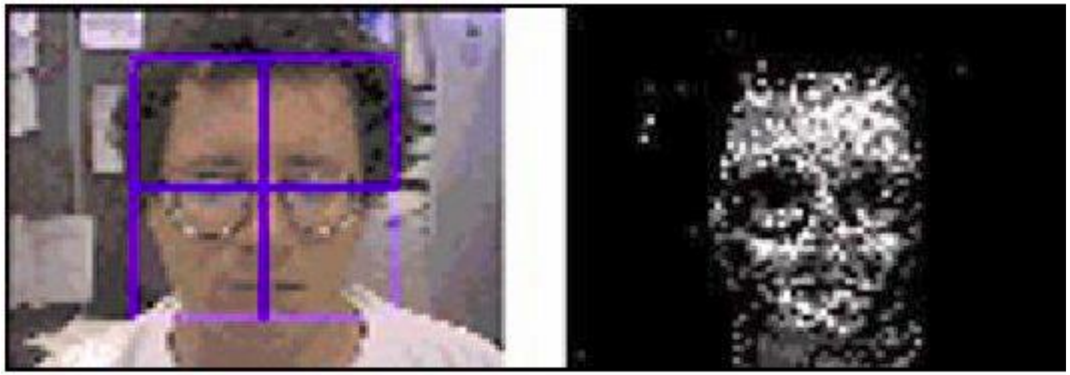
Sum over all pixels $\mathbf{a}$ under kernel K

Normalization term

$\Rightarrow$ *Mean-shift computes the weighted mean of all shifts (offsets), weighted by the point likelihood and the kernel function centered at* $\mathbf{x}$.



Recap: Explicit Weight Images



- **Histogram backprojection**
  - Histogram is an empirical estimate of $p(color \mid object) = p(c \mid o)$
  - Bayes' rule says: $$p(o|c) = \frac{p(c|o)p(o)}{p(c)}$$
  - Simplistic approximation: assume $p(o)/p(c)$ is constant.

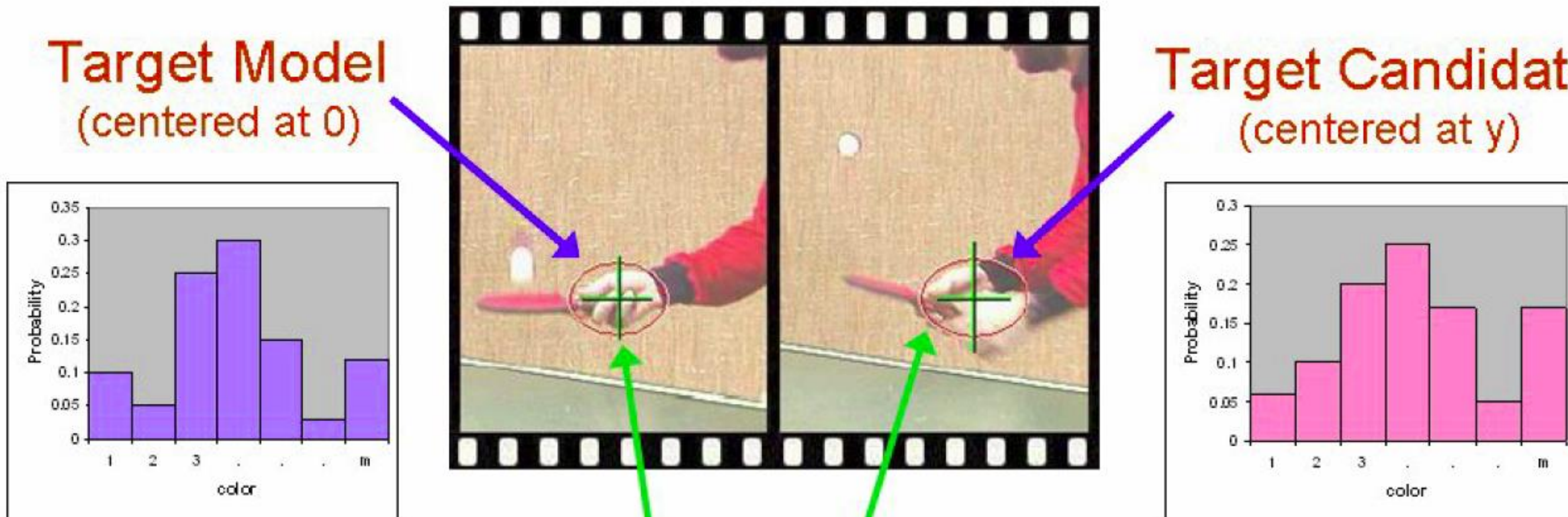
  $\Rightarrow$ Use histogram h as a lookup table to set pixel values in the weight image.
  - If pixel maps to histogram bucket i, set weight for pixel to $h(i)$.

Recap: Scale Adaptation in CAMshift

Mean shift window initialization

Image source: http://docs.opencv.org/trunk/doc/py_tutorials/py_video/py_meanshift/py_meanshift.html

Recap: Tracking with Implicit Weight Images



$$\vec{q} = \{q_u\}_{u=1..m} \qquad \sum_{u=1}^{m} q_u = 1$$

$$\vec{p}(y) = \{p_u(y)\}_{u=1..m} \qquad \sum_{u=1}^{m} p_u = 1$$

Similarity Function: $f(y) = f\left[\vec{q}, \vec{p}(y)\right]$

Recap: Comaniciu's Mean-Shift

- **Color histogram representation**

$$\text{target model:} \quad \hat{\mathbf{q}} = \{\hat{q}_u\}_{u=1\ldots m} \qquad \sum_{u=1}^{m} \hat{q}_u = 1$$

$$\text{target candidate:} \quad \hat{\mathbf{p}}(\mathbf{y}) = \{\hat{p}_u(\mathbf{y})\}_{u=1\ldots m} \qquad \sum_{u=1}^{m} \hat{p}_u = 1\,.$$

- **Measuring distances between histograms**
  - Distance as a function of window location $\mathbf{y}$

$$d(\mathbf{y}) = \sqrt{1 - \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right]}\,,$$

  - where $\hat{\rho}(\mathbf{y})$ is the Bhattacharyya coefficient

$$\hat{\rho}(\mathbf{y}) \equiv \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\mathbf{y})\hat{q}_u}\,,$$

Recap: Comaniciu's Mean-Shift

- Compute histograms via Parzen estimation

$$\hat{q}_u = C \sum_{i=1}^{n} k(\|\mathbf{x}_i^\star\|^2)\delta\left[b(\mathbf{x}_i^\star) - u\right] ,$$

$$\hat{p}_u(\mathbf{y}) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{y}-\mathbf{x}_i}{h}\right\|^2\right)\delta\left[b(\mathbf{x}_i) - u\right] ,$$

  - where $k(\cdot)$ is some radially symmetric smoothing kernel profile, $\mathbf{x}_i$ is the pixel at location i, and $b(\mathbf{x}_i)$ is the index of its bin in the quantized feature space.

- Consequence of this formulation
  - Gathers a histogram over a neighborhood
  - Also allows interpolation of histograms centered around an off-lattice location.

Recap: Result of Taylor Expansion

- Simple update procedure: At each iteration, perform

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\|\frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\|\frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h}\right\|^2\right)} \quad \text{where } g(x) = -k'(x),$$

  - which is just standard mean-shift on (implicit) weight image w_i.
  - Let's look at the weight image more closely. For each pixel $\mathbf{x}_i$

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}} \boxed{\delta\left[b(\mathbf{x}_i) - u\right]}.$$

This is only 1 once in the summation

⇒ If pixel $\mathbf{x}_i$'s value maps to histogram bucket B, then

$$w_i = \sqrt{q_B / p_B(\mathbf{y}_0)}$$

Today: Contour based Tracking

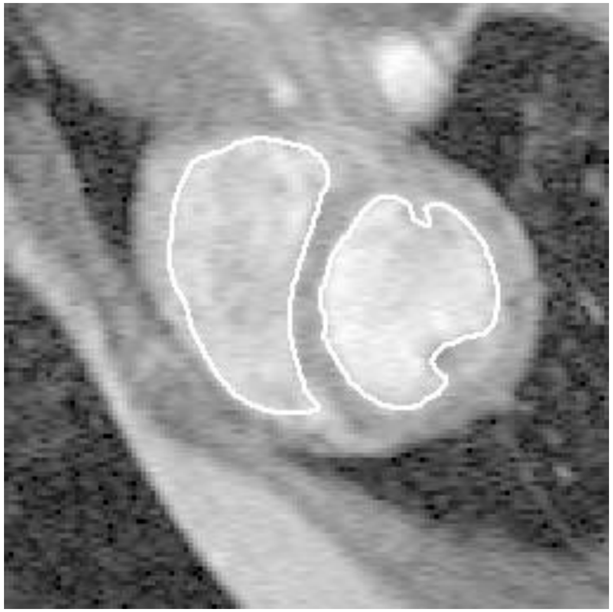


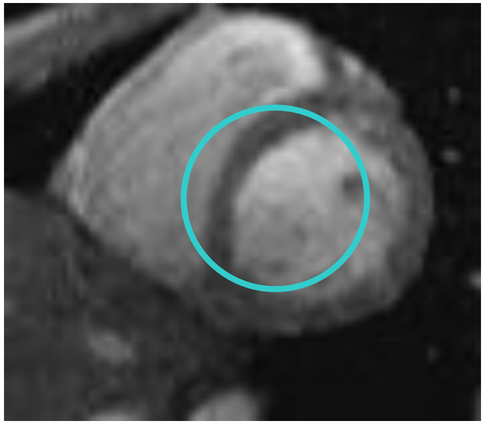
Image source: Yuri Boykov

Topics of This Lecture

- Deformable contours
  - Motivation
  - Contour representation
- Defining the energy function
  - External energy
  - Internal energy
- Energy minimization
  - Greedy approach
  - Dynamic Programming approach
- Extensions
  - Tracking
  - Level Sets

Deformable Contours

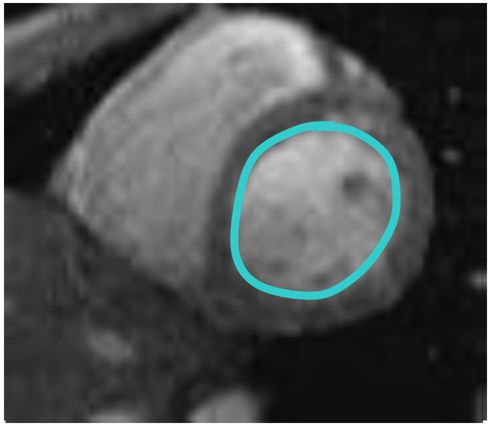
- Given
  - Initial contour (model) near desired object



M. Kass, A. Witkin, D. Terzopoulos. Snakes: Active Contour Models, IJCV1988.

Deformable Contours

- **Given**
  - Initial contour (model) near desired object
- **Goal**
  - Evolve the contour to fit the exact object boundary

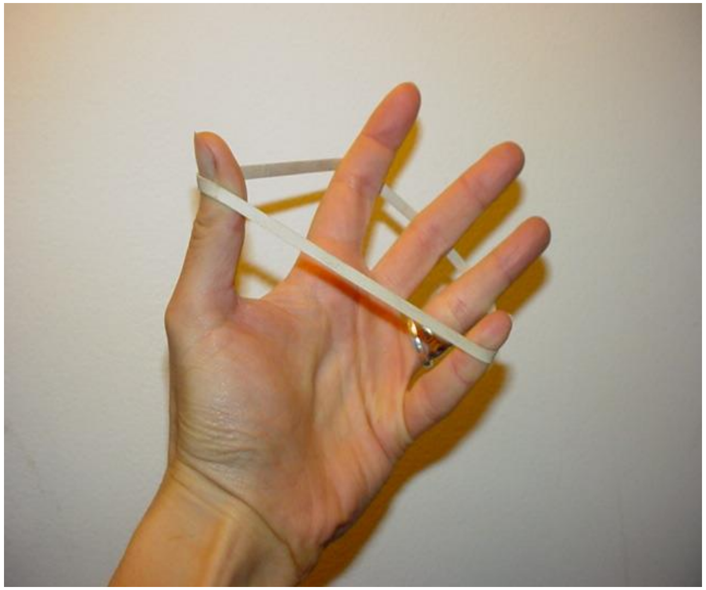


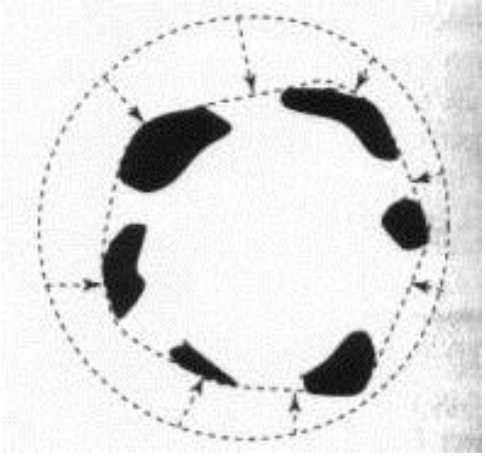
- **Main ideas**
  - Iteratively adjust the elastic band so as to be near image positions with high gradients, and
  - Satisfy shape "preferences" or contour priors
  - Formulation as energy minimization problem.

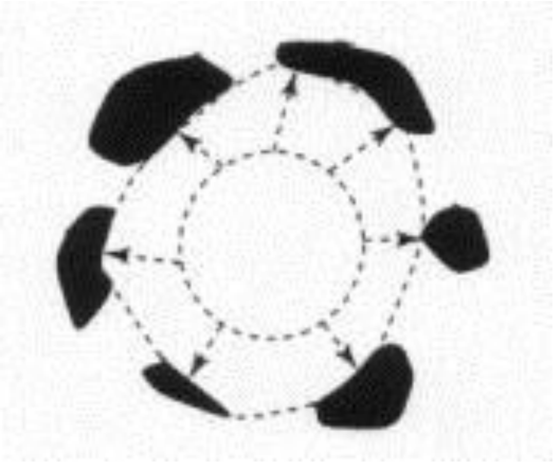
M. Kass, A. Witkin, D. Terzopoulos. Snakes: Active Contour Models, IJCV1988.

Image source: Yuri Boykov

Deformable Contours: Intuition







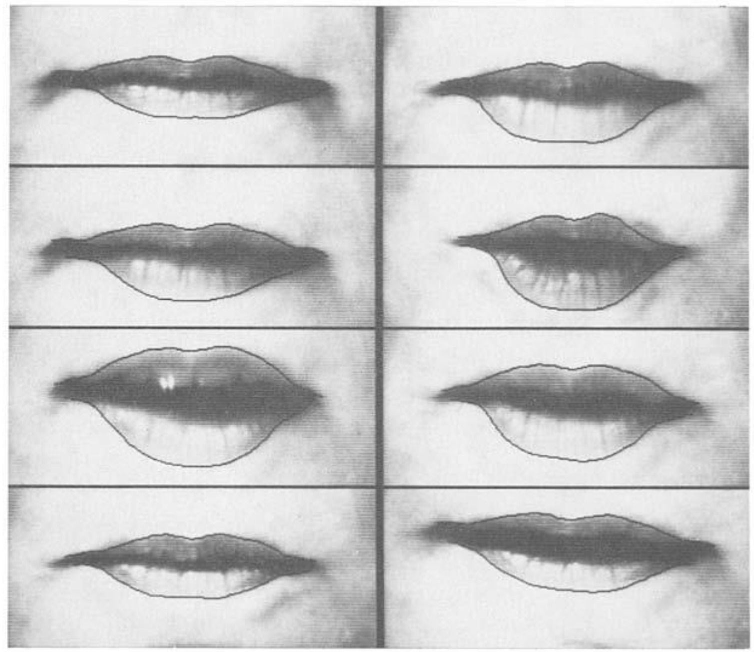
Slide credit: Kristen Grauman

Image source: http://www.healthline.com/blogs/exercise_fitness/uploaded_images/HandBand2-795868.JPG

Why Do We Want Deformable Shapes?

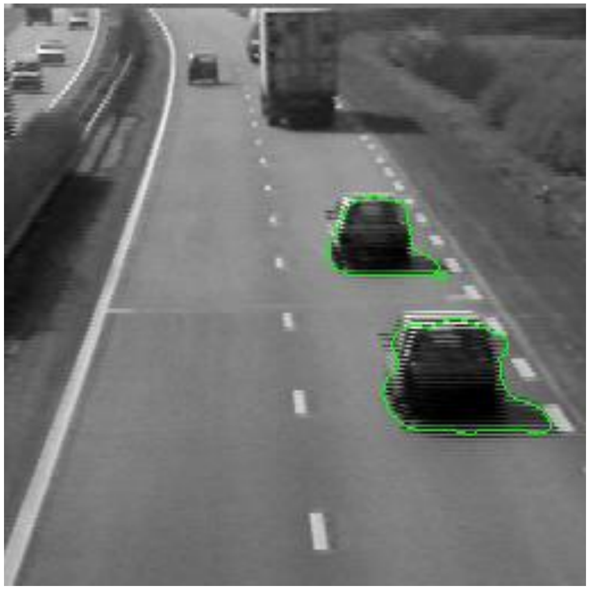
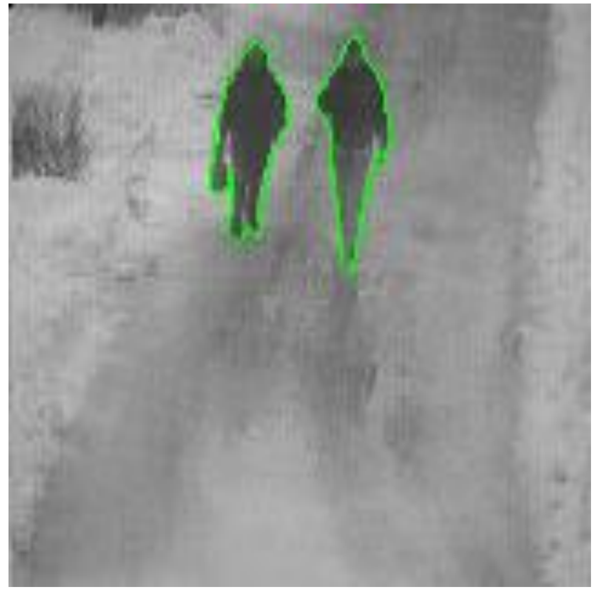
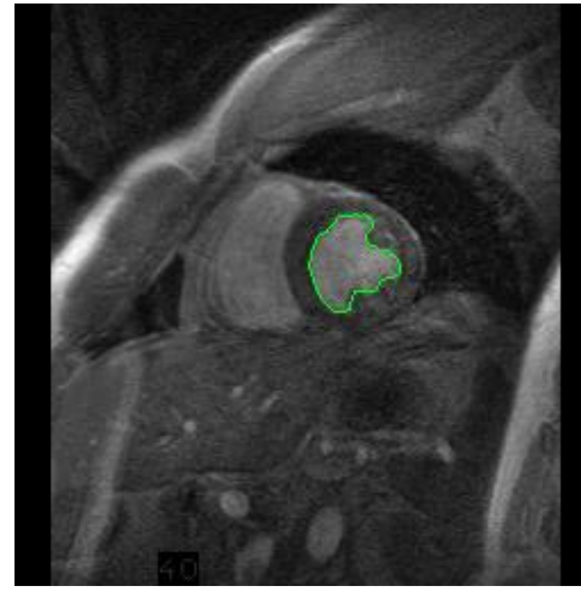
- **Motivations**
  - **Some objects have similar basic form, but some variety in contour shape.**

Why Do We Want Deformable Shapes?



- **Motivations**
  - **Some objects have similar basic form, but some variety in contour shape.**
  - **Non-rigid, deformable objects can change their shape over time, e.g. lips, hands…**

Why Do We Want Deformable Shapes?

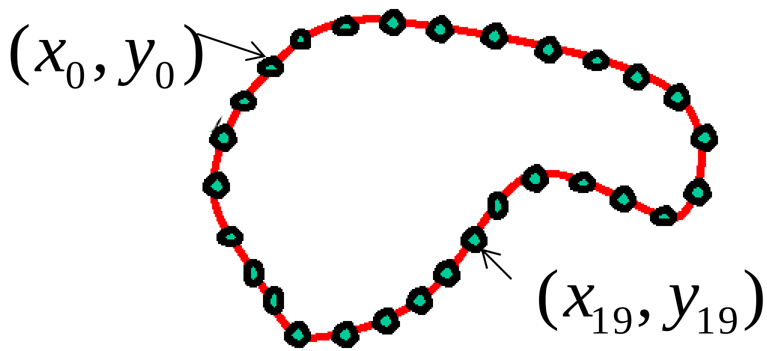
- **Motivations**
  - **Some objects have similar basic form, but some variety in contour shape.**
  - **Non-rigid, deformable objects can change their shape over time, e.g. lips, hands...**
  - **Contour shape may be an important cue for tracking**

Topics of This Lecture

- Deformable contours
  - Motivation
  - Contour representation
- Defining the energy function
  - External energy
  - Internal energy
- Energy minimization
  - Greedy approach
  - Dynamic Programming approach
- Extensions
  - Tracking
  - Level Sets

Contour Representation

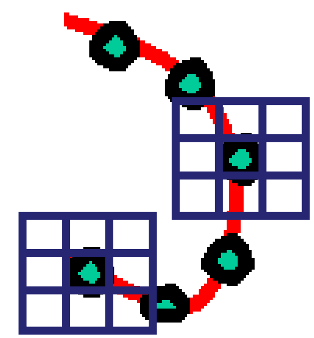
- **Discrete representation**
  - **We'll consider a discrete representation of the contour, consisting of a list of 2D point positions ("vertices").**



$$v_i = (x_i, y_i),$$

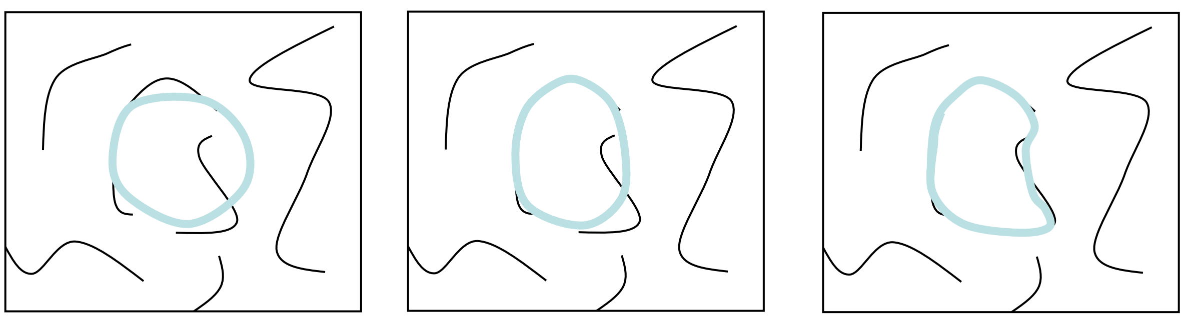
for $i = 0, 1, \ldots, n-1$

  - **At each iteration, we'll have the option to move each vertex to another nearby location ("state").**



Fitting Deformable Contours

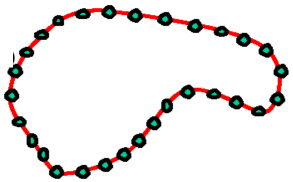
- **How to adjust the current contour to form the new contour at each iteration?**
  - **Define a cost function ("energy" function) that says how good a candidate configuration is.**
  - **Seek next configuration that minimizes that cost function.**



initial intermediate final

Energy Function

- Definition
  - Total energy (cost) of the current snake

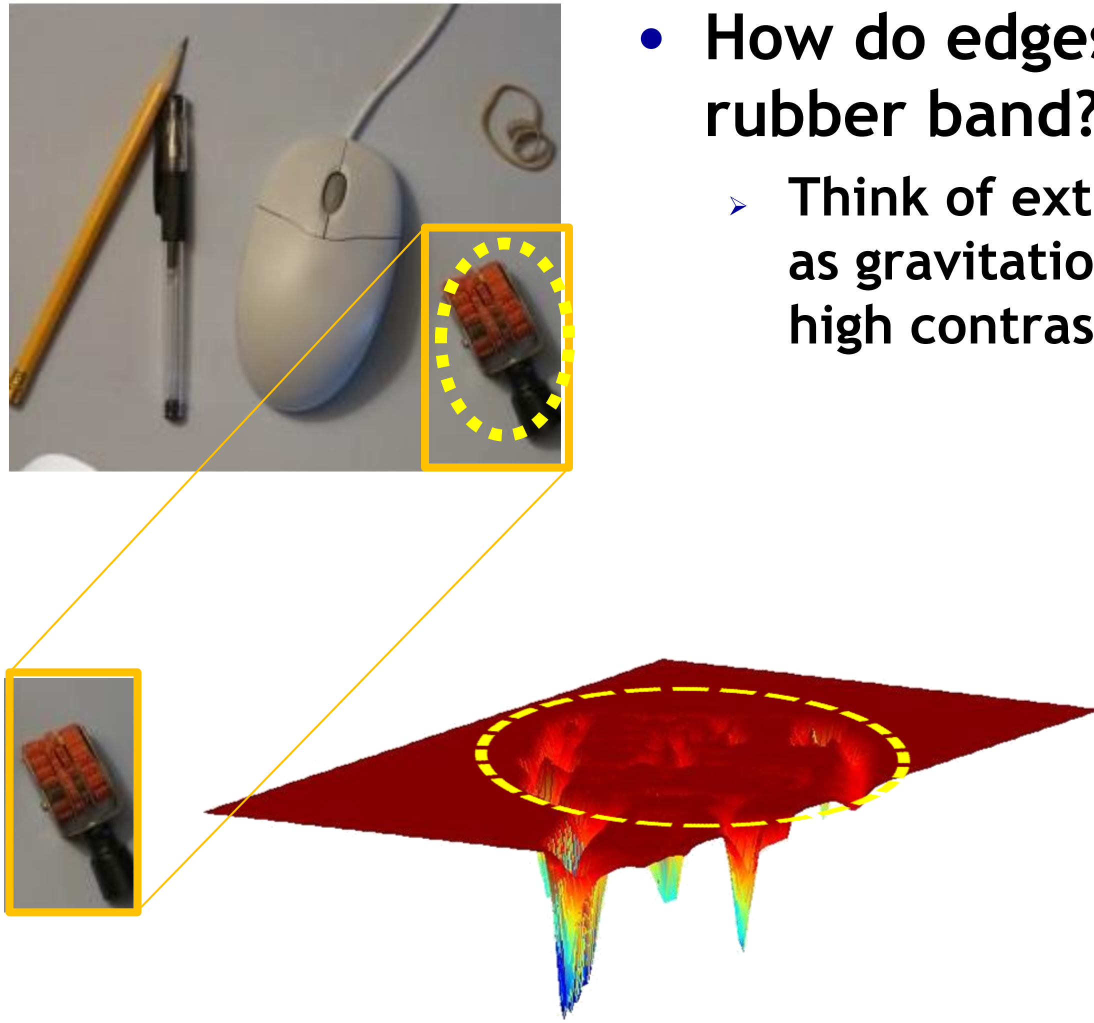


$$E_{total} = E_{internal} + E_{external}$$

- Internal energy
  - Encourage prior shape preferences: e.g., smoothness, elasticity, particular known shape.
- External energy
  - Encourage contour to fit on places where image structures exist, e.g., edges.

⇒ Good fit between current deformable contour and target shape in the image will yield a low value for this cost function.

External Image Energy



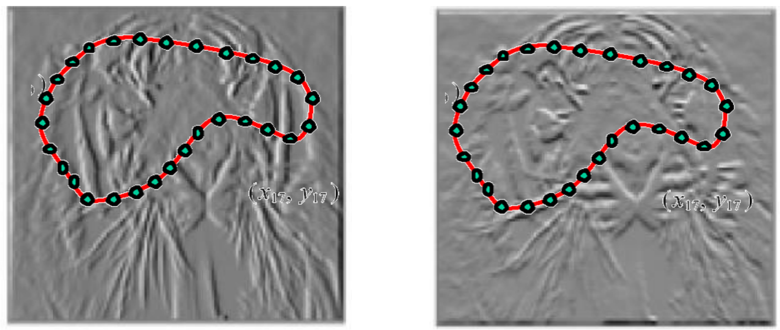
- How do edges affect snap of rubber band?
  - Think of external energy from image as gravitational pull towards areas of high contrast.

- (Magnitude of gradient)

$$-\left(G_x(I)^2 + G_y(I)^2\right)$$

External Image Energy

- **Gradient images** $G_x(x, y)$ **and** $G_y(x, y)$



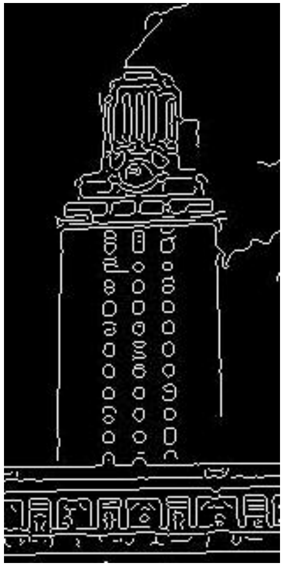
- **External energy at a point on the curve is:**

$$E_{external}(\nu) = -(\,|\,G_x(\nu)\,|^2 + |\,G_y(\nu)\,|^2\,)$$

- **External energy for the whole curve:**

$$E_{external} = \quad -\sum_{i=0}^{n-1} |\,G_x(x_i, y_i)\,|^2 + |\,G_y(x_i, y_i)\,|^2$$

Internal Energy: Intuition



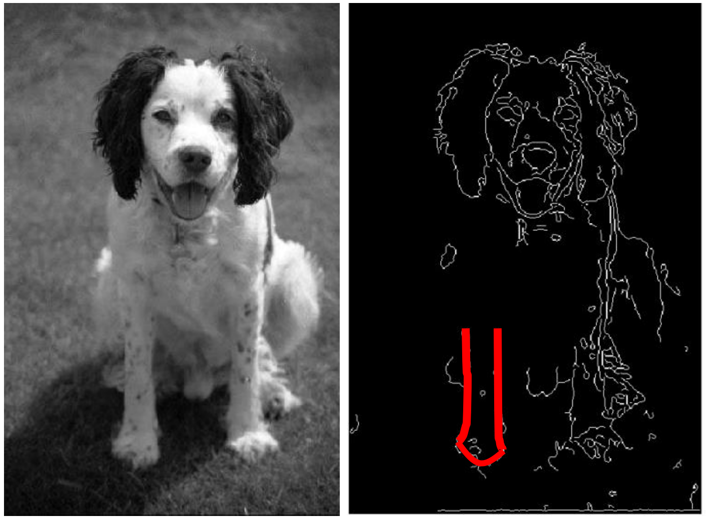
What are the underlying boundaries in this fragmented edge image?

And in this one?

Slide credit: Kristen Grauman

Internal Energy: Intuition

- *A priori*, we want to favor
  - Smooth shapes
  - Contours with low curvature
  - Contours similar to a known shape, etc. to balance what is actually observed (i.e., in the gradient image).



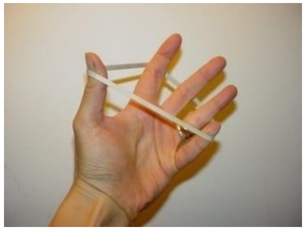
Internal Energy

- **Common formulatoin**
  - For a *continuous* curve, a common internal energy term is the "bending energy".
  - At some point *v(s)* on the curve, this is:

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

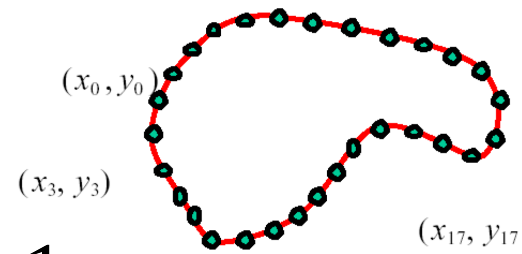
Tension, Elasticity

Stiffness, Curvature



Internal Energy

- **For our discrete representation,**

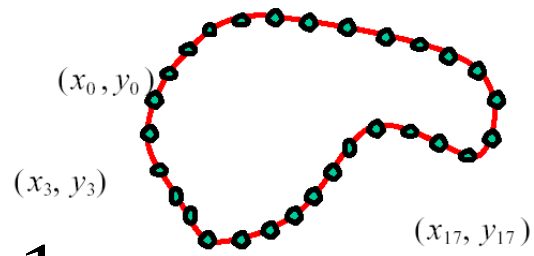


$$v_i = (x_i, y_i) \qquad i = 0 \ldots n-1$$

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

  - ***Note these are derivatives relative to position - not spatial image gradients.***

Internal Energy



- **For our discrete representation,**

$$v_i = (x_i, y_i) \qquad i = 0 \ldots n-1$$

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

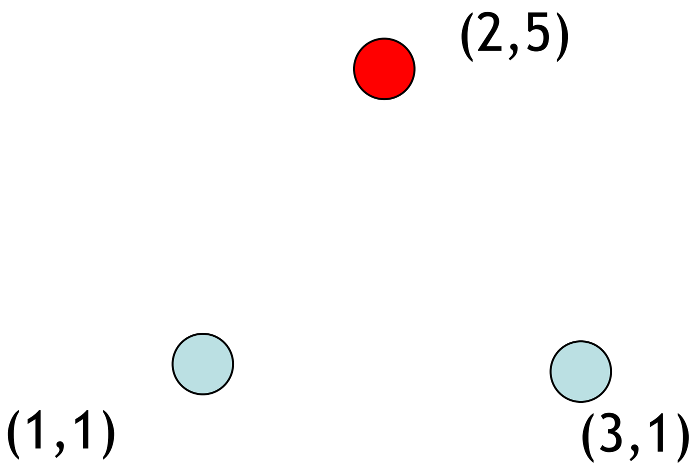
- **Internal energy for the whole curve:**

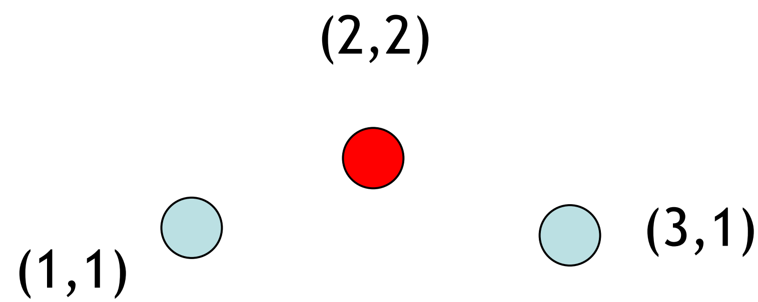
$$E_{internal} = \sum_{i=0}^{n-1} \alpha \left\| v_{i+1} - v_i \right\|^2 + \beta \left\| v_{i+1} - 2v_i + v_{i-1} \right\|^2$$

  - ***Why do these reflect tension and curvature?***

Example: Compare Curvature

$$E_{curvature}(v_i) = \left\| v_{i+1} - 2v_i + v_{i-1} \right\|^2$$
$$= (x_{i+1} - 2x_i + x_{i-1})^2 + (y_{i+1} - 2y_i + y_{i-1})^2$$





$$(3 - 2(2) + 1)^2 + (1 - 2(5) + 1)^2$$
$$= (-8)^2 = 64$$

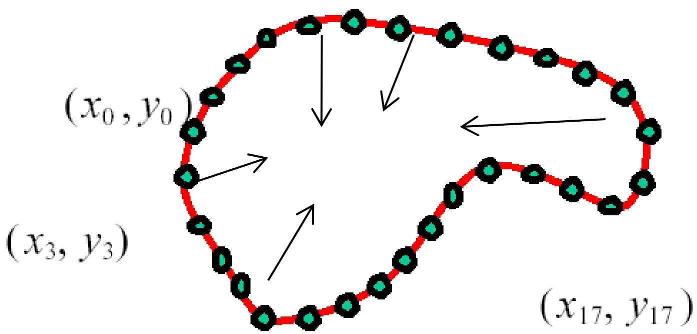
$$(3 - 2(2) + 1)^2 + (1 - 2(2) + 1)^2$$
$$= (-2)^2 = 4$$

Penalizing Elasticity

- **Current elastic energy definition uses a discrete estimate of the derivative:**

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

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



What is a possible problem with this definition?

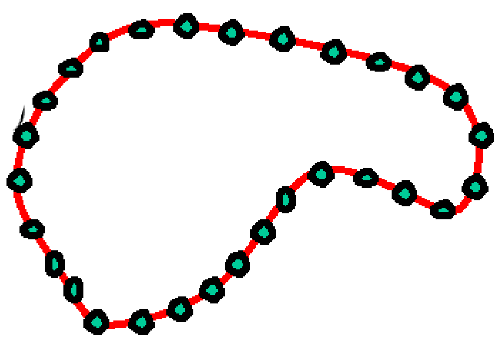
Penalizing Elasticity

- **Current elastic energy definition uses a discrete estimate of the derivative:**

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

- **Instead:**

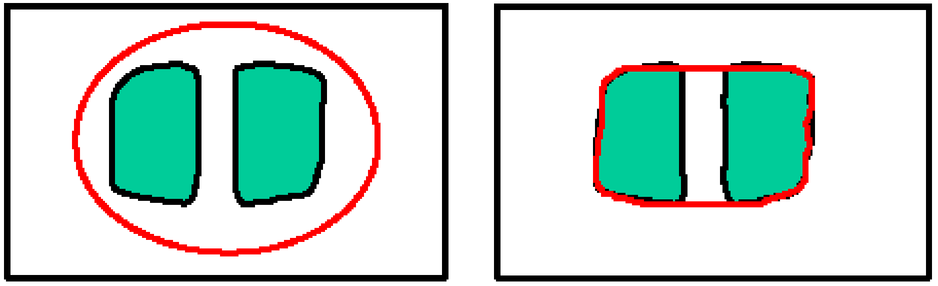
$$= \alpha \cdot \sum_{i=0}^{n-1} \left((x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 - \overline{d} \right)^2$$

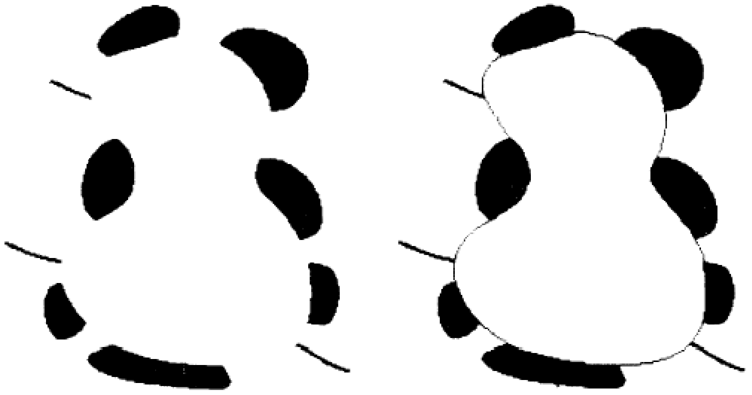


where d is the average distance between pairs of points - updated at each iteration.

Dealing with Missing Data

- Effect of Internal Energy
  - Preference for low-curvature, smoothness helps dealing with missing data





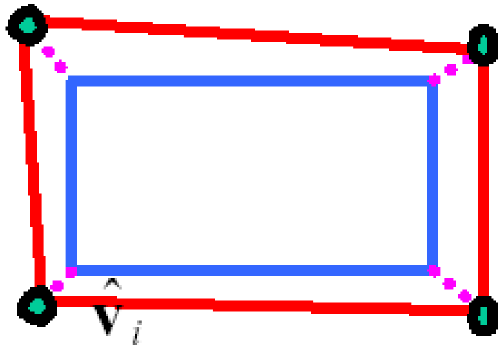
Illusory contours found!

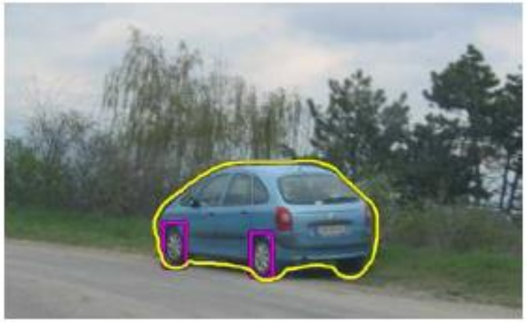
Extending the Internal Energy: Shape Priors

- Shape priors
  - If object is some smooth variation on a known shape, we can use a term that will penalize deviation from that shape:

$$E_{internal} += \alpha \cdot \sum_{i=0}^{n-1} (v_i - \hat{v}_i)^2$$

where $\{\hat{v}_i\}$ are the points of the known shape.





Putting Everything Together...

- **Total energy**

$$E_{total} = E_{internal} + \gamma E_{external}$$

  - **with the component terms**

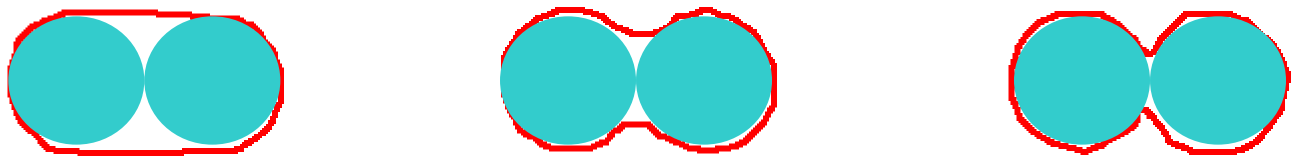
$$E_{external} = -\sum_{i=0}^{n-1} |G_x(x_i, y_i)|^2 + |G_y(x_i, y_i)|^2$$

$$E_{internal} = \sum_{i=0}^{n-1} \alpha\left(\bar{d} - \|v_{i+1} - v_i\|\right)^2 + \beta\|v_{i+1} - 2v_i + v_{i-1}\|^2$$

Behavior can be controlled by adapting the weights α, β, γ.

Total Energy

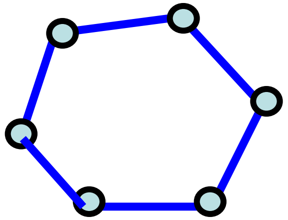
- **Behavior varies as a function of the weights**
  - **E.g., α weight controls the penalty for internal elasticity.**



large α **medium α** **small α**

Summary: Deformable Contours

- A simple elastic snake is defined by:
  - A set of N points,
  - An internal energy term (tension, bending, plus optional shape prior)
  - An external energy term (gradient-based)



- To use to segment an object:
  - Initialize in the vicinity of the object
  - Modify the points to minimize the total energy
  - *How can we do this minimization?*

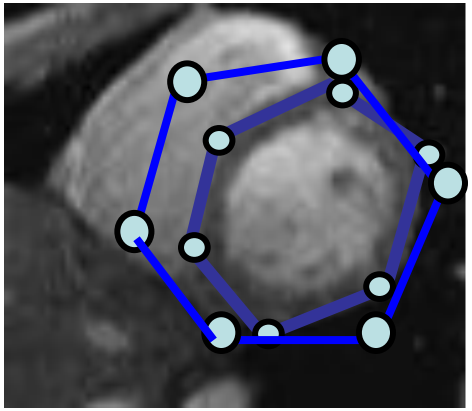


Figure source: Yuri Boykov

Topics of This Lecture

- Deformable contours
  - Motivation
  - Contour representation
- Defining the energy function
  - External energy
  - Internal energy
- **Energy minimization**
  - **Greedy approach**
  - **Dynamic Programming approach**
- Extensions
  - Tracking
  - Level Sets

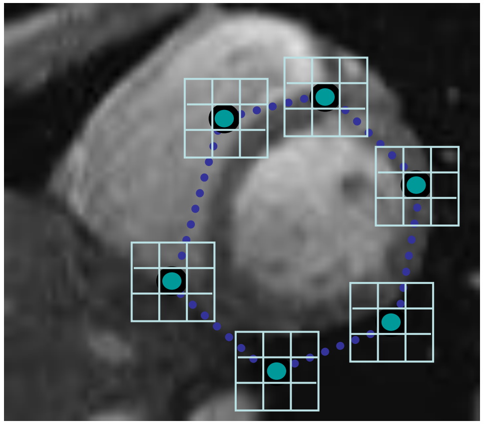
Energy Minimization

- Several algorithms have been proposed to fit deformable contours
  - Greedy search
  - Variational approaches
  - Dynamic programming (for 2D snakes)
  - ...

- We'll look at two of them in the following...

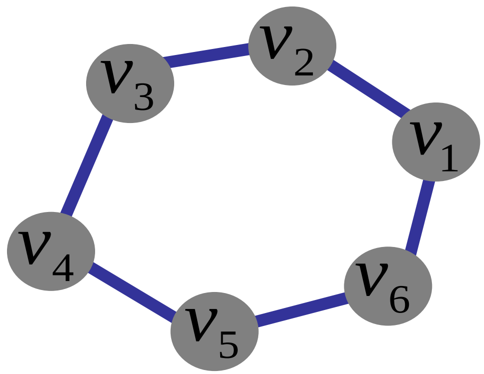
Energy Minimization: Greedy

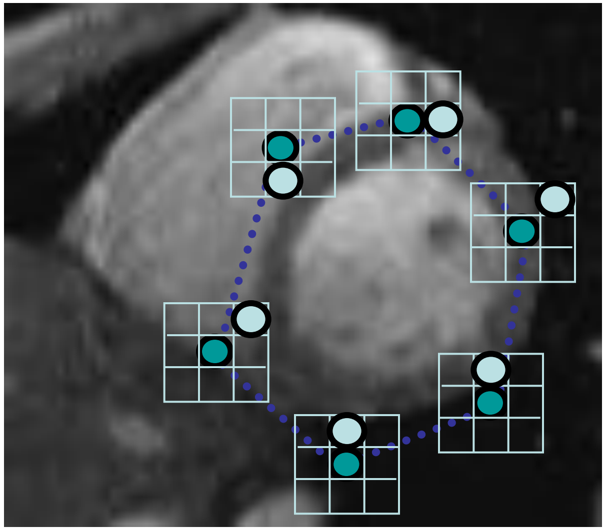
- Greedy optimization
  - For each point, search window around it and move to where energy function is minimal.
  - Typical window size, e.g., 5×5 pixels



- Stopping criterion
  - Stop when predefined number of points have not changed in last iteration, or after max number of iterations.
- Note:
  - Local optimization - need decent initialization!
  - Convergence not guaranteed

Energy Minimization: Dynamic Programming





- **Constraining the search space**
  - **Limit possible moves to neighboring pixels**
  - **With this form of the energy function, we can minimize using dynamic programming, with the Viterbi algorithm.**

  ⇒ Optimal results in the local search space defined by the box.

A. Amini, T.E. Weymouth, R.C. Jain. Using Dynamic Programming for Solving Variational Problems in Vision, PAMI, Vol. 12(9), 1990.

Slide credit: Kristen Grauman

Figure source: Yuri Boykov

Energy Minimization: Dynamic Programming

- Dynamic Programming optimization
  - Possible because snake energy can be rewritten as a sum of pair-wise interaction potentials:

$$E_{total}(v_1,\ldots,v_n) = \sum_{i=1}^{n-1} E_i(v_i, v_{i+1})$$

  - Or sum of triple interaction potentials

$$E_{total}(v_1,\ldots,v_n) = \sum_{i=1}^{n-1} E_i(v_{i-1}, v_i, v_{i+1})$$

Snake Energy: Pairwise Interactions

- **Total energy**

$$E_{total}(x_1,\ldots,x_n,y_1,\ldots,y_n) = -\sum_{i=1}^{n-1} |G_x(x_i,y_i)|^2 + |G_y(x_i,y_i)|^2 + \alpha \cdot \sum_{i=1}^{n-1} (x_{i+1}-x_i)^2 + (y_{i+1}-y_i)^2$$

  - Rewriting the above with $v_i = (x_i, y_i)$:

$$E_{total}(v_1,\ldots,v_n) = -\sum_{i=1}^{n-1} \| G(v_i) \|^2 + \alpha \cdot \sum_{i=1}^{n-1} \| v_{i+1} - v_i \|^2$$

  - Pairwise formulation

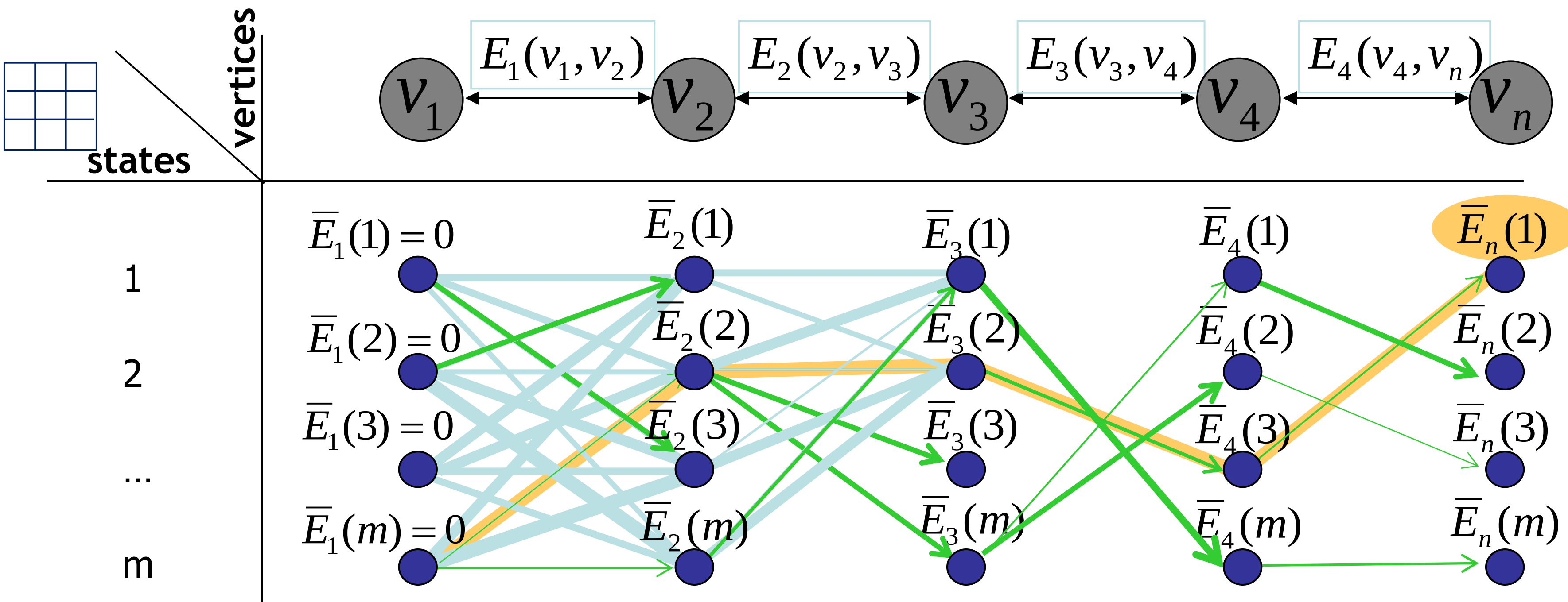
$$E_{total}(v_1,\ldots,v_n) = E_1(v_1,v_2) + E_2(v_2,v_3) + \ldots + E_{n-1}(v_{n-1},v_n)$$

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

Viterbi Algorithm

- **Main idea:**
  - **Determine optimal state of predecessor, for each possible state**
  - **Then backtrack from best state for last vertex**

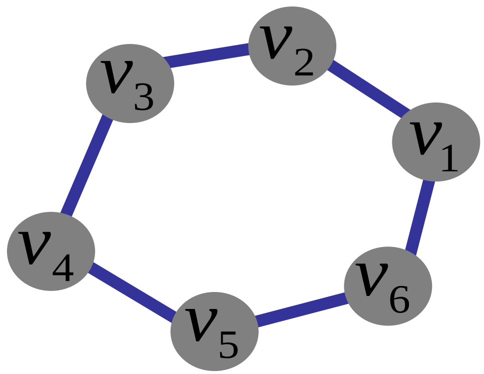
$$E_{total} = E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$

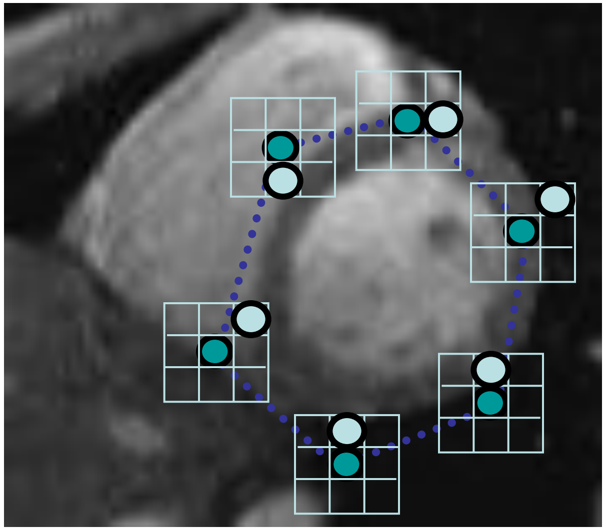


Complexity: $\mathcal{O}(nm^2)$ **vs. brute force search ______?**

Slide credit: Kristen Grauman, adapted from Yuri Boykov

Summary: Dynamic Programming





- **Dynamic Programming solution**
  - **Limit possible moves to neighboring pixels (discrete states).**
  - **Find the best joint move of all points using Viterbi algorithm.**
  - **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.**

Slide credit: Kristen Grauman

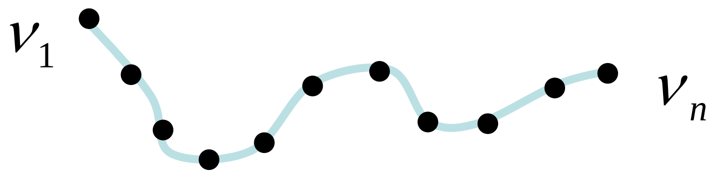
[Amini, Weymouth, Jain, 1990]

Figure source: Yuri Boykov

Energy Minimization: Dynamic Programming

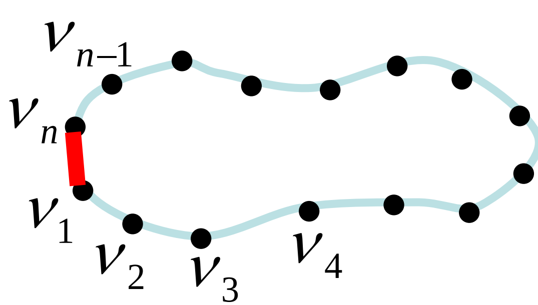
- **Limitations**
  - DP can be applied to optimize an open-ended snake

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



  - For a closed snake, a loop is introduced into the energy

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



Workaround:

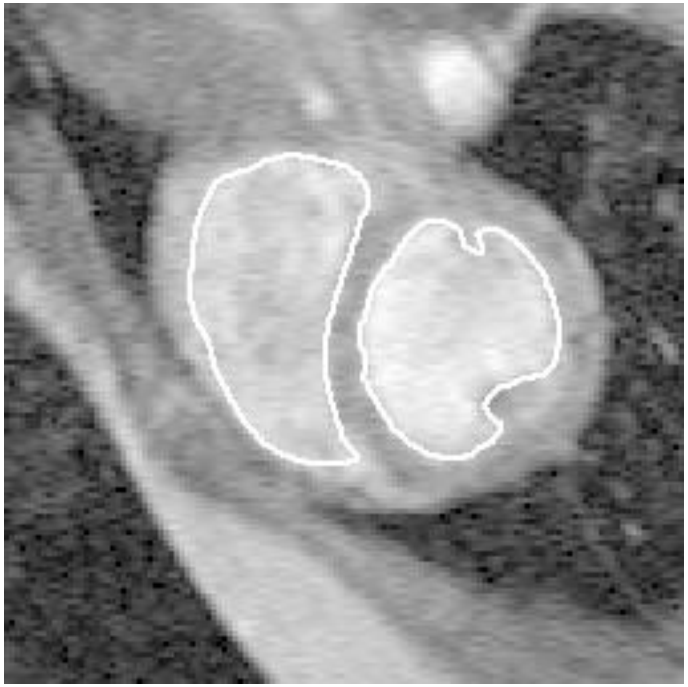
1) Fix v_1 and solve for rest .
2) Fix an intermediate node at its position found in (1), solve for rest.

Topics of This Lecture

- Deformable contours
  - Motivation
  - Contour representation
- Defining the energy function
  - External energy
  - Internal energy
- Energy minimization
  - Greedy approach
  - Dynamic Programming approach
- **Extensions**
  - **Tracking**
  - **Level Sets**

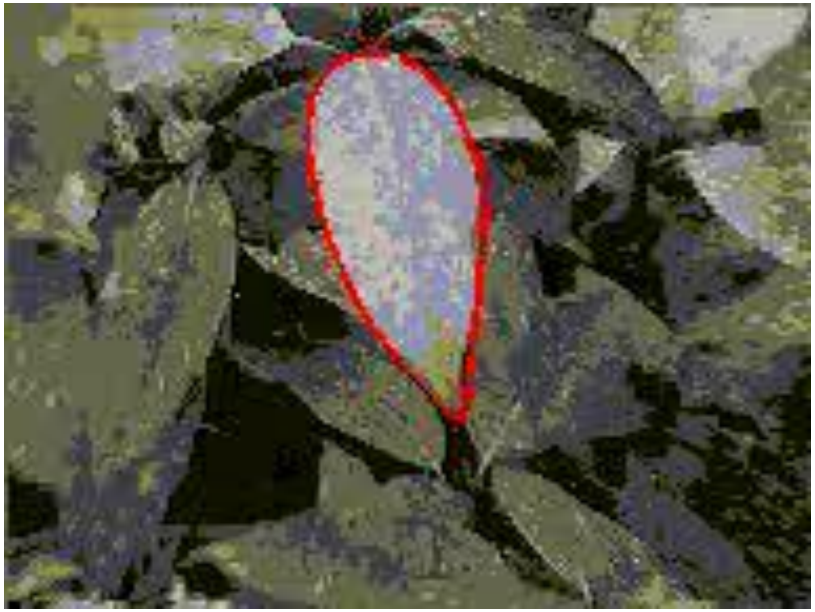
Tracking via Deformable Contours

- **Idea**
  1. Use final contour/model extracted at frame t as an initial solution for frame $t+1$
  2. Evolve initial contour to fit exact object boundary at frame $t+1$
  3. Repeat, initializing with most recent frame.



Tracking Heart Ventricles (multiple frames)

Tracking via Deformable Contours



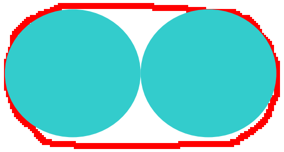
- Many applications
  - Traffic monitoring, surveillance
  - Human-computer interaction
  - Animation
  - Computer assisted diagnosis in medical imaging
  - ...

Slide credit: Kristen Grauman

Video source: M. Isard, B. Bascle, Univ. of Oxford

Limitations

- **Limitations of Dynamic Contours**
  - May over-smooth the boundary



  - Cannot follow topological changes of objects

Limitations

- External energy
  - Snake does not really "see" object boundaries in the image unless it gets very close to them.

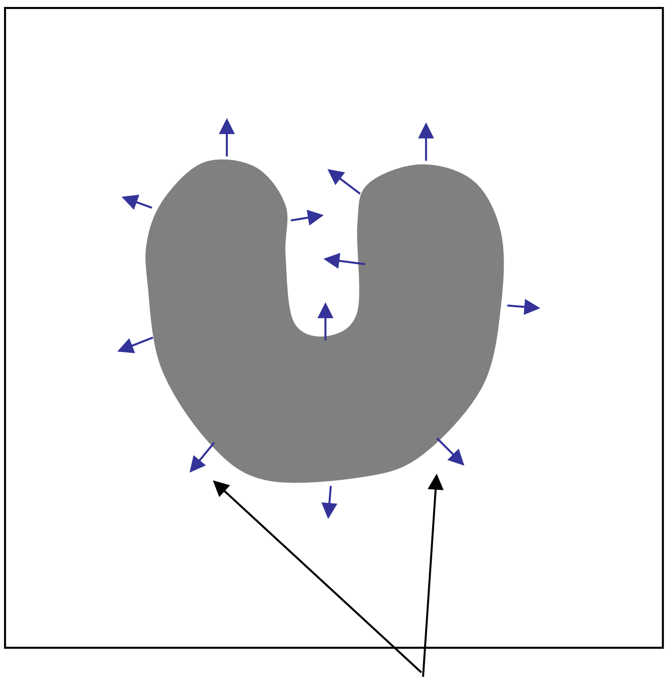
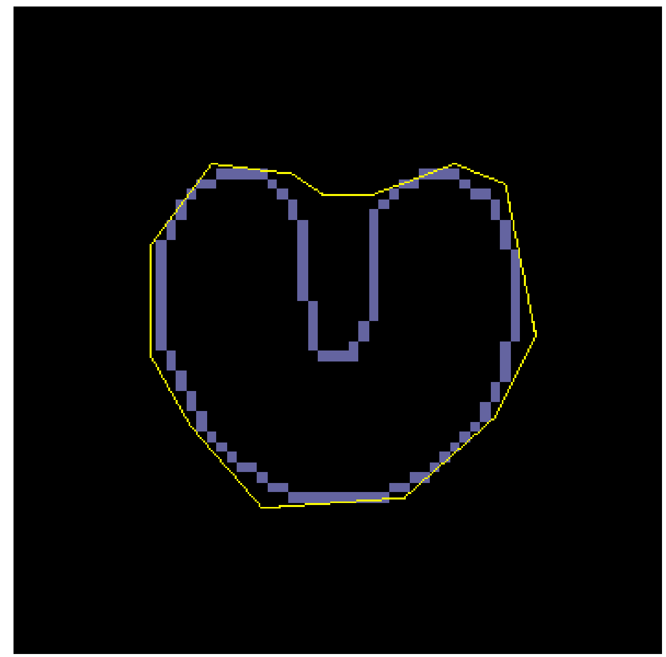


image gradients ∇I are large only directly on the boundary

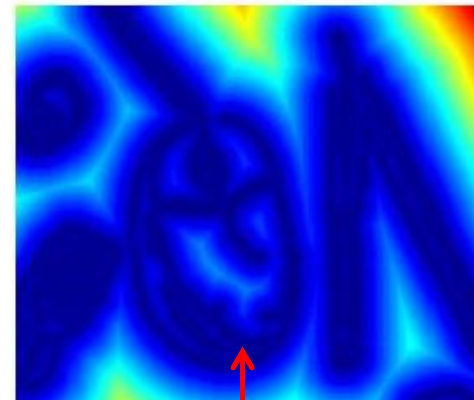


Workaround: Distance Transform

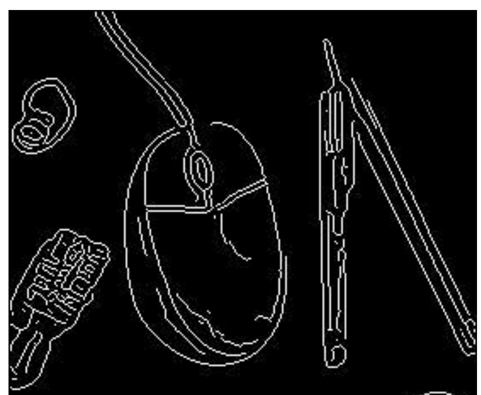
- External energy can instead be taken from the **distance transform** of the edge image.

Original

Gradient



Distance transform



Edges

Value at (x,y) tells how far that position is from the nearest edge point (or other binary image structure)

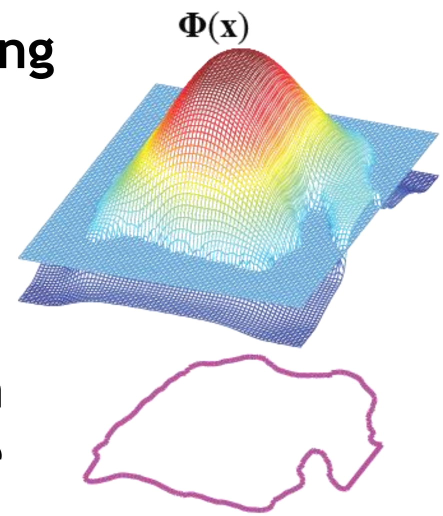
```
>> help bwdist
```

Discussion

- Pros:
  - Useful to track and fit non-rigid shapes
  - Contour remains connected
  - Possible to fill in "subjective" contours
  - Flexibility in how energy function is defined, weighted.
- Cons:
  - Must have decent initialization near true boundary, may get stuck in local minimum.
  - Parameters of energy function must be set well based on prior information
  - Discrete optimization
  - Unable to handle topological changes

Extension: Level Sets

- **Main idea**
  - Instead of explicitly representing the contour to track, model it implicitly as the zero-level set of a continuous embedding function $\Phi(\mathbf{x})$.
  - Evolve the embedding function in order to better fit the image content.
  - Leads to variational approaches.



- **Advantages**
  - Continuous optimization, easier to handle
  - Can naturally cope with topological changes
  - Not restricted to contour information, can also incorporate region information (color, texture, motion, disparity, etc.)

Region-based Level Set Tracking

- Using a color model to separate fg and bg regions

C. Bibby, I. Reid, Robust Real-Time Visual Tracking using Pixel-Wise Posteriors, *ECCV'08*.

[Bibby & Reid, ECCV'08]

Summary

- Deformable shapes and active contours are useful for
  - Segmentation: fit or “snap” to boundary in image
  - Tracking: previous frame’s estimate serves to initialize the next
- Fitting active contours:
  - Define terms to encourage certain shapes, smoothness, low curvature, push/pulls, …
  - Use weights to control relative influence of each component cost
  - Can optimize 2d snakes with Viterbi algorithm.
- Image structure (esp. gradients) can act as attraction force for *interactive* segmentation methods.

References and Further Reading

- The original Snakes paper

  - M. Kass, A. Witkin, D. Terzopoulos. Snakes: Active Contour Models, IJCV1988.

- The Dynamic Programming extension

  - A. Amini, T.E. Weymouth, R.C. Jain. Using Dynamic Programming for Solving Variational Problems in Vision, PAMI, Vol. 12(9), 1990.