

# Mathematical Foundations of Computer Graphics and Vision

## Variational Methods II

Luca Ballan

# Last Lecture

- If we have a topological vector space  $A$  with an inner product
- and functionals of type  $L : A \rightarrow \mathbb{R}$  sufficiently regular
- it is possible to define an object called **directional derivative**

$$\frac{\partial L}{\partial \lambda}(u) = \lim_{\epsilon \rightarrow 0} \frac{L(u + \epsilon \cdot \lambda) - L(u)}{\epsilon} \quad (\text{that not always exists})$$

- But when this exists for all possible directions, and there exist a function

$$\nabla L : A \rightarrow A$$

such that

$$\frac{\partial L}{\partial \lambda}(u) = \langle \nabla L(u), \lambda \rangle_A$$

- then this function is called **gradient** (and it exists very rarely)

# Last Lecture

- If the directional derivative for  $L$  exists for all **points and directions** in  $A$  then the set of stationary point is defined as

$$S_L = \left\{ u \in A \mid \frac{\partial L}{\partial \lambda}(u) = 0_{\mathbb{R}}, \forall \lambda \in A \right\}$$

The set of points in  $A$  where small variations in  $A$  coincide with zero variations in  $L$

- If the gradient for  $L$  exists for all points in  $A$  then this set can be equivalently defined as

$$S_L = \{u \in A \mid \nabla L(u) = 0_A\}$$

# The Space $L^2$

- Given

$$A \subseteq \mathbb{F}(\Omega \subseteq \mathbb{R}^k, \mathbb{R}^m)$$

such that

$$A = \left\{ \forall f : (\Omega \subseteq \mathbb{R}^k) \rightarrow \mathbb{R}^m \mid \int_{\Omega} \|f(x)\|^2 dx \text{ exists} \right\}$$

\*

- Therefore for any functional of type

$$L : L^2(\Omega \subseteq \mathbb{R}^k, \mathbb{R}^m) \rightarrow \mathbb{R}$$

the directional derivative and gradient can be defined (when they exist).

# The functional $L : A \rightarrow \mathbb{R}$

- the **directional derivative**

$$\begin{aligned} \frac{\partial L}{\partial \lambda}(u) &= \lim_{\epsilon \rightarrow 0} \frac{L(u + \epsilon \cdot \lambda) - L(u)}{\epsilon} \\ &= \lim_{\epsilon \rightarrow 0} \frac{\partial}{\partial \epsilon} L(u + \epsilon \cdot \lambda) \end{aligned}$$

(Gâteaux derivative)

where

$$\begin{aligned} u &\in \mathbb{F}(\Omega \subseteq \mathbb{R}^k, \mathbb{R}^m) \\ \lambda &\in \mathbb{F}(\Omega \subseteq \mathbb{R}^k, \mathbb{R}^m) \end{aligned}$$

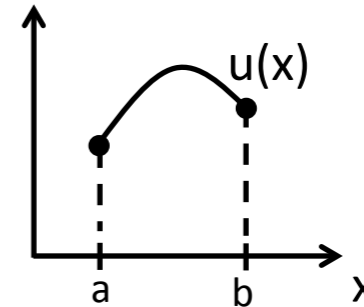
- and the **gradient** is defined as that object of type  $\nabla L : \mathbb{F}(\Omega \subseteq \mathbb{R}^k, \mathbb{R}^m) \rightarrow \mathbb{F}(\Omega \subseteq \mathbb{R}^k, \mathbb{R}^m)$  that satisfies this relationship (if it exists)

$$\begin{aligned} \frac{\partial L}{\partial \lambda}(u) &= \langle \nabla L(u), \lambda \rangle \\ &= \int_{\Omega} \langle \nabla L(u)(x), \lambda(x) \rangle_{\mathbb{R}^m} dx \end{aligned}$$

# A simple Loss Functional

- Given  $L : C^2([a, b], \mathbb{R}) \rightarrow \mathbb{R}$

$$L(u) = \int_a^b \psi(x, u(x), \dot{u}(x)) dx$$

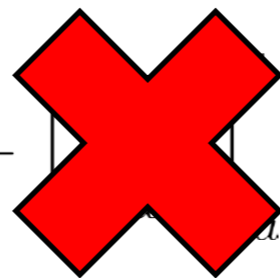


- The directional derivative:

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left( \frac{\partial \psi}{\partial u}(x, u(x), \dot{u}(x)) - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}}(x, u(x), \dot{u}(x)) \right) \lambda(x) dx + \left[ \frac{\partial \psi}{\partial \dot{u}}(x, u(x), \dot{u}(x)) \lambda(x) \right]_a^b$$

- or shortly:

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}} \right) \lambda dx + \left[ \frac{\partial \psi}{\partial \dot{u}} \lambda \right]_a^b$$



In some situations, this second term is zero.

# A simple Loss Functional

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}} \right) \lambda dx$$

Directional derivative

- In these situations the gradient of  $L$  exists (sufficient and necessary condition for its existence)

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}} \right)$$

Gradient  $\in C^0([a, b], \mathbb{R})$



- $u$  is a stationary point/function for  $L$  if and only if

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}} \right) = \mathbf{0}$$

(PDE)

**Euler-Lagrange equation**  
(necessary condition for optimality,  
not in general sufficient)

↑  
has to be equal to the null function!!



# Boundary Conditions

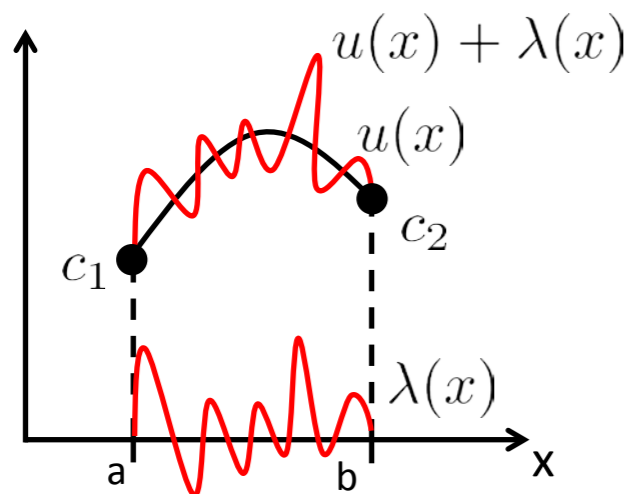
- Sufficient and necessary condition for the gradient to exist is that the second term of the directional derivative is zero

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}} \right) \lambda dx + \left[ \frac{\partial \psi}{\partial \dot{u}}(x, u(x), \dot{u}(x)) \lambda(x) \right]_a^b$$



- If the problem to minimize has a constraint of type 
$$\begin{cases} u(a) = c_1 \\ u(b) = c_2 \end{cases}$$

- it is logical to consider only variations/directions  $\lambda$  which maintain the extremes of the curve



$$\begin{cases} \lambda(a) = 0 \\ \lambda(b) = 0 \end{cases}$$

**Dirichlet boundary condition**



$$\left[ \frac{\partial \psi}{\partial \dot{u}}(x, u(x), \dot{u}(x)) \lambda(x) \right]_a^b = 0$$



# Boundary Conditions

$$\left[ \frac{\partial \psi}{\partial \dot{u}}(x, u(x), \dot{u}(x)) \lambda(x) \right]_a^b = 0$$

$$\Rightarrow \frac{\partial L}{\partial \lambda}(u) = \int_a^b \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial x} \frac{\partial \psi}{\partial \dot{u}} \right) \lambda dx$$

$$\left\{ \begin{array}{l} \lambda(a) = 0 \\ \lambda(b) = 0 \end{array} \right.$$

**Dirichlet boundary condition**

$$\left\{ \begin{array}{l} \frac{\partial \psi}{\partial \dot{u}}(a, u(a), \dot{u}(a)) = 0 \\ \frac{\partial \psi}{\partial \dot{u}}(b, u(b), \dot{u}(b)) = 0 \end{array} \right.$$

**Von Neumann boundary condition**

$$\left\{ \begin{array}{l} u(a) = u(b) \\ \dot{u}(a) = \dot{u}(b) \\ \frac{\partial \psi}{\partial \dot{u}} \text{ does not depend on } x \\ \lambda(a) = \lambda(b) \end{array} \right.$$

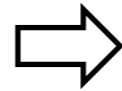
**(no name) boundary condition**  
(but it is the most useful)

\*

# Application Example: Sky-Line detection



Input



Segmented image

$$L(u) = \int_0^1 \underbrace{-E(x, u(x))}^2 + \dot{u}(x)^2 dx$$

$$E(x, y) = \frac{\partial I}{\partial y}(x, y)$$

it penalizes curves passing through non-edge pixels

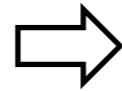


**Edge Map**  
(it is an image)

# Application Example: Sky-Line detection



Input



Segmented image

$$L(u) = \int_0^1 \underbrace{-E(x, u(x))}^2 + \underbrace{\dot{u}(x)}^2 dx$$

$$E(x, y) = \frac{\partial I}{\partial y}(x, y)$$

it penalizes curves passing through non-edge pixels

**Snakes model, or Active Contour Model**

[Kass, Witkin, Terzopoulos IJCV'88]

it penalizes curves which are non-smooth

# Application Example: Sky-Line detection

- To solve this problem, first compute the edge map  $E(x, y) = \frac{\partial I}{\partial y}(x, y)$
- then solve the following minimization problem

$$u^* = \arg \min_{u \in \mathbb{C}^2([0,1], \mathbb{R})} \int_0^1 -E(x, u(x))^2 + \dot{u}(x)^2 dx$$

- compute the Euler-Lagrange Equation

$$\nabla L(u)(x) \left( \frac{\partial \psi}{\partial u} 2E(x, u(x)) \right) \frac{\partial E}{\partial y}(x, u(x)) - 2\ddot{u}(x) = \mathbf{0}$$


\*

- This is still a PDE, but the solution is not obvious, since we do not have a close formula for  $E(x, y)$
- But, we can always resort to a **gradient descent** approach.

# Gradient Descent

- First we need to add a time dimension to our problem  $\longrightarrow u^*(t, x)$
- Solve the dynamical system

$$\begin{cases} u^*(0, x) = u_0(x) \\ \frac{\partial u^*}{\partial t}(t, x) = -\nabla L(u^*(t, x)) \end{cases}$$

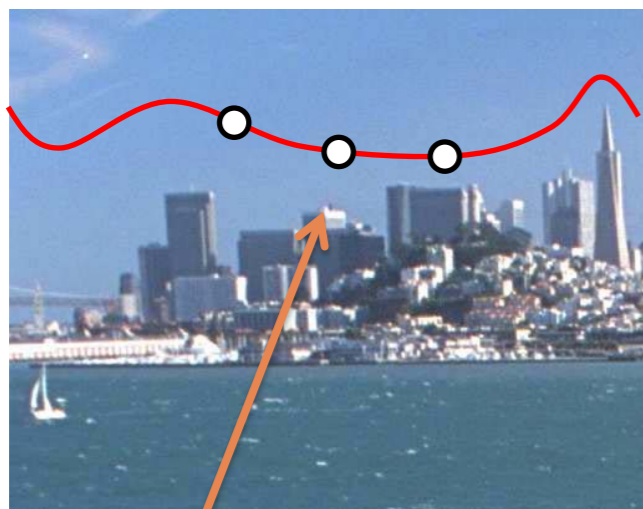
  $u_0(x)$

$$= 2E(x, u^*(t, x)) \frac{\partial E}{\partial y}(x, u^*(t, x)) + 2\ddot{u}^*(t, x)$$

- We know that the solution of our problem is (if it converges)

$$u^*(x) = \lim_{t \rightarrow \infty} u^*(t, x)$$

# Gradient Descent

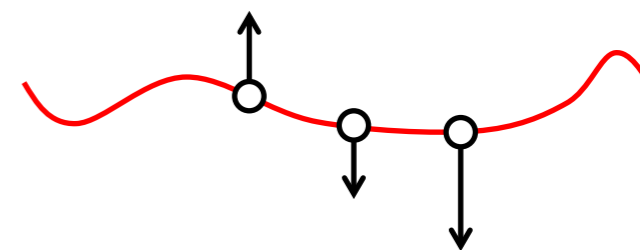


$$u^*(t, x)$$

- Let fix an  $\bar{x}$ ,  $u^*(t, \bar{x})$  represents the y-position of a particle (the particle  $\bar{x}$ ) evolving over time. (something falling down)

- This particle starts from  $u_0(\bar{x})$  at time  $t = 0$

- It then falls down in the image with speed  $\frac{\partial u^*}{\partial t}(t, \bar{x})$





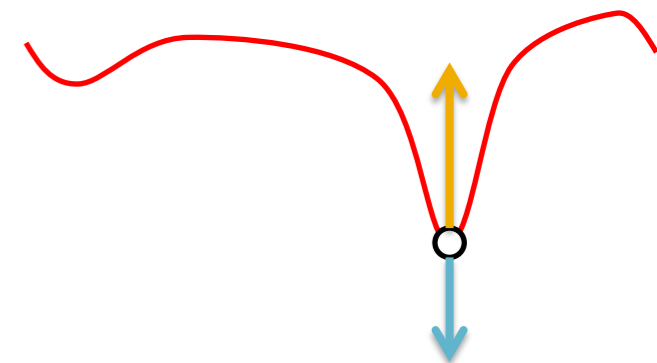
# Gradient Descent

- The speed of this particle

$$\frac{\partial u^*}{\partial t}(t, \bar{x}) = \underbrace{2E(\bar{x}, u^*(t, \bar{x})) \frac{\partial E}{\partial y}(\bar{x}, u^*(t, \bar{x}))}_{\text{depends by the edge map } E} + \underbrace{2\ddot{u}^*(t, \bar{x})}_{\text{and by the second derivative of the curve in } \bar{x}}$$

- These two terms can be thought as

$$\frac{\partial u^*}{\partial t}(t, \bar{x}) = \text{External force on } \bar{x} + \text{Internal force on } \bar{x}$$



# Numerical Implementation

$$\begin{cases} u^*(0, x) = u_0(x) \\ \frac{\partial u^*}{\partial t}(t, x) = -\nabla L(u^*(t, x)) \end{cases}$$

- The curve  $u^*(t, x)$  needs to be discretized in space and time

$$\begin{cases} u^*(0, x) = u_0(x) \\ u^*(t + 1, x) = u^*(t, x) - \beta \nabla L(u^*(t, x)) \end{cases}$$

time discretization

$$x \in \{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$$

space discretization

- $\beta$  is a constant guiding the convergence of the dynamical system: a too high  $\beta$  makes the system unstable; a too small  $\beta$  makes the convergence too slow. From the Finite Element theory, the choice  $\beta$  depends (also) on the used spatial parameterization/sampling  $\{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$
- $\beta$  can be computed in each step using the line search algorithm (no more a curve evolving but a pure optimization technique)



# Numerical Implementation

$$\begin{cases} u^*(0, x) = u_0(x) \\ \frac{\partial u^*}{\partial t}(t, x) = -\nabla L(u^*(t, x)) \end{cases}$$

- The curve  $u^*(t, x)$  needs to be discretized in space and time

$$\begin{cases} u^*(0, x) = u_0(x) \\ u^*(t + 1, x) = u^*(t, x) - \beta \nabla L(u^*(t, x)) \end{cases}$$

time discretization

$$x \in \{0, \frac{1}{n}, \frac{2}{n}, \dots, 1\}$$

space discretization

- for a sufficiently high  $t$ ,  $u^*(t, x)$  will lie on a **local minima** of  $L$

# Gradient Evaluation

$$\nabla L(u^*(t, x)) = \underbrace{-2E(x, u^*(t, x)) \frac{\partial E}{\partial y}(x, u^*(t, x))}_{\text{external force}} - 2\ddot{u}^*(t, x)$$

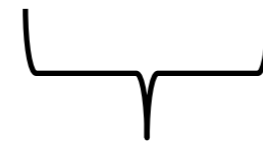
- To compute the external force for a point in the curve (i.e. a particle), we just need to evaluate a point in the function (floating point image)

$$E(x, y) \frac{\partial E}{\partial y}(x, y) \quad \text{where} \quad E(x, y) = \frac{\partial I}{\partial y}(x, y)$$

- this function/image can be pre-computed before running the optimization (e.g. using Sobel filters).
- Btw, better smooth  $I$  before otherwise  $\frac{\partial I}{\partial y}$  is full of Dirac deltas (which are useless for a continuous evolution)
- the evaluation is independent for each particle (to estimate the external force of one particle, we do not need to know the state of the other particles)
- therefore, this operation is parallelizable

# Gradient Evaluation

$$\nabla L(u^*(t, x)) = -2E(x, u^*(t, x)) \frac{\partial E}{\partial y}(x, u^*(t, x)) - 2\underbrace{\ddot{u}^*(t, x)}_{\text{internal force}}$$



internal force

- To compute the internal force for a point in the curve (i.e. a particle), we can approximate the second derivative of the curve using **finite differences**

$$\ddot{u}^*(t, u) = \frac{u^*(t, x + h) - 2u^*(t, x) + u^*(t, x - h)}{h^2}$$

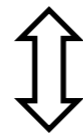
- the evaluation is dependent only on the state of neighboring particles  $x \pm h$
- therefore, this operation is easily parallelizable on a grid (e.g. on GPU)
- (Variational approaches are very suitable for GPU implementations)

# Boundary conditions

$$L(u) = \int_0^1 -E(x, u(x))^2 + \dot{u}(x)^2 dx$$

- We need to make sure that the boundary conditions are satisfied

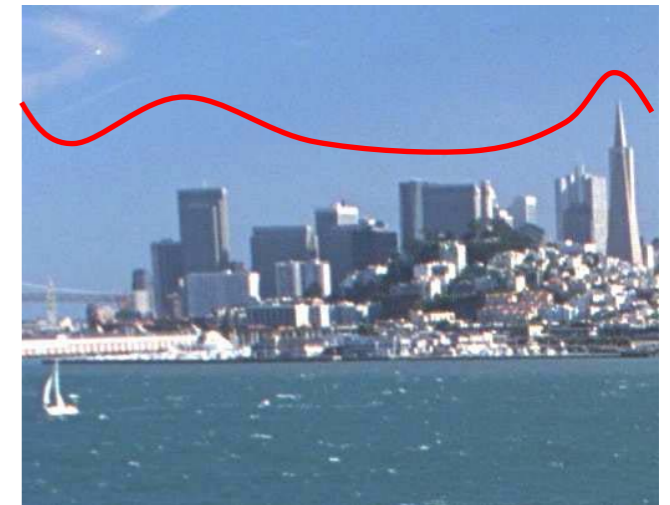
$$\left[ \frac{\partial \psi}{\partial \dot{u}}(x, u(x), \dot{u}(x)) \lambda(x) \right]_a^b = 0$$



$$[2\dot{u}(x)\lambda(x)]_0^1 = 0$$



$$2\dot{u}(1)\lambda(1) - 2\dot{u}(0)\lambda(0) = 0$$



**Dirichlet boundary condition?**

$$\lambda(1) = \lambda(0) = 0$$

Fix the extremes at the starting position: bad

**Von Neumann boundary condition?**

$$\dot{u}(1) = \dot{u}(0) = 0$$

Fix the steepness of the extremes: ok

**No name boundary condition?**

$$u(1) = u(0) \quad \dots \text{ Etc...}$$

Both the extremes has to be at the same height: bad

# Boundary conditions

$$L(u) = \int_0^1 -E(x, u(x))^2 + \dot{u}(x)^2 dx$$

- What if the boundary conditions are not satisfied?
- So, what will this object be?

$$\nabla L(u)(x) = -2E(x, u(x)) \frac{\partial E}{\partial y}(x, u(x)) - 2\ddot{u}(x)$$

- Will it work the same?
- Yes, if  $\nabla L(u)$  is still a descent direction for L  
(instead of a gradient descent technique we will have a descent technique)



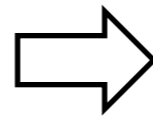
One has to verify this.. And the verification depends from problem to problem...

# Image Segmentation

- The goal of image segmentation is to partition an image into “meaningful” components.
- What is “meaningful” depends on the application. In case of an image representing a real scene, one may want that each of the segmented components corresponds to different physical objects.



Input



Segmented image

# Image Segmentation

Typical approaches:

- **Edge-based methods:** running an edge detector to identify the contours on the color or on the texture. Then group the output into connected curves.
- **Region-based methods:** identify regions in the image for which some criterion is more or less uniform (brightness, color, textures,...) (e.g. by thresholding or clustering). Then use region growing, region merging, connected components etc.. to refine the segmentation.

# Image Segmentation: a Variational Appr.



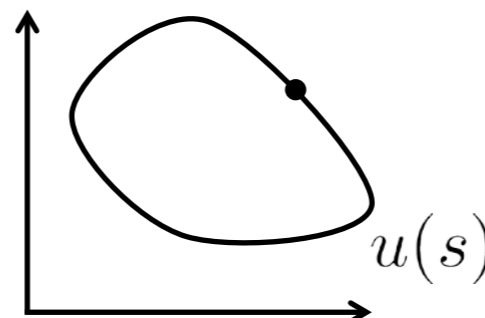
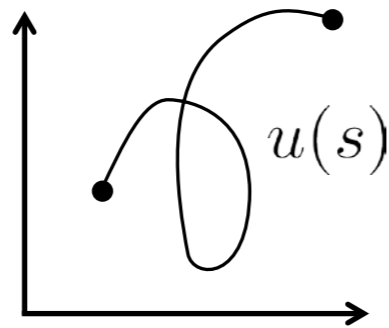
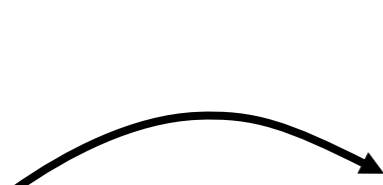
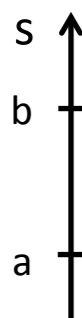
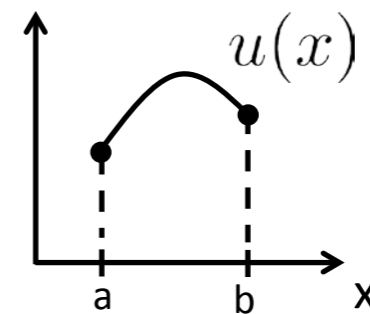
D. Mumford, J. Shah '89

- To get these segmentations we need something more than a curve of type

$$u \in C^2([0, 1], \mathbb{R})$$



$$u \in C^2([0, 1], \mathbb{R}^2)$$



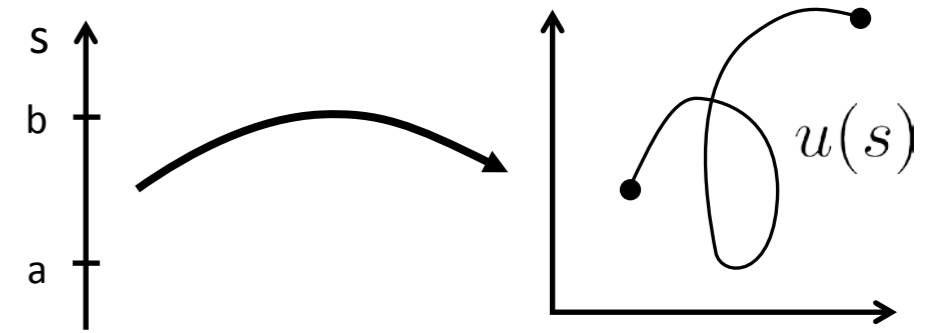
possibly closed



# Snake Model v2.0

$$u \in C^2([0, 1], \mathbb{R}^2)$$

$$L : C^2([0, 1], \mathbb{R}^2) \rightarrow \mathbb{R}$$



$$u^* = \arg \min_{u \in C^2([0, 1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 ds$$

- The edge map  $E(x, y) = \|\nabla I(x, y)\|$

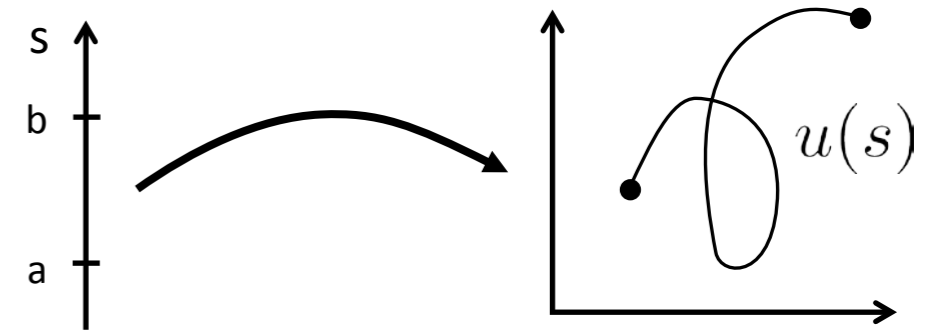
$$L(u) = \int_0^1 -E(x, u(x))^2 + \dot{u}(x)^2 dx$$

Snake 1.0

# Snake Model v2.0

$$u \in C^2([0, 1], \mathbb{R}^2)$$

$$L : C^2([0, 1], \mathbb{R}^2) \rightarrow \mathbb{R}$$



$$u^* = \arg \min_{u \in C^2([0, 1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 ds$$

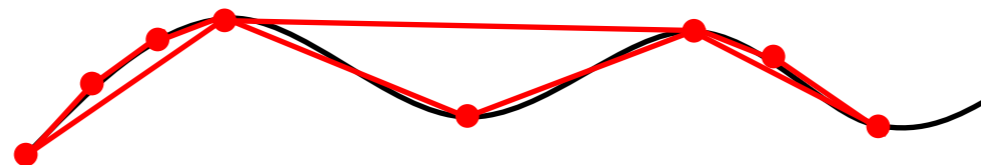
$$L(u) = \int_0^1 -E(x, u(x))^2 + \dot{u}(x)^2 dx$$

Snake 1.0

- The edge map  $E(x, y) = \|\nabla I(x, y)\|$

- $\dot{u}(s)$  in this case does not enforce the curve to be smooth but, instead, it forces the parameterization to be smooth.

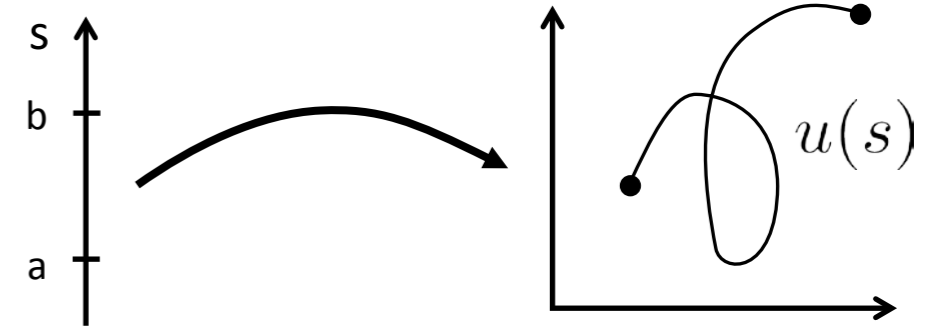
- if  $u(s)$  is discretized, the term  $\|\dot{u}(s)\|^2$  would penalize non-uniform sampling of the curves



# Snake Model v2.1

$$u \in C^2([0, 1], \mathbb{R}^2)$$

$$L : C^2([0, 1], \mathbb{R}^2) \rightarrow \mathbb{R}$$

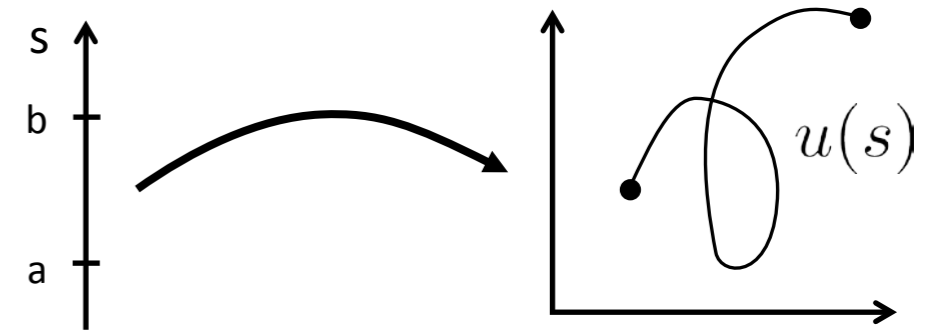


$$u^* = \arg \min_{u \in C^2([0, 1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 ds$$

# Snake Model v2.1

$$u \in \underline{C^4([0, 1], \mathbb{R}^2)}$$

$$L : \underline{C^4([0, 1], \mathbb{R}^2)} \rightarrow \mathbb{R}$$



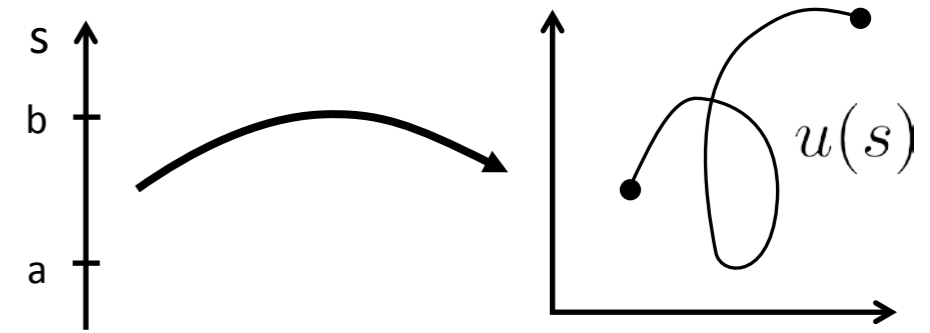
$$u^* = \arg \min_{u \in C^4([0, 1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 + \underline{\frac{\beta}{2} \|\ddot{u}(s)\|^2} ds$$

- the term  $\|\ddot{u}(s)\|$  is equal to the curvature of the curve, in case of uniform parameterization

# Snake Model v2.1

$$u \in \underline{C^4([0, 1], \mathbb{R}^2)}$$

$$L : \underline{C^4([0, 1], \mathbb{R}^2)} \rightarrow \mathbb{R}$$



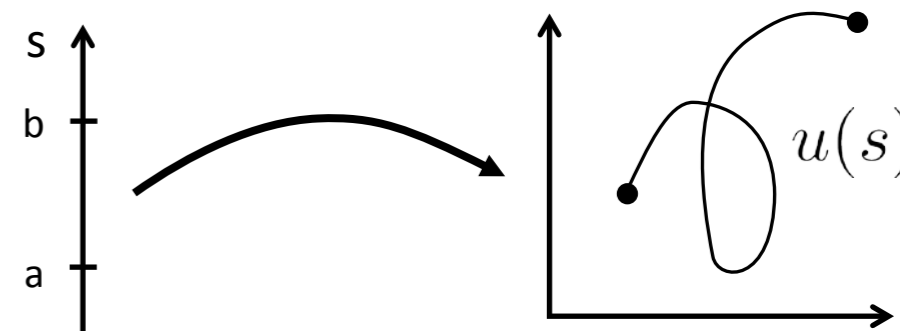
$$u^* = \arg \min_{u \in C^4([0, 1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 + \underline{\frac{\beta}{2} \|\ddot{u}(s)\|^2} ds$$

- the term  $\|\ddot{u}(s)\|$  is equal to the curvature of the curve, in case of uniform parameterization
- The functional we have now is more complex than the one before

$$L(u) = \int_a^b \psi(s, u(s), \dot{u}(s), \underline{\ddot{u}(s)}) ds \quad u : [0, 1] \rightarrow \underline{\mathbb{R}^2}$$

# Another Loss Functional #2

- Given  $L : C^2([a, b], \mathbb{R}^m) \rightarrow \mathbb{R}$



$$L(u) = \int_a^b \psi(s, u_1(s), \dots, u_m(s), \dot{u}_1(s), \dots, \dot{u}_m(s)) ds$$

- The directional derivative is in this case

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left\langle \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} \right) (s), \lambda(s) \right\rangle_{\mathbb{R}^m} ds + \left[ \text{Red X} \right]_a^b$$

$\uparrow$   
 $[a, b] \rightarrow \mathbb{R}^m$

$\uparrow$   
 $[a, b] \rightarrow \mathbb{R}^m$

# Another Loss Functional #2: Summary

$$L(u) = \int_a^b \psi(s, u(s), \dot{u}(s)) ds$$

Functional  $L : C^2([a, b], \mathbb{R}^m) \rightarrow \mathbb{R}$

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left\langle \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} \right) (s), \lambda(s) \right\rangle_{\mathbb{R}^m} ds$$

Directional derivative

$$\frac{\partial L}{\partial \lambda}(u) = \langle \nabla L(u), \lambda \rangle = \int_a^b \langle \nabla L(u)(s), \lambda(s) \rangle_{\mathbb{R}^m} ds$$

# Another Loss Functional #2: Summary

$$L(u) = \int_a^b \psi(s, u(s), \dot{u}(s)) ds$$

Functional  $L : C^2([a, b], \mathbb{R}^m) \rightarrow \mathbb{R}$

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left\langle \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} \right) (s), \lambda(s) \right\rangle_{\mathbb{R}^m} ds$$

Directional derivative

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} \right)$$

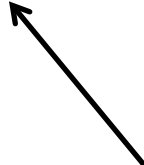
Gradient  $\in C^0([a, b], \mathbb{R}^m)$

- $u$  is a stationary point/function for  $L$  if and only if

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} \right) = \mathbf{0}$$

**Euler-Lagrange equation**

this is the null function in  $C^2([a, b], \mathbb{R}^m)$





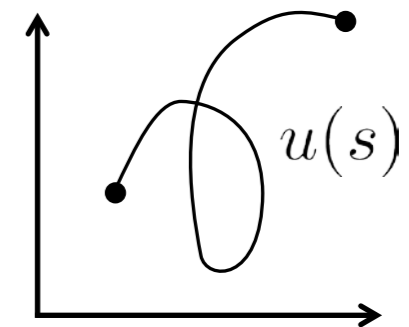
# Boundary Conditions

$$\left[ \left\langle \frac{\partial \psi}{\partial \dot{u}}, \lambda \right\rangle \right]_a^b = 0 \quad \iff \quad \left[ \sum_{i=1}^m \frac{\partial \psi}{\partial \dot{u}_i}(s, u_1(s), \dots, u_m(s), \dot{u}_1(s), \dots, \dot{u}_m(s)) \lambda_i(s) \right]_a^b = 0$$

$$\left\{ \begin{array}{l} \lambda(a) = 0 \\ \lambda(b) = 0 \end{array} \right.$$

## Dirichlet boundary condition

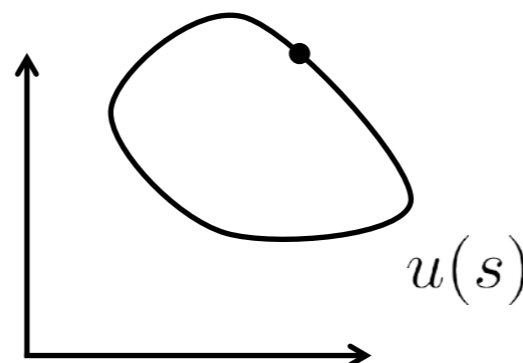
(lambda is just noise on the curve  $u(t)$  which does not affect the extremes  $u(a)$  and  $u(b)$ )



$$\left\{ \begin{array}{l} u(a) = u(b) \\ \dot{u}(a) = \dot{u}(b) \\ \frac{\partial \psi}{\partial \dot{u}} \text{ does not depend on s} \\ \lambda(a) = \lambda(b) \end{array} \right.$$

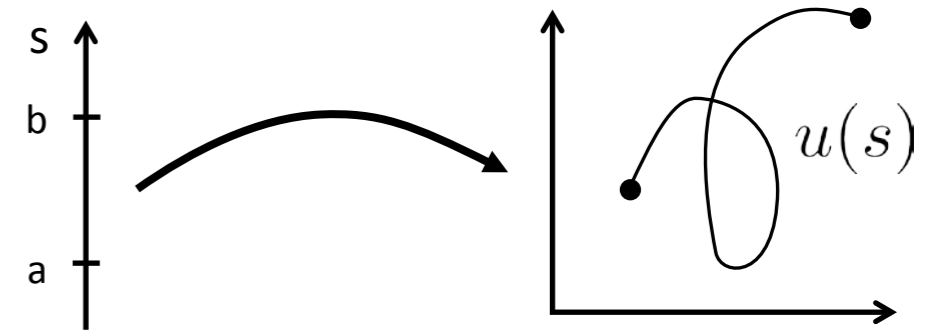
## (no name) boundary condition

The curve must be close and with continuous first derivative at the closure point



# Another Loss Functional #3

- Given  $L : C^4([a, b], \mathbb{R}^m) \rightarrow \mathbb{R}$



$$L(u) = \int_a^b \psi(s, u(s), \dot{u}(s), \ddot{u}(s)) ds$$

- The directional derivative is in this case

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left\langle \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} + \frac{\partial^2}{\partial s^2} \frac{\partial \psi}{\partial \ddot{u}} \right), \lambda \right\rangle ds + \left[ \text{red X} \right]_a^b + \left[ \text{red X} \right]_a^b - \left[ \text{red X} \right]_a^b$$

# Another Loss Functional #2: Summary

$$L(u) = \int_a^b \psi(s, u(s), \dot{u}(s), \ddot{u}(s)) ds$$

Functional  $L : C^4([a, b], \mathbb{R}^m) \rightarrow \mathbb{R}$

$$\frac{\partial L}{\partial \lambda}(u) = \int_a^b \left\langle \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} + \frac{\partial^2}{\partial s^2} \frac{\partial \psi}{\partial \ddot{u}} \right), \lambda \right\rangle ds$$

Directional derivative

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} + \frac{\partial^2}{\partial s^2} \frac{\partial \psi}{\partial \ddot{u}} \right)$$

Gradient  $\in C^0([a, b], \mathbb{R}^m)$

- $u$  is a stationary point/function for  $L$  if and only if

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} + \frac{\partial^2}{\partial s^2} \frac{\partial \psi}{\partial \ddot{u}} \right) = \mathbf{0}$$

**Euler-Lagrange equation**

# Boundary Conditions

$$\left[ \left\langle \frac{\partial \psi}{\partial \dot{u}}, \lambda \right\rangle \right]_a^b + \left[ \left\langle \frac{\partial \psi}{\partial \ddot{u}}, \dot{\lambda} \right\rangle \right]_a^b - \left[ \left\langle \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \ddot{u}} (s, u(s), \dot{u}(s), \ddot{u}(s)), \lambda \right\rangle \right]_a^b = 0$$

$$\left\{ \begin{array}{l} \lambda(a) = 0 \\ \lambda(b) = 0 \\ \dot{\lambda}(a) = 0 \\ \dot{\lambda}(b) = 0 \end{array} \right.$$

## Dirichlet boundary condition

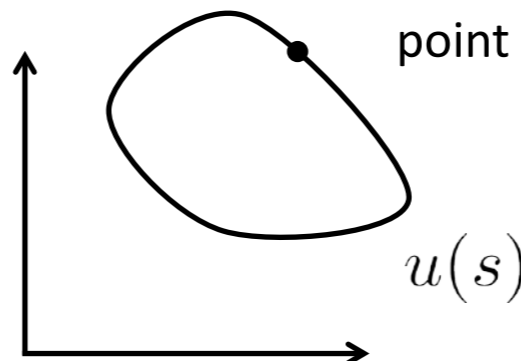
(lambda is just noise on the curve  $u(t)$  which does not affect the extremes  $u(a)$  and  $u(b)$  and their first derivatives)

(bad for our case)

$$\left\{ \begin{array}{l} u(a) = u(b) \\ \dot{u}(a) = \dot{u}(b) \\ \ddot{u}(a) = \ddot{u}(b) \\ \dddot{u}(a) = \dddot{u}(b) \\ \frac{\partial \psi}{\partial \dot{u}} \text{ does not depend on } s \\ \frac{\partial \psi}{\partial \ddot{u}} \text{ does not depend on } s \\ \lambda(a) = \lambda(b) \quad \dot{\lambda}(a) = \dot{\lambda}(b) \end{array} \right.$$

## (no name) boundary condition

The curve must be close and with continuous first, second and third derivatives at the closure point



# Snake Model v2.1

- Now we can solve for the snake model

$$u^* = \arg \min_{u \in \mathbb{C}^4([0,1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 + \frac{\beta}{2} \|\ddot{u}(s)\|^2 ds$$

$$\begin{aligned} \text{subject to } u(0) &= u(1) \\ \dot{u}(0) &= \dot{u}(1) \\ \ddot{u}(0) &= \ddot{u}(1) \\ \dddot{u}(0) &= \dddot{u}(1) \end{aligned}$$

- The gradient is

$$\nabla L(u) = \left( \frac{\partial \psi}{\partial u} - \frac{\partial}{\partial s} \frac{\partial \psi}{\partial \dot{u}} + \frac{\partial^2}{\partial s^2} \frac{\partial \psi}{\partial \ddot{u}} \right)$$



$$\nabla L(u) = -2E(u(s)) \nabla E(u(s)) - \alpha \ddot{u}(s) + \beta \dddot{u}(s)$$

$$E(x, y) = \|\nabla I(x, y)\|$$

# Snake Model v2.1

$$u^* = \arg \min_{u \in \mathbb{C}^4([0,1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 + \frac{\beta}{2} \|\ddot{u}(s)\|^2 ds$$

subject to  $u(0) = u(1)$

$$\dot{u}(0) = \dot{u}(1)$$

$$\ddot{u}(0) = \ddot{u}(1)$$

$$\dddot{u}(0) = \dddot{u}(1)$$

$$\left\{ \begin{array}{l} u(a) = u(b) \\ \dot{u}(a) = \dot{u}(b) \\ \ddot{u}(a) = \ddot{u}(b) \\ \dddot{u}(a) = \dddot{u}(b) \\ \frac{\partial \psi}{\partial \dot{u}} \text{ does not depend on } s \\ \frac{\partial \psi}{\partial \ddot{u}} \text{ does not depend on } s \\ \lambda(a) = \lambda(b) \quad \dot{\lambda}(a) = \dot{\lambda}(b) \end{array} \right.$$

$\Rightarrow$  (no name) boundary condition

# Snake Model v2.1

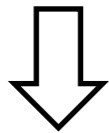
$$u^* = \arg \min_{u \in \mathbb{C}^4([0,1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 + \frac{\beta}{2} \|\ddot{u}(s)\|^2 ds$$

subject to  $u(0) = u(1)$

$$\dot{u}(0) = \dot{u}(1)$$

$$\ddot{u}(0) = \ddot{u}(1)$$

$$\dddot{u}(0) = \dddot{u}(1)$$



$$\nabla L(u) = \underbrace{-2E(u(s))\nabla E(u(s))}_{\text{External "force" } \in \mathbb{R}^2} - \underbrace{\alpha\ddot{u}(s) + \beta\dddot{u}(s)}_{\text{Internal "force" } \in \mathbb{R}^2}$$

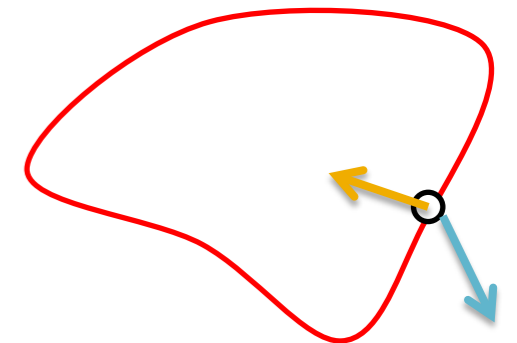
External "force"

$$\in \mathbb{R}^2$$

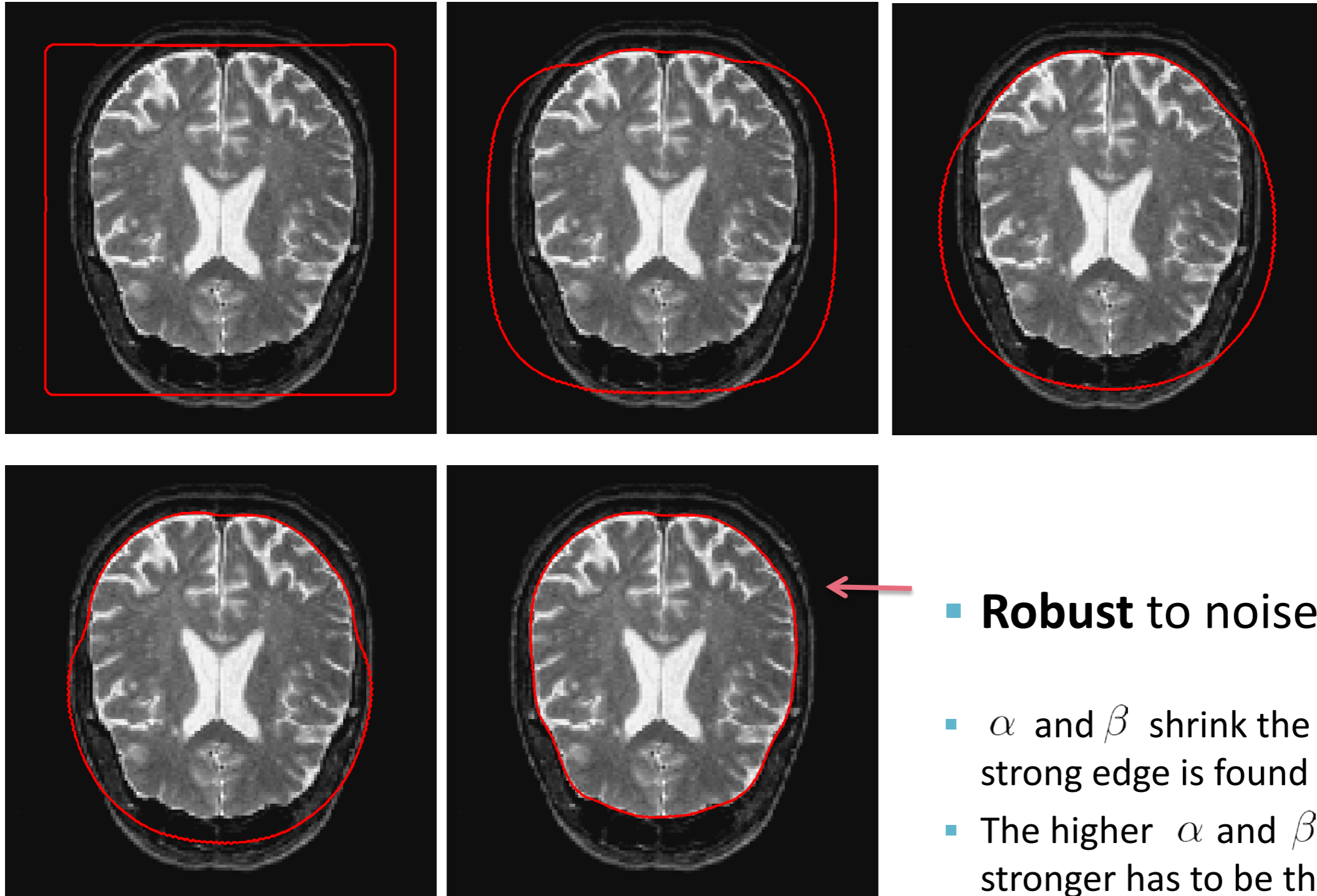
Internal "force"

$$\in \mathbb{R}^2$$

2D vector field in the  
image space



# Snake Model v2.1: Evolution example

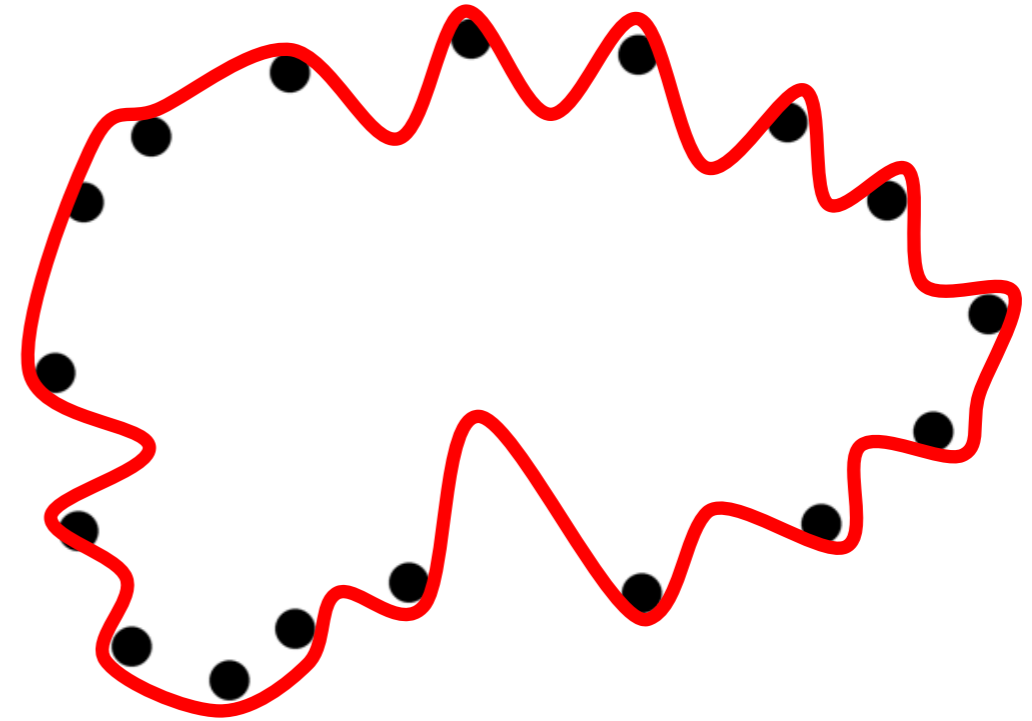
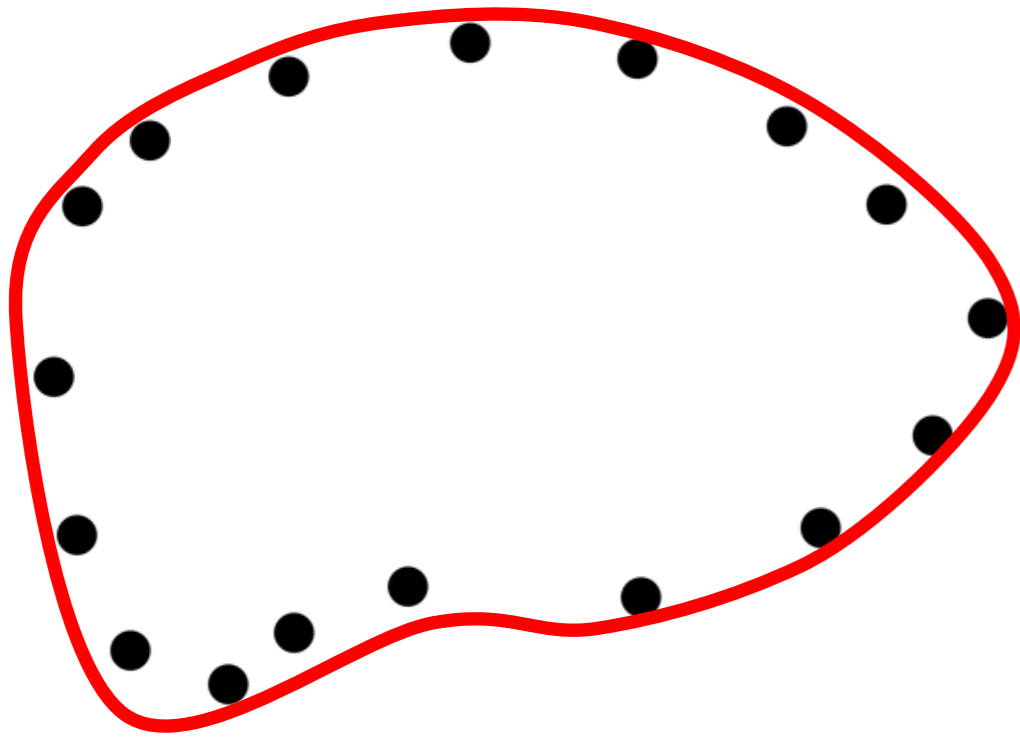


- **Robust to noise**

- $\alpha$  and  $\beta$  shrink the snake till a strong edge is found
- The higher  $\alpha$  and  $\beta$  are, the stronger has to be the edge (these have to be tuned)



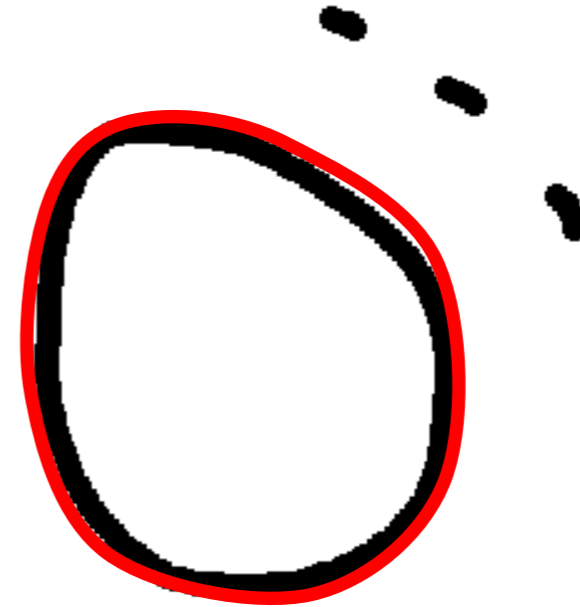
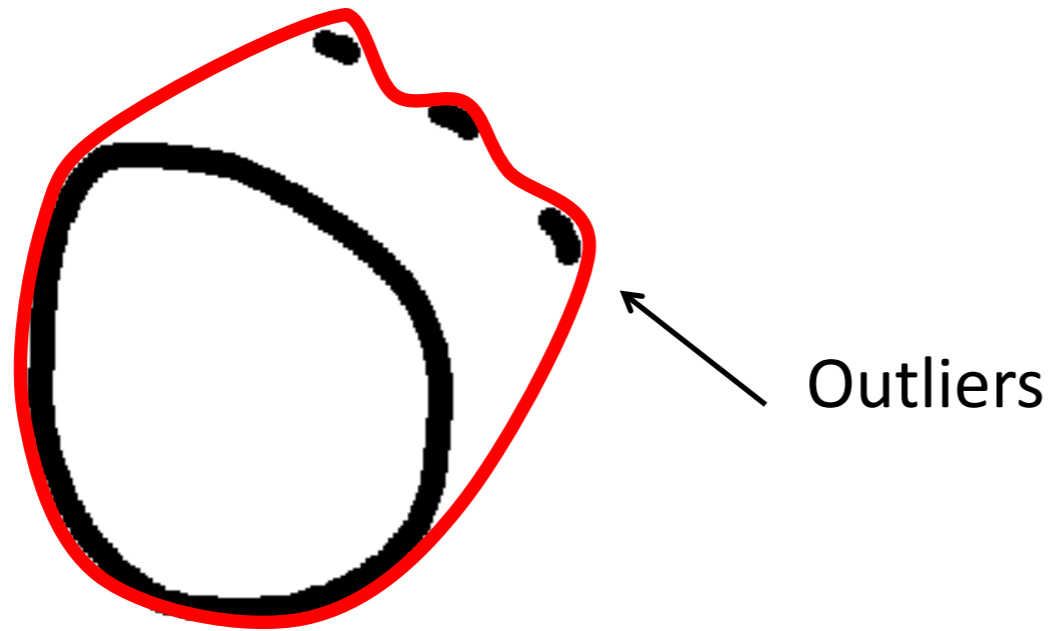
# Lack of information



The result here depends on the chosen prior, i.e. the values of  $\alpha$  and  $\beta$

The higher they are, the more robust the method is to lack of information

# Outliers



Again, the result depends on the choice of  $\alpha$  and  $\beta$

The higher they are, the more robust the method is to outliers

# Outliers, Noise, Lack of Information

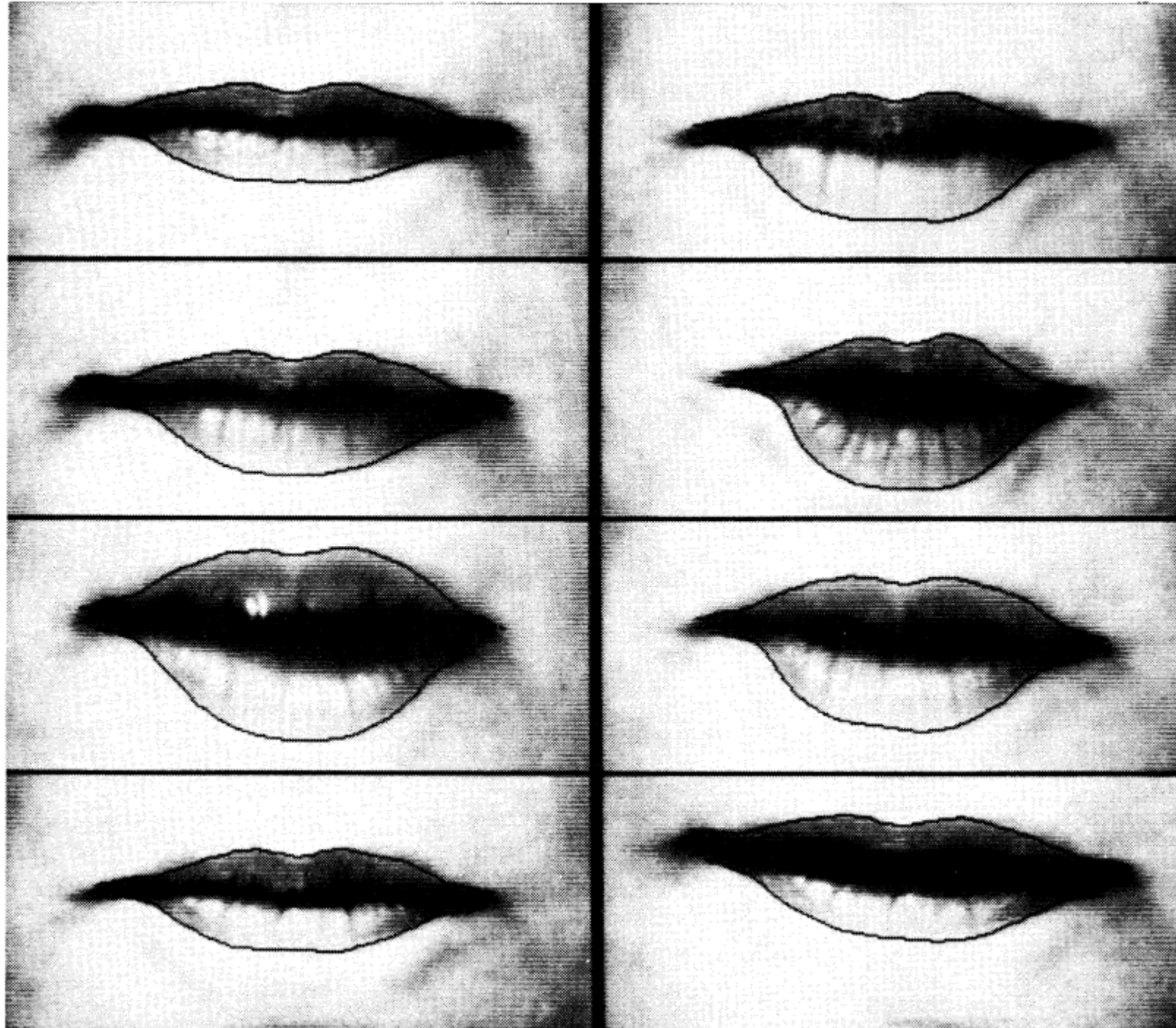


# Outliers, Noise, Lack of Information

- Smooth curves make the approach robust to noise, “outliers” (still an L2), and lack of information
- Too smooth!! is this what we want? No more details
- real detail      vs.      outlier/noise      (L2 smoothness term)

# Tracking dynamic object

- Very fast to track dynamic elements, using the previous frame as initialization of the snake
- with very few iterations it will converge



# Problems

- It inherits all the problems of a gradient descent technique
  - it converges to a local minima
  - these local minima can densely populate the space of solution
    - especially for high dimensional problems
    - therefore, once we get out from one local minimum it will fall right after into another one.
  - convergence can slow down in case of flat areas of the functional (line search)

\*

# Problems

- It inherits all the problems of a gradient descent technique
  - it converges to a local minima
  - these local minima can densely populate the space of solution
    - especially for high dimensional problems
    - therefore, once we get out from one local minimum it will fall right after into another one.
  - convergence can slow down in case of flat areas of the functional (line search)
- The way the problem is formulated can introduce
  - multiple global minima
  - The term  $-E(u(s))^2$  introduces a lot of flat areas in the functional (gradient information is only local not global) [Resort to descent technique, not gradient] \*
  - $\alpha$  and  $\beta$  have to be decided a priori
    - internal forces tend to shrink the snake into a single point (the higher are these constant the higher is this effect)



# Problems

$$u^* = \arg \min_{u \in \mathbb{C}^4([0,1], \mathbb{R}^2)} \int_0^1 -E(u(s))^2 + \frac{\alpha}{2} \|\dot{u}(s)\|^2 + \frac{\beta}{2} \|\ddot{u}(s)\|^2 ds$$

subject to

$$u(0) = u(1)$$
$$\dot{u}(0) = \dot{u}(1)$$
$$\ddot{u}(0) = \ddot{u}(1)$$
$$\dddot{u}(0) = \dddot{u}(1)$$
$$u \neq \emptyset$$

- The global/local minimum of this functional is the **empty set**  $\rightarrow L(u) = 0$
- Our gradient descent approach implicitly excluded it as a possible solution by just stopping to the first local minimum, but

