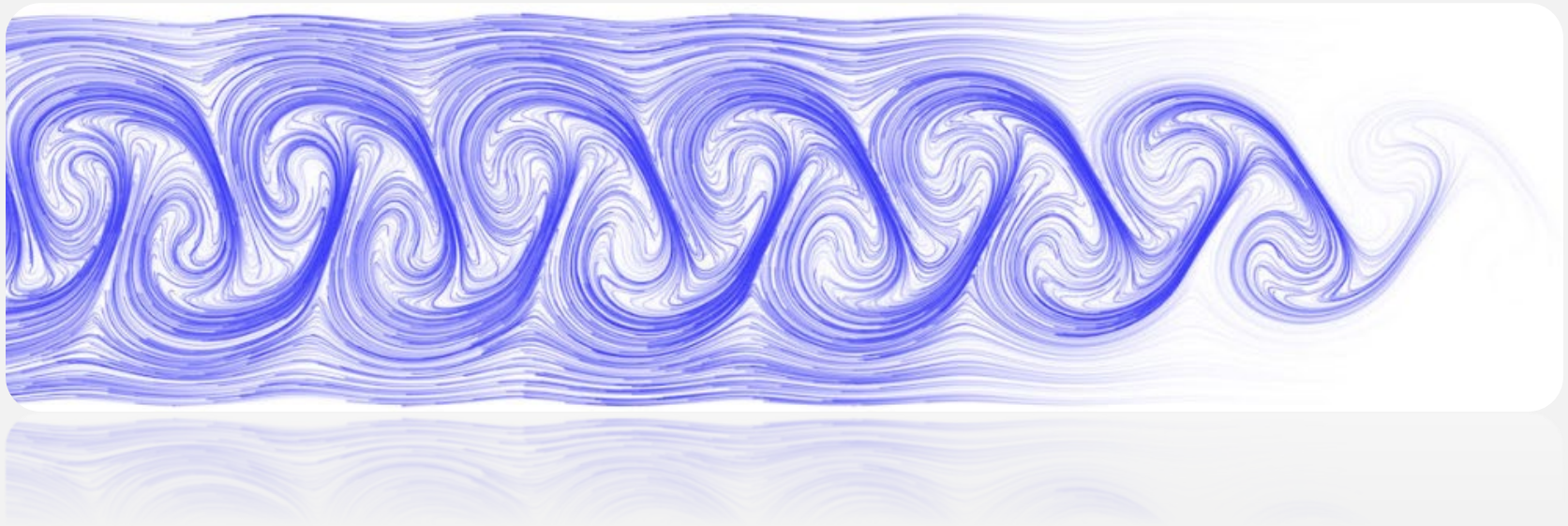
IFT 6113

APPLICATIONS OF VECTOR FIELDS

tiny.cc/6113



Mikhail Bessmeltsev

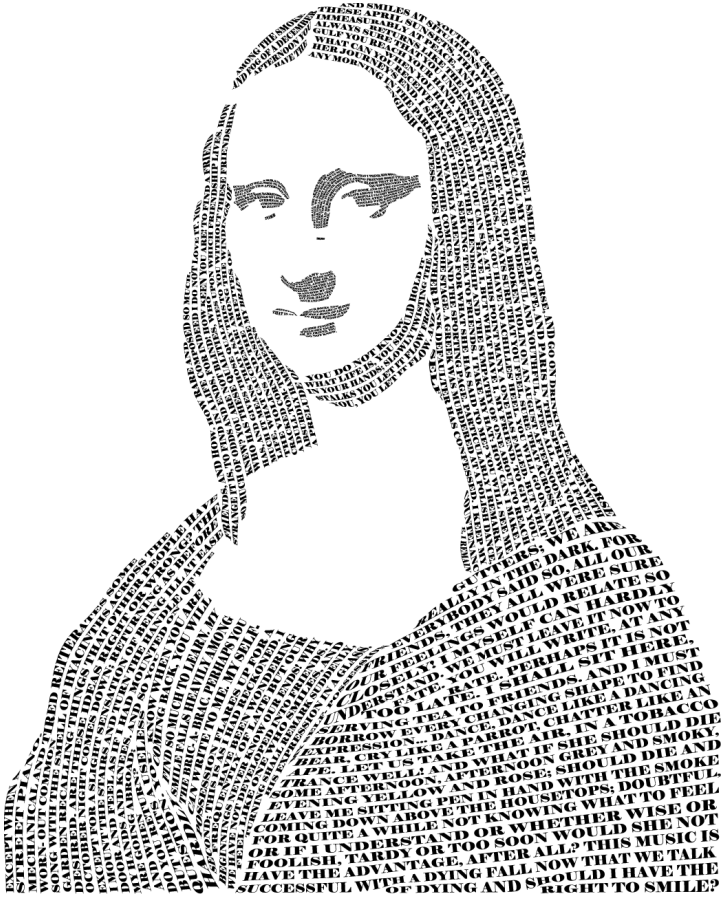
Outline

- Geometry processing
  - Mesh Generation
  - Deformation
  - Texture mapping and synthesis
- Misc
  - Non-photorealistic rendering
  - Crowd simulation

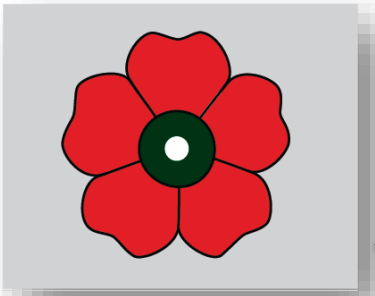
Outline

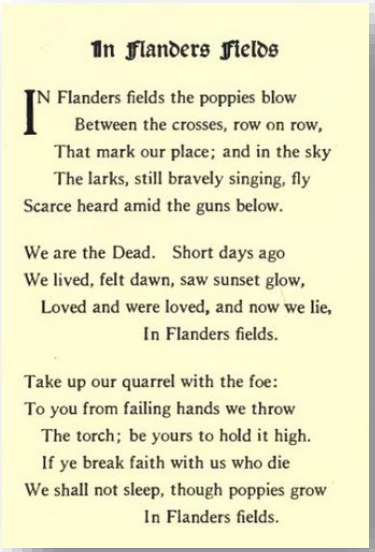
- Geometry processing
  - Mesh Generation
  - Deformation
  - Texture mapping and synthesis
- **Misc**
  - Non-photorealistic rendering
  - Crowd simulation

2D: Digital Micrography



2D: Digital Micrography



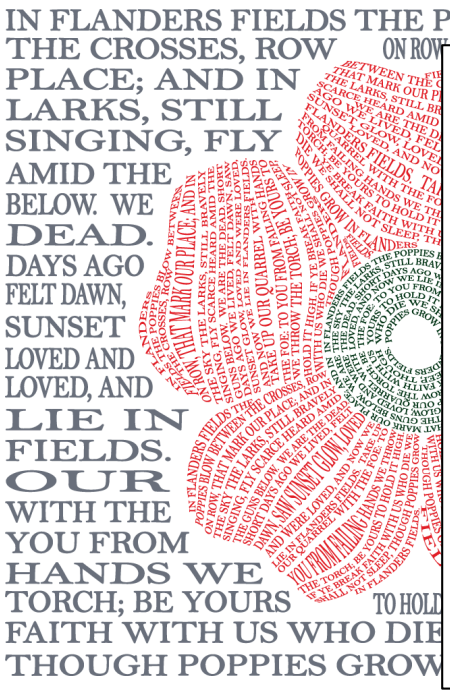


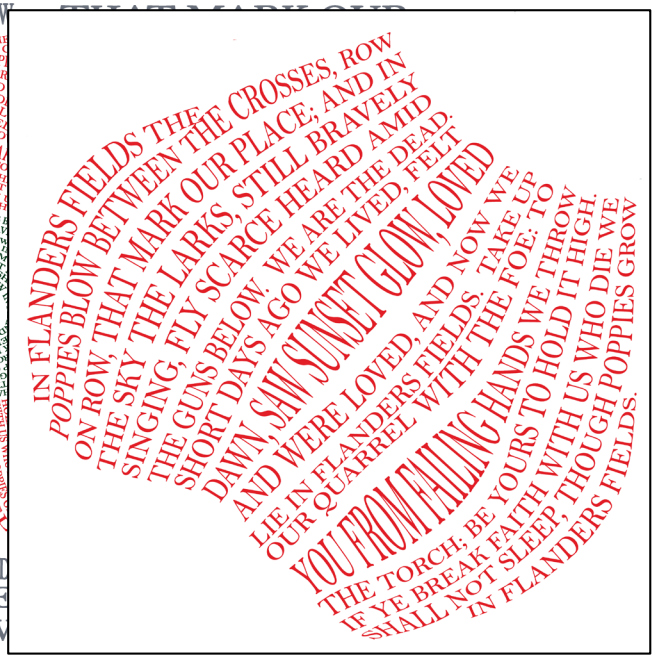
In Flanders Fields

IN Flanders fields the poppies blow
Between the crosses, row on row,
That mark our place; and in the sky
The larks, still bravely singing, fly
Scarce heard amid the guns below.

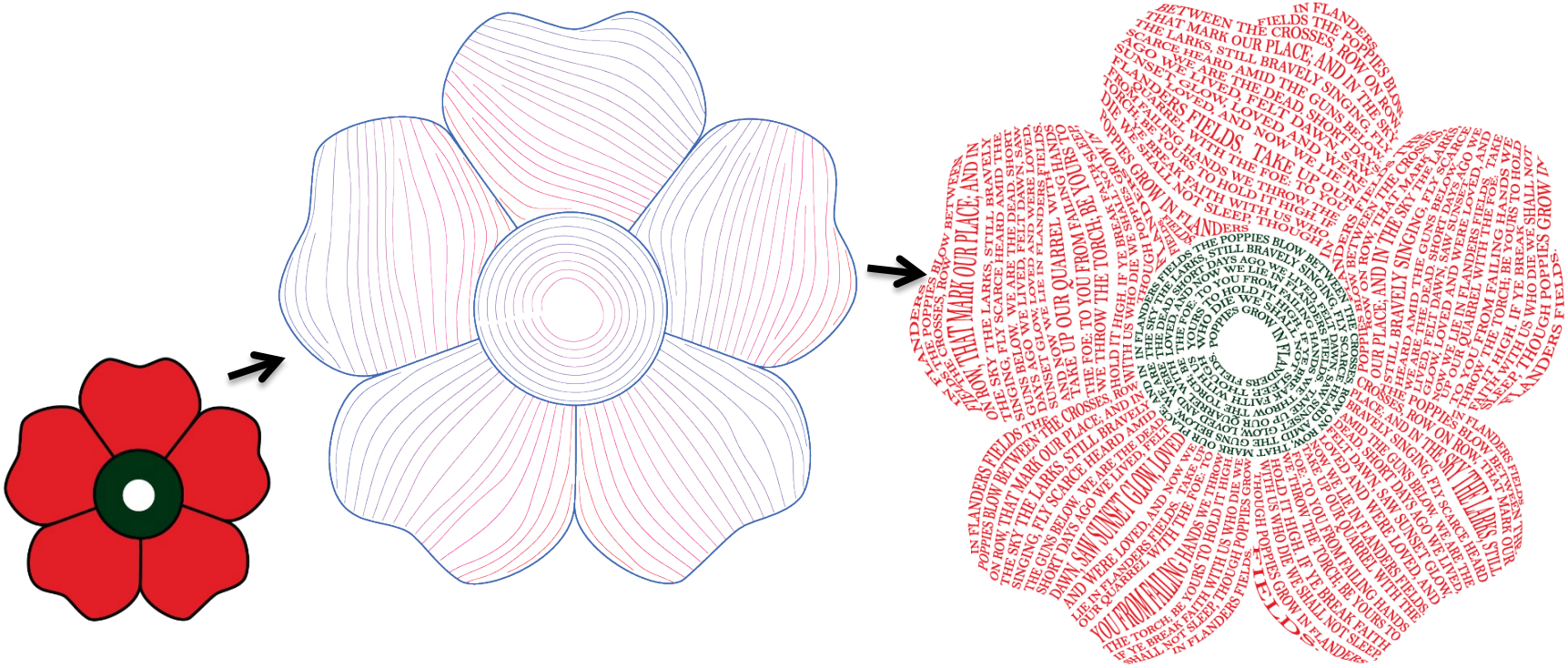
We are the Dead. Short days ago
We lived, felt dawn, saw sunset glow,
Loved and were loved, and now we lie,
In Flanders fields.

Take up our quarrel with the foe:
To you from failing hands we throw
The torch; be yours to hold it high.
If ye break faith with us who die
We shall not sleep, though poppies grow
In Flanders fields.



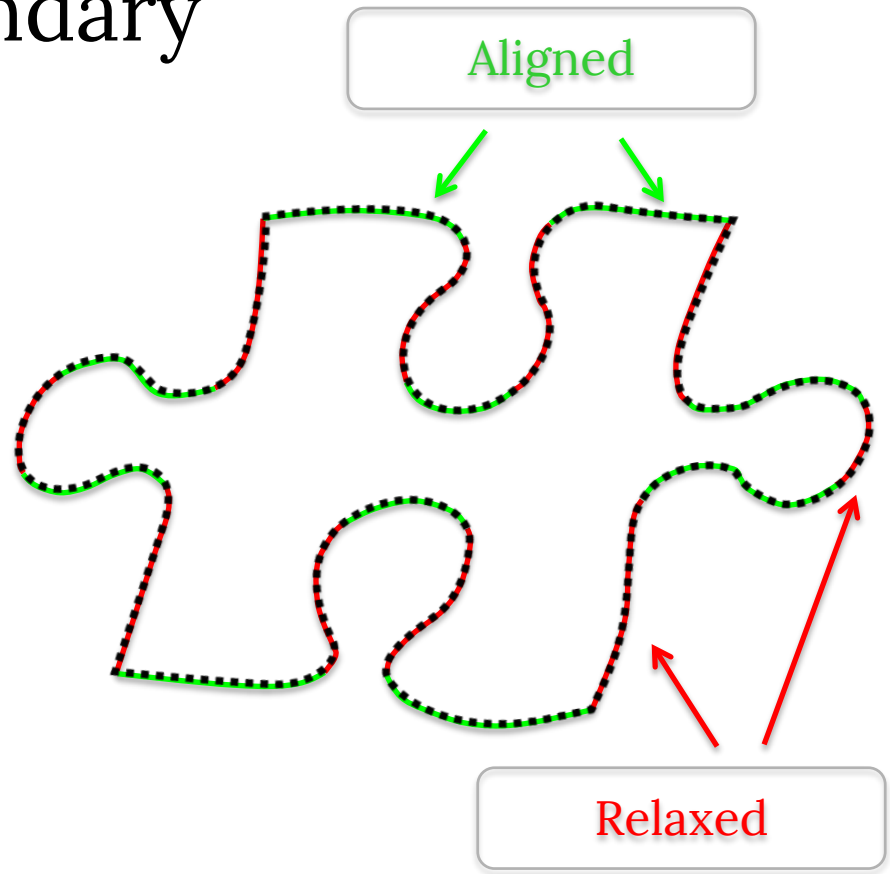


2D: Digital Micrography

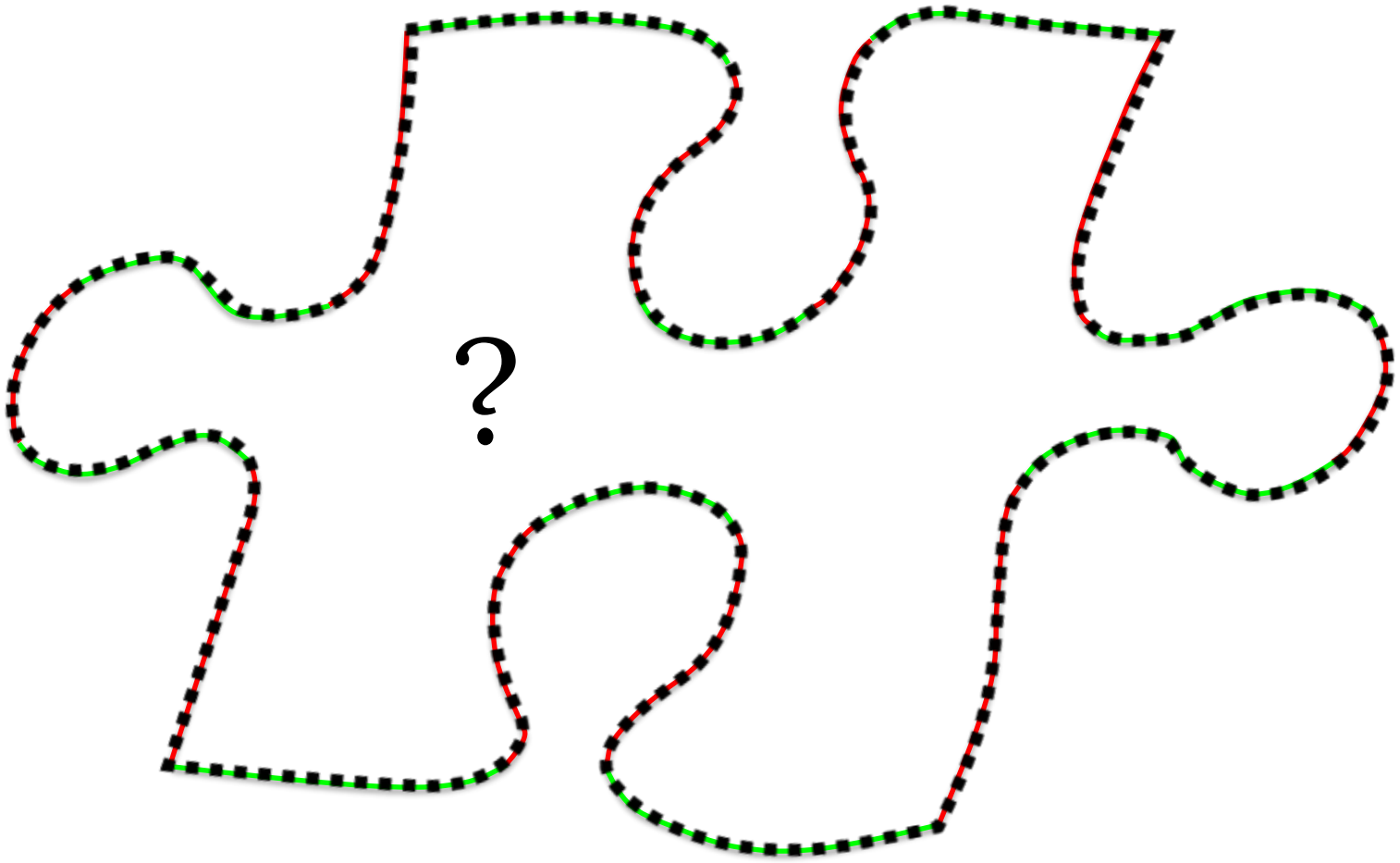


Boundary conditions

- Vector field is parallel or perpendicular to the boundary



Inside?



Inside?

- Smoothest interpolation of boundary values
- Laplace equation with Dirichlet boundary conditions
- Discretization?
- Representation?

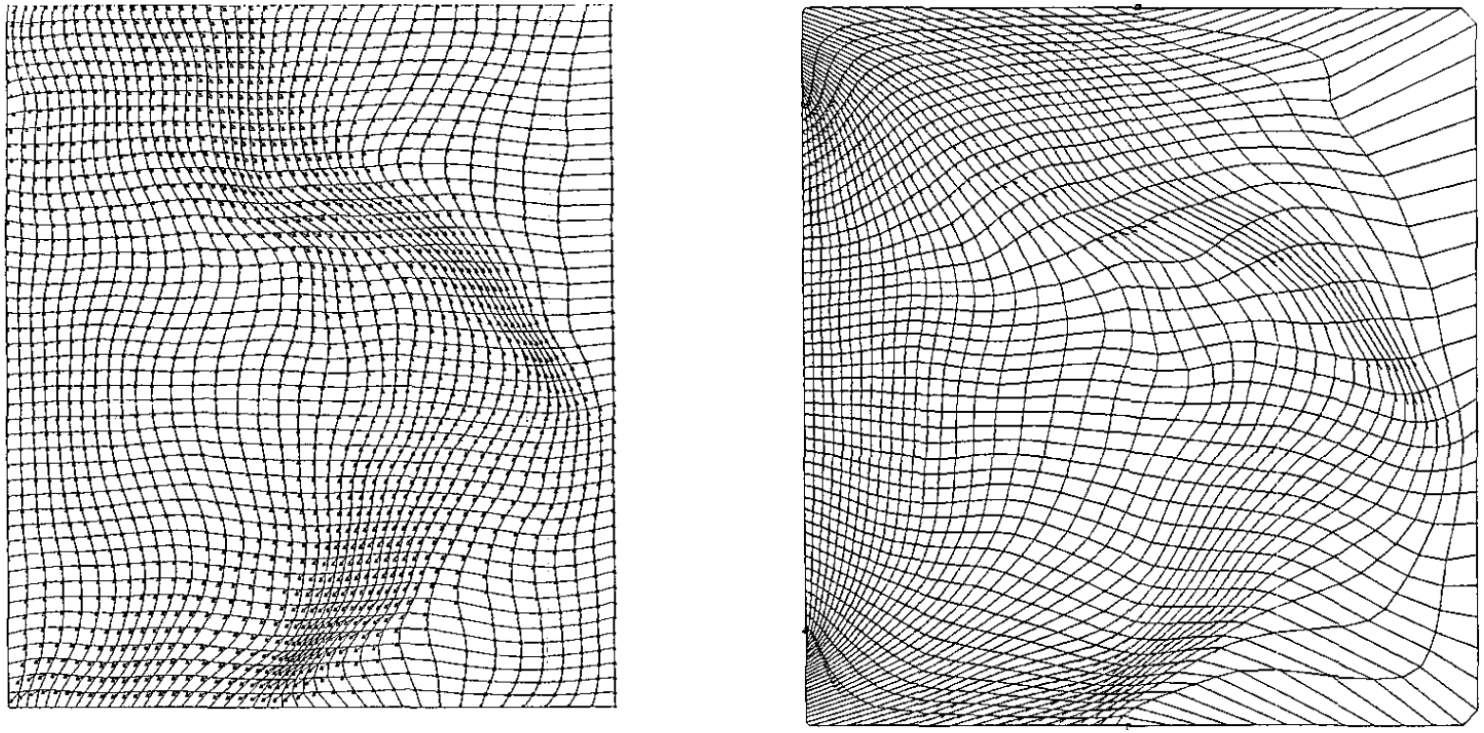
$$\Delta u = 0$$
$$u\big|_{\partial\Omega} = v$$

Outline

- Geometry processing
  - **Mesh Generation**
  - Deformation
  - Texture mapping and synthesis
- Misc
  - Non-photorealistic rendering
  - Crowd simulation

2D Mesh Generation

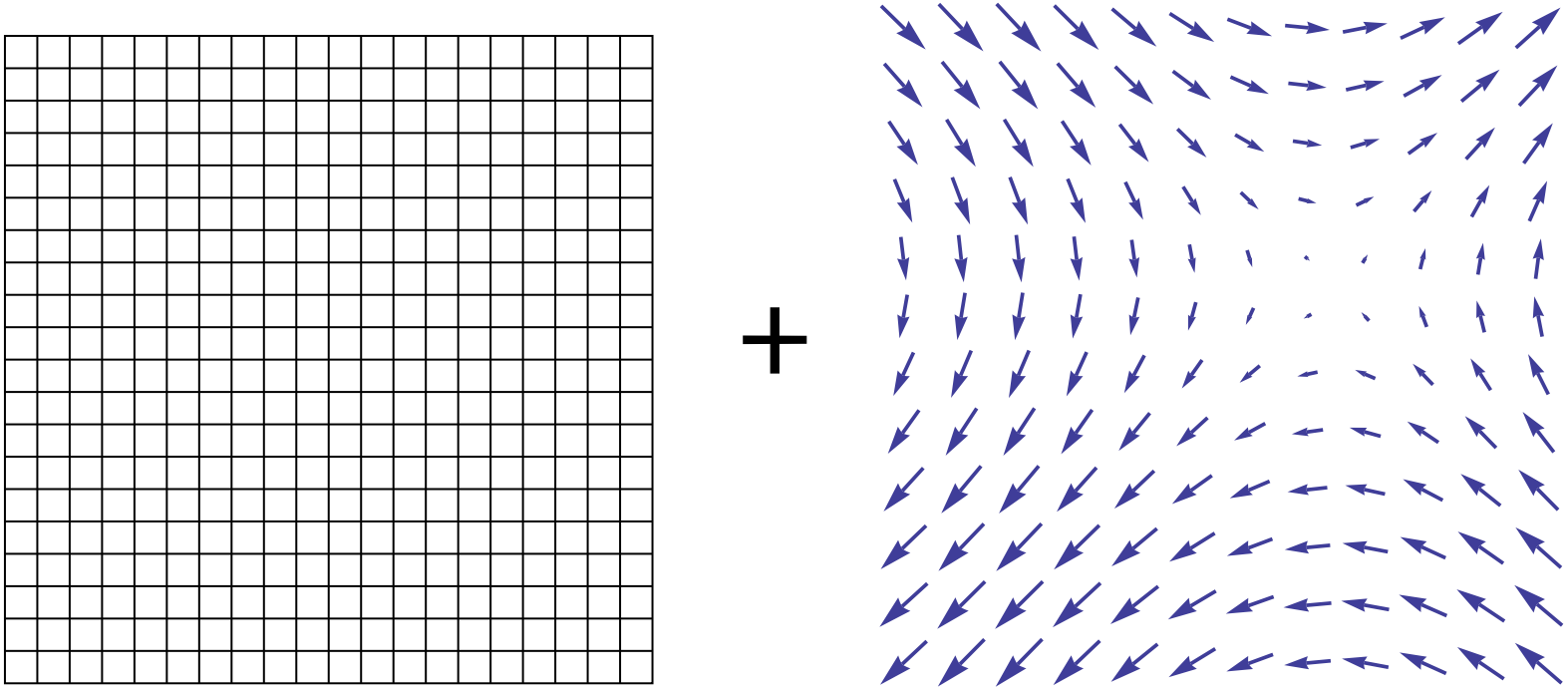
- Input: mesh topology + vector field (VF)
- Task: Align the mesh with the VF



‘Mesh Generation Using Vector Fields’ by P.Knupp, 1994

2D Mesh Generation

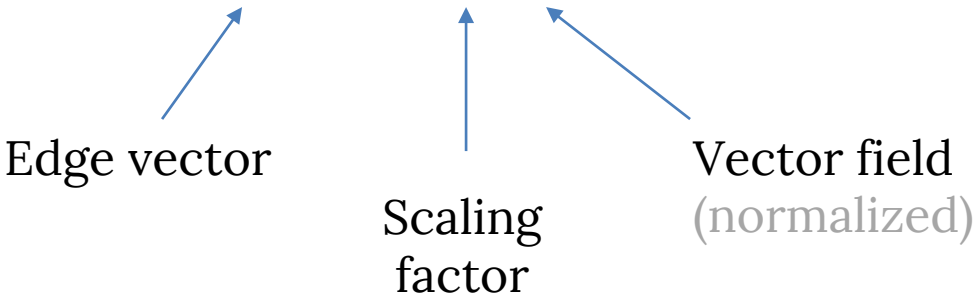
- User chooses which edge should align to VF



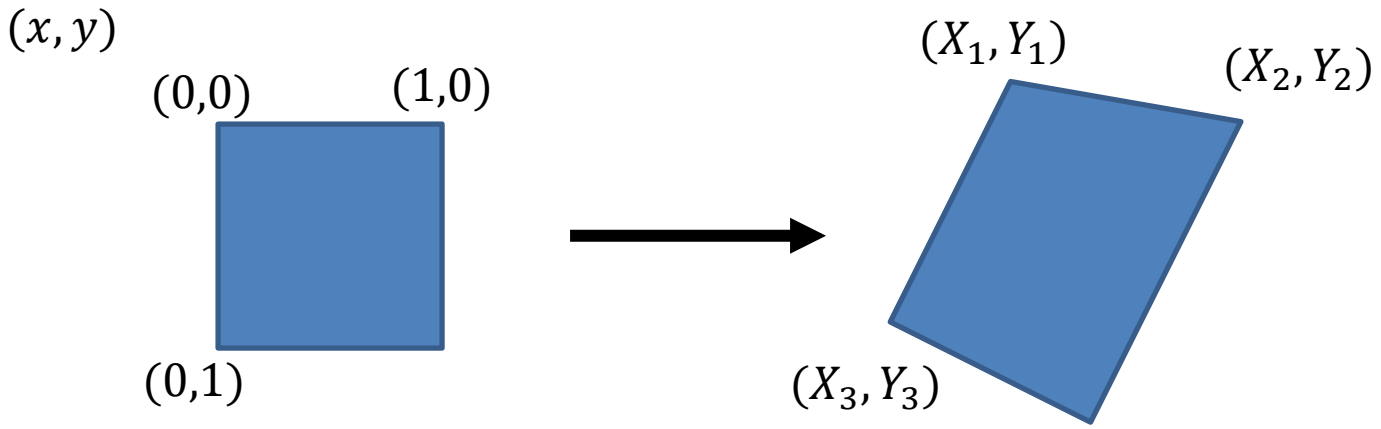
- How to formulate alignment?

Alignment

$$X_e = l_e u_i$$



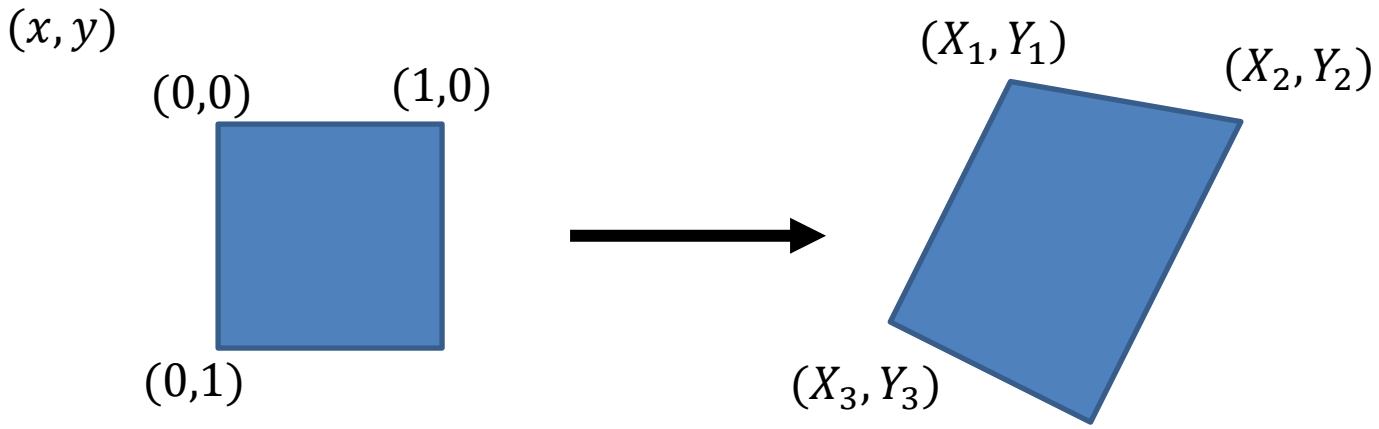
Equivalent formulation



Jacobian: $J = \begin{pmatrix} \frac{\partial X}{\partial x} & \frac{\partial X}{\partial y} \\ \frac{\partial Y}{\partial x} & \frac{\partial Y}{\partial x} \end{pmatrix}$

Discretized: $J = \begin{pmatrix} X_2 - X_1 & X_3 - X_1 \\ Y_2 - Y_1 & Y_3 - Y_1 \end{pmatrix}$

Equivalent formulation



$$J = \begin{pmatrix} X_2 - X_1 & X_3 - X_1 \\ Y_2 - Y_1 & Y_3 - Y_1 \end{pmatrix}$$

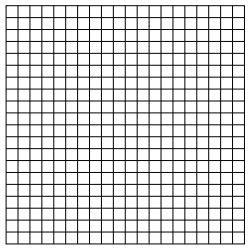
$$J = U \cdot \begin{pmatrix} l_1 & \\ & l_2 \end{pmatrix} = T_U$$

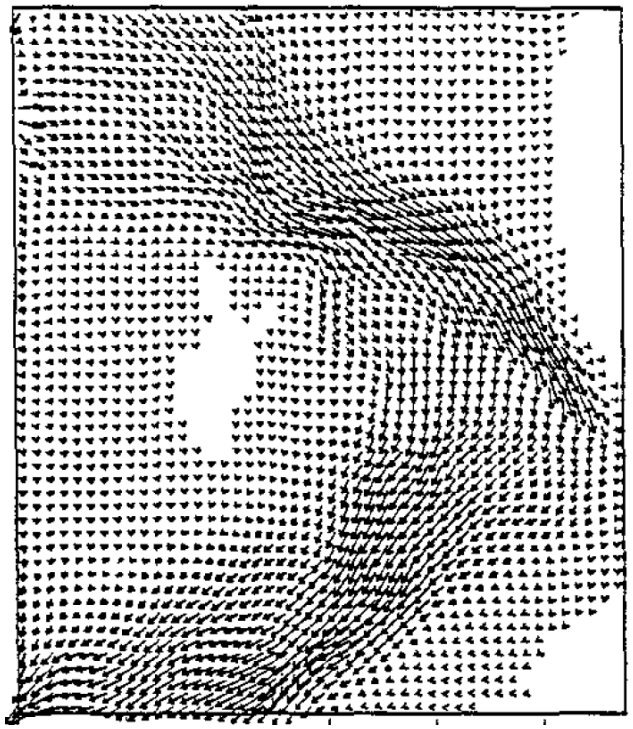
Final statement

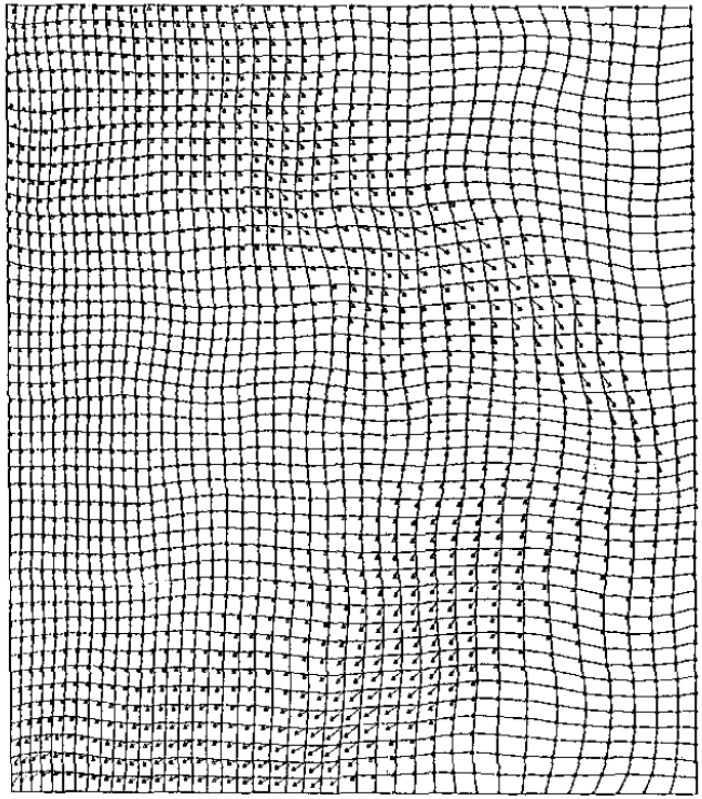
- Constrain inverses instead

$$min \int \det(J^{-1} - T^{-1})^2 dxdy$$

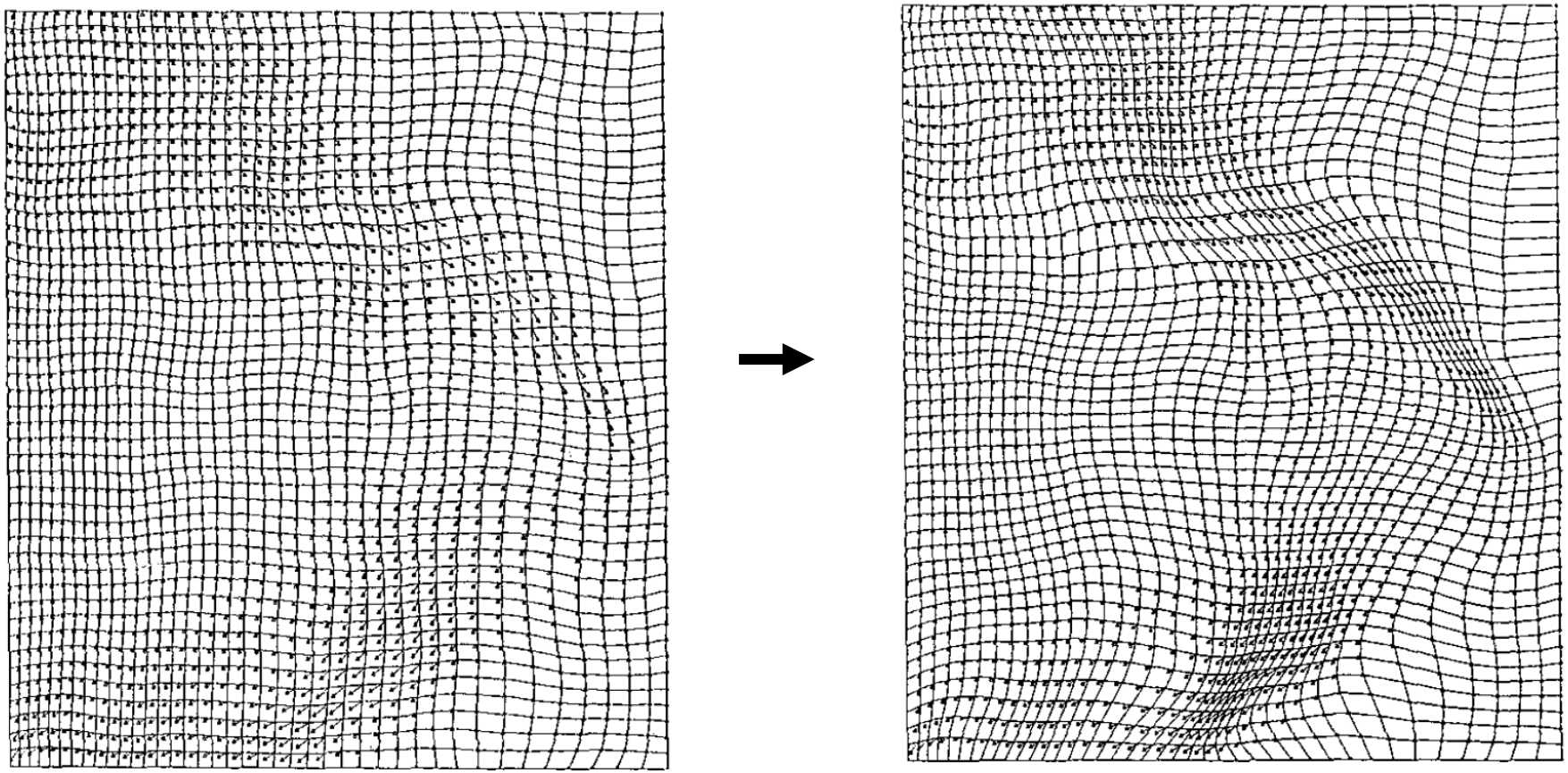
Issues?







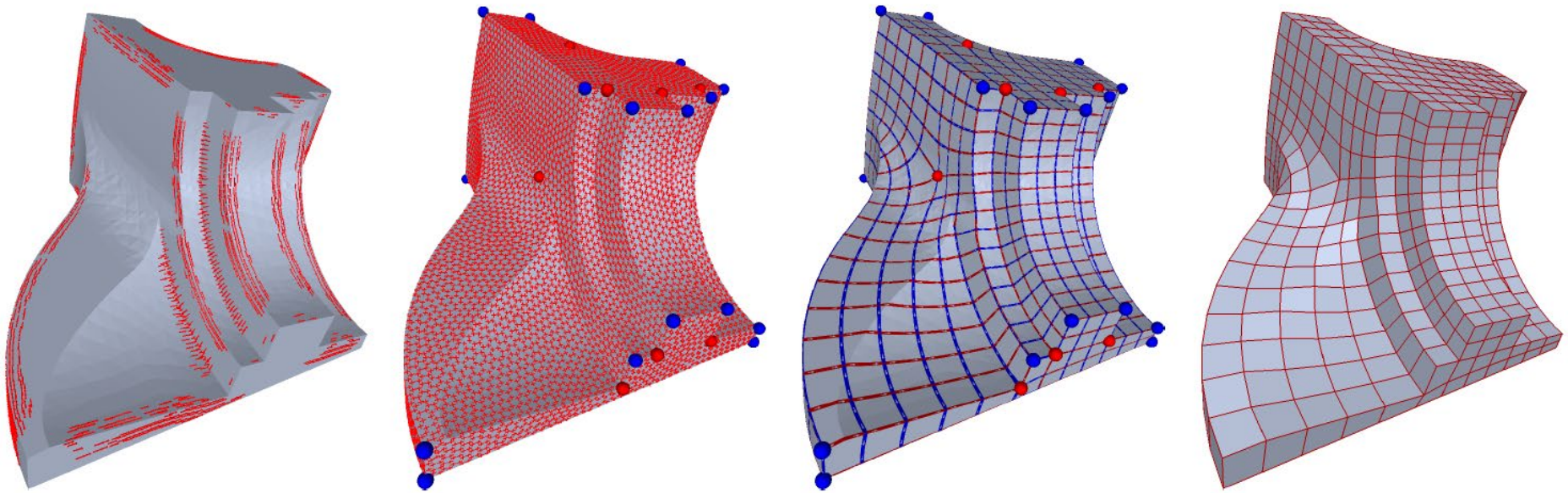
Adding non-uniform sizing



Mesh Quadrangulation

Input: Triangle mesh + sparse directions

Output: Quad mesh aligned with the directions



‘Mixed-Integer Quadrangulation’ by Bommes et al., 2009

Mesh Quadrangulation

1. Compute two vector fields
2. Align a quad mesh with them

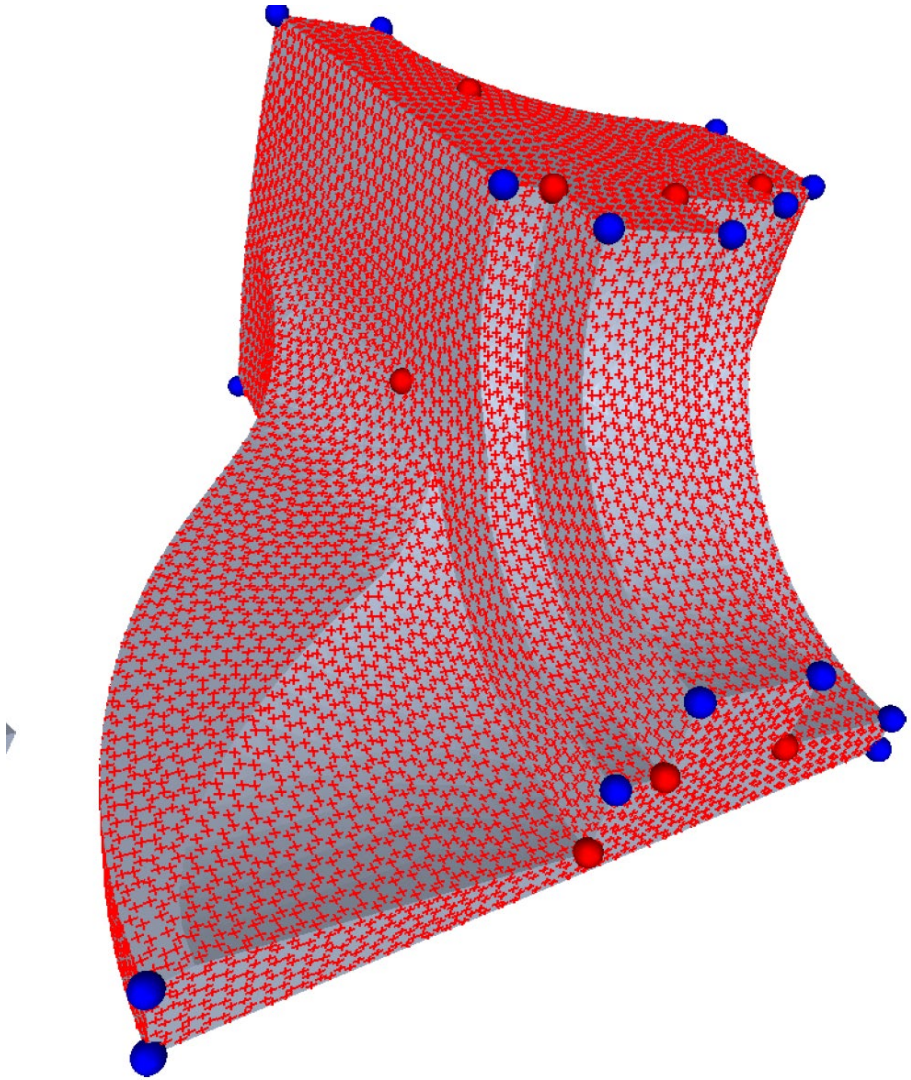
Mesh Quadrangulation

1. Compute a cross field
2. For all points on a surface, compute (u, v)

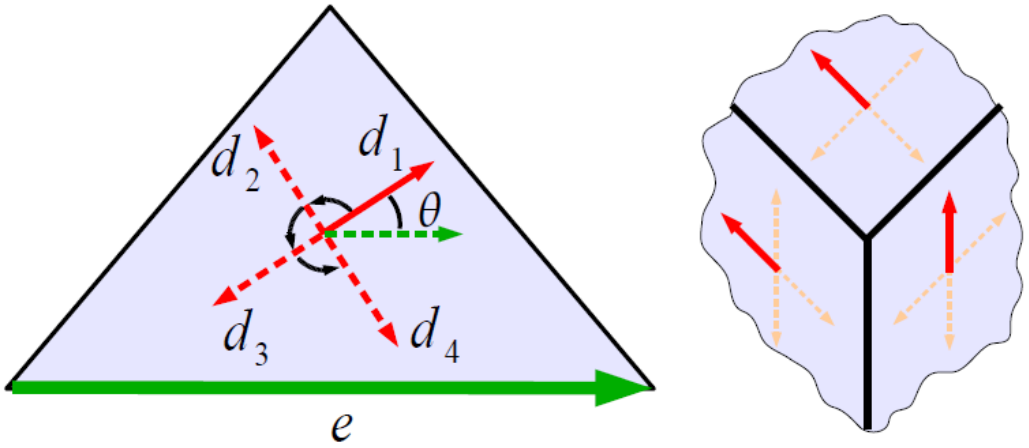
Parameterization!
More on that later

Cross Fields

- 4 coupled vectors =
  2 directions



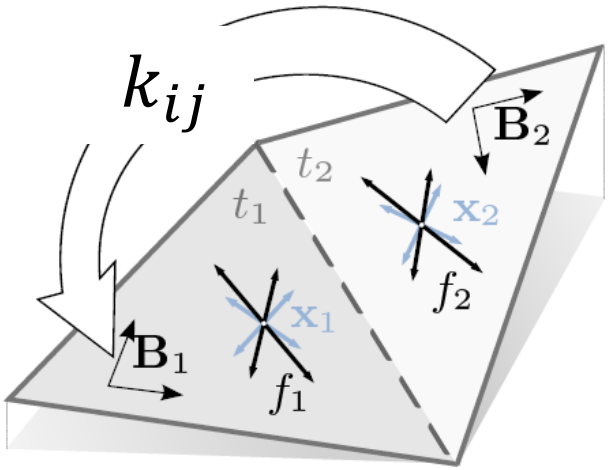
Representation and singularities



Smoothness?

- Associate tangent spaces
- Add period jumps

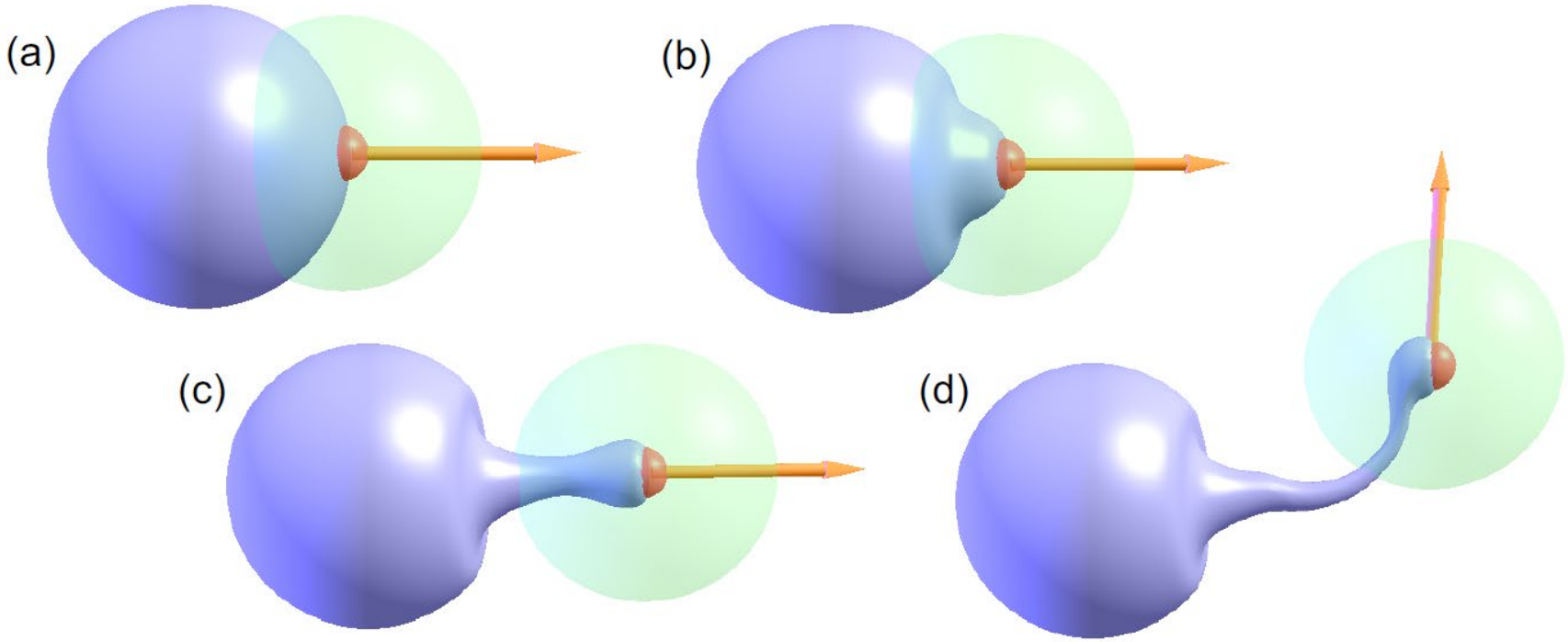
$$E_{smooth} = \sum_{e_{ij} \in E} (\underbrace{\theta_i + \kappa_{ij} + \frac{\pi}{2} p_{ij}}_{\theta_i \text{ w.r.t. frame } j} - \theta_j)^2 \quad p_{ij} \in \mathbb{Z}$$



Outline

- Geometry processing
  - Mesh Generation
  - **Deformation**
  - Texture mapping and synthesis
- Misc
  - Non-photorealistic rendering
  - Crowd simulation

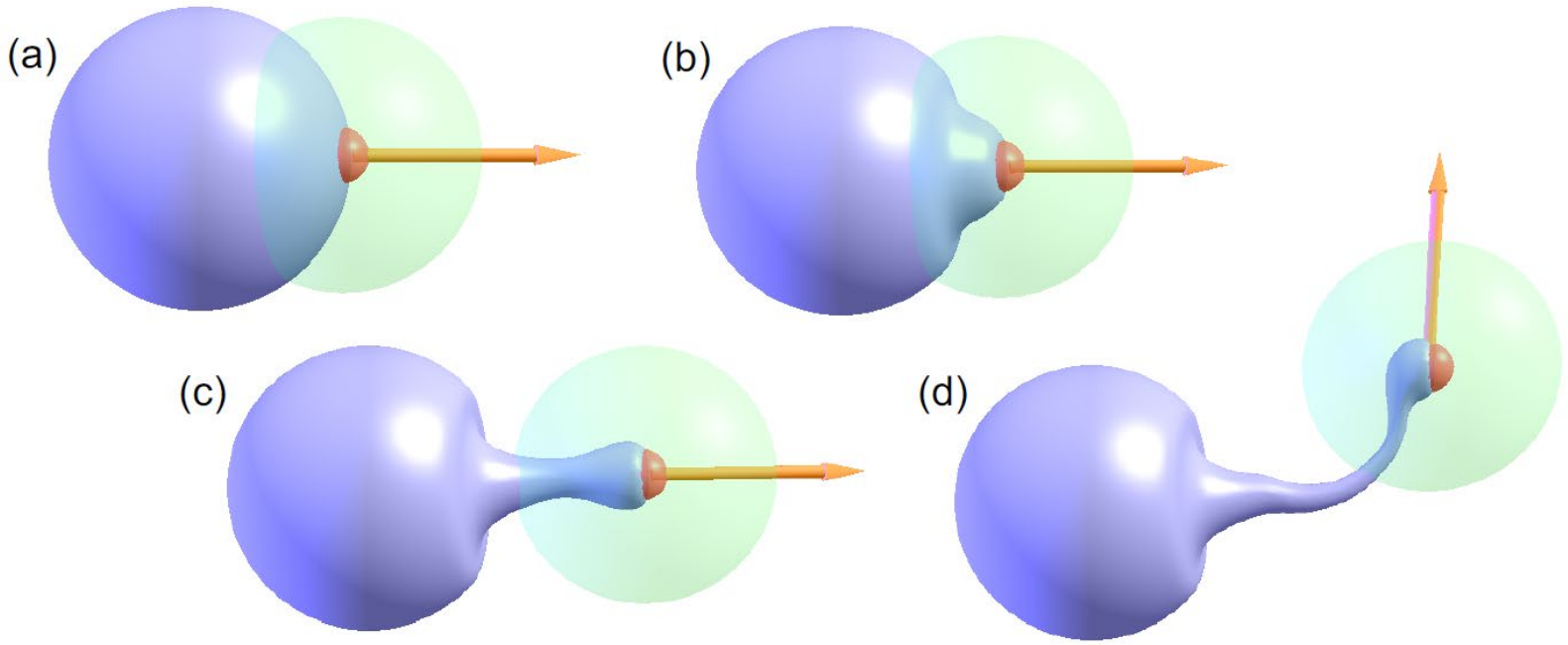
Mesh Deformation



Mesh Deformation

Find a *divergence-free* vector field

$$div\ v = 0$$



'Vector Field Based Shape Deformations' by Funck et al., 2006

Before:

Helmholtz-Hodge Decomposition

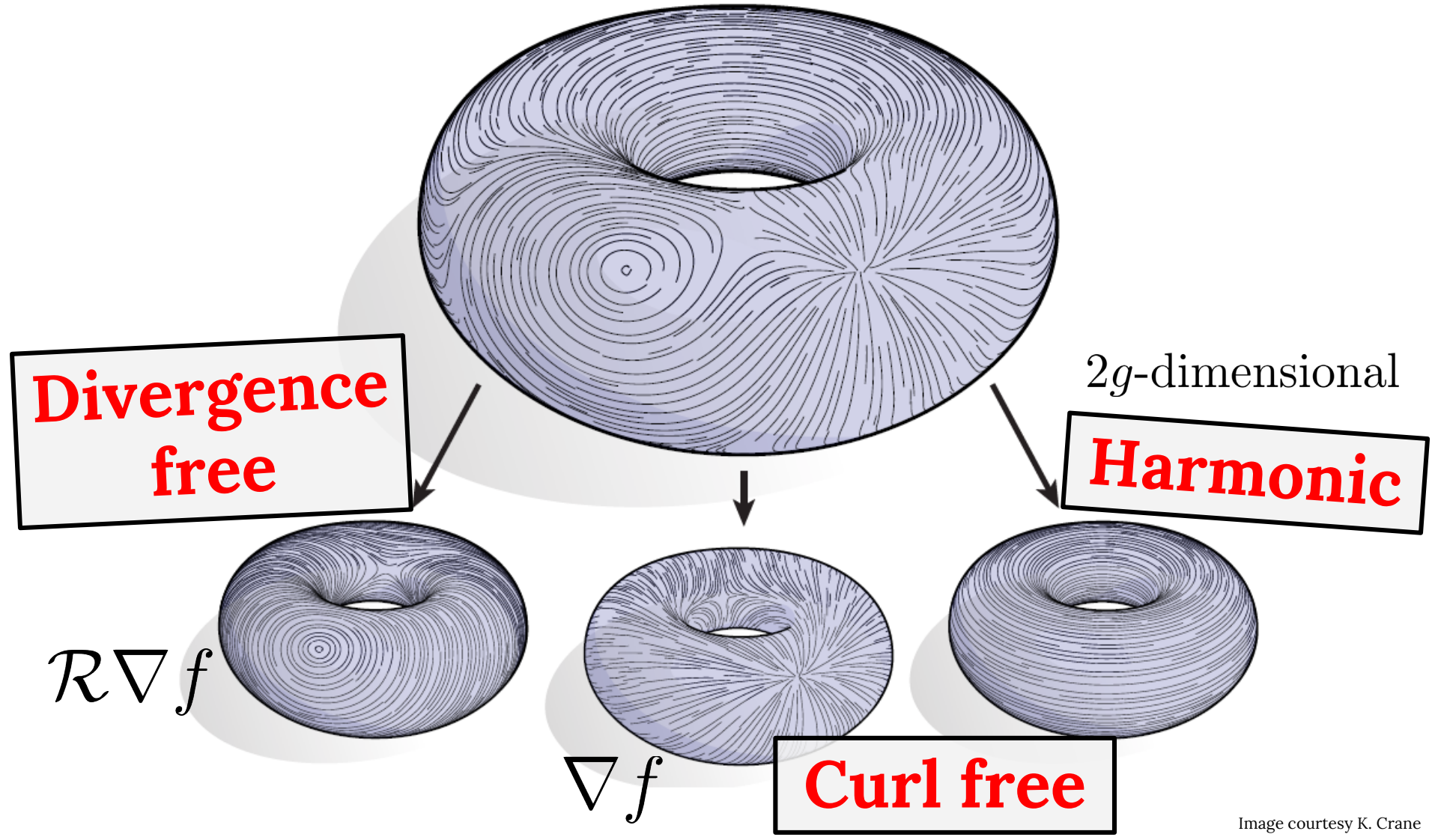
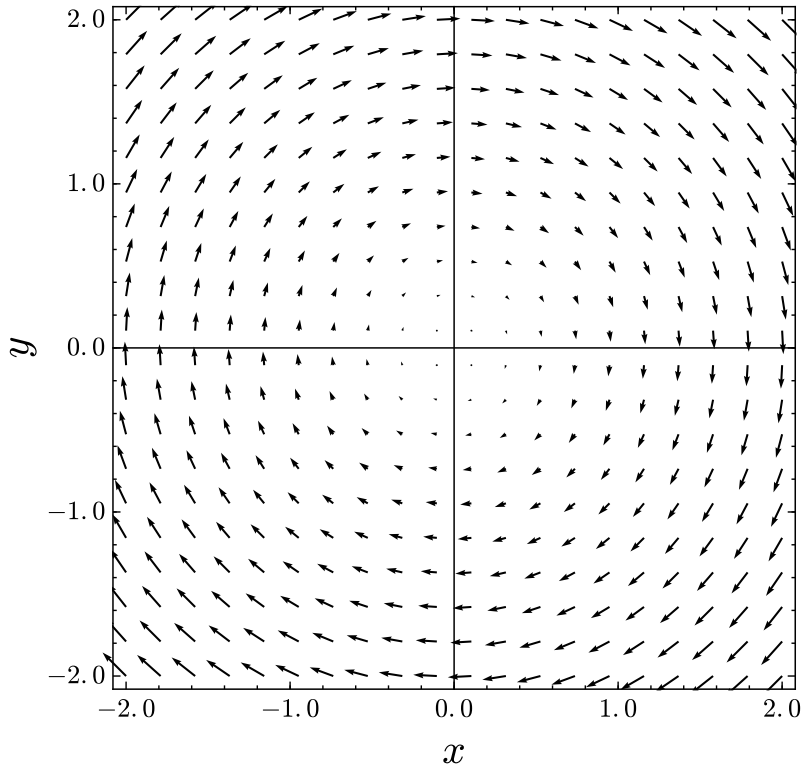


Image courtesy K. Crane

Divergence-Free VF

$$div\ v = \nabla \cdot v = \frac{\partial v}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial v}{\partial z} = 0$$



Divergence-free => No stretch/squash!

Tangent Vector Fields

- Rotated gradient fields have zero divergence

$$div\ R\nabla u = 0$$

(proof for 2D case on the board)

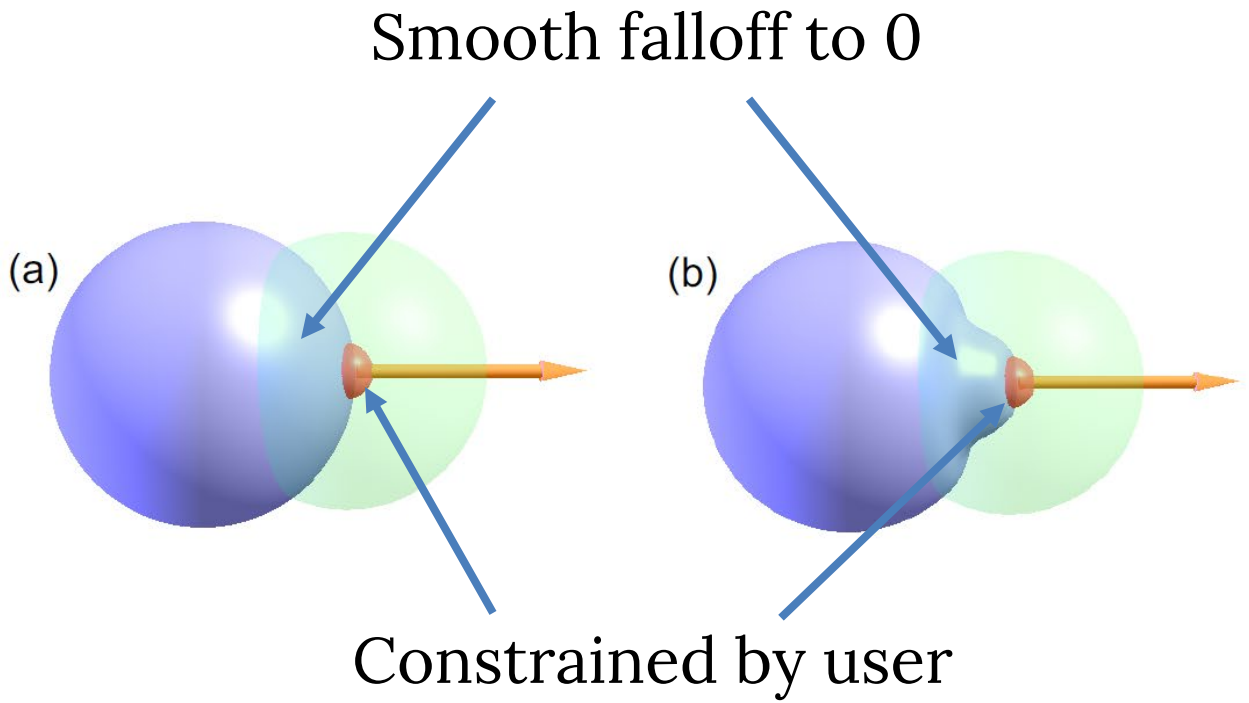
Normal Vector Fields

Cross product of two gradients has zero divergence

$$\mathbf{v}(x,y,z) = \nabla p(x,y,z) \times \nabla q(x,y,z)$$

$$div\ v = 0$$

Mesh Deformation



Can specify twist!

Outline

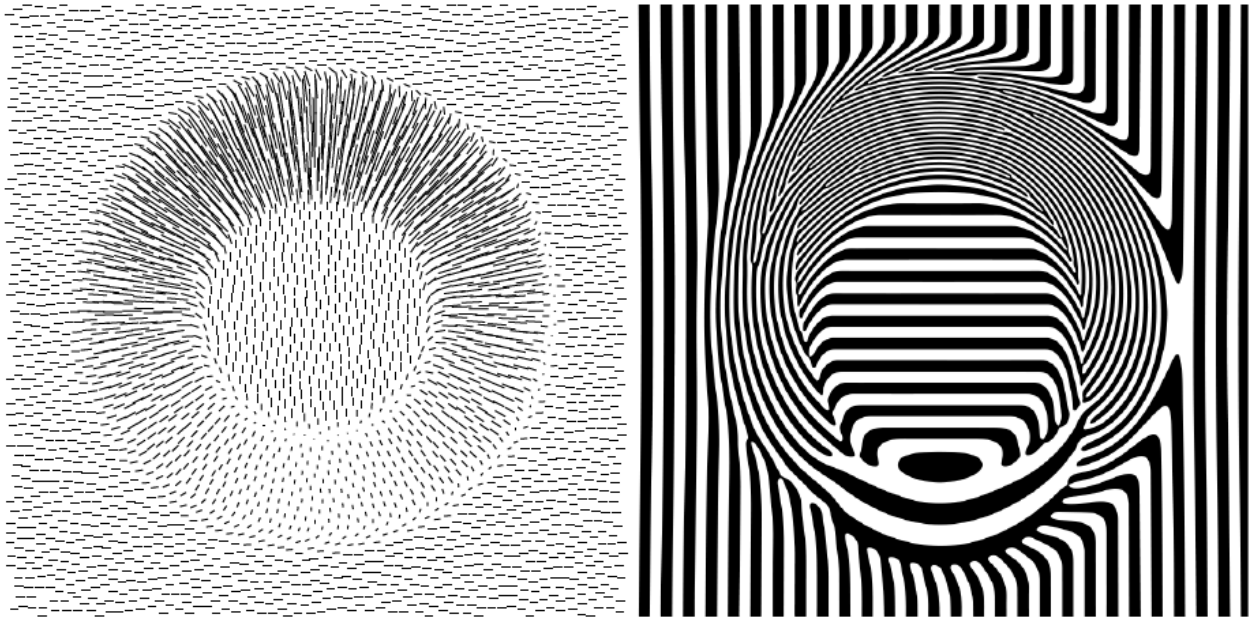
- Geometry processing
  - Mesh Generation
  - Deformation
  - **Texture mapping and synthesis**
- Misc
  - Non-photorealistic rendering
  - Crowd simulation

Texture Synthesis

‘Stripe Patterns on Surfaces’ by F. Knoppel et al., 2015

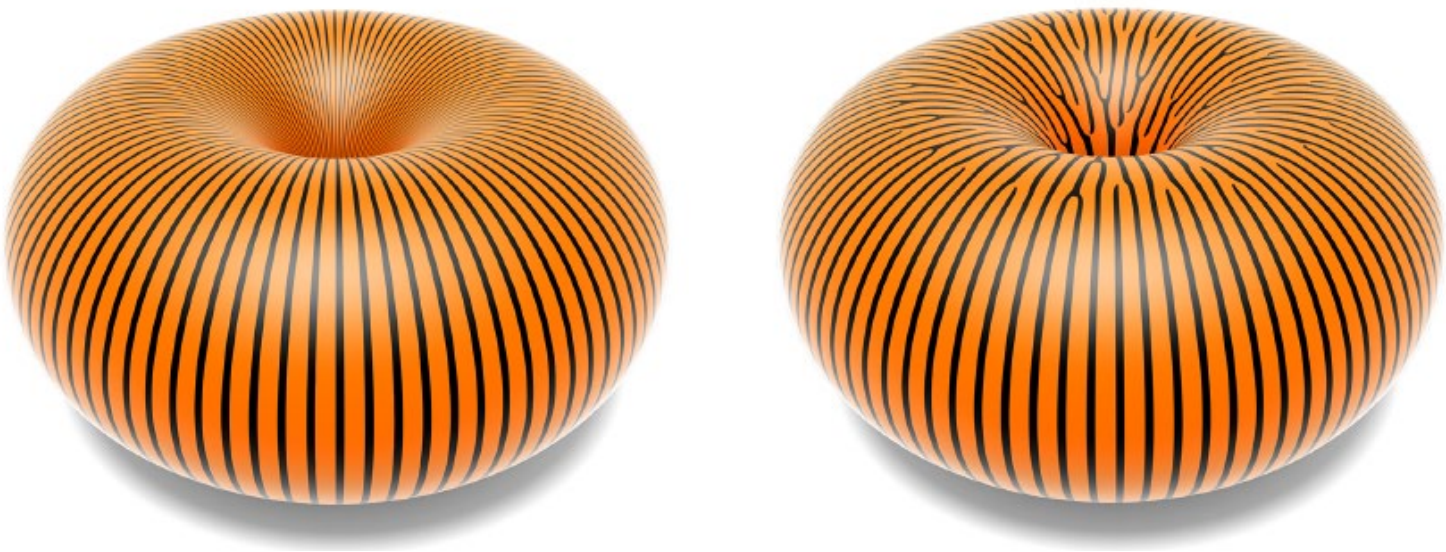
Idea

- Input: mesh + vector field (+scale)
- Output: scalar field controlling periodic texture
  - Imagine periodic texture as $f = \sin(\boldsymbol{\alpha})$



Idea

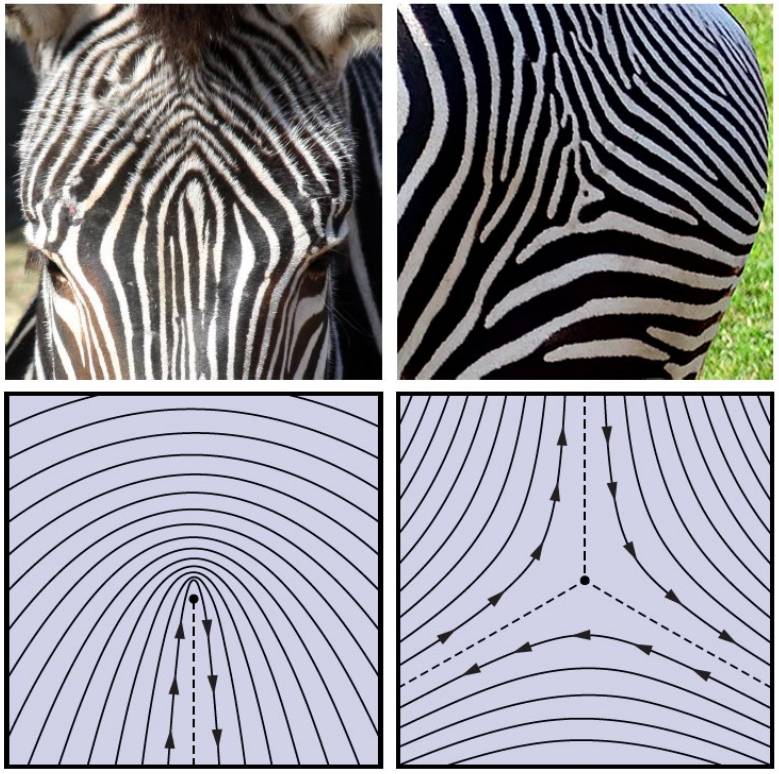
Singularities => more even spacing



Idea

Singularities => more even spacing

Also occurring in nature



Useless fact

"It was previously believed that zebras were white animals with black stripes, since some zebras have white underbellies. Embryological evidence, however, shows that the animal's background colour is black and the white stripes and bellies are additions."

Prothero D.R.; Schoch R. M. (2003). *Horns, Tusks, and Flippers: The Evolution of Hoofed Mammals*.

Familiar components!

- Representation
- Connection
- Singularities
- Dirichlet Energy

… but also some other notions
beyond the scope of this course

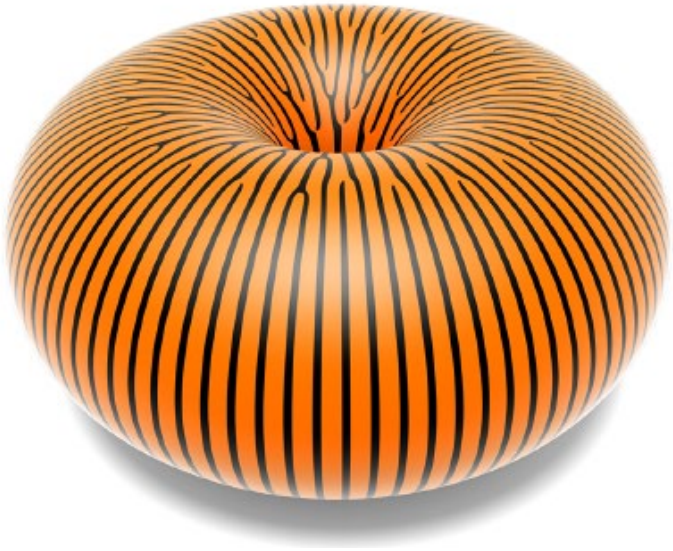
Familiar components!

- **Representation**
- Connection
- Singularities
- Dirichlet Energy

… but also some other notions
beyond the scope of this course

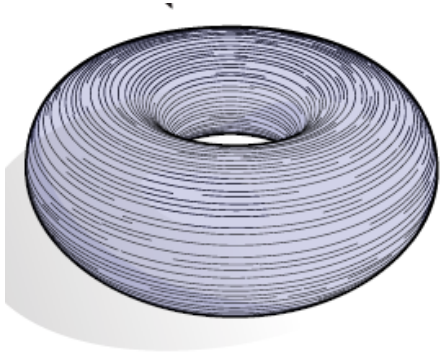
How to optimize for α

$\nabla\alpha$ should be perpendicular
to the vector field?

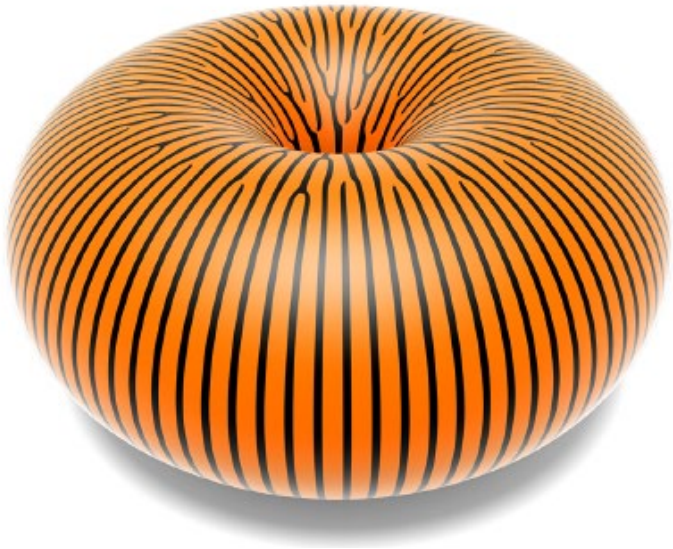


How to optimize for α

$\nabla\alpha$ should be perpendicular to the vector field?

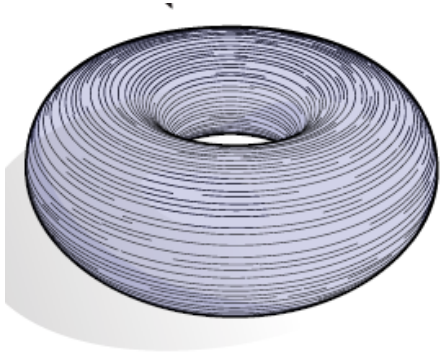


Can we integrate that?

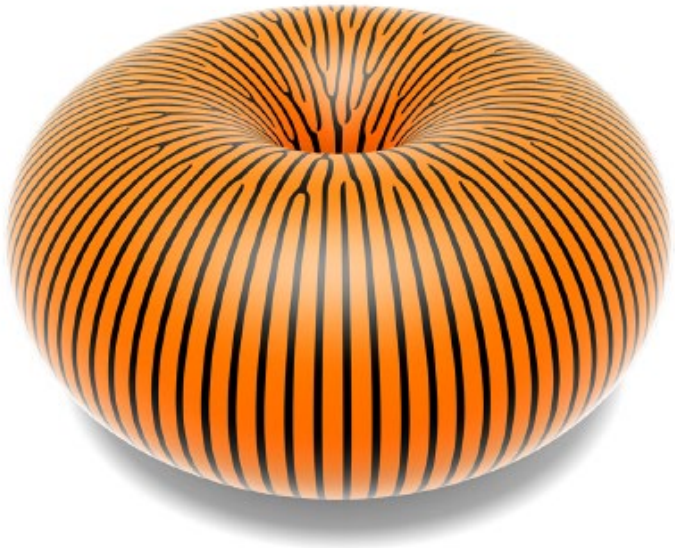


How to optimize for α

Look for $\psi = e^{i\alpha}$ instead: it can be smooth



Can we integrate that?

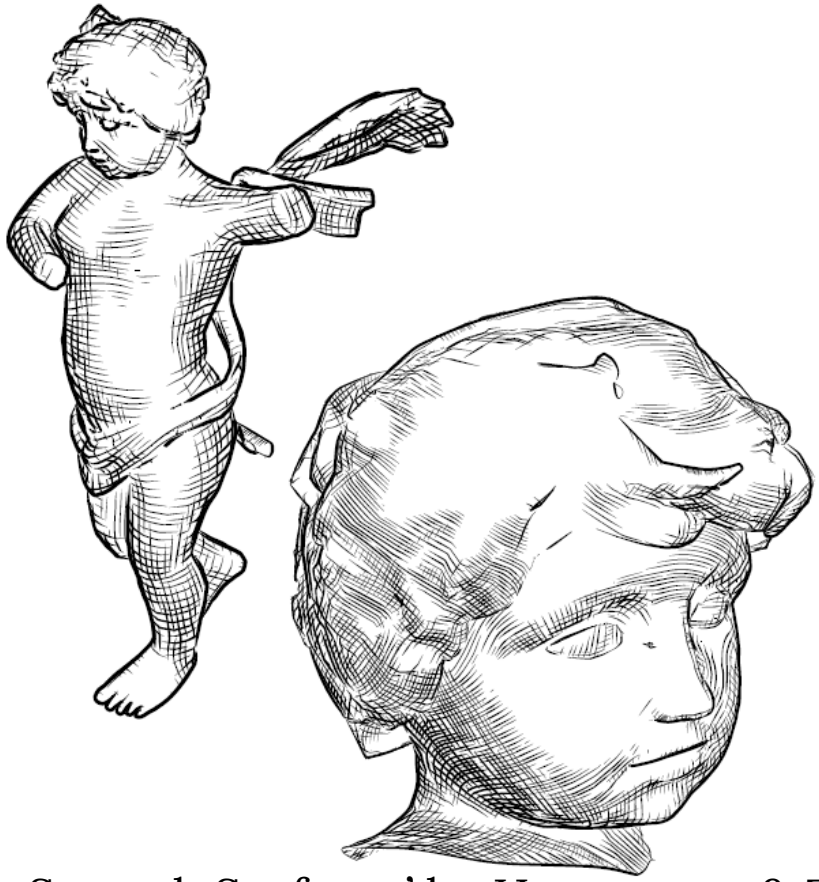


Outline

- Geometry processing
  - Mesh Generation
  - Deformation
  - Texture mapping and synthesis
- Misc
  - **Non-photorealistic rendering**
  - Crowd simulation

Non-photorealistic rendering

- Input: mesh
- Output:

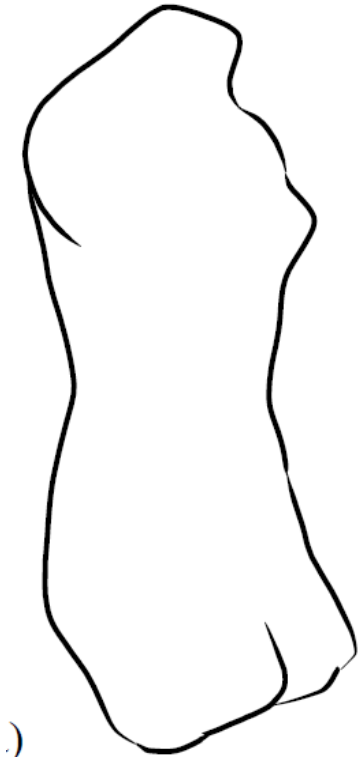


‘Illustrating Smooth Surfaces’ by Hertzmann & Zorin, 2001

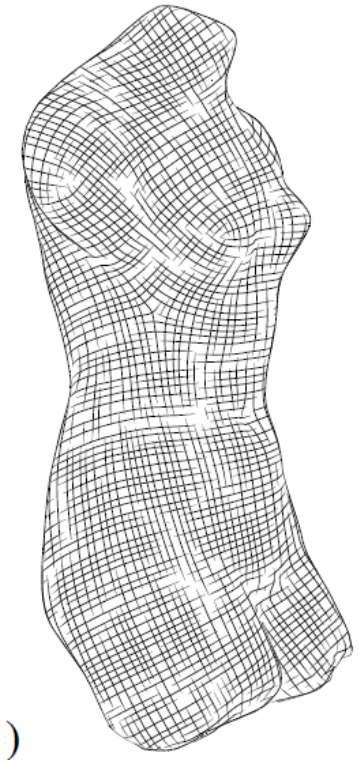
Components

Silhouettes

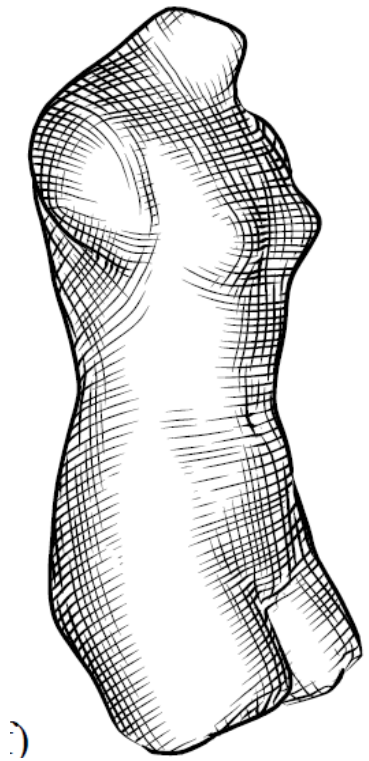
‘hatching’



+

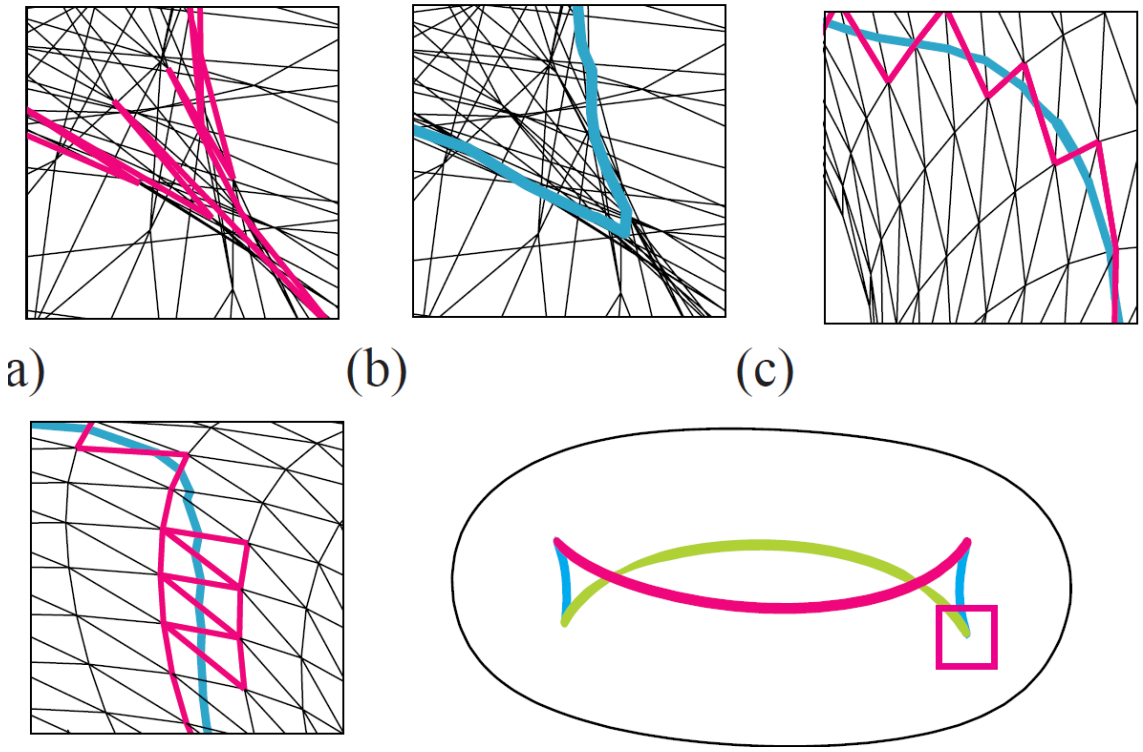


=



Silhouettes

- Mesh silhouettes are unreliable



Silhouettes

Better idea:

Silhouettes = zeros of a scalar field

Point p is on silhouette $\Leftrightarrow$

$$n \cdot (c - p) = 0$$

Camera position

Silhouettes

Better idea:

Silhouettes = zeros of a scalar field

Point p is on silhouette $\Leftrightarrow$

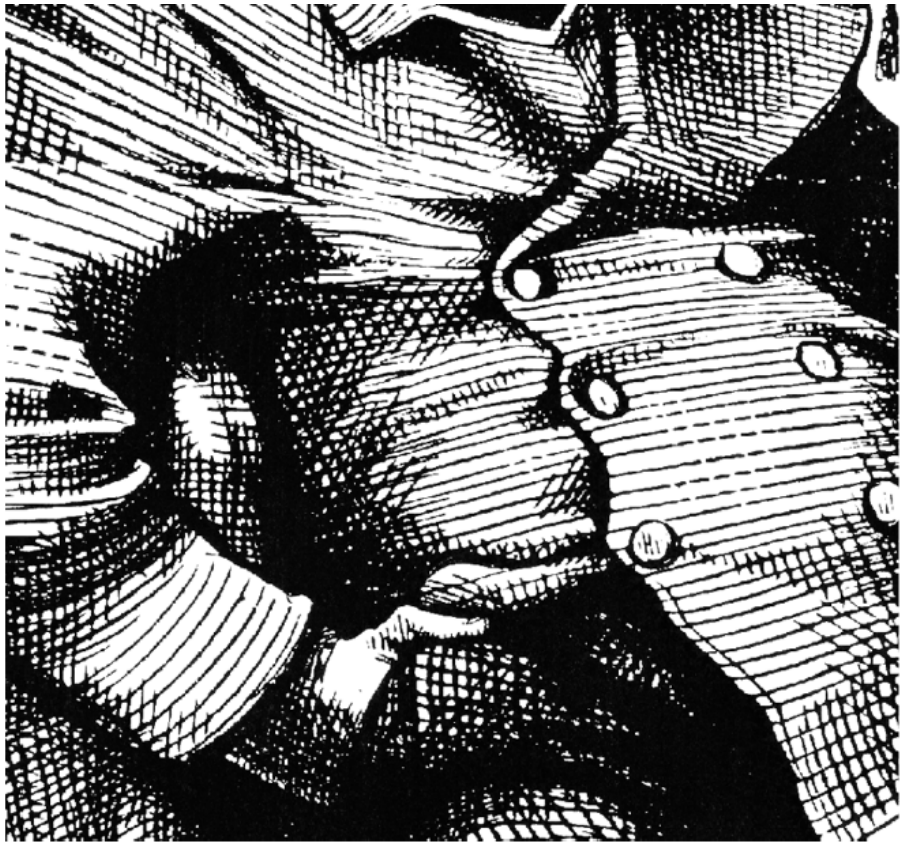
$$n \cdot (c - p) = 0$$

Compute at every vertex, interpolate, find zero-crossings

Camera position

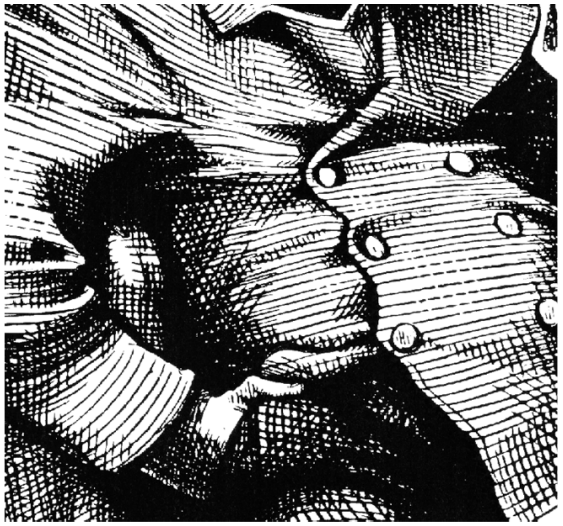
Hatching

- Principle curvature directions!



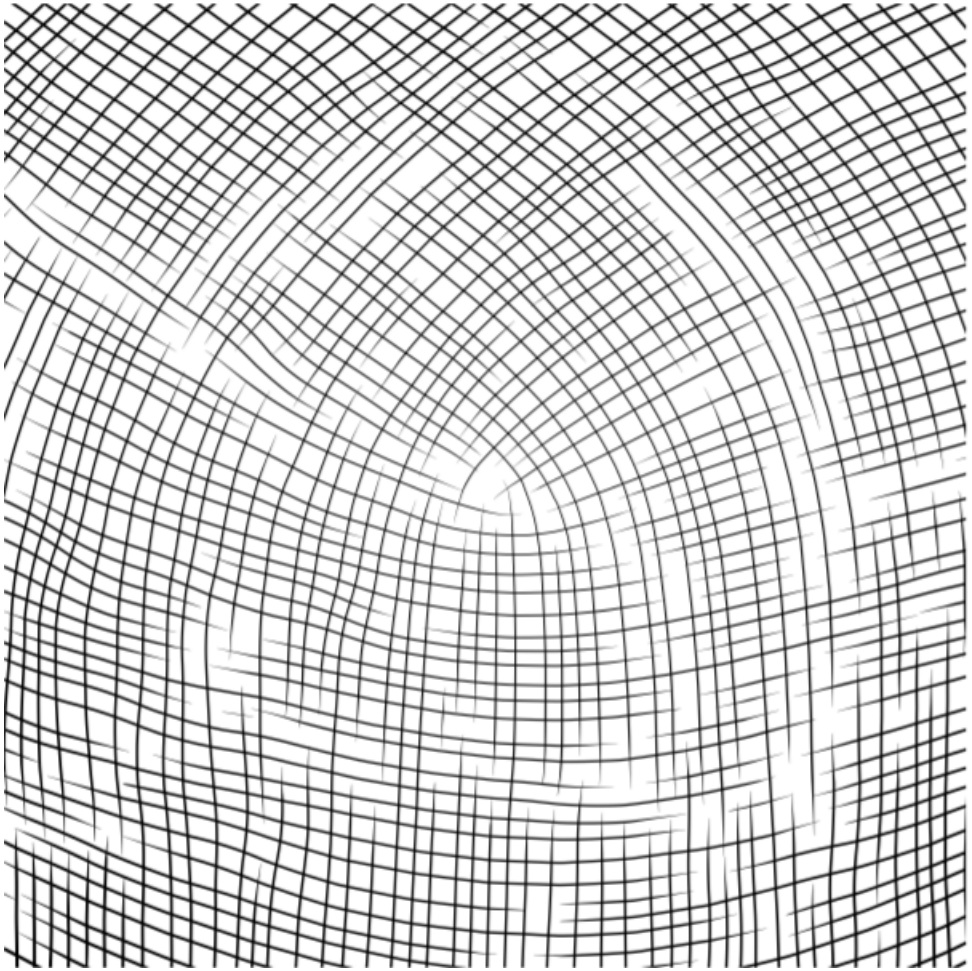
Hatching

- Principle curvature directions!



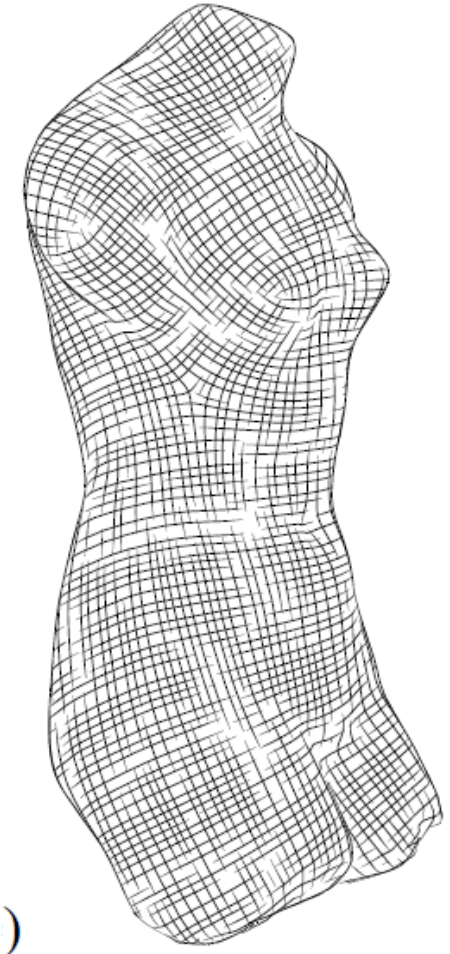
- Except those are not defined for *umbilics* (equal principal curvatures)
- At umbilics, draw geodesics!

2 directional fields?



Hatching

- Find parabolic areas of the mesh
- Constrain cross field to align with principle curvatures
- The rest should be smooth
  - Smoothness term uses a connection



Result