Signal Processing
From Images to Surfaces

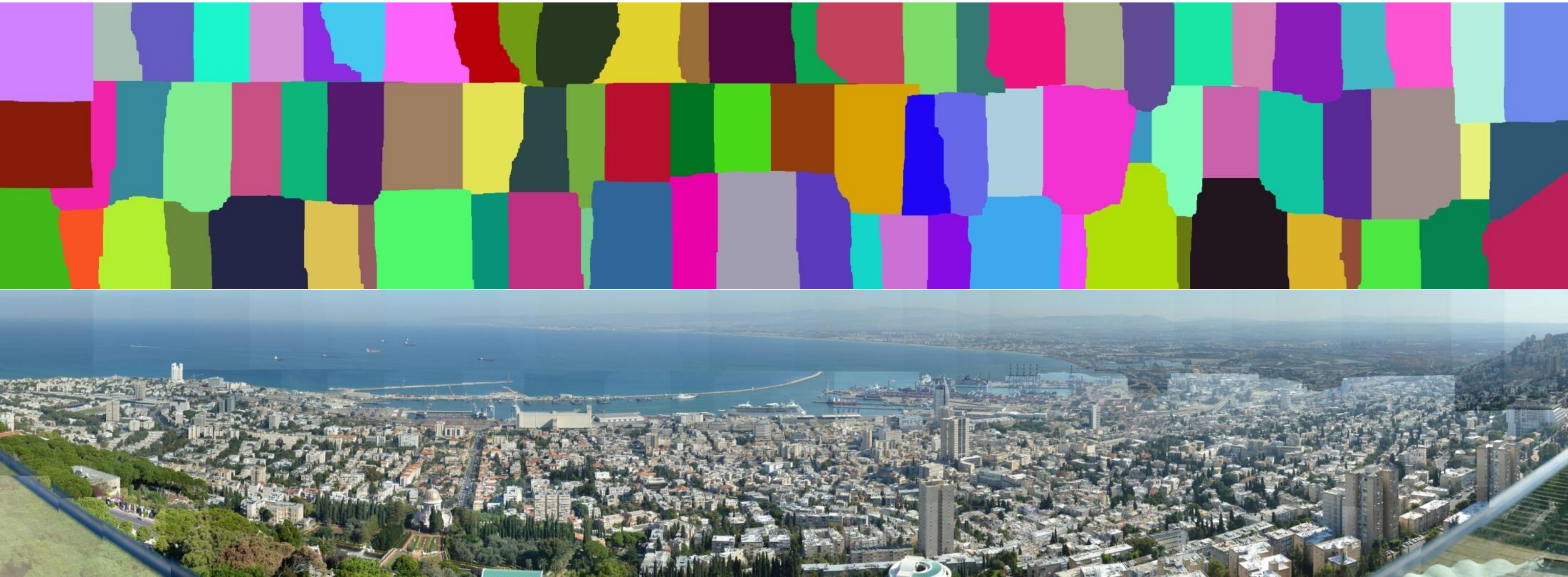
Ming Chuang , Fabian Prada, Alvaro Collet, Linjie Luo,
Benedict Brown, Szymon Rusinkiewicz, Hugues Hoppe

Estimating the Laplace-Beltrami Operator by Restricting 3D Functions. [Chuang et al., 2009]
Fast Mean-Curvature Flow via Finite Elements Tracking. [Chuang et al., 2011]
Interactive and Anisotropic Geometry Processing Using the Screened Poisson Equation. [Chuang et al., 2011]
Unconditionally Stable Shock Filters for Image and Geometry Processing. [Prada et al., 2015]
Motion Graphs for Unstructured Textured Meshes. [Prada et al., 2016]

Goal

Extend image-processing techniques to surfaces:

1. Gradient Domain



Goal

Extend image-processing techniques to surfaces:

1. Gradient Domain

Goal

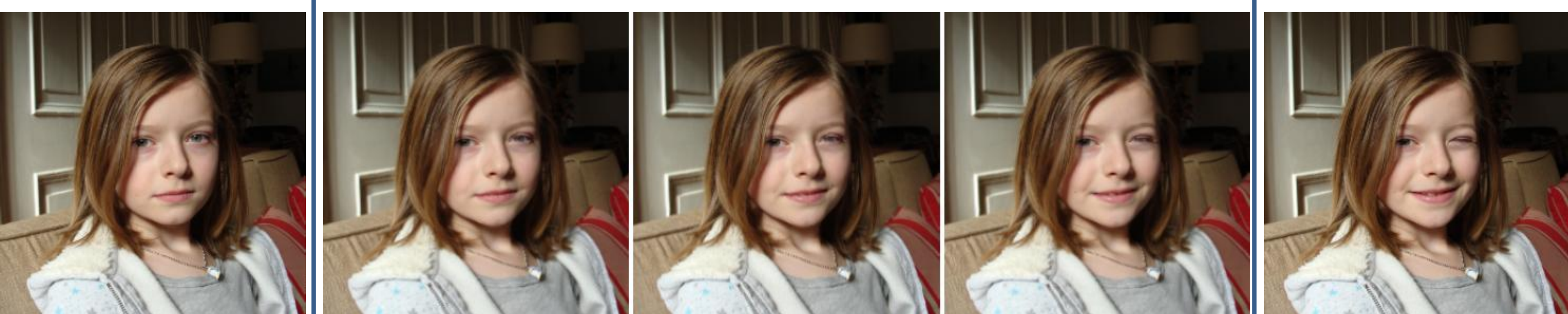
Extend image-processing techniques to surfaces:

1. Gradient Domain
2. Shock Filters

Goal

Extend image-processing techniques to surfaces:

1. Gradient Domain
2. Shock Filters
3. Optical Flow



Source — Optical Flow Interpolation — Target

Moving Gradients: A Path Based Method for Plausible Image Interpolation. [Mahajan et al., 2009]

Outline

- Motivation
- **Processing Tools**
  - **Screened Poisson Equation**
  - **Flow Fields/Lines**
- Extensions to Signals on Surfaces
- Conclusion

Image-Processing Tools

1. Screened Poisson Equation:

Given a 2D domain Ω, a function g, and a vector field $\vec{v}$, *solve for the function* f *minimizing*:

$$E(f) = \int_\Omega \alpha \underbrace{\|f - g\|^2}_{\text{value-fitting}} + \underbrace{\|\nabla f - \vec{v}\|^2}_{\text{gradient-fitting}}$$

Image-Processing Tools

1. Screened Poisson Equation:

Given a 2D domain Ω, a function g, and a vector field $\vec{v}$, *solve for the function* f *minimizing*:

$$E(f) = \int_\Omega \alpha \|f - g\|^2 + \|\nabla f - \vec{v}\|^2$$

$$\Downarrow$$

$$(\alpha \cdot \mathbf{1} - \boldsymbol{\Delta}) f = \alpha \cdot \mathbf{1} \cdot g - \mathbf{div}(\vec{v})$$

Image-Processing Tools

2a. Flow Fields/Lines:

Given a 2D domain Ω and a vector field $\vec{v}$, a *flow-line* of $\vec{v}$ is a curve γ_p such that:

$$\gamma_p(0) = p \quad \text{and} \quad \gamma_p'(t) = \vec{v}\left(\gamma_p(t)\right).$$

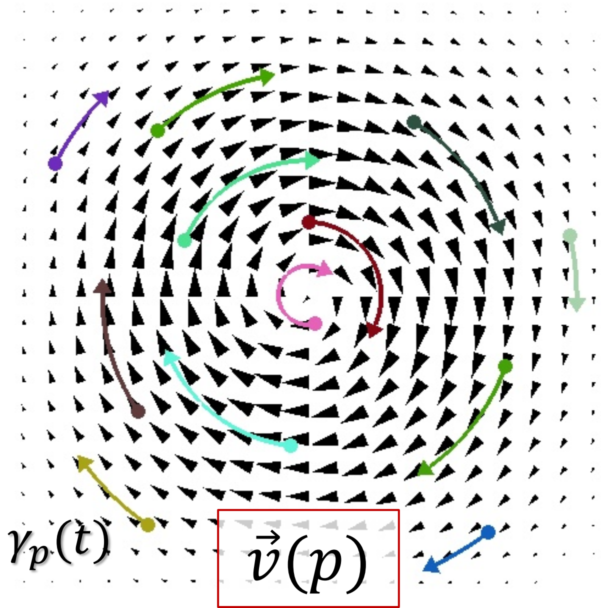
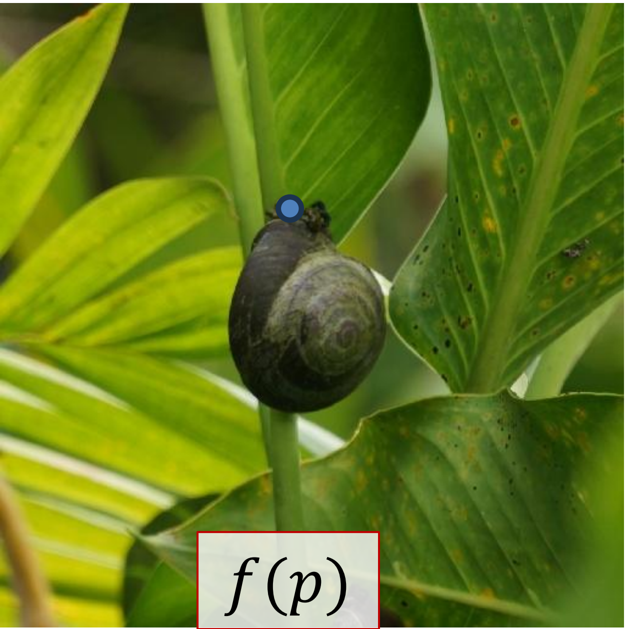


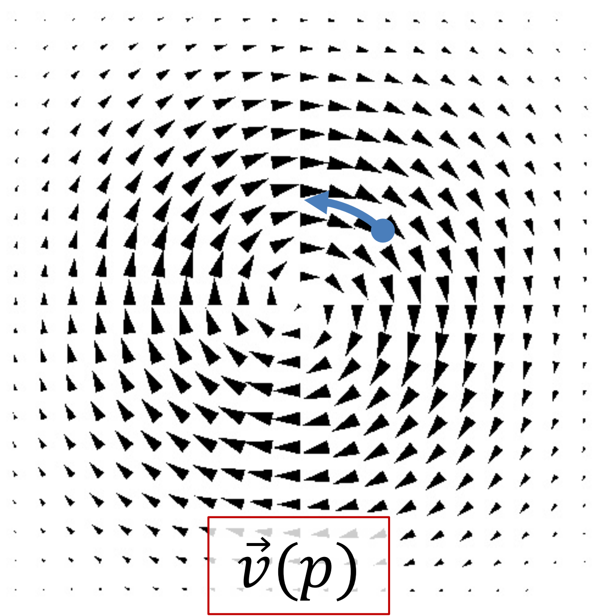
Image-Processing Tools

2b. Flow Fields/Lines:

Given a 2D domain Ω and a vector field $\vec{v}$, the *advection* of a function f along $\vec{v}$ is the function:

$$[\text{Adv}_{\vec{v}}(f)](p) = f\left(\gamma_p(-1)\right).$$





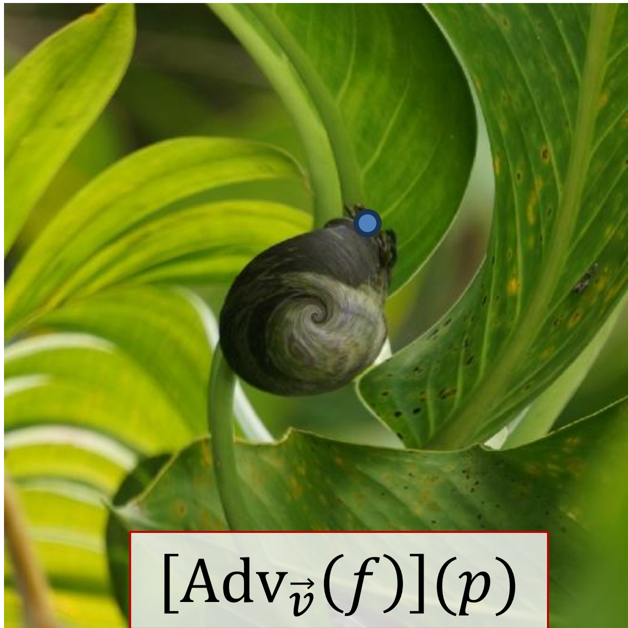
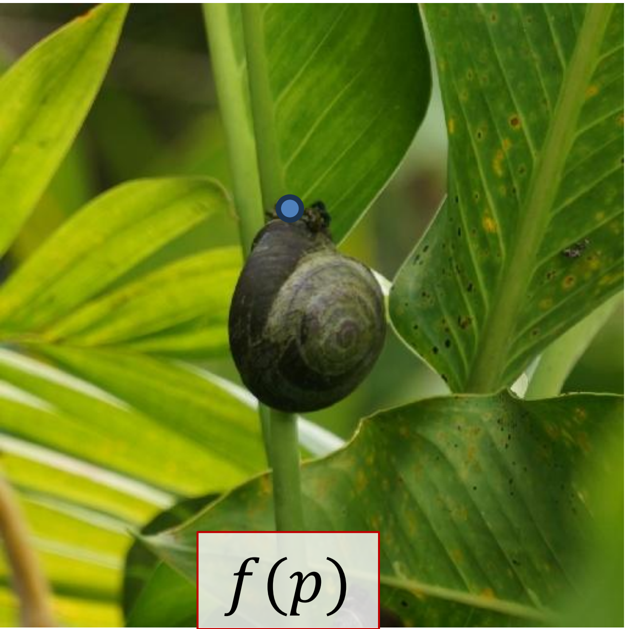


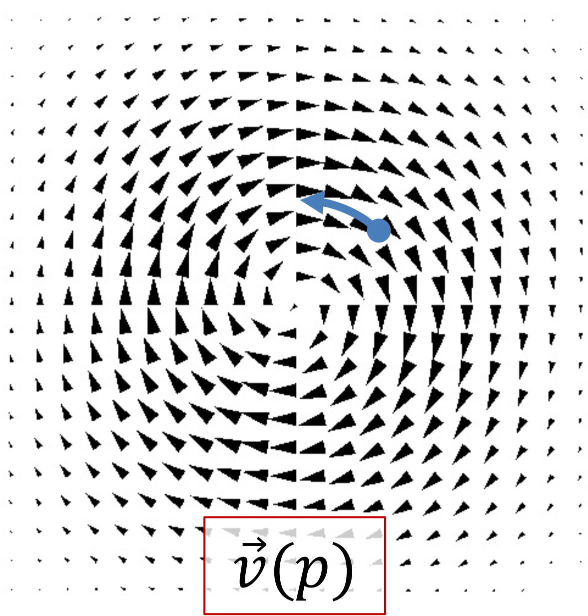
Image-Processing Tools

2c. Flow Fields/Lines:

Given..., for small t we have:

$$[\mathrm{Adv}_{t\cdot\vec{v}}(f)](p) - f(p) \approx f(p - t\vec{v}) - f(p)$$
$$\approx -t \cdot \langle \nabla f(p), \vec{v} \rangle$$





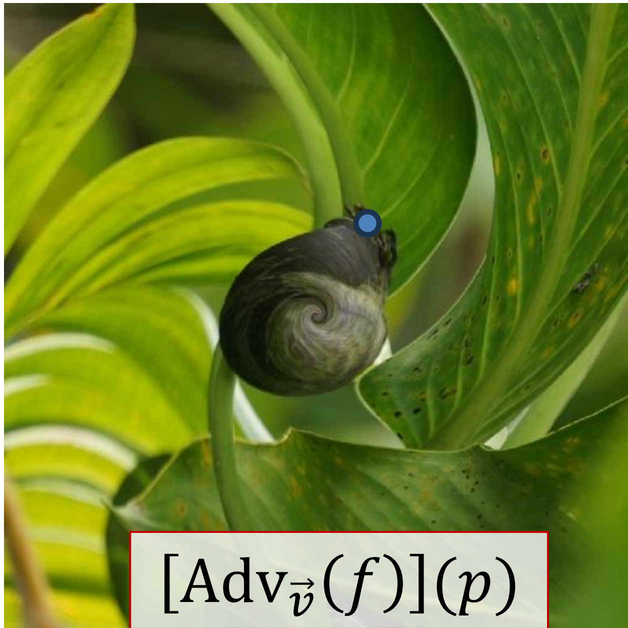


Image-Processing Tools

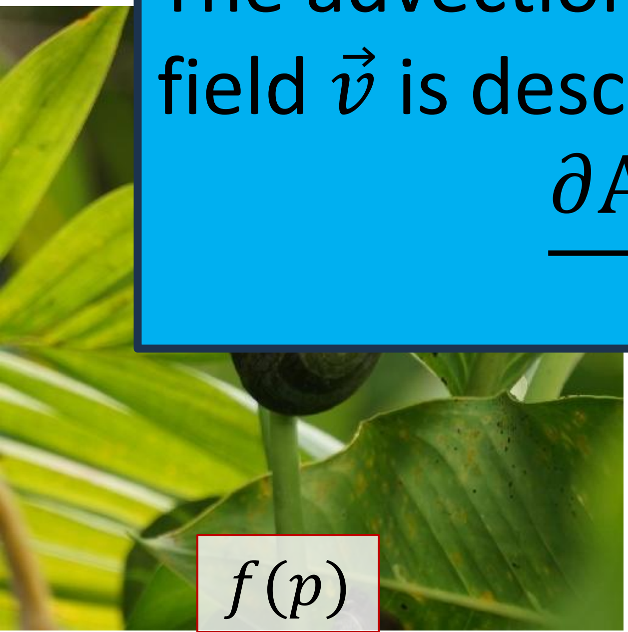
2c. Flow Fields/Lines:

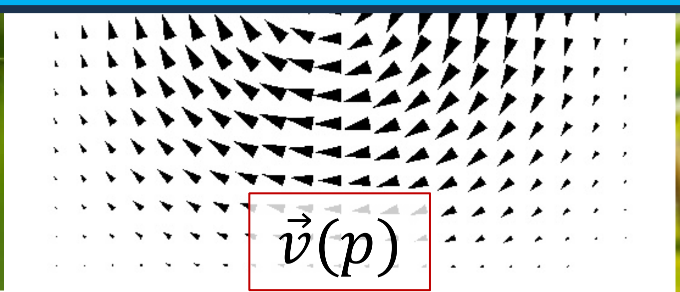
Given..., for small t we have:

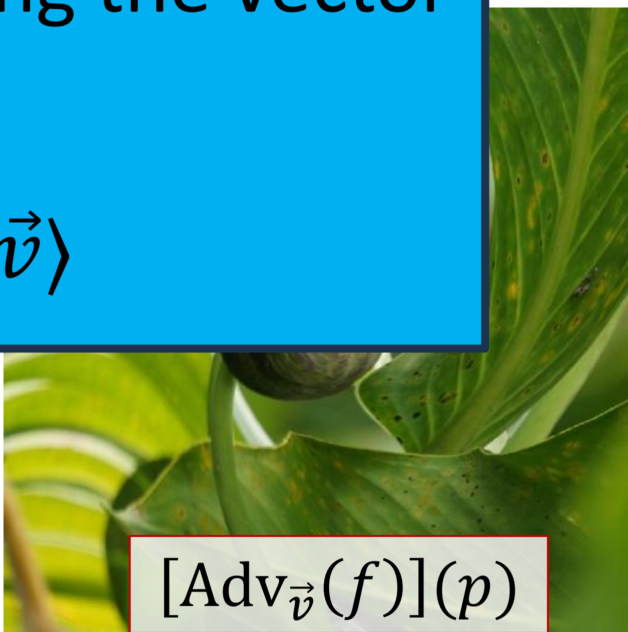
$$[\mathrm{Adv}_{t\cdot\vec{v}}(f)](p) - f(p) \approx f(p - t\vec{v}) - f(p)$$
$$\approx -t \cdot \langle \nabla f(p), \vec{v} \rangle$$

The advection of the signal f along the vector field $\vec{v}$ is described by the PDE:

$$\frac{\partial \mathrm{Adv}_{t\cdot\vec{v}}(f)}{\partial t} = -\langle \nabla f, \vec{v} \rangle$$





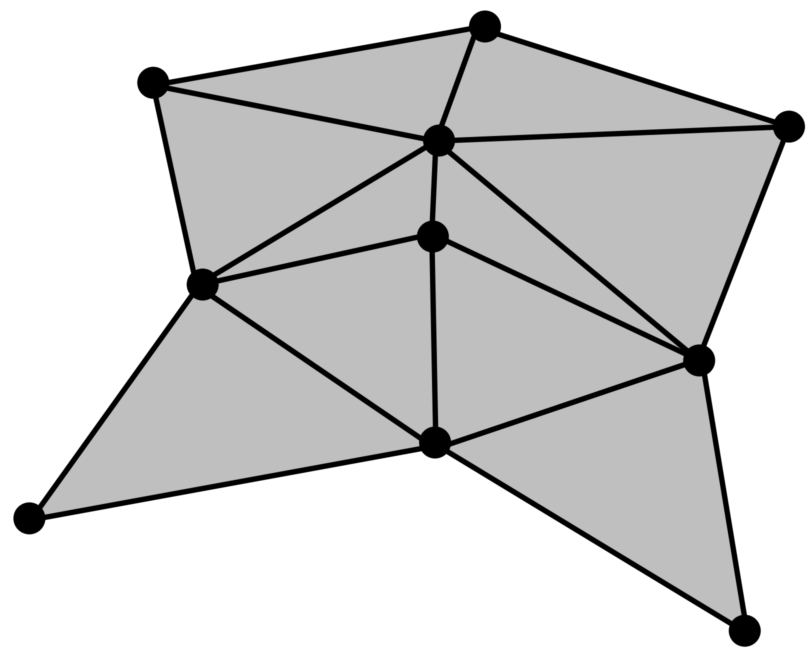


Geometry-Processing Tools

1. Screened Poisson Equation:

$$(\alpha \cdot \mathbf{1} - \mathbf{\Delta})f = \alpha \cdot \mathbf{1} \cdot g - \mathbf{div}(\vec{v})$$

On a mesh:



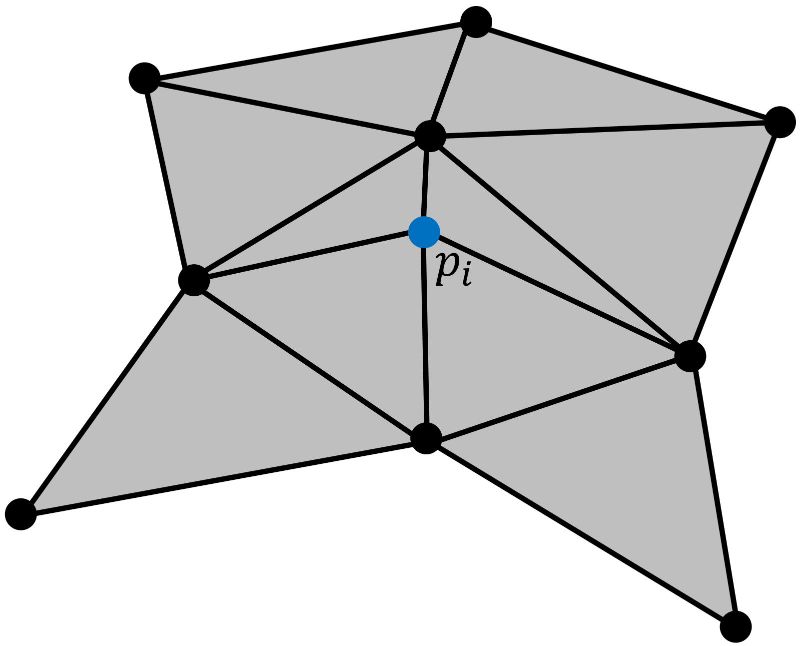
Geometry-Processing Tools

1. Screened Poisson Equation:

$$(\alpha \cdot \mathbf{1} - \boldsymbol{\Delta}) \boxed{f} = \alpha \cdot \mathbf{1} \cdot \boxed{g} - \mathbf{div}(\vec{v})$$

On a mesh:

- $f, g \rightarrow$ maps from vertices to real values



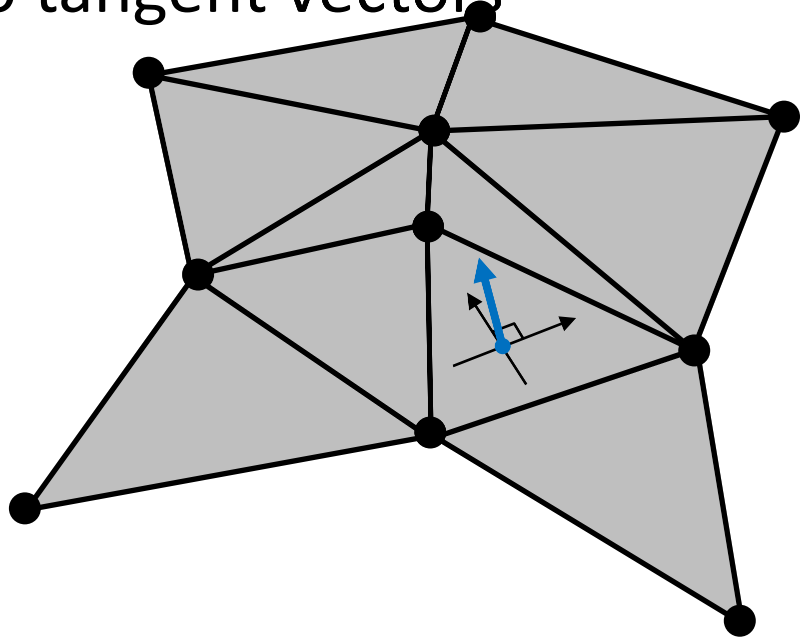
Geometry-Processing Tools

1. Screened Poisson Equation:

$$(\alpha \cdot \mathbf{1} - \mathbf{\Delta})f = \alpha \cdot \mathbf{1} \cdot g - \mathbf{div}(\vec{v})$$

On a mesh:

- $f, g \rightarrow$ maps from vertices to real values
- $\vec{v} \rightarrow$ a map from triangles to tangent vectors



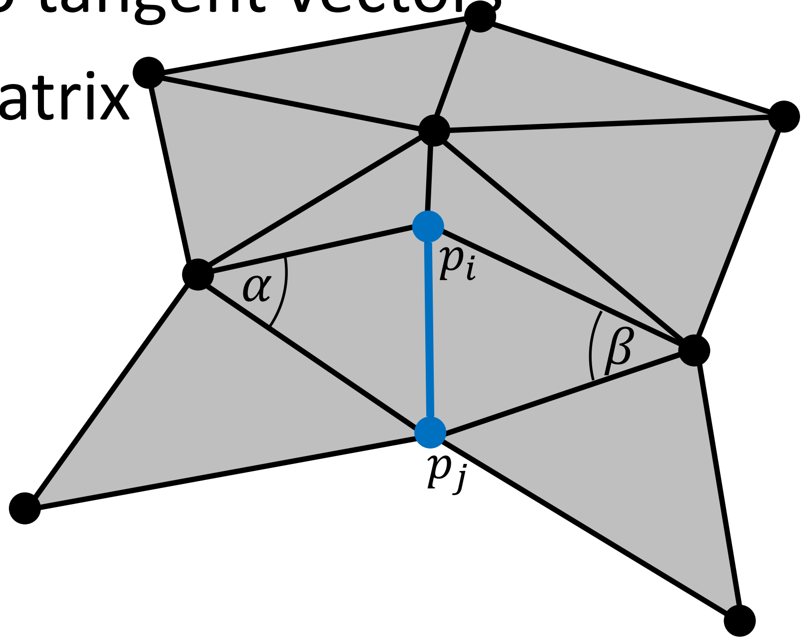
Geometry-Processing Tools

1. Screened Poisson Equation:

$$(\alpha \cdot \mathbf{1} - \mathbf{\Delta})f = \alpha \cdot \mathbf{1} \cdot g - \mathbf{div}(\vec{v})$$

On a mesh:

- $f, g \rightarrow$ maps from vertices to real values
- $\vec{v} \rightarrow$ a map from triangles to tangent vectors
- $\mathbf{\Delta} \rightarrow$ the cotan. Laplacian matrix



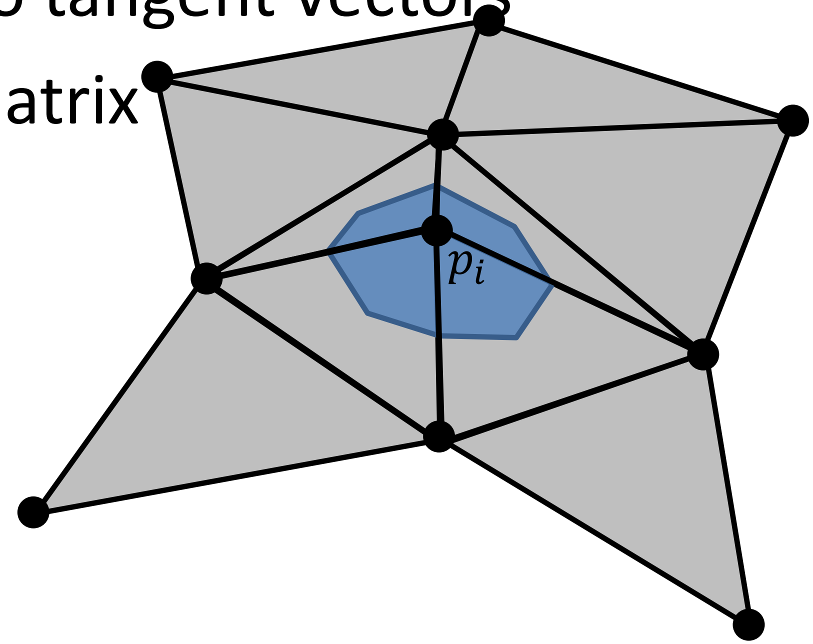
Geometry-Processing Tools

1. Screened Poisson Equation:

$$(\alpha \cdot \boxed{\mathbf{1}} - \mathbf{\Delta})f = \alpha \cdot \boxed{\mathbf{1}} \cdot g - \mathbf{div}(\vec{v})$$

On a mesh:

- $f, g \rightarrow$ maps from vertices to real values
- $\vec{v} \rightarrow$ a map from triangles to tangent vectors
- $\mathbf{\Delta} \rightarrow$ the cotan. Laplacian matrix
- $\mathbf{1} \rightarrow$ the mass-matrix



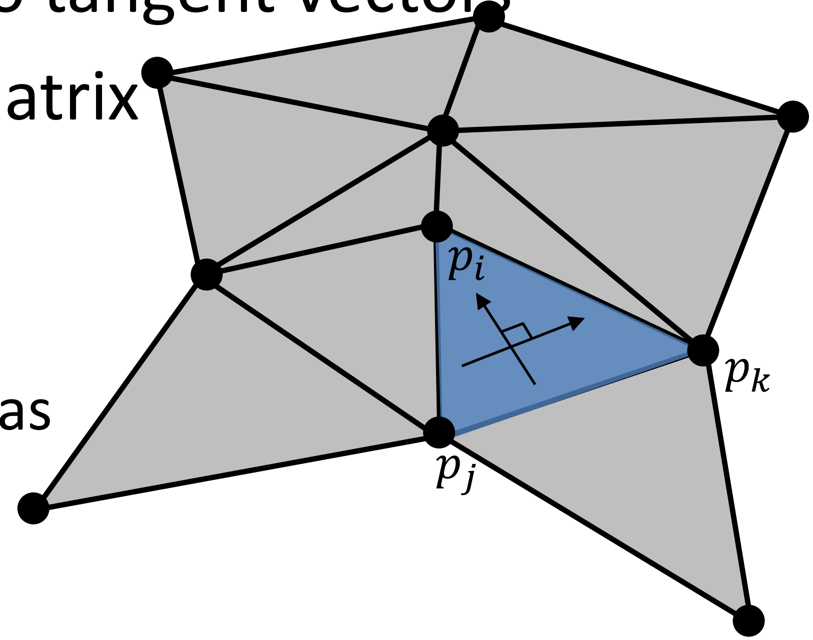
Geometry-Processing Tools

1. Screened Poisson Equation:

$$(\alpha \cdot \mathbf{1} - \mathbf{\Delta})f = \alpha \cdot \mathbf{1} \cdot g - \boxed{\mathbf{div}}(\vec{v})$$

On a mesh:

- $f, g \rightarrow$ maps from vertices to real values
- $\vec{v} \rightarrow$ a map from triangles to tangent vectors
- $\mathbf{\Delta} \rightarrow$ the cotan. Laplacian matrix
- $\mathbf{1} \rightarrow$ the mass-matrix
- $\mathbf{div} \rightarrow \mathbf{\nabla}^{\top} \cdot \mathbf{\Lambda}$:
  - $\mathbf{\Lambda}$: diagonal with triangle areas
  - $\mathbf{\nabla}$: the gradient operator



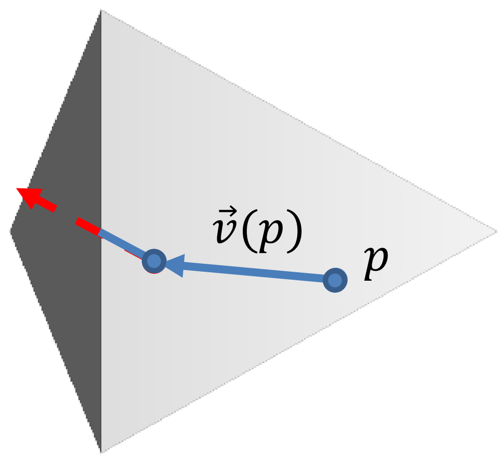
Geometry-Processing Tools

2. Flow Fields/Lines:

$$\gamma_p(0) = p \quad \text{and} \quad \gamma_p'(t) = \vec{v}\left(\gamma_p(t)\right)$$

Iteratively:

- Sample the flow field at p.
- Take a small step in a straight line along the flow direction.



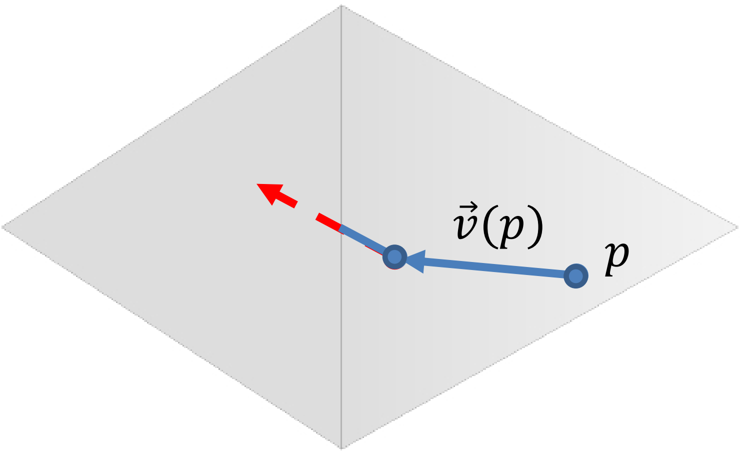
Geometry-Processing Tools

2. Flow Fields/Lines:

$$\gamma_p(0) = p \quad \text{and} \quad \gamma_p'(t) = \vec{v}\left(\gamma_p(t)\right)$$

Iteratively:

- Sample the flow field at p.
- Take a small step in a straight line along the flow direction.



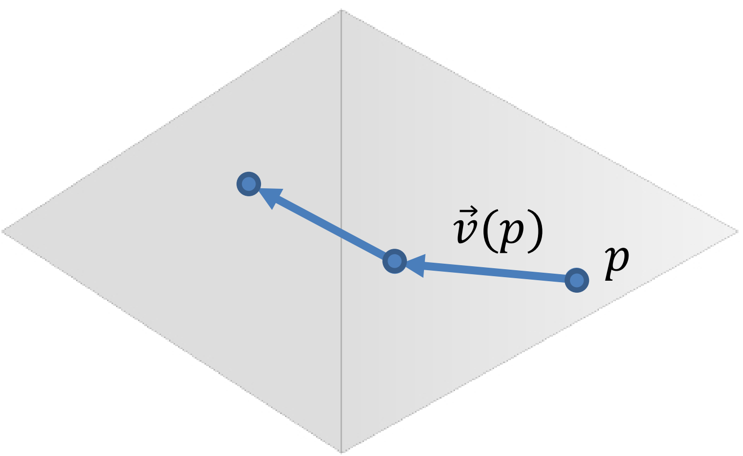
Geometry-Processing Tools

2. Flow Fields/Lines:

$$\gamma_p(0) = p \quad \text{and} \quad \gamma_p'(t) = \vec{v}\left(\gamma_p(t)\right)$$

Iteratively:

- Sample the flow field at p.
- Take a small step in a straight line along the flow direction.



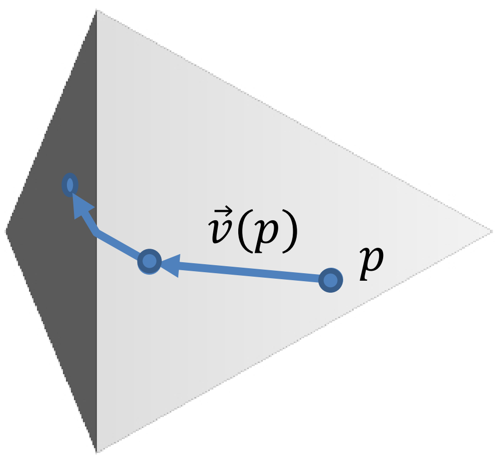
Geometry-Processing Tools

2. Flow Fields/Lines:

$$\gamma_p(0) = p \quad \text{and} \quad \gamma_p'(t) = \vec{v}\left(\gamma_p(t)\right)$$

Iteratively:

- Sample the flow field at p.
- Take a small step in a straight line along the flow direction.

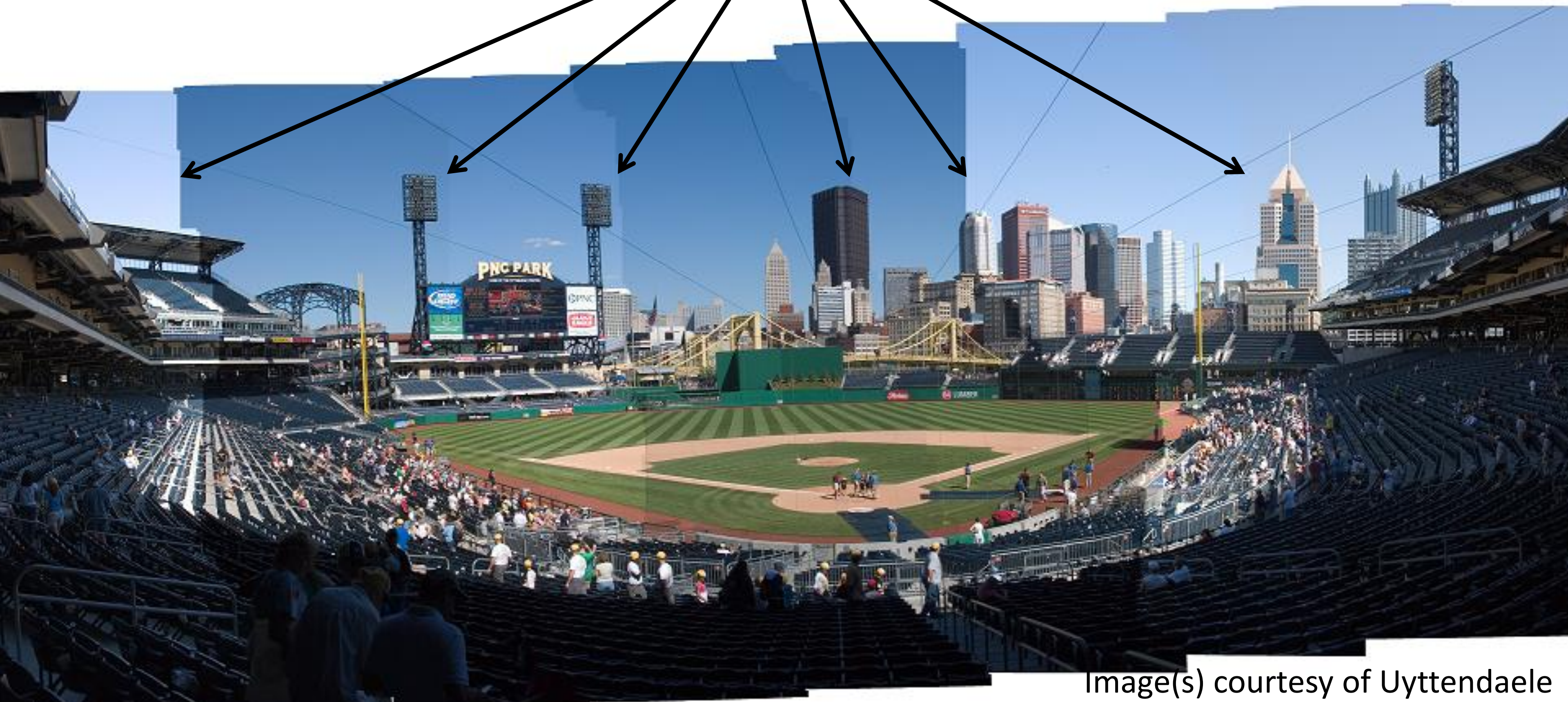


Outline

- Motivation
- Tools of the Trade
- **Extensions to Signals on Surfaces**
  - **Gradient Domain [Poisson]**
  - Shock Filters [Advection]
  - Optical Flow [Poisson + Advection]
- Conclusion

Gradient Domain (Stitching)

Different exposures ⇒ Seams in the panorama



Image(s) courtesy of Uyttendaele

Gradient Domain (Stitching)

- Copy interior gradients into $\vec{v}$
- Set seam-crossing gradients to zero

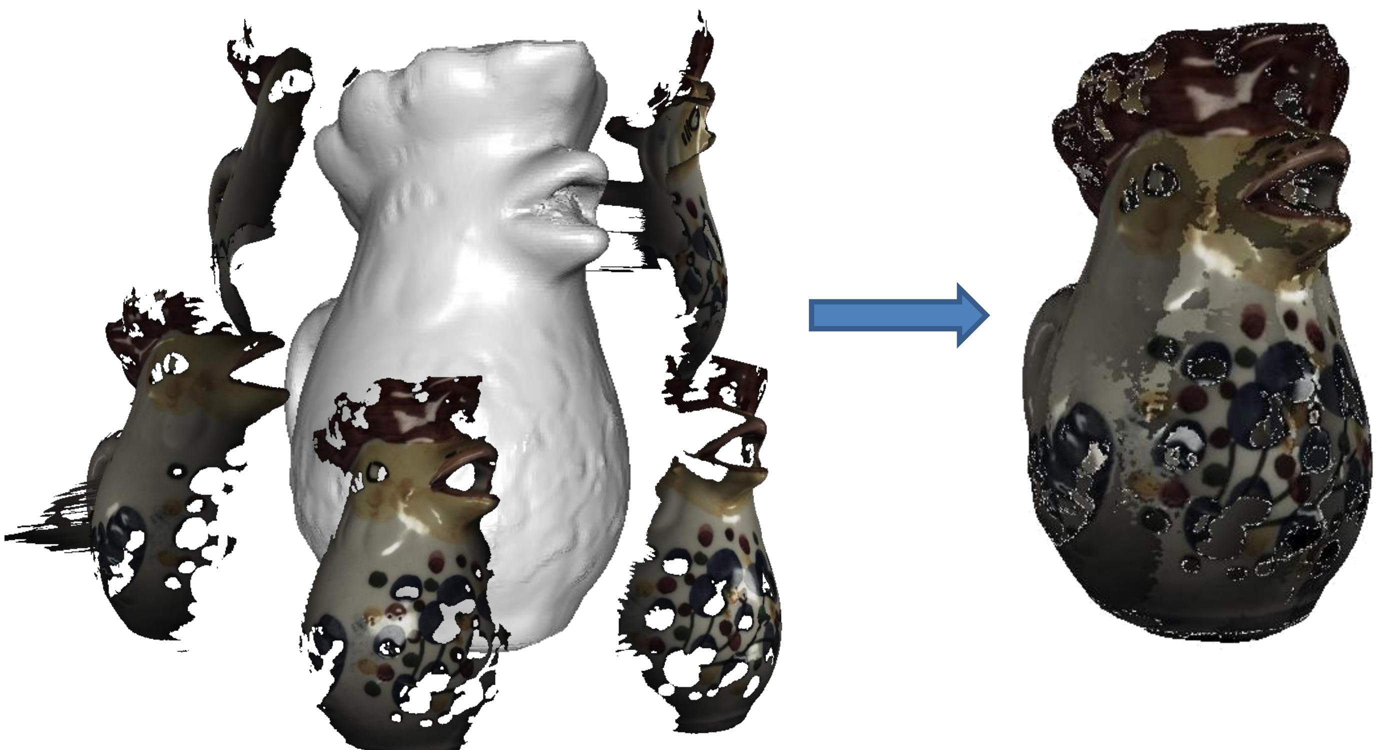
$$f^{out} = \underset{f:\Omega\to\mathbb{R}}{\operatorname{argmin}} \int \|\nabla f - \vec{v}\|^2 dp$$

Image(s) courtesy of Uyttendaele

Gradient Domain (Stitching)

- Copy interior gradients into $\vec{v}$
- Set seam-crossing gradients to zero

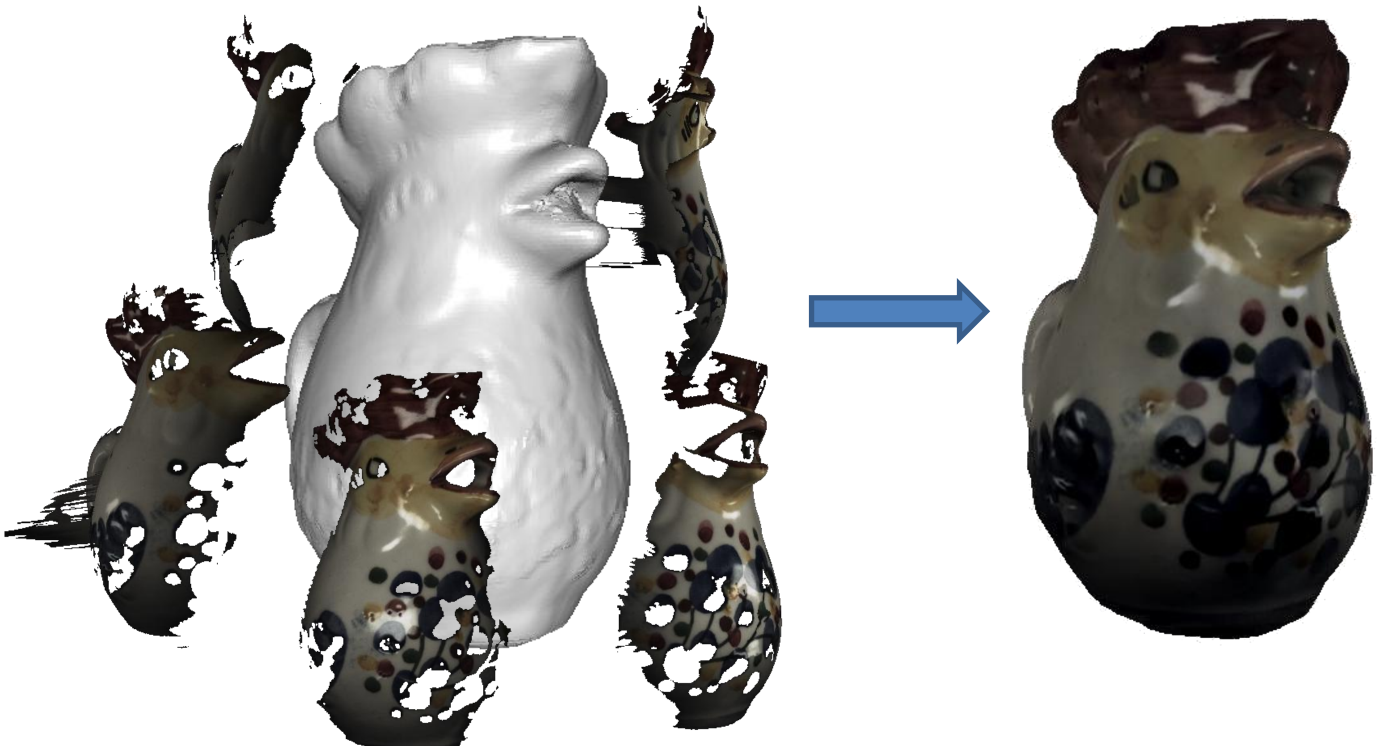
$$f^{out} = \underset{f:\Omega\to\mathbb{R}}{\operatorname{argmin}} \int \|\nabla f - \vec{v}\|^2 dp$$



Gradient Domain (Stitching)

- Copy interior gradients into $\vec{v}$
- Set seam-crossing gradients to zero

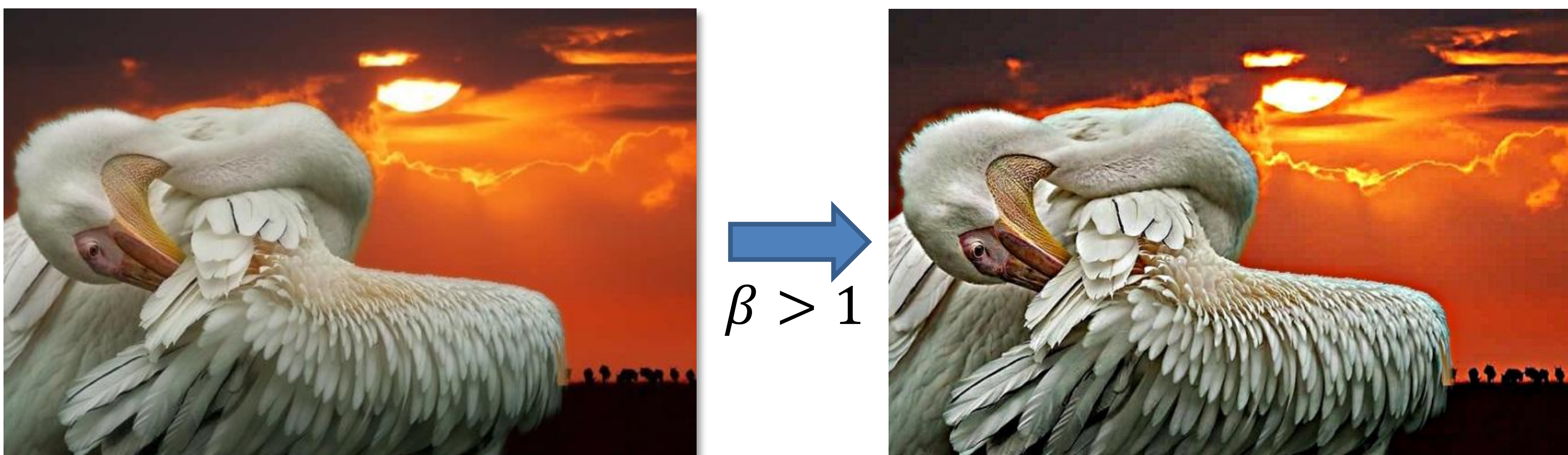
$$f^{out} = \underset{f:\Omega\to\mathbb{R}}{\operatorname{argmin}} \int \|\nabla f - \vec{v}\|^2 dp$$



Gradient Domain (Sharpening)

- Fit input colors: $g = f^{in}$
- Scale (amplify) input gradients: $\vec{v} = \beta \cdot \nabla f^{in}$

$$f^{out} = \operatorname*{argmin}_{f:\Omega \to \mathbb{R}} \int_\Omega \underbrace{\alpha \left\| f - f^{in} \right\|^2}_{\text{value-fitting}} + \underbrace{\left\| \nabla f - \beta \cdot \nabla f^{in} \right\|^2}_{\text{gradient-fitting}}$$

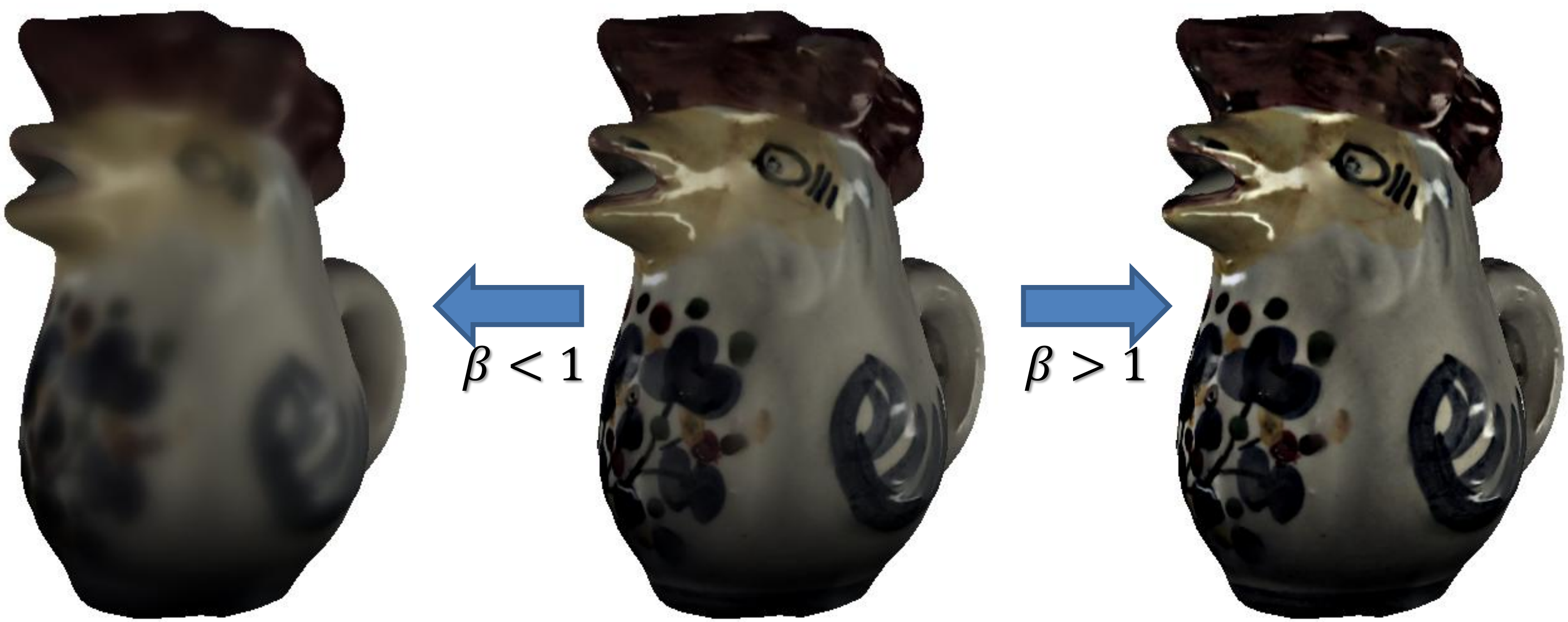


Fourier Analysis of the 2D Screened Poisson Equation for Gradient Domain Problems. [Bhat *et al*. 2008]

Gradient Domain (Sharpening)

- Fit input colors: $g = f^{in}$
- Scale (amplify) input gradients: $\vec{v} = \beta \cdot \nabla f^{in}$

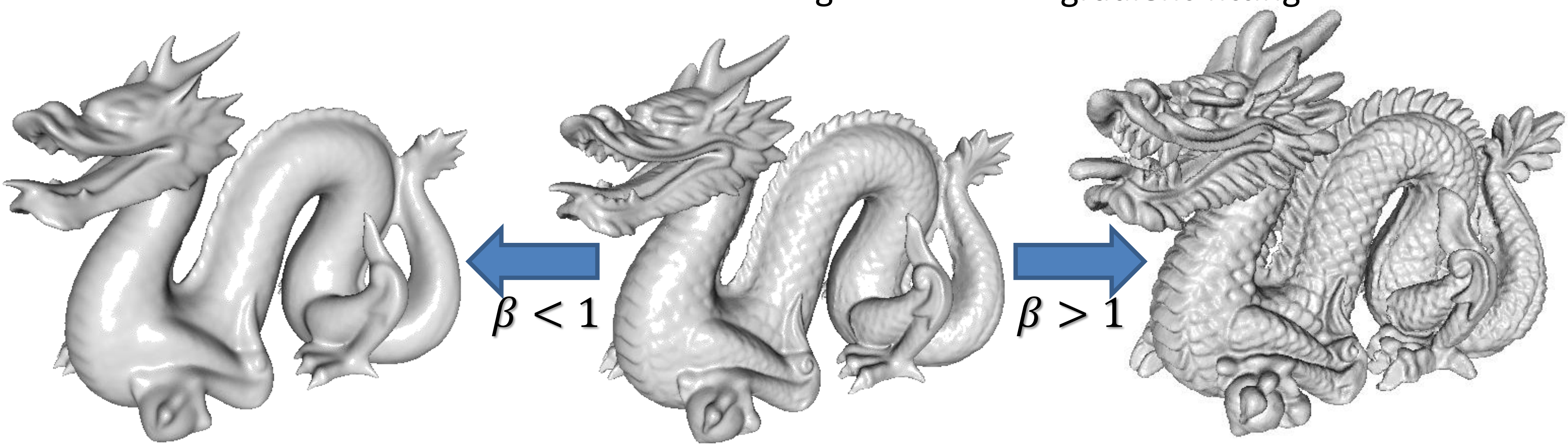
$$f^{out} = \underset{f:\Omega\to\mathbb{R}}{\operatorname{argmin}} \int_\Omega \alpha \underbrace{\left\| f - f^{in} \right\|^2}_{\text{value-fitting}} + \underbrace{\left\| \nabla f - \beta \cdot \nabla f^{in} \right\|^2}_{\text{gradient-fitting}}$$



Gradient Domain (Sharpening)

- Fit input colors: $g = f^{in}$
- Scale (amplify) input gradients: $\vec{v} = \beta \cdot \nabla f^{in}$

$$f^{out} = \underset{f:\Omega\to\mathbb{R}}{\operatorname{argmin}} \int_\Omega \alpha \underbrace{\left\| f - f^{in} \right\|^2}_{\text{value-fitting}} + \underbrace{\left\| \nabla f - \beta \cdot \nabla f^{in} \right\|^2}_{\text{gradient-fitting}}$$



Setting f^{in} to the positions of the vertices in 3D

Outline

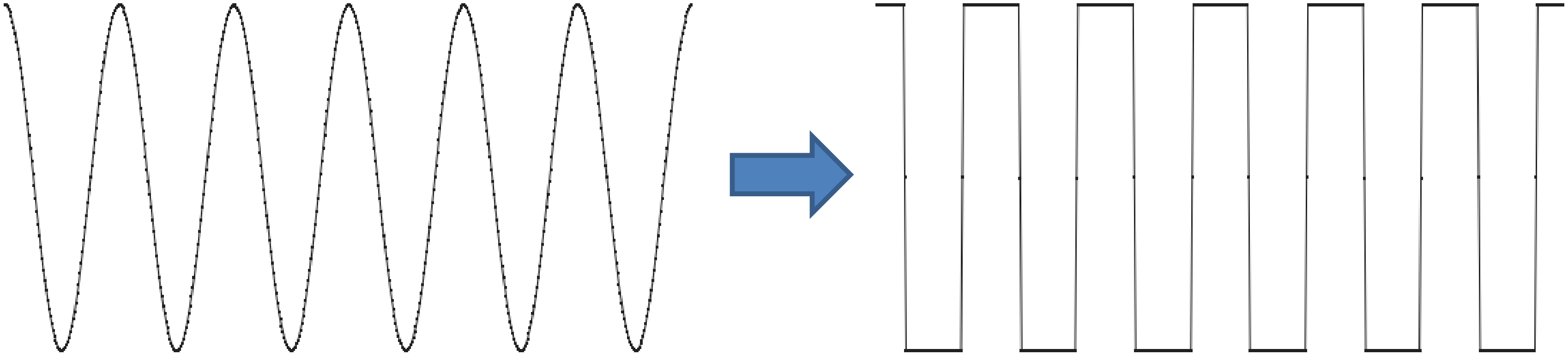
- Motivation
- Tools of the Trade
- **Extensions to Signals on Surfaces**
  - Gradient Domain [Poisson]
  - **Shock Filters [Advection]**
  - Optical Flow [Poisson + Advection]
- Conclusion

Shock Filters

[Osher and Rudin, 1990]:

Progressively sharpen a signal so that:

- Extrema preserved
  - Zero derivative → value fixed
- Edges pronounced
  - Concave up → value decreases
  - Concave down → value increases



Shock Filters

[Osher and Rudin, 1990]:

Progressively sharpen a signal so that:

- Extrema preserved $\rightarrow \mathcal{F}$ vanishes with the gradient
- Edges pronounced $\rightarrow \mathcal{G}$ gives the sign w.r.t. the edge

[Osher and Rudin, 1990] Solve the PDE by carefully adapting step-sizes.

$$\frac{df}{dt} = \mathcal{F}(f) \cdot \mathcal{G}(f)$$

$$\mathcal{F}(f) = \|\nabla f\|^2$$

$$\mathcal{G}(f) = -\frac{\partial^2 f}{\partial(\nabla f / \|\nabla f\|)^2}$$

(Second derivative in the gradient direction)

Shock Filters

Method of Characteristics:

We can re-write the PDE:

$$\frac{df}{dt} = \mathcal{F}(f) \cdot \mathcal{G}(f)$$
$$= -\langle \nabla f, H_f \cdot \nabla f \rangle$$

This describes the advection of f along the flow:

$$\vec{v} = H_f \cdot \nabla f$$

$$\mathcal{F}(f) = \|\nabla f\|^2$$

$$\mathcal{G}(f) = -\frac{\partial^2 f}{\partial(\nabla f / \|\nabla f\|)^2} = -\frac{1}{\|\nabla f\|^2} \langle \nabla f, H_f \cdot \nabla f \rangle$$

Shock Filters

Method of Characteristics:

We can re-write the PDE:

$$\frac{df}{dt} = \mathcal{F}(f) \cdot \mathcal{G}(f)$$
$$= -\langle \nabla f, H_f \cdot \nabla f \rangle$$

This describes the advection of f along the flow:

$$\vec{v} = H_f \cdot \nabla f = \frac{1}{2} \nabla \|\nabla f\|^2$$

$$\mathcal{F}(f) = \|\nabla f\|^2$$

$$\mathcal{G}(f) = -\frac{\partial^2 f}{\partial(\nabla f / \|\nabla f\|)^2} = -\frac{1}{\|\nabla f\|^2} \langle \nabla f, H_f \cdot \nabla f \rangle$$

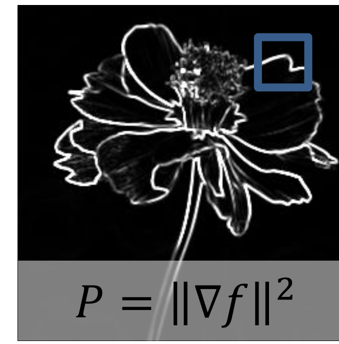
Shock Filters

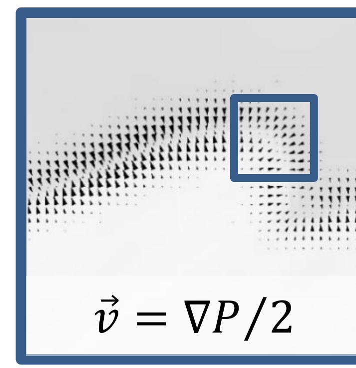
Sharpen f by advecting along the flow:

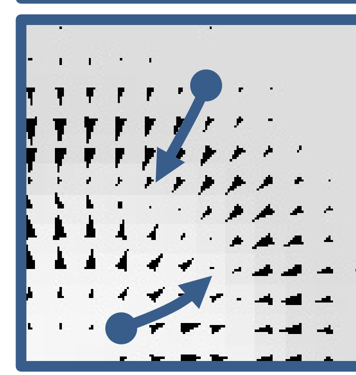
$$\vec{v} = \frac{1}{2}\nabla P \quad \text{w/} \quad P = \|\nabla f\|^2$$

<u>ShockAdvect</u>(f , t)

1. $P \leftarrow \|\nabla f\|^2$ // potential
2. $\vec{v} \leftarrow \frac{1}{2}\nabla P$ // flow field
3. return Advect(f , $\vec{v}$, t)







Shock Filters

Sharpen f by advecting along the flow:

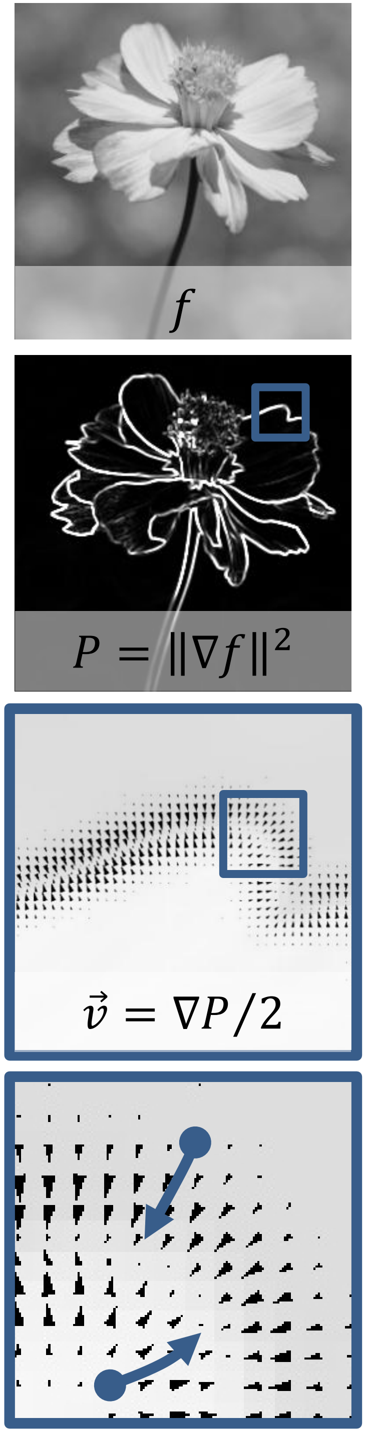
$$\vec{v} = \frac{1}{2}\nabla P \quad \text{w/} \quad P = \|\nabla f\|^2$$

Values are transported:

– From regions where gradient has small magnitude to regions where the gradient has large magnitude

⇔ From regions where the signal is (nearly) constant towards edges

⇒ “Piecewise constant” image with input extrema pushed to the edges.



Shock Filters

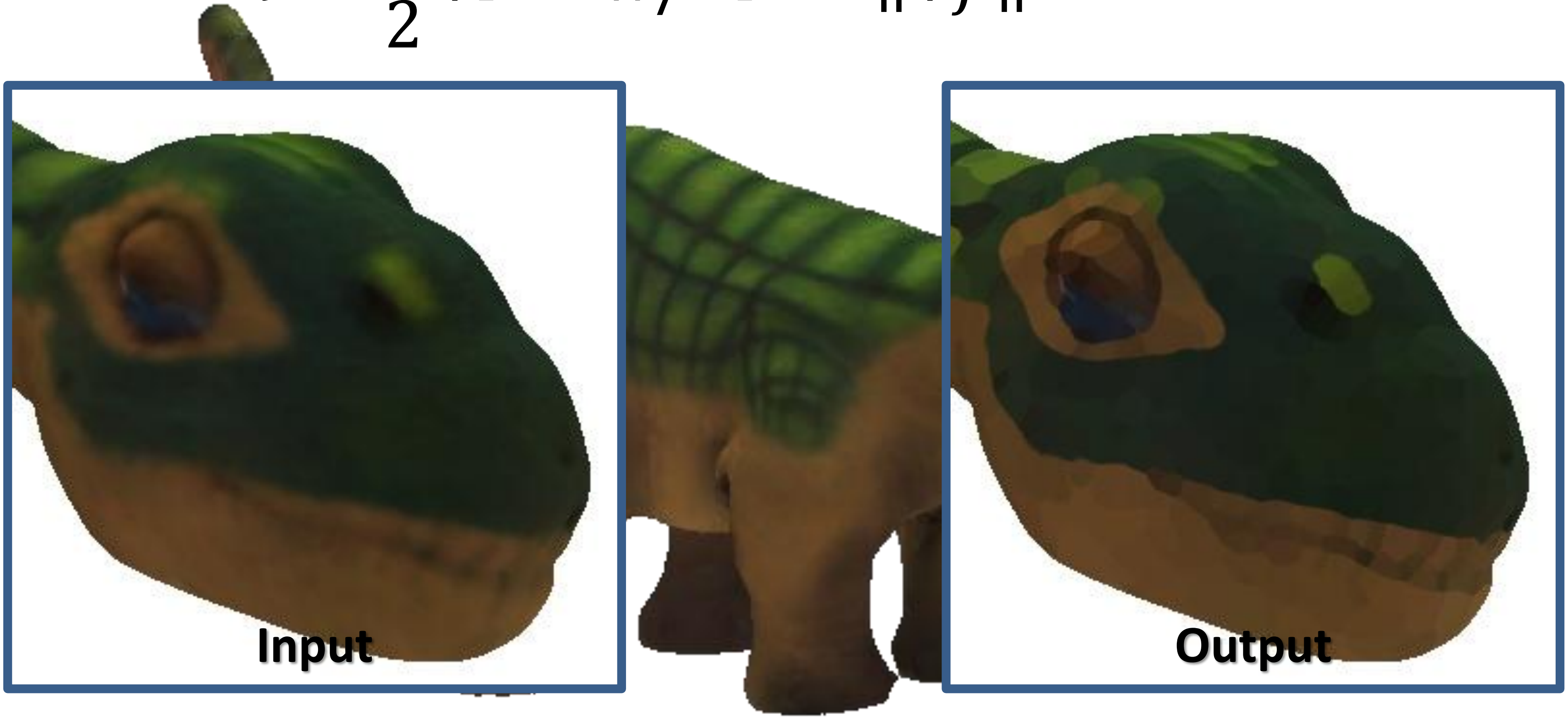
Sharpen f by advecting along the flow:

$$\vec{v} = \frac{1}{2}\nabla P \quad \text{w/} \quad P = \|\nabla f\|^2$$

Shock Filters

Sharpen f by advecting along the flow:

$$\vec{v} = \frac{1}{2}\nabla P \quad \text{w/} \quad P = \|\nabla f\|^2$$



Shock Filters

Sharpen f by advecting along the flow:

$$\vec{v} = \frac{1}{2}\nabla P \quad \text{w/} \quad P = \|\nabla f\|^2$$

Setting f to the <u>normals</u> of the vertices

Outline

- Motivation
- Tools of the Trade
- **Extensions to Signals on Surfaces**
  - Gradient Domain [Poisson]
  - Shock Filters [Advection]
  - **Optical Flow** **[Poisson + Advection]**
- Conclusion

Optical Flow

Video Textures:

Given a video, generate "a continuous, infinitely varying stream of video images".

Video Textures. [Schödl, Szeliski, Salesin, and Essa, 2000]

Optical Flow

Extend video textures [Schödl et al., 2000] to 3D:

Optical Flow

Extend video textures [Schödl et al., 2000] to 3D:

- Identify similar windows in the video
- Interpolate geometries
- Interpolate textures

Optical Flow

<u>Image Interpolation</u>:

Target

Optical Flow

Image Interpolation (Linear):

– Linear interpolation causes ghosting.

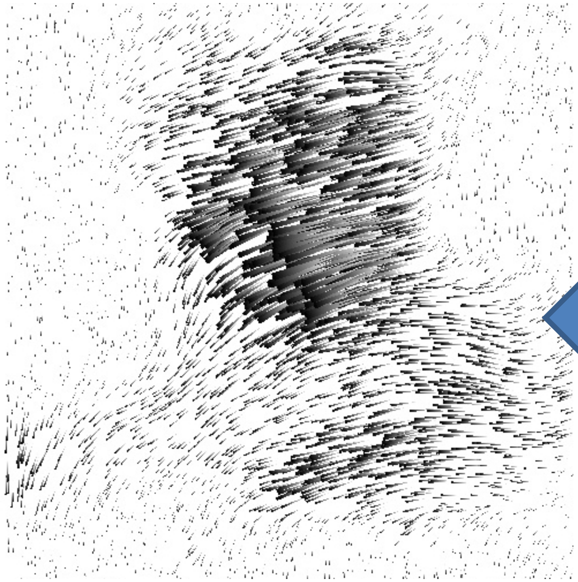
Target

Linear Blend

Optical Flow

<u>Image Interpolation (Advected)</u>:

– Estimate optical flow field.



Optical Flow Field

Target

Linear Blend

Optical Flow

<u>Image Interpolation (Advected)</u>:

– Estimate optical flow field.

– Advect forward/backward and blend.


Optical Flow Blend


Target


Linear Blend

Optical Flow

Brightness Constancy [Lucas and Kanade, 1981]:

Solve for $\vec{v}$ that advects the source/target towards each other by minimizing:

$$
\begin{aligned}
E(\vec{v}) &= \|\text{Adv}_{-\vec{v}}(f^t) - \text{Adv}_{\vec{v}}(f^s)\|^2 \\
&\approx \|(f^t - f^s) + \langle \nabla(f^t + f^s), \vec{v} \rangle\|^2
\end{aligned}
$$

Estimate $\vec{v}$ hierarchically (coarse-to-fine):

- Advance the source/target along $\vec{v}$
- Solve for the correcting flow
- Incorporate the correcting flow into $\vec{v}$
- Advance to the next level of the hierarchy

Optical Flow

Brightness Constancy [Lucas and Kanade, 1981]:

Solve for $\vec{v}$ that advects the source/target towards each other by minimizing:

Smooth signals/vector-fields are implicitly mandated by working in a space that does not have high-frequencies.

Estimate $\vec{v}$ hierarchically (coarse-to-fine):

- Advance the source/target along $\vec{v}$
- Solve for the correcting flow
- Incorporate the correcting flow into $\vec{v}$
- Advance to the next level of the hierarchy

Optical Flow

Scale Space Formulation:

Smooth solutions are explicitly encouraged by:

– Smoothing the source and target at each level:

$$E\left(\tilde{f}^{s/t}\right) = \underbrace{\left\|\tilde{f}^{s/t} - f^{s/t}\right\|^2}_{\text{fitting term}} + \underbrace{\frac{\alpha}{4^l}\left\|\nabla\tilde{f}^{s/t}\right\|^2}_{\text{level-weighted smoothness term}}$$

Optical Flow

Scale Space Formulation:

Smooth solutions are explicitly encouraged by:

– Smoothing the source and target at each level:

$$E\left(\tilde{f}^{s/t}\right) = \left\|\tilde{f}^{s/t} - f^{s/t}\right\|^2 + \frac{\alpha}{4^l}\left\|\nabla\tilde{f}^{s/t}\right\|^2$$

– Incorporating a smoothness term in the energy:

$$E(\vec{v}) = \underbrace{\left\|\mathrm{Adv}_{\vec{v}}\left(\tilde{f}^s\right) - \mathrm{Adv}_{-\vec{v}}\left(\tilde{f}^t\right)\right\|^2}_{\text{brightness constancy term}} + \underbrace{\frac{\alpha}{4^l}\|\nabla\vec{v}\|^2}_{\text{level-weighted smoothness term}}$$

Optical Flow

Scale Space Formulation:

Smooth solutions are explicitly encouraged by:

– Smoothing the source and target at each level:

$$E\left(\tilde{f}^{s/t}\right) = \left\|\tilde{f}^{s/t} - f^{s/t}\right\|^2 + \frac{\alpha}{4^l}\left\|\nabla \tilde{f}^{s/t}\right\|^2$$

– Incorporating a smoothness term in the energy:

$$E(\vec{v}) = \left\|\mathrm{Adv}_{\vec{v}}\left(\tilde{f}^s\right) - \mathrm{Adv}_{-\vec{v}}\left(\tilde{f}^t\right)\right\|^2 + \frac{\alpha}{4^l}\|\nabla\vec{v}\|^2$$

Solve two Poisson equations per level.*

*In the second, the Laplacian is the vector-field (Hodge) Laplacian.

Optical Flow

<u>Texture Interpolation</u>:

Source

Target

Synthesized

Optical Flow

<u>Texture Interpolation</u>:

Source

Target

Synthesized

Optical Flow

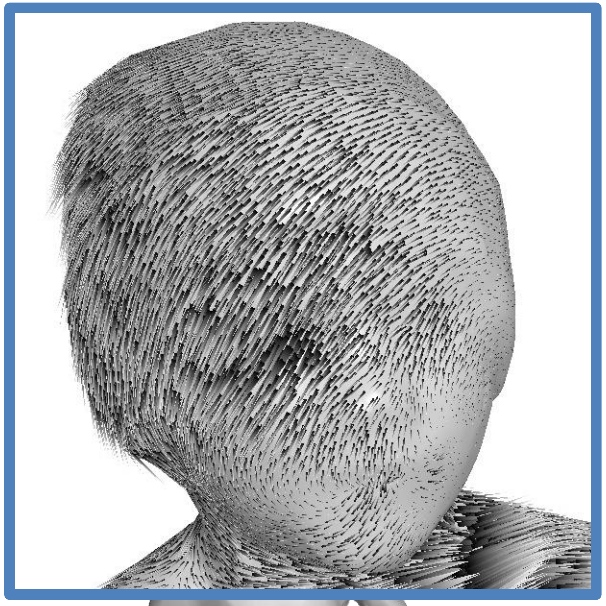
<u>Texture Interpolation</u>:

Source Target Synthesized

Optical Flow

Texture Interpolation:

Optical Flow Field



Source

Target

Synthesized

Optical Flow

<u>Texture Interpolation</u>:

Linear Blend

Optical Flow Blend

Outline

- Motivation
- Tools of the Trade
- Extensions to Signals on Surfaces
- **Conclusion**

Conclusion

Extend fundamental image-processing operators to the context of surfaces

[Differential Operators / Flows]

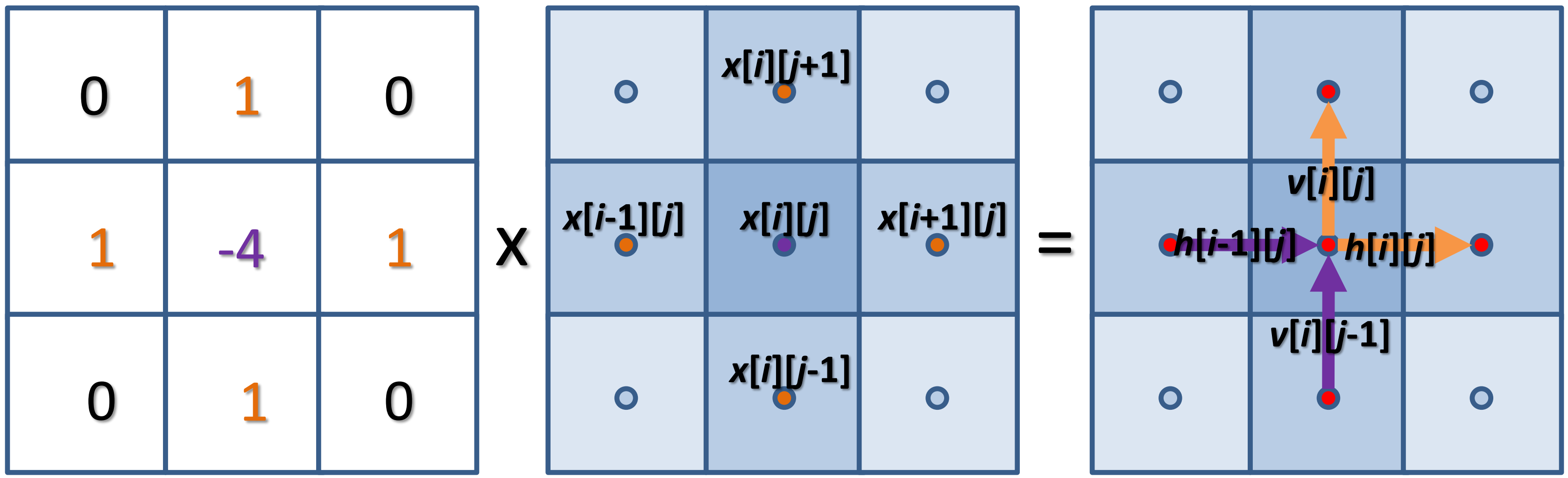
Much of the heavy lifting has already been done for us

Well-established image-processing techniques carry over

[Gradient Domain / Shock Filters / Optical Flow]

Conclusion

Working with images makes simple things easier.



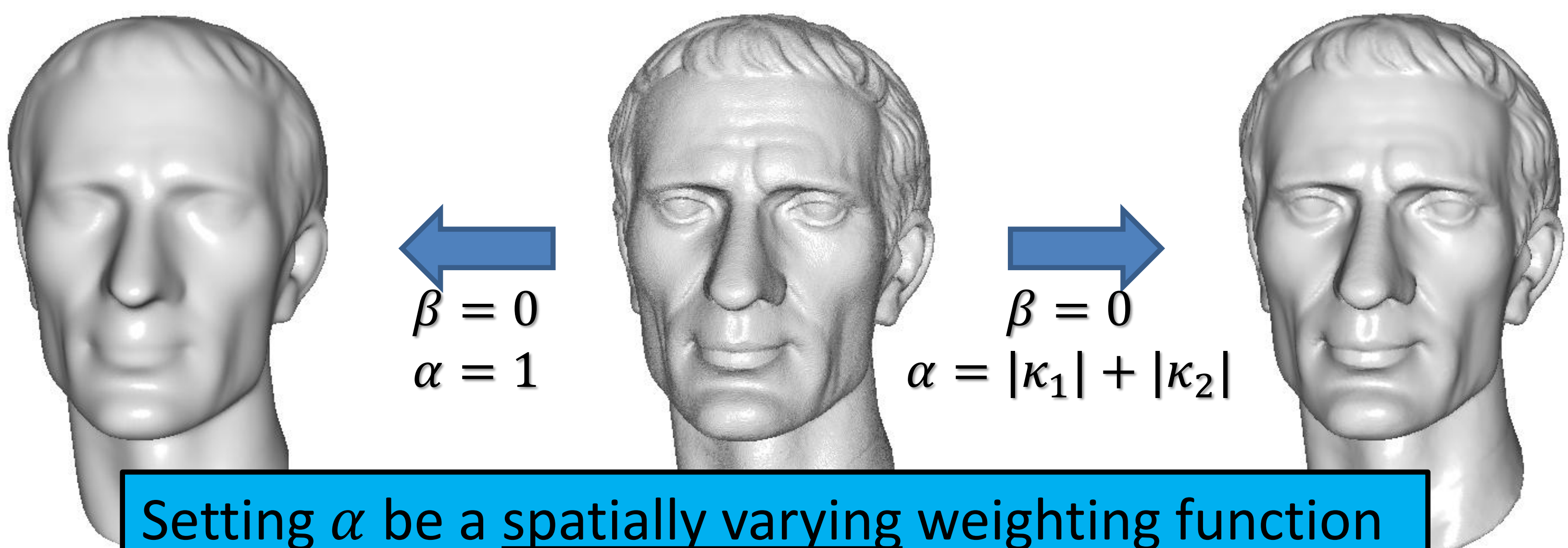
Fixed stencil Laplacian / Gradient / Divergence
Parallelization / Out-of-Core Streaming
Fast Fourier Transform

Conclusion

Working with images makes simple things easier.

Working with meshes makes hard things easier.

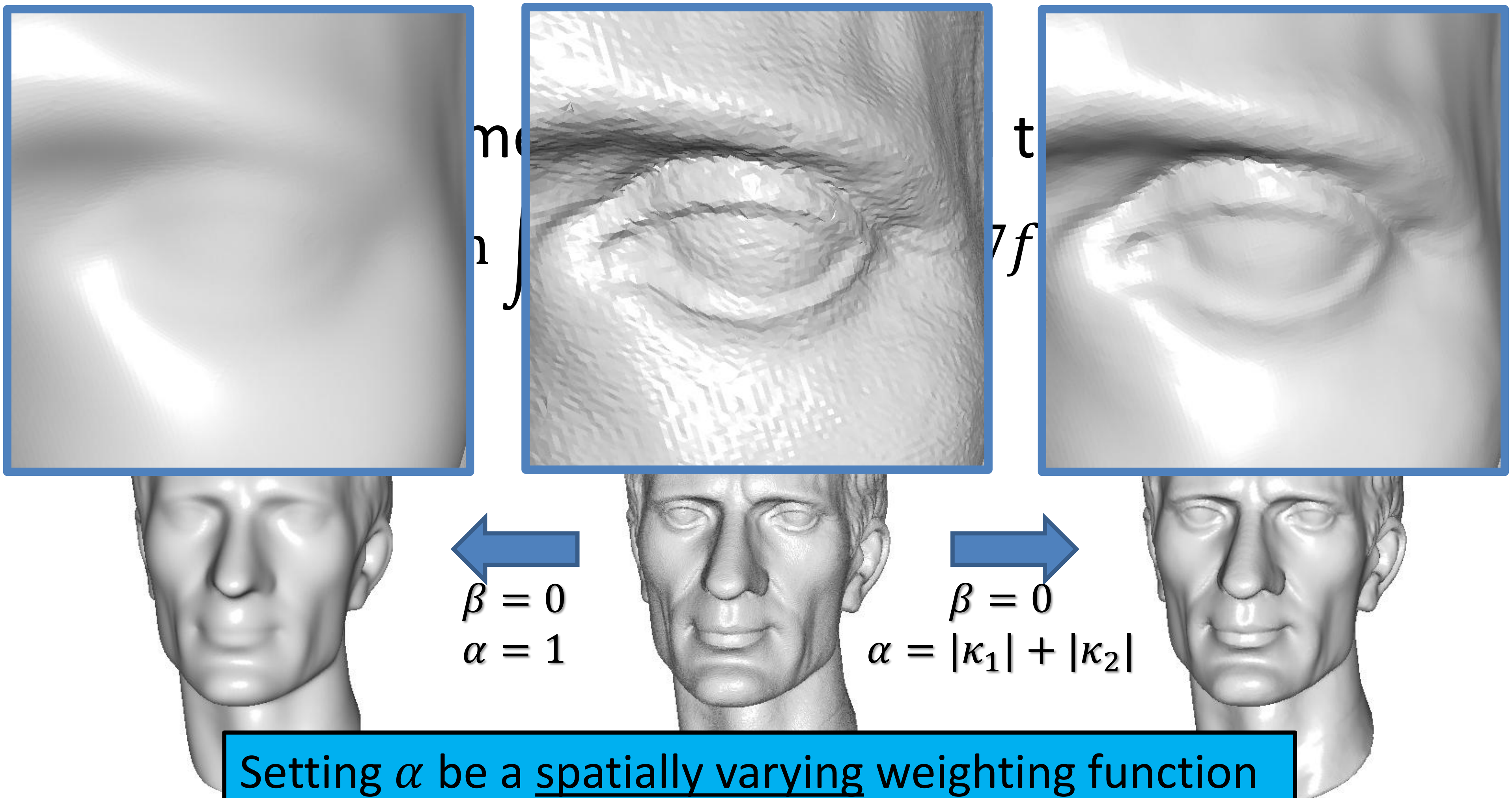
$$f^{out} = \underset{f:\Omega\to\mathbb{R}}{\operatorname{argmin}} \int_\Omega \alpha \left\| f - f^{in} \right\|^2 + \left\| \nabla f - \beta \cdot \nabla f^{in} \right\|^2$$



Setting α be a <u>spatially varying</u> weighting function

Conclusion

Working with images makes simple things easier.



Setting α be a spatially varying weighting function