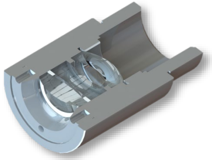
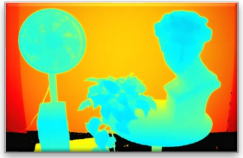








Computational Imaging

Lecture 11: Computing Toolkit: Image Gradient and Processing I

Qilin Sun（孙启霖）

School of Data Science

The Chinese University of Hong Kong, Shenzhen

Today's Topic

- Introduction
- Basics of Gradients and Fields
- Integrable vector fields.
- Poisson blending.

Introduction to
Gradient-Domain Image Processing

Applications

Poisson Blending

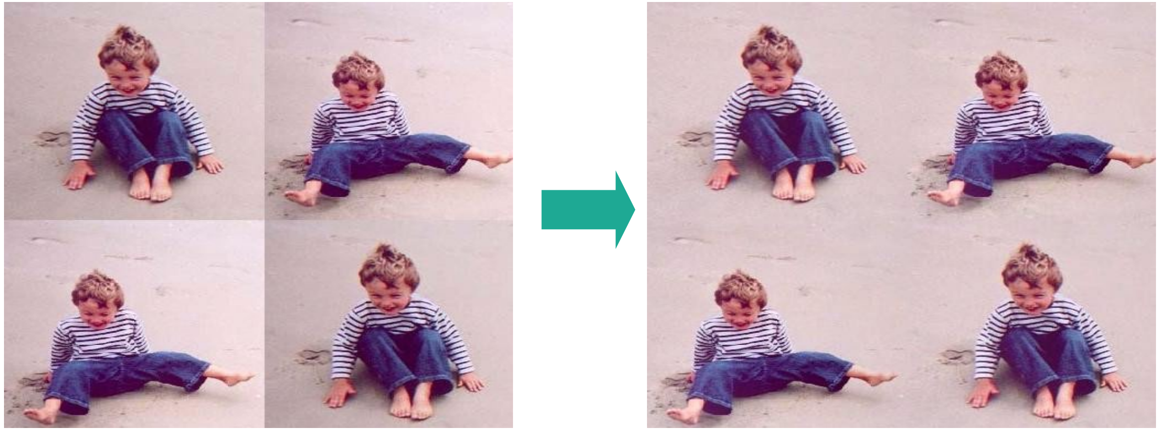
Copy-paste

Poisson blending

http://www.alanyao.com/school/CS194/proj4/

Applications

Glass Reflections Removal



Seamless Image Stitching

Applications

Fusing day and night photos

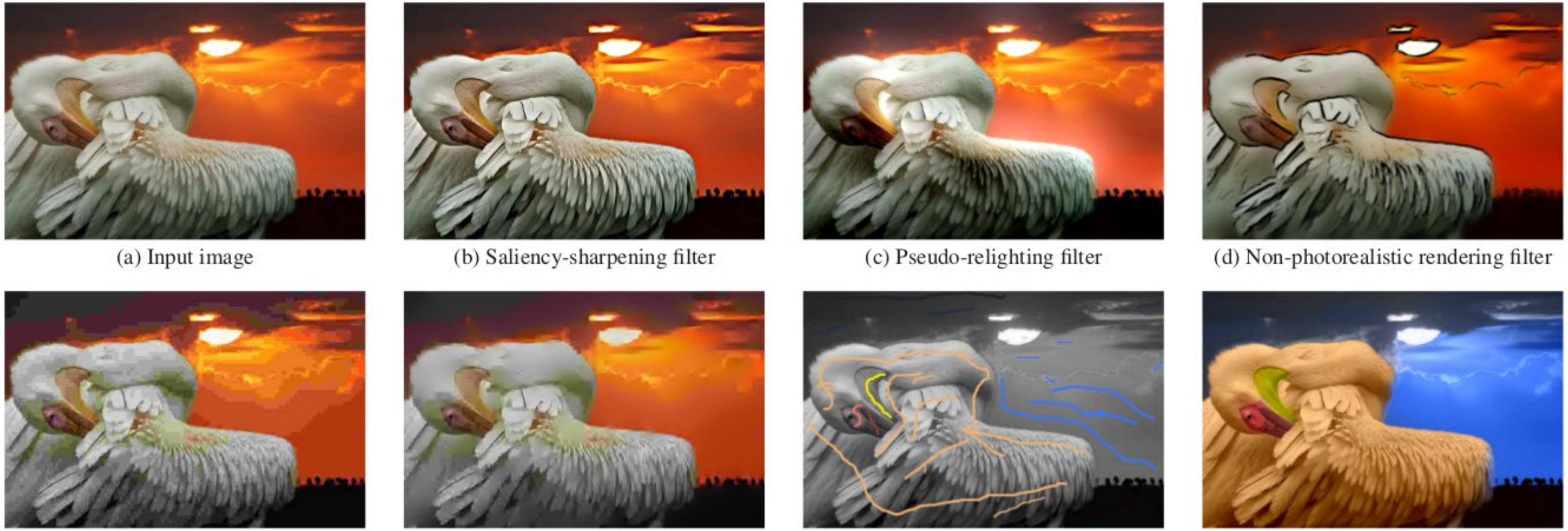
Tonemapping

Entire Suite of Image Editing Tools

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

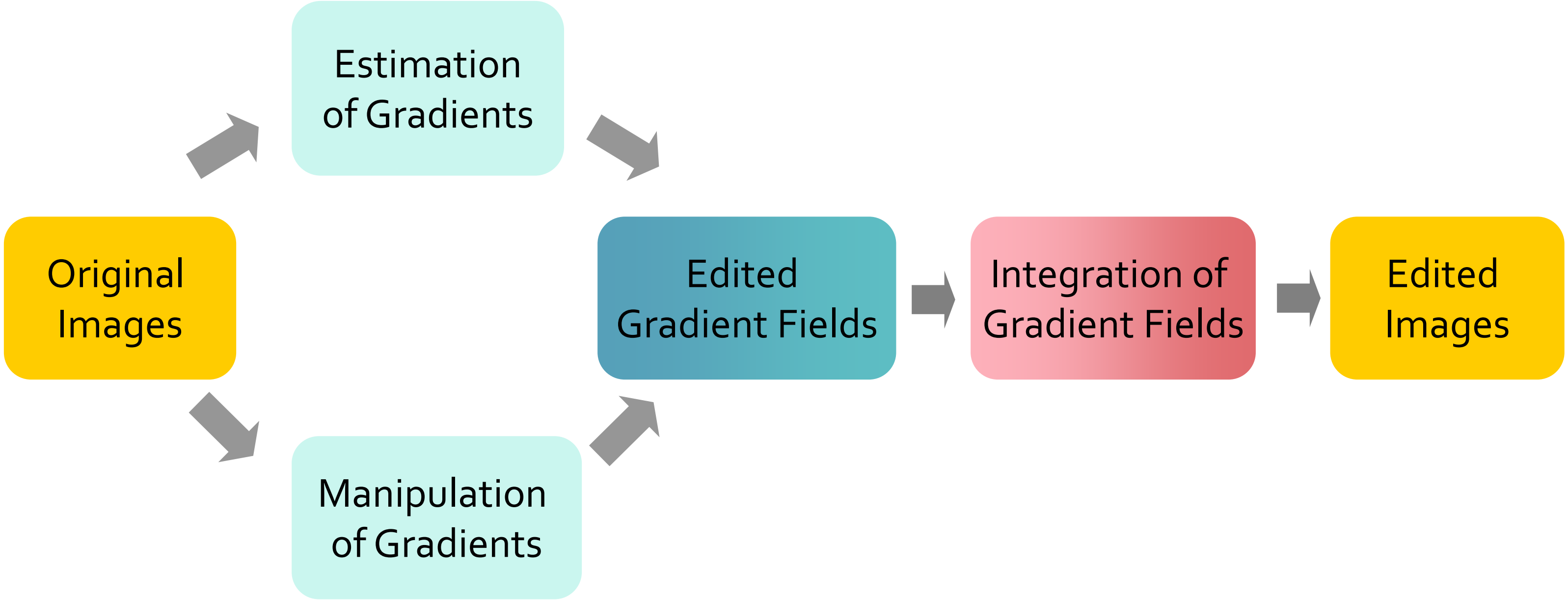
GradientShop: A Gradient-Domain Optimization Framework for Image and Video Filtering

Pravin Bhat[1] C. Lawrence Zitnick[2] Michael Cohen[1,2] Brian Curless[1]
[1]University of Washington [2]Microsoft Research



(a) Input image
(b) Saliency-sharpening filter
(c) Pseudo-relighting filter
(d) Non-photorealistic rendering filter
(e) Compressed input-image
(f) De-blocking filter
(g) User input for colorization
(h) Colorization filter

Main Pipeline



Basics of Gradients and Fields

Some Vector Calculus Definitions in 2D

Scalar field: a function assigning a scalar to every point in space.

$$I(x, y): \mathbb{R}^2 \rightarrow \mathbb{R}$$

Vector field: a function assigning a vector to every point in space.

$$[u(x, y) \quad v(x, y)]: \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

Can you think of examples of scalar fields and vector fields?

- A grayscale image is a scalar field.
- A two-channel image is a vector field.
- A three-channel (e.g., RGB) image is also a vector field, but of higher-dimensional range than what we will consider here.

Vector Calculus Definitions in 2D

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Nabla (or del): vector differential operator.

$$\nabla = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix}$$

Think of this as a 2D vector.

Gradient (grad): product of nabla with a scalar field.

$$\nabla I(x,y) = \begin{bmatrix} \frac{\partial I}{\partial x}(x,y) & \frac{\partial I}{\partial y}(x,y) \end{bmatrix}$$

This is a vector field.

Divergence: inner product of nabla with a vector field.

$$\nabla \cdot [u(x,y) \quad v(x,y)] = \frac{\partial u}{\partial x}(x,y) + \frac{\partial v}{\partial y}(x,y)$$

This is a scalar field.

Curl: cross product of nabla with a vector field.

$$\nabla \times [u(x,y) \quad v(x,y)] = \left(\frac{\partial v}{\partial x}(x,y) - \frac{\partial u}{\partial y}(x,y) \right) \hat{k}$$

This is a vector field.

This is a scalar field.

Vector Calculus Definitions in 2D

Nabla (or del): vector differential operator.

$$\nabla = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix}$$

Think of this as a 2D vector.

Gradient (grad): product of nabla with a scalar field.

$$\nabla I(x, y) = \begin{bmatrix} \frac{\partial I}{\partial x}(x, y) & \frac{\partial I}{\partial y}(x, y) \end{bmatrix}$$

This is a <u>vector</u> field.

Divergence: inner product of nabla with a vector field.

$$\nabla \cdot [u(x, y) \quad v(x, y)] = \frac{\partial u}{\partial x}(x, y) + \frac{\partial v}{\partial y}(x, y)$$

This is a <u>scalar</u> field.

Curl: cross product of nabla with a vector field.

$$\nabla \times [u(x, y) \quad v(x, y)] = \boxed{\left(\frac{\partial v}{\partial x}(x, y) - \frac{\partial u}{\partial y}(x, y)\right)} \hat{k}$$

This is a <u>scalar</u> field

This is a <u>vector</u> field.

Combinations

Curl of the gradient:

$$\nabla \times \nabla I(x,y) = \frac{\partial^2}{\partial y \partial x} I(x,y) - \frac{\partial^2}{\partial x \partial y} I(x,y)$$

Divergence of the gradient:

$$\nabla \cdot \nabla I(x,y) = \frac{\partial^2}{\partial x^2} I(x,y) + \frac{\partial^2}{\partial y^2} I(x,y) \equiv \Delta I(x,y)$$

Laplacian: scalar differential operator.

$$\Delta \equiv \nabla \cdot \nabla = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

Inner product of del with itself!

Simplified Notation

Nabla (or del): vector differential operator.

$$\nabla = [\quad {}_x \qquad {}_y]$$

Think of this as a 2D vector.

Gradient (grad): product of nabla with a scalar field.

$$\nabla I \;= [I_x \quad I_y]$$

This is a vector field.

Divergence: inner product of nabla with a vector field.

$$\nabla \cdot [u \quad v] = u_x + v_y$$

This is a scalar field.

Curl: cross product of nabla with a vector field.

$$\nabla \times [u \quad v] = (v_x - u_y)\hat{k}$$

This is a scalar field

Simplified Notation

Curl of the gradient:

$$\nabla \times \nabla I = I_{yx} - I_{xy}$$

Divergence of the gradient:

$$\nabla \cdot \nabla I = I_{xx} + I_{yy} \equiv \Delta I$$

Laplacian: scalar differential operator.

$$\Delta \equiv \nabla \cdot \nabla = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

Inner product of del with itself!

Image Representation

- We can treat grayscale images as scalar fields (i.e., two dimensional functions)

$$I(x, y): \mathbb{R}^2 \rightarrow \mathbb{R}$$

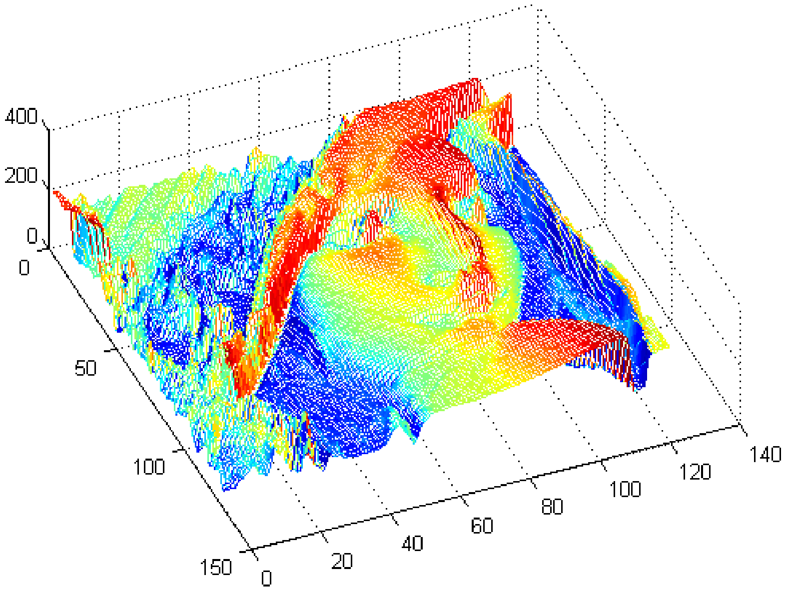
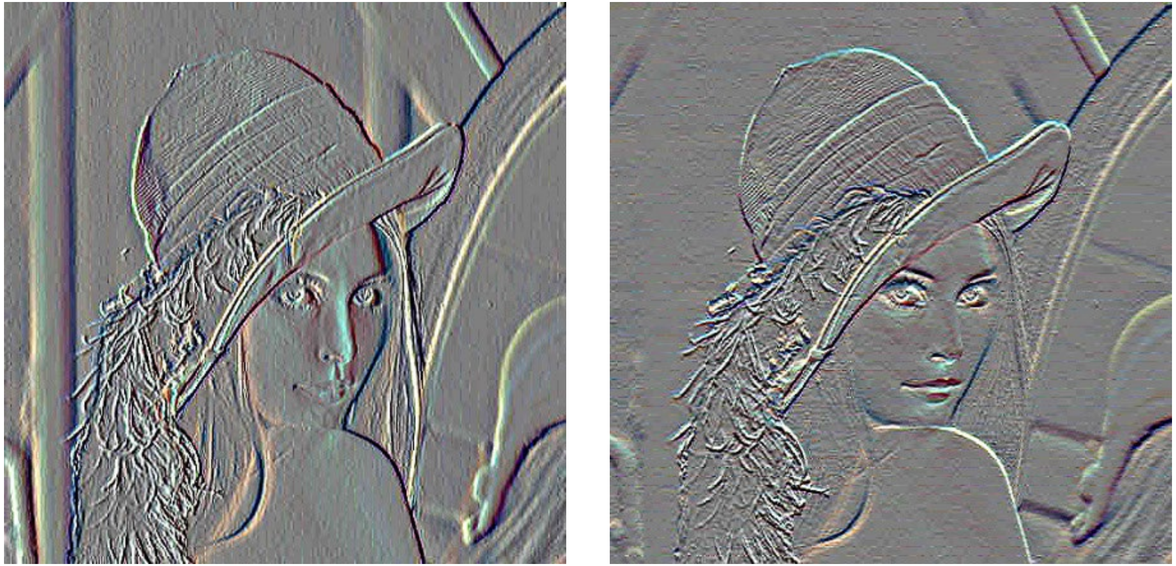


Image Gradients

- Convert the ***scalar*** field into a ***vector*** field through differentiation.

scalar field

$$I(x,y): \mathbb{R}^2 \rightarrow \mathbb{R}$$



vector field

$$\nabla I(x,y) = \left[\frac{\partial I}{\partial x}(x,y) \quad \frac{\partial I}{\partial y}(x,y) \right]$$

Finite Differences

Definition of a derivative using forward difference.

$$\frac{\partial I}{\partial x}(x, y) = \lim_{h \to 0} \frac{I(x+h, y) - I(x, y)}{h}$$

For discrete scalar fields: remove limit and set h = 1.

$$\frac{\partial I}{\partial x}(x, y) = I(x+1, y) - I(x, y)$$

What <u>convolution</u> kernel does this correspond to?

Finite Differences

Definition of a derivative using forward difference.

$$\frac{\partial I}{\partial x}(x, y) = \lim_{h \to 0} \frac{I(x+h, y) - I(x, y)}{h}$$

For discrete scalar fields: remove limit and set h = 1.

$$\frac{\partial I}{\partial x}(x, y) = I(x+1, y) - I(x, y)$$

-1	1	?
1	-1	?

Finite Differences

Definition of a derivative using forward difference.

$$\frac{\partial I}{\partial x}(x,y) = \lim_{h \to 0} \frac{I(x+h,y) - I(x,y)}{h}$$

For discrete scalar fields: remove limit and set h = 1.

$$\frac{\partial I}{\partial x}(x,y) = I(x+1,y) - I(x,y)$$

partial-x derivative filter

1	-1

?

Note: common to use central difference, but we will *not* use it in this lecture.

$$\frac{\partial I}{\partial x}(x,y) = \frac{I(x+1,y) - I(x-1,y)}{2}$$

Finite Differences

Definition of a derivative using forward difference.

$$\frac{\partial I}{\partial x}(x,y) = \lim_{h \to 0} \frac{I(x+h,y) - I(x,y)}{h}$$

For discrete scalar fields: remove limit and set h = 1.

$$\frac{\partial I}{\partial x}(x,y) = I(x+1,y) - I(x,y)$$

partial-x derivative filter

1	-1

?

Similarly for partial-y derivative.

$$\frac{\partial I}{\partial y}(x,y) = I(x,y+h) - I(x,y)$$

partial-y derivative filter

1
-1

Discrete Laplacian

How do we compute the image Laplacian?

$$\Delta I(x,y) = \frac{\partial^2 I}{\partial x^2}(x,y) + \frac{\partial^2 I}{\partial y^2}(x,y)$$

Note:
- ➢ use consistent derivative and Laplacian filters.
- ➢ account for boundary shifting and padding from convolution.

Use multiple applications of the discrete derivative filters:

$$\underbrace{\begin{bmatrix} 1 & -1 \end{bmatrix} * \begin{bmatrix} 1 & -1 \end{bmatrix}}_{\frac{\partial^2 I}{\partial x^2}(x,y)} + \underbrace{\begin{bmatrix} 1 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 \\ -1 \end{bmatrix}}_{\frac{\partial^2 I}{\partial y^2}(x,y)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Laplacian filter

Warning!

A correct implementation of differential operators should pass the following test:

Note:
- use consistent derivative and Laplacian filters.
- account for boundary shifting and padding from convolution.

Equality holds at all pixels except boundary (first and last row, first and last column).

$\nabla \cdot ($ $\nabla($  $)$ $)$ $=$ $\Delta($  $)$

gradient operator

Laplacian operator

divergence operator

Typically requires implementing derivatives in various differential operators differently.

Image Gradients

Convert the ***scalar*** field into a ***vector*** field through differentiation.

scalar field $I(x, y): \mathbb{R}^2 \to \mathbb{R}$

vector field $\nabla I(x, y) = \left[\frac{\partial I}{\partial x}(x, y) \quad \frac{\partial I}{\partial y}(x, y)\right]$

- How do we do this differentiation in real ***discrete*** images?
- Can we go in the opposite direction, from gradients to images?

Vector Field Integration

Two fundamental questions:

- When is integration of a vector field possible?
- How can integration of a vector field be performed?

Integrable Vector Fields

Integrable Fields

- Given an arbitrary vector field (u, v), can we always integrate it into a scalar field I?

?

$I(x, y)\colon \mathbb{R}^2 \to \mathbb{R}$

$u(x, y)\colon \mathbb{R}^2 \to \mathbb{R}$

$v(x, y)\colon \mathbb{R}^2 \to \mathbb{R}$

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

such that

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

Property of Twice-differentiable Functions

Curl of the gradient field should be zero:

$$\nabla \times \nabla I = I_{yx} - I_{xy} = 0$$

What does that mean intuitively?

- Same result independent of order of differentiation.

$$I_{yx} = I_{xy}$$

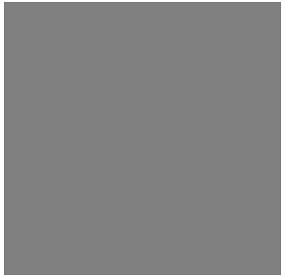
Demonstration

image I

I_x

I_y

ΔI



$\nabla \times \nabla I$

I_{xy}

=

I_{yx}

Property of Twice-differentiable Functions

Curl of the gradient field should be zero:

$$\nabla \times \nabla I = I_{yx} - I_{xy} = 0$$

What does that mean intuitively?

- Same result independent of order of differentiation.

$$I_{yx} = I_{xy}$$

Can you use this property to derive an integrability condition?

Integrable Fields

- Given an arbitrary vector field (u, v), can we always integrate it into a scalar field I?

?

$I(x,y)\colon\ \mathbb{R}^2 \to \mathbb{R}$

$u(x,y)\colon\ \mathbb{R}^2 \to \mathbb{R}$

$v(x,y)\colon\ \mathbb{R}^2 \to \mathbb{R}$

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

such that

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

Only if:

$$\nabla \times \begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} = 0 \Rightarrow \frac{\partial u}{\partial y}(x,y) = \frac{\partial v}{\partial x}(x,y)$$

Vector Field Integration

Two fundamental questions:

- When is integration of a vector field possible?
  - -Use curl to check for equality of mixed partial second derivatives.
- How can integration of a vector field be performed?

Integration Problems

- Reconstructing height fields from gradients

  Applications: shape from shading, photometric stereo

- Manipulating image gradients

  Applications: tonemapping, image editing, matting, fusion, mosaics

- Manipulation of 3D gradients

  Applications: mesh editing, video operations

Key challenge: Most vector fields in applications are not integrable.

- Integration must be done ***approximately***.

Prototypical Integration Problem: Poisson Blending

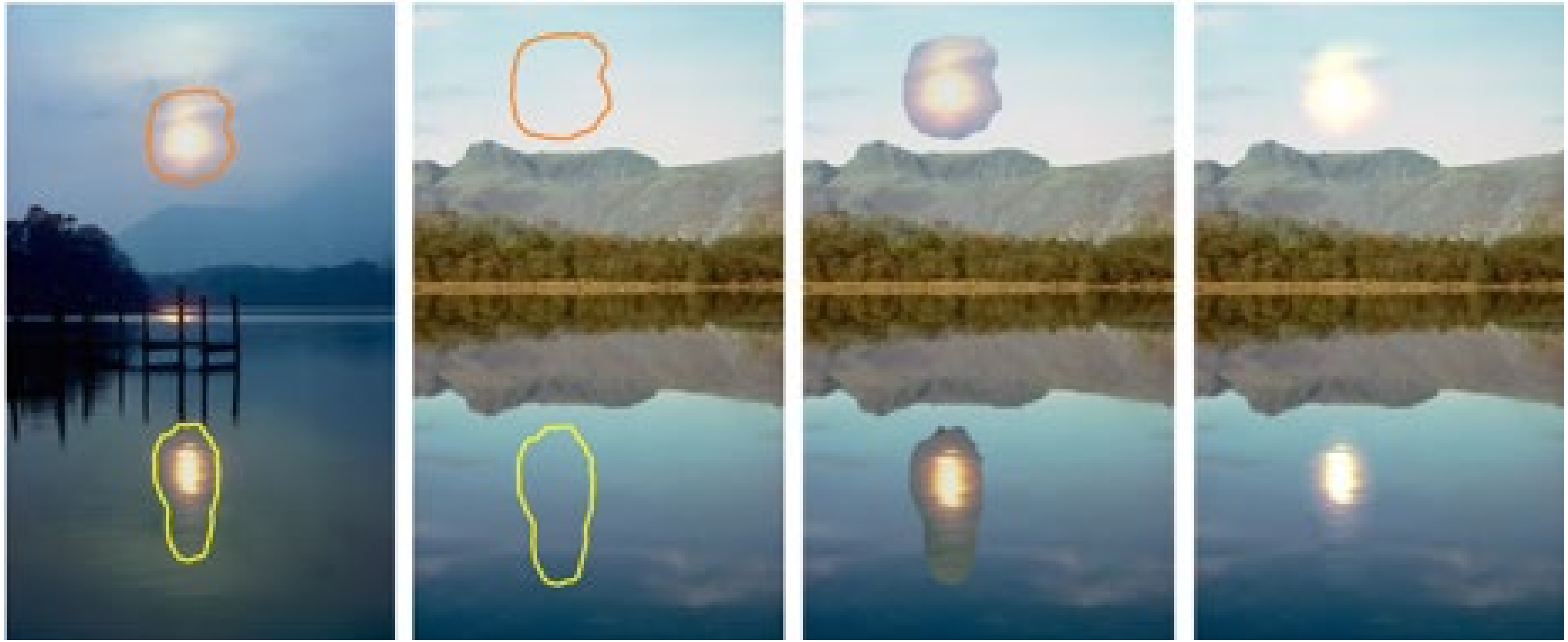
Application: Poisson Blending

Copy-paste

Poisson blending

Key Idea

When blending, retain the gradient information as best as possible



Source Destination Copy-paste Poisson blending

Definitions and Notation

Notation

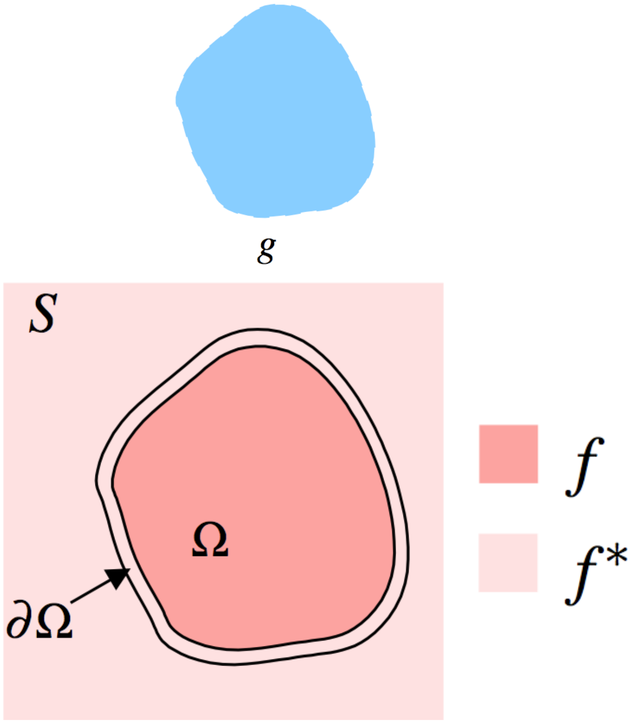
g: source function

S: destination

Ω: destination domain

f: interpolant function

f^*: destination function



Which one is the unknown?

Definitions and Notation

Notation

g: source function

S: destination

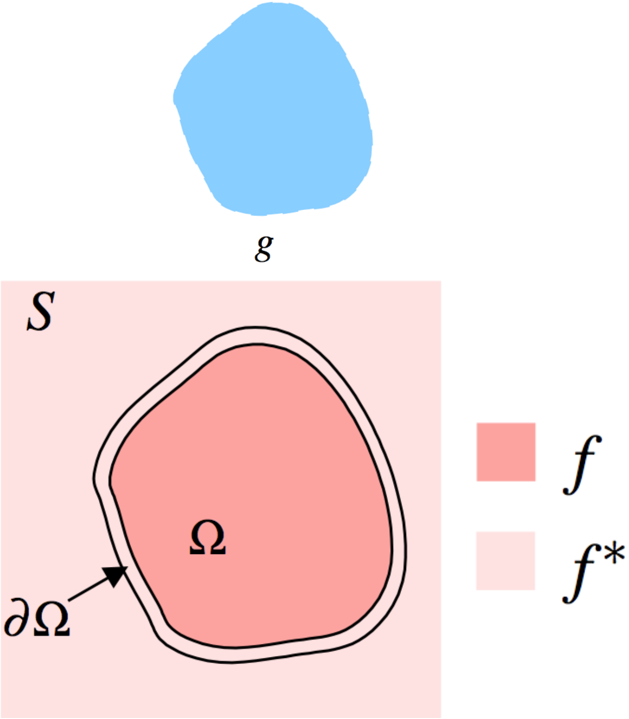
Ω: destination domain

$\boxed{f\text{: interpolant function}}$

f^*: destination function

How should we determine f?

- Should it be similar to g?
- Should it be similar to f^*?



Definitions and Notation

Notation

g: source function

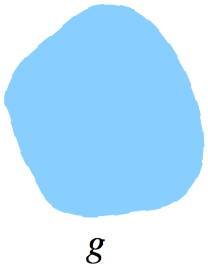
S: destination

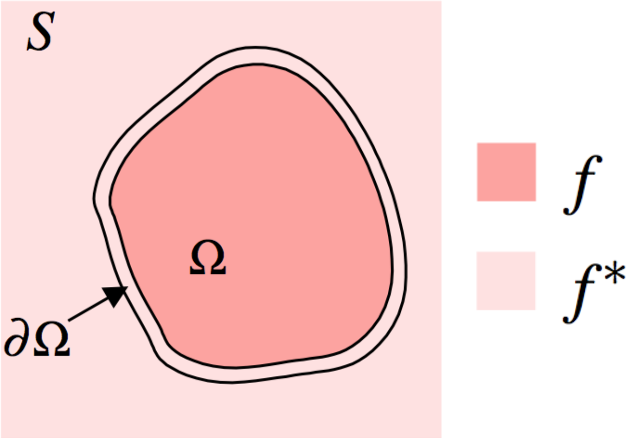
Ω: destination domain

$\boxed{f\text{: interpolant function}}$

f^*: destination function

Find f such that:
- $\nabla f = \nabla g$ inside Ω.
- $f = f^*$ at the boundary $\partial\Omega$





Poisson blending: <u>integrate</u> vector field ∇g with Dirichlet boundary conditions f^*.

Least-Squares Integration and The Poisson Problem

Least-Squares Integration

"Variational" means optimization where the unknown is an entire function

Variational problem

$$\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

what does this term do?

what does this term do?

Recall ...

Nabla operator definition

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

is this known?

$$\mathbf{v} = (u, v)$$

Least-Squares Integration

Why do we need boundary conditions for least-squares integration?

"Variational" means optimization where the unknown is an entire function

Variational problem

$$\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

gradient of f looks like vector field v

f is equivalent to f* at the boundaries

Recall ...

Nabla operator definition

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Yes, this is the vector field we are integrating

$$\mathbf{v} = (u, v)$$

Equivalently

The ***stationary point*** of the variational loss is the solution to the:

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \text{div } \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

what does this term do?

This can be derived using the *Euler-Lagrange equation*.

Recall ...

Laplacian $\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

Divergence $\text{div } \mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$

Input vector field:

$$\mathbf{v} = (u, v)$$

Equivalently

The ***stationary point*** of the variational loss is the solution to the:

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \text{div }\mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

Laplacian of f same as divergence of vector field v

This can be derived using the *Euler-Lagrange equation*.

Recall …

Laplacian $\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

Divergence $\text{div }\mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$

Input vector field:

$$\mathbf{v} = (u, v)$$

Poisson Blending Example...

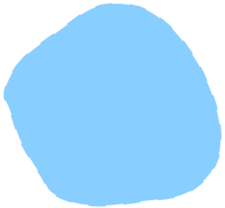
The ***stationary point*** of the variational loss is the solution to the:

Poisson equation (with Dirichlet boundary conditions)

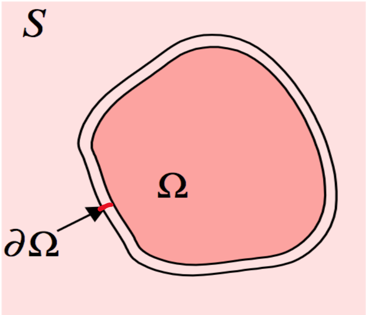
$$\Delta f = \text{div}\ \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

Find f such that:

- $\nabla f = \nabla g$ inside Ω.
- $f = f^*$ at the boundary $\partial\Omega$.



g



What does the input vector field equal in Poisson blending?

$$\mathbf{v} = (u, v) =$$

Poisson Blending Example...

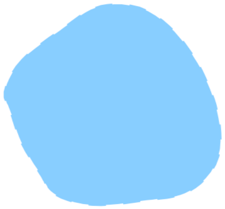
The ***stationary point*** of the variational loss is the solution to the:

Poisson equation (with Dirichlet boundary conditions)

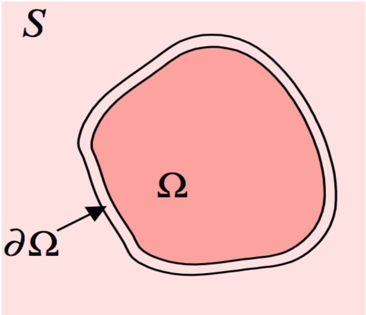
$$\Delta f = \text{div } \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

Find f such that:

- $\nabla f = \nabla g$ inside Ω.
- $f = f^*$ at the boundary $\partial\Omega$.



g



What does the input vector field equal in Poisson blending?

$$\mathbf{v} = (u, v) = \nabla g$$

What does the divergence of the input vector field equal in Poisson blending?

$$\text{div } \mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} =$$

Poisson Blending Example...

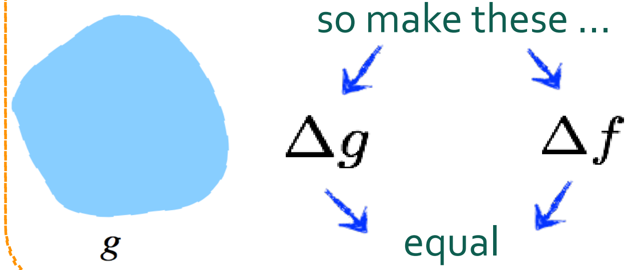
The ***stationary point*** of the variational loss is the solution to the:

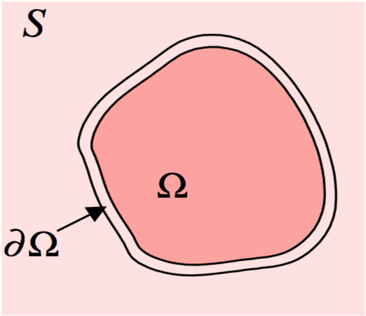
Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \text{div } \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

Find f such that:

- $\nabla f = \nabla g$ inside Ω.
- $f = f^*$ at the boundary $\partial\Omega$.





What does the input vector field equal in Poisson blending?

$$\mathbf{v} = (u, v) = \nabla g$$

What does the divergence of the input vector field equal in Poisson blending?

$$\text{div } \mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = \Delta g$$

Equivalently

The ***stationary point*** of the variational loss is the solution to the:

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \text{div}\ \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

How to solve the Poisson equation?

Recall ...

Laplacian $\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

Divergence $\text{div}\ \mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$

Input vector field:

$$\mathbf{v} = (u, v)$$

Discretization of the Poisson Equation

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \text{div}\,\mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

Recall ...

Laplacian filter

0	1	0
1	-4	1
0	1	0

partial-x derivative filter

1	-1

partial-y derivative filter

1
-1

So for each pixel, do:

$$(\Delta f)(x, y) = (\nabla \cdot \mathbf{v})(x, y)$$

Or for discrete images:

$$-4f(x,y) + f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) = u(x+1,y) - u(x,y) + v(x,y+1) - v(x,y)$$

Discretization of the Poisson Equation

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \operatorname{div} \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

Recall ...

Laplacian filter

0	1	0
1	-4	1
0	1	0

partial-x derivative filter

1	-1

partial-y derivative filter

1
-1

So for each pixel, do (more compact notation):

$$(\Delta f)_p = (\nabla \cdot \mathbf{v})_p$$

Or for discrete images (more compact notation):

$$-4f_p + \sum_{q \in N_p} f_q = (u_x)_p + \left(v_y\right)_p$$

Rewrite this as

linear equation of P variables

$$-4f_p + \sum_{q \in N_p} f_q = (u_x)_p + (v_y)_p$$

one for each pixel p = 1, ..., P

In vector form:

(each pixel adds another 'sparse' row here)

Laplacian matrix →

$$\underbrace{\begin{bmatrix} & & & & & \vdots & & & & & \\ 0 & \cdots & 1 & \cdots & 1 & -4 & 1 & \cdots & 1 & \cdots & 0 \\ & & & & & \vdots & & & & & \end{bmatrix}}_{\boldsymbol{A}} \cdot \underbrace{\begin{bmatrix} f_1 \\ \vdots \\ f_{q_1} \\ \vdots \\ f_{q_2} \\ f_p \\ f_{q_3} \\ \vdots \\ f_{q_4} \\ \vdots \\ f_P \end{bmatrix}}_{\boldsymbol{f}} = \underbrace{\begin{bmatrix} (\nabla \cdot \mathrm{v})_1 \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_1} \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_2} \\ (\nabla \cdot \mathrm{v})_p \\ (\nabla \cdot \mathrm{v})_{q_3} \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_4} \\ \vdots \\ (\nabla \cdot \mathrm{v})_P \end{bmatrix}}_{\boldsymbol{b}}$$

what are the sizes of these?

Laplacian Matrix

For a $m \times n$ image, we can re-organize this matrix into *block tridiagonal form* as:

$$A_{mn\times mn} = \begin{bmatrix} D & I & 0 & 0 & 0 & \cdots & 0 \\ I & D & I & 0 & 0 & \cdots & 0 \\ 0 & I & D & I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & I & D & I & 0 \\ 0 & \cdots & \cdots & 0 & I & D & I \\ 0 & \cdots & \cdots & \cdots & 0 & \boxed{I} & \boxed{D} \end{bmatrix}$$

This requires ordering pixels in column-major order.

$I_{m\times m}$ is the $m \times m$ identity matrix

$$D_{m\times m} = \begin{bmatrix} -4 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & -4 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & -4 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & -4 & 1 & 0 \\ 0 & \cdots & \cdots & 0 & 1 & -4 & 1 \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & -4 \end{bmatrix}$$

Discrete Poisson Equation

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \mathrm{div}\ \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

After discretization, equivalent to:

$$\begin{bmatrix} D & I & 0 & 0 & 0 & \cdots & 0 \\ I & D & I & 0 & 0 & \cdots & 0 \\ 0 & I & D & I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & I & D & I & 0 \\ 0 & \cdots & \cdots & 0 & I & D & I \\ 0 & \cdots & \cdots & \cdots & 0 & I & D \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_{q_1} \\ \vdots \\ f_{q_2} \\ f_p \\ f_{q_3} \\ \vdots \\ f_{q_4} \\ \vdots \\ f_P \end{bmatrix} = \begin{bmatrix} (\nabla \cdot \mathrm{v})_1 \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_1} \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_2} \\ (\nabla \cdot \mathrm{v})_p \\ (\nabla \cdot \mathrm{v})_{q_3} \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_4} \\ \vdots \\ (\nabla \cdot \mathrm{v})_P \end{bmatrix}$$

$\Rightarrow$

Linear system of equations:

$$Af = b$$

How would you solve this?

WARNING: requires special treatment at the borders (target boundary values are same as source)

Solving the Linear System

Convert the system to a linear least-squares problem:

$$E_{LLS} = ||\mathbf{A}f - \boldsymbol{b}||^2$$

Expand the error:

$$E_{LLS} = f^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{A})f - 2f^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{b}) + ||\boldsymbol{b}||^2$$

Minimize the error:

Set derivative to 0 $\quad (\mathbf{A}^{\mathrm{T}}\mathbf{A})f = \mathbf{A}^{\mathrm{T}}\mathbf{b}$

Solve for x $\quad f = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}$

Note: You almost <u>never</u> want to compute the inverse of a matrix.

Discrete the Poisson Equation

Poisson equation (with Dirichlet boundary conditions)

$$\Delta f = \operatorname{div} \mathbf{v} \quad \text{over} \quad \Omega, \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega}$$

After discretization, equivalent to:

$$\begin{bmatrix} D & I & 0 & 0 & 0 & \cdots & 0 \\ I & D & I & 0 & 0 & \cdots & 0 \\ 0 & I & D & I & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & I & D & I & 0 \\ 0 & \cdots & \cdots & 0 & I & D & I \\ 0 & \cdots & \cdots & \cdots & 0 & I & D \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_{q_1} \\ \vdots \\ f_{q_2} \\ f_p \\ f_{q_3} \\ \vdots \\ f_{q_4} \\ \vdots \\ f_P \end{bmatrix} = \begin{bmatrix} (\nabla \cdot \mathrm{v})_1 \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_1} \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_2} \\ (\nabla \cdot \mathrm{v})_p \\ (\nabla \cdot \mathrm{v})_{q_3} \\ \vdots \\ (\nabla \cdot \mathrm{v})_{q_4} \\ \vdots \\ (\nabla \cdot \mathrm{v})_P \end{bmatrix} \Rightarrow$$

Linear system of equations:

$$Af = b$$

Matrix is $P \times P$ → billions of entries

WARNING: requires special treatment at the borders (target boundary values are same as source)

Integration Procedures

- Poisson solver (i.e., least squares integration)
  - + Generally applicable.
  - - Matrices A can become very large.
- Acceleration techniques:
  - + (Conjugate) gradient descent solvers.
  - + Multi-grid approaches.
  - + Pre-conditioning.
  - ...
- Alternative solvers: projection procedures.
  - We will discuss one of these in the next slide.

Today's Topic

- Introduction
- Basics of Gradients and Fields
- Integrable vector fields.
- Poisson blending.

Thank You!

Qilin Sun（孙启霖）

School of Data Science

The Chinese University of Hong Kong, Shenzhen