CMPT 732-G200 Practices for Visual Computing

Ali Mahdavi Amiri

Image Inpainting

- Restoring, manipulating, completing an image in an unnoticeable way.

a b c

Applications

- Image editing

- Nikolai Yezhov is removed.

Applications

- Image restoration.

Poisson and Laplace

Partial Derivatives

- Both Laplace and Poisson equations work based on partial derivatives.

Partial Derivatives

- $f(x, y)$ samples a continuously differentiable function defined on the plane (Image)
  - Its partial derivatives are

$$f_x(x, y) = f(x + 1, y) - f(x, y) \qquad f_y(x, y) = f(x, y + 1) - f(x, y)$$

Partial Derivatives

- $f(x, y)$ samples a continuously differentiable function defined on the plane (Image)
  - Its partial derivatives are

$$f_x(x, y) = f(x+1, y) - f(x, y) \qquad f_y(x, y) = f(x, y+1) - f(x, y)$$

- The vector consisting these derivatives is gradient $\nabla f(x, y)$

$$\nabla f(x, y) = \left[f_x(x, y), f_y(x, y)\right] = f_x(x, y)\vec{\imath} + f_y(x, y)\vec{\jmath}$$

Review of Divergence

- A vector field is a function that assigns a vector to every point in space.
  - Example: external forces in active contour, or partial derivatives of an image.

Review of Divergence

- A vector field is a function that assigns a vector to every point in space.
  - Example: external forces in active contour, or partial derivatives of an image.

- The divergence of a vector field $\mathrm{V} = [u(x, y), v(x, y)]$ is defined as partial derivative of $u(x, y)$ with respect to x plus partial derivative of $v(x, y)$with respect to y.

Review of Divergence

- A vector field is a function that assigns a vector to every point in space.
  - Example: external forces in active contour, or partial derivatives of an image.

- The divergence of a vector field $\mathrm{V} = [u(x,y), v(x,y)]$ is defined as partial derivative of $u(x,y)$ with respect to x plus partial derivative of $v(x,y)$with respect to y.

$$divV = \frac{\partial u(x,y)}{\partial x} + \frac{\partial v(x,y)}{\partial y}$$

Example

- Compute the divergence of $V(x, y) = [x^3, xy] = x^3\vec{\imath} + xy\vec{\jmath}$.

$$divV = \frac{\partial u(x, y)}{\partial x} + \frac{\partial v(x, y)}{\partial y}$$

Example

- Compute the divergence of $V(x, y) = [x^3, xy] = x^3\vec{i} + xy\vec{j}$.

$$divV = \frac{\partial u(x,y)}{\partial x} + \frac{\partial v(x,y)}{\partial y} \qquad u(x,y) = x^3 \qquad v(x,y) = xy$$

Example

- Compute the divergence of $V(x, y) = [x^3, xy] = x^3\vec{\imath} + xy\vec{\jmath}$.

$$divV = \frac{\partial u(x, y)}{\partial x} + \frac{\partial v(x, y)}{\partial y} \qquad u(x, y) = x^3 \qquad v(x, y) = xy$$

$$divV = 3x^2 + x$$

Question?

- What is the divergence of gradient $f(x, y)$?

Question?

- What is the divergence of gradient $f(x, y)$?

Question?

- What is the divergence of gradient $f(x, y)$?

Question?

- What is the divergence of gradient $f(x, y)$?

Question?

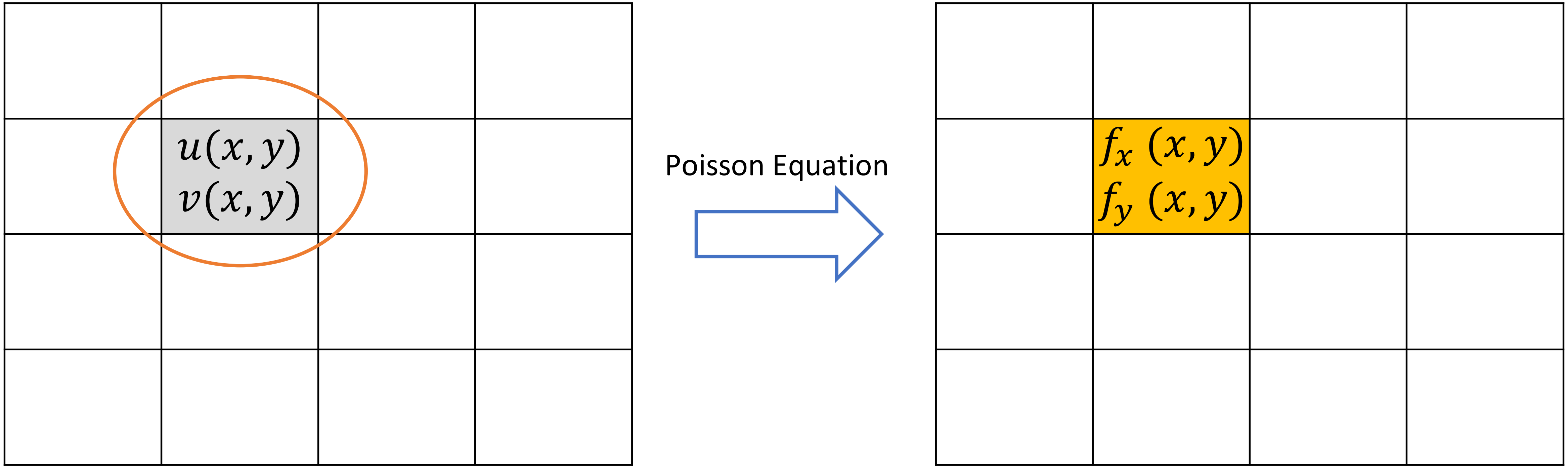
- What is the divergence of gradient $f(x, y)$?

Question?

- What is the divergence of gradient $f(x, y)$?

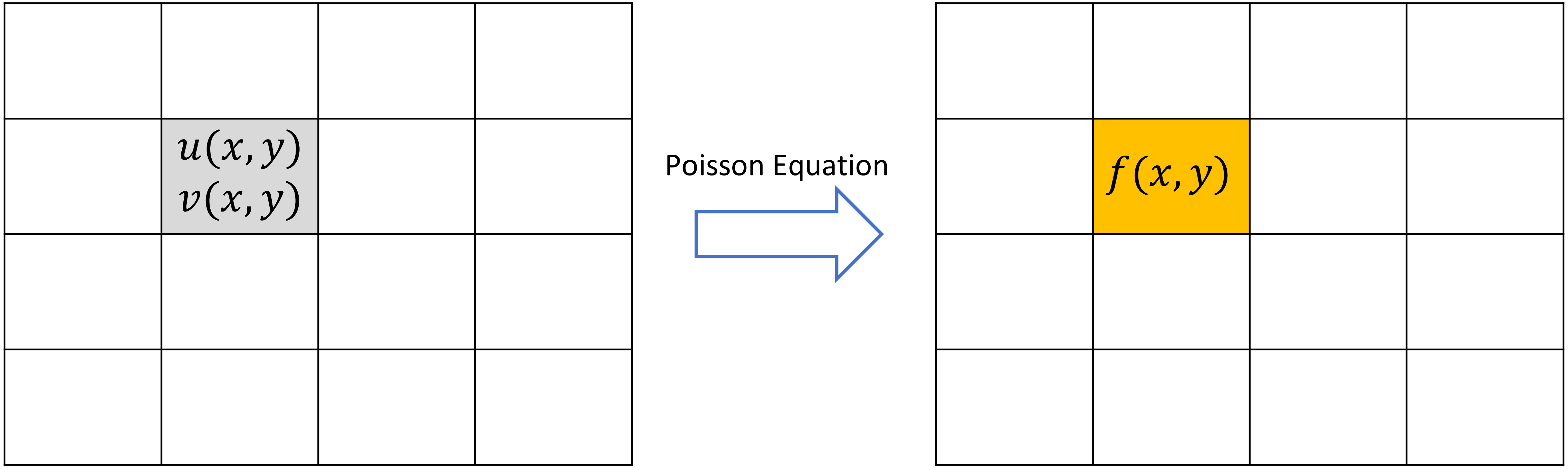
Poisson Equation

- Given $f_x(x, y) = u(x, y)$ and $f_y(x, y) = v(x, y)$, we are looking to reconstruct $f(x, y)$ through Poisson equation.



Poisson Equation

- For each pixel (x,y), there are two constraints $u(x,y)$ and $v(x,y)$. Therefore there is no definite answer and it is overconstrained.



Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x, y) - u(x, y))^2 + \left(f_y(x, y) - v(x, y)\right)^2$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x, y) - u(x, y))^2 + \left(f_y(x, y) - v(x, y)\right)^2$$

- Differentiate and equal to zero

$$\begin{gathered} f(x+1, y) - 2f(x, y) + f(x-1, y) + u(x-1, y) - u(x, y) + \\ f(x, y+1) - 2f(x, y) + f(x, y-1) + v(x, y-1) - v(x, y) = 0 \end{gathered}$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)} + u(x-1,y) - u(x,y) +$$
$$f(x,y+1) - 2f(x,y) + f(x,y-1) + v(x,y-1) - v(x,y) = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_{f} \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

f_{xx}

$$\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)} + u(x-1,y) - u(x,y) +$$
$$f(x,y+1) - 2f(x,y) + f(x,y-1) + v(x,y-1) - v(x,y) = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_{f} \sum_{x,y} (f_x(x, y) - u(x, y))^2 + \left(f_y(x, y) - v(x, y)\right)^2$$

- Differentiate and equal to zero

f_{xx}

$$\boxed{f(x+1, y) - 2f(x, y) + f(x-1, y)} + \boxed{u(x-1, y) - u(x, y)} +$$
$$f(x, y+1) - 2f(x, y) + f(x, y-1) + v(x, y-1) - v(x, y) = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\underbrace{\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)}}_{f_{xx}} + \underbrace{\boxed{u(x-1,y) - u(x,y)}}_{-u_x} +$$
$$f(x,y+1) - 2f(x,y) + f(x,y-1) + v(x,y-1) - v(x,y) = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\underbrace{f(x+1,y) - 2f(x,y) + f(x-1,y)}_{f_{xx}} + \underbrace{u(x-1,y) - u(x,y)}_{-u_x} +$$
$$\boxed{f(x,y+1) - 2f(x,y) + f(x,y-1)} + v(x,y-1) - v(x,y) = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\underbrace{\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)}}_{f_{xx}} + \overbrace{\boxed{u(x-1,y) - u(x,y)}}^{-u_x} +$$
$$\underbrace{\boxed{f(x,y+1) - 2f(x,y) + f(x,y-1)}}_{f_{yy}} + v(x,y-1) - v(x,y) = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_{f} \sum_{x,y} (f_x(x, y) - u(x, y))^2 + \left(f_y(x, y) - v(x, y)\right)^2$$

- Differentiate and equal to zero

$$\underbrace{f(x+1, y) - 2f(x, y) + f(x-1, y)}_{f_{xx}} + \overbrace{u(x-1, y) - u(x, y)}^{-u_x} +$$

$$\underbrace{f(x, y+1) - 2f(x, y) + f(x, y-1)}_{f_{yy}} + \boxed{v(x, y-1) - v(x, y)} = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_{f} \sum_{x,y} (f_x(x, y) - u(x, y))^2 + \left(f_y(x, y) - v(x, y)\right)^2$$

- Differentiate and equal to zero

$$\underbrace{f(x+1, y) - 2f(x, y) + f(x-1, y)}_{f_{xx}} + \underbrace{u(x-1, y) - u(x, y)}_{-u_x} +$$

$$\underbrace{f(x, y+1) - 2f(x, y) + f(x, y-1)}_{f_{yy}} + \underbrace{v(x, y-1) - v(x, y)}_{-v_y} = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_{f} \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\underbrace{f(x+1,y) - 2f(x,y) + f(x-1,y)}_{f_{xx}} + \underbrace{u(x-1,y) - u(x,y)}_{-u_x} +$$

$$\underbrace{f(x,y+1) - 2f(x,y) + f(x,y-1)}_{f_{yy}} + \underbrace{v(x,y-1) - v(x,y)}_{-v_y} = 0$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_f \sum_{x,y} \left(f_x(x,y) - u(x,y)\right)^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\begin{aligned} &\underbrace{\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)}}_{f_{xx}} + \underbrace{\boxed{u(x-1,y) - u(x,y)}}_{-u_x} + \\ &\underbrace{\boxed{f(x,y+1) - 2f(x,y) + f(x,y-1)}}_{f_{yy}} + \underbrace{\boxed{v(x,y-1) - v(x,y)}}_{-v_y} = 0 \end{aligned}$$

$$\boxed{f_{xx} + f_{yy}} \rightarrow \Delta f(x,y)$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)} + \boxed{u(x-1,y) - u(x,y)} +$$
$$\boxed{f(x,y+1) - 2f(x,y) + f(x,y-1)} + \boxed{v(x,y-1) - v(x,y)} = 0$$

$$\boxed{\begin{array}{c} f_{xx} \\ + \\ f_{yy} \end{array}} \qquad \boxed{\begin{array}{c} -u_x \\ + \\ -v_y \end{array}}$$

$$\Delta f(x,y)$$

Least Square

- We need an optimization to find f whose gradients are close to $u(x,y)$ and $v(x,y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\boxed{f_{xx} + f_{yy}} \qquad \boxed{-u_x + -v_y}$$

$$\boxed{f(x+1,y) - 2f(x,y) + f(x-1,y)} + \boxed{u(x-1,y) - u(x,y)} +$$

$$\boxed{f(x,y+1) - 2f(x,y) + f(x,y-1)} + \boxed{v(x,y-1) - v(x,y)} = 0$$

$$\Delta f(x,y) - div[u(x,y), v(x,y)]$$

Poisson Equation

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x,y) - u(x,y))^2 + \left(f_y(x,y) - v(x,y)\right)^2$$

- Differentiate and equal to zero

$$\Delta f(x,y) - div[u(x,y), v(x,y)] = 0$$

Poisson Equation

- We need an optimization to find f whose gradients are close to $u(x, y)$ and $v(x, y)$.

$$\min_f \sum_{x,y} (f_x(x, y) - u(x, y))^2 + \left(f_y(x, y) - v(x, y)\right)^2$$

- Differentiate and equal to zero

$$\Delta f(x, y) - div[u(x, y), v(x, y)] = 0$$

$$\Delta f(x, y) = div[u(x, y), v(x, y)]$$

Summary

- Poisson equation $\Delta f(x, y) = div[u(x, y), v(x, y)]$
  - Tries to find a function whose gradient is close to a vector field.

Summary

- Poisson equation $\Delta f(x, y) = div[u(x, y), v(x, y)]$
  - Tries to find a function whose gradient is close to a vector field.

- Laplace Equation $\Delta f(x, y) = 0$
  - Tries to find a function with minimal gradient field.

Laplace

- We explain Laplace First as it is easier

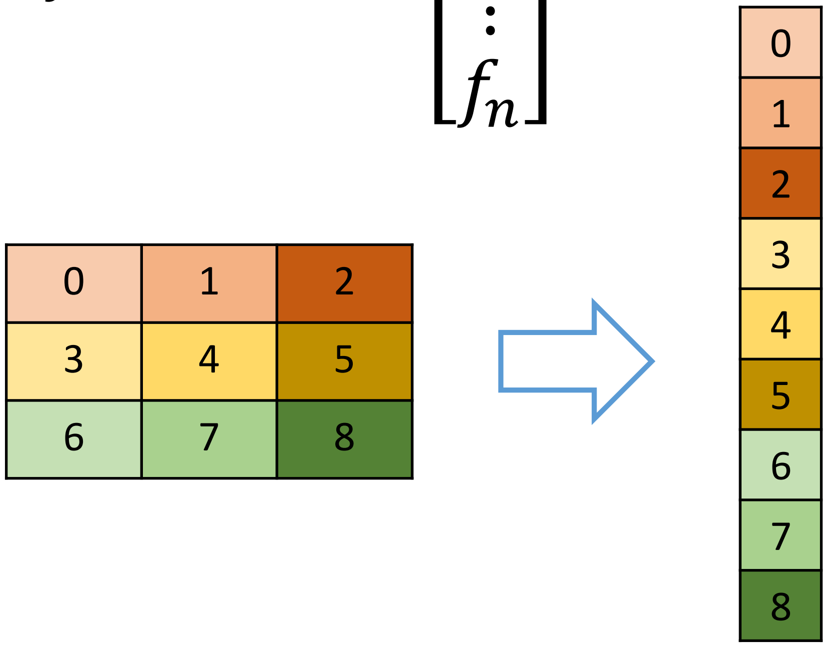
$$\Delta f(x,y) = 0 \qquad f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Laplace

- We explain Laplace First as it is easier

$\Delta f(x, y) = 0 \quad f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$

- Convert image $f(x, y)$ to vector $\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}$ in which $n = h \times w$.



Laplace

- We explain Laplace First as it is easier

$$\Delta f(x, y) = 0 \qquad f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

- Now we can convert $\Delta f(x, y) = 0$ into a matrix notation:

$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

Laplace

- Let's say, we don't have any known values for f_i, what is the answer?

$$\begin{pmatrix} 1 & & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & & \vdots & & & \\ & & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

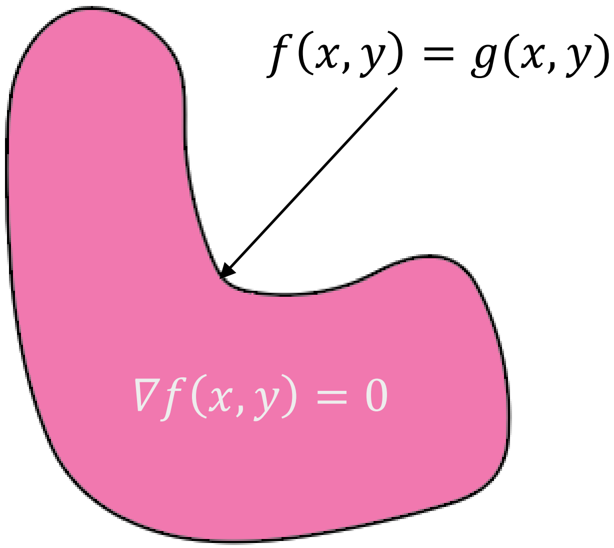
Laplace

- Therefore, we need to have some known values or boundary

$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

Boundaries

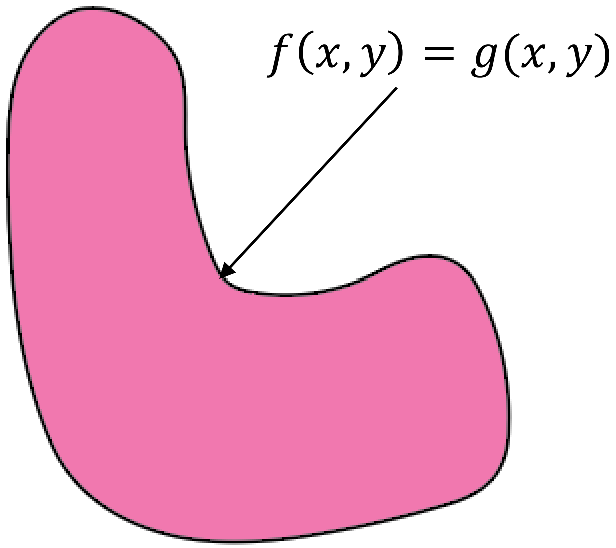
- We want our function $f(x,y)$ interpolates known values $g(x,y)$.



Laplace

- Therefore, our equation is changed

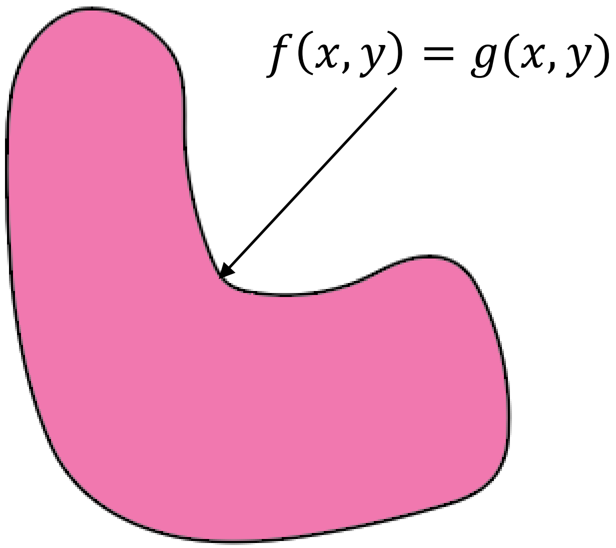
$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$



Laplace

- Therefore, our equation is changed

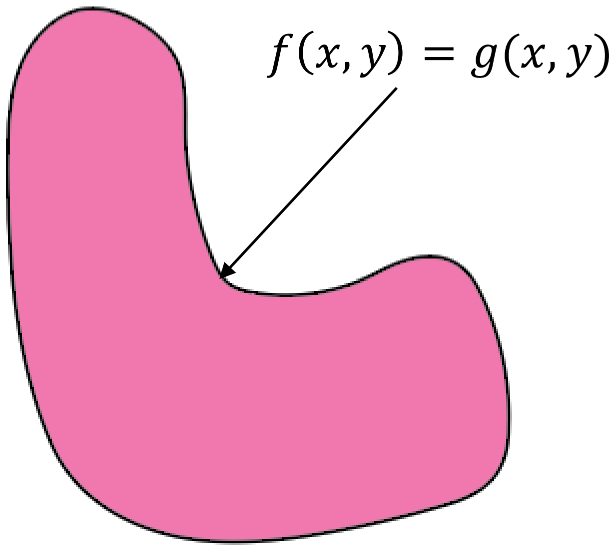
$$\underbrace{\boxed{\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix}}}_{\mathbf{L}} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$



Laplace

- We need to interpolate values at the boundary

$$\underbrace{\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & \vdots & & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix}}_{\mathbf{L}} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

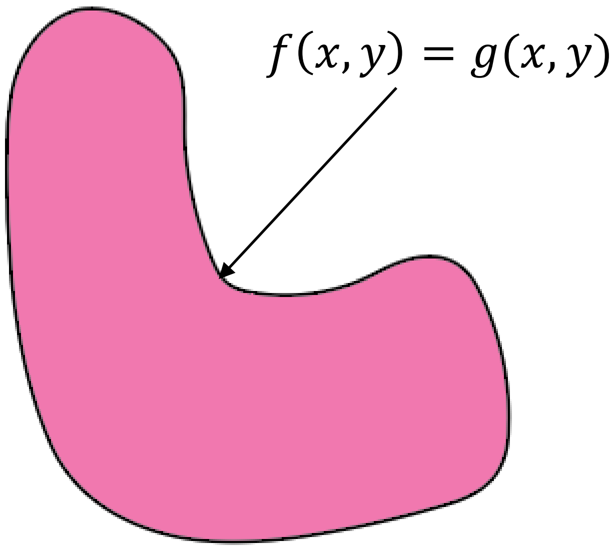


Laplace

- We need to interpolate values at the boundary

$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

$\mathbf{L}$



Laplace

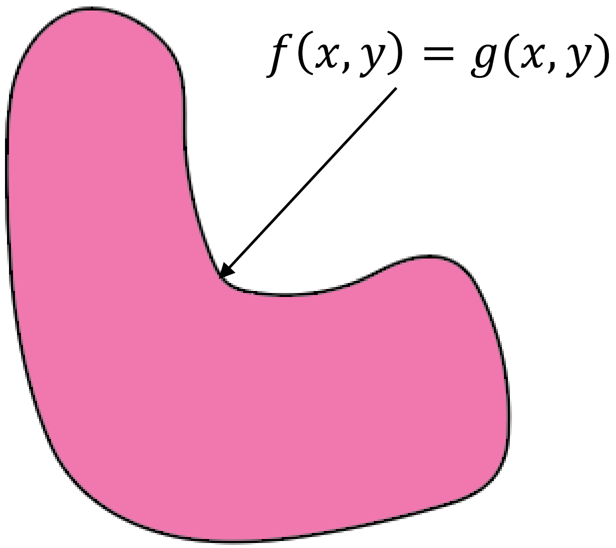
- We need to interpolate values at the boundary

$$\mathbf{L} = \boxed{\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix}} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

$$\Downarrow$$

$$\begin{pmatrix} & L & \\ \hline 0 & | & I_{k\times k} \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ \hline c_1 \\ \vdots \\ c_k \end{pmatrix}$$

$g(x, y)$



Laplace

- Assuming boundary values are k end values.

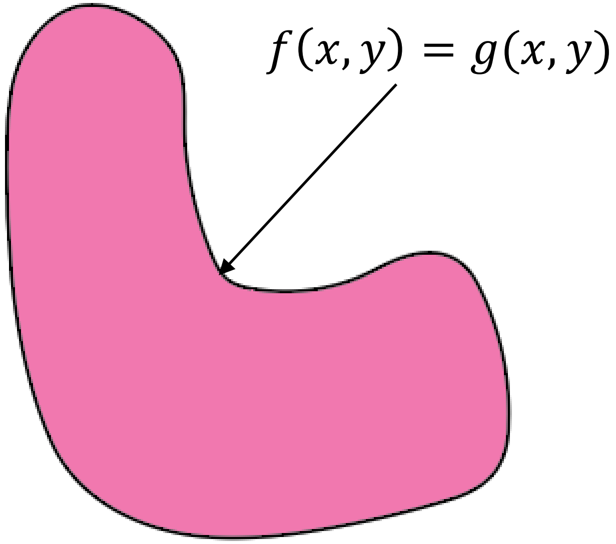
$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots & & \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & & \\ & & 1 & 1 & -4 & 1 & 1 & \cdots & & \\ & & & & \vdots & & & & & \\ & & & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = 0$$

$\mathbf{L}$

$$\Downarrow$$

$$\begin{pmatrix} & L & \\ - - - & - - - & - - - \\ 0 & | & I_{k\times k} \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ - - - \\ c_1 \\ \vdots \\ c_k \end{pmatrix}$$

$g(x,y)$

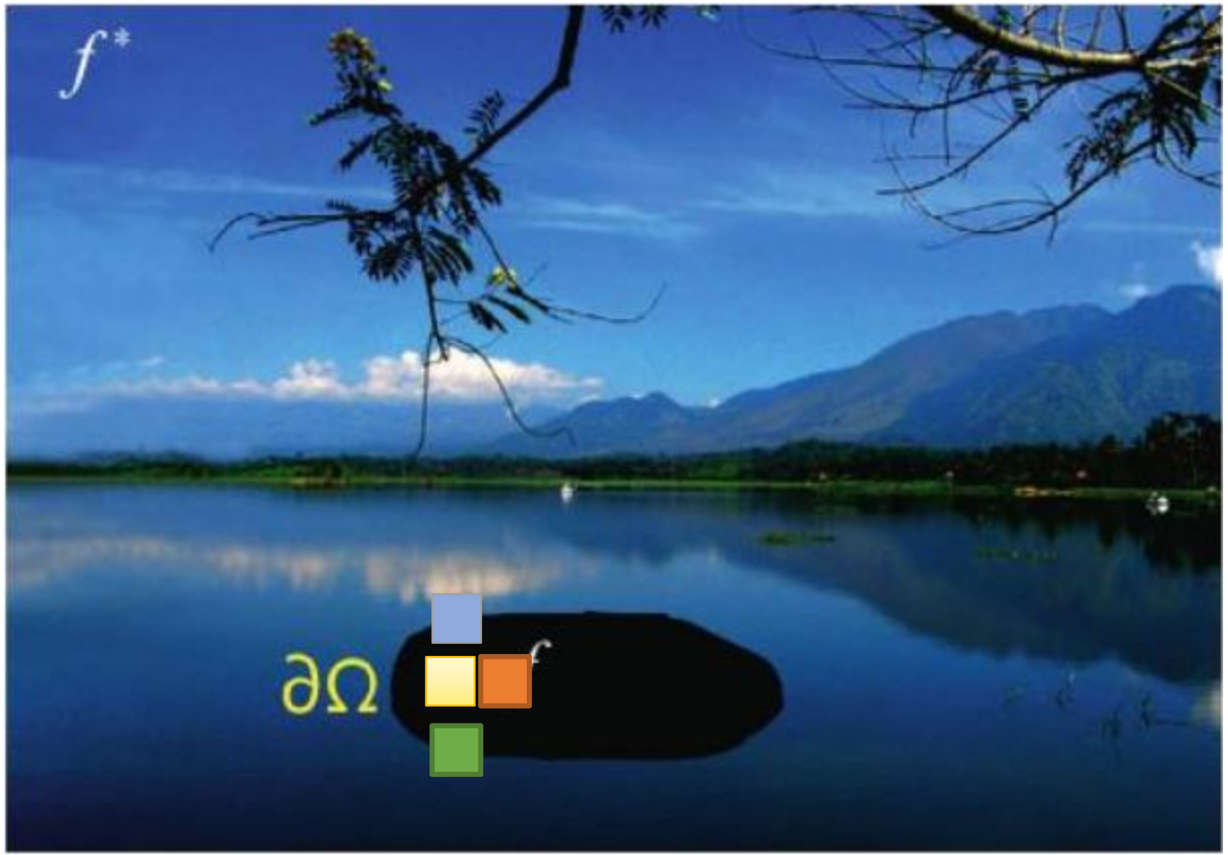


Laplace Image inpainting

- Part of the image is missing, we want to fill the hole.

Unknown values we are looking for

$$\left(\begin{array}{c} L \\ \hline 0 \;|\; I_{k\times k} \end{array}\right)\begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ \hline c_1 \\ \vdots \\ c_k \end{pmatrix}$$

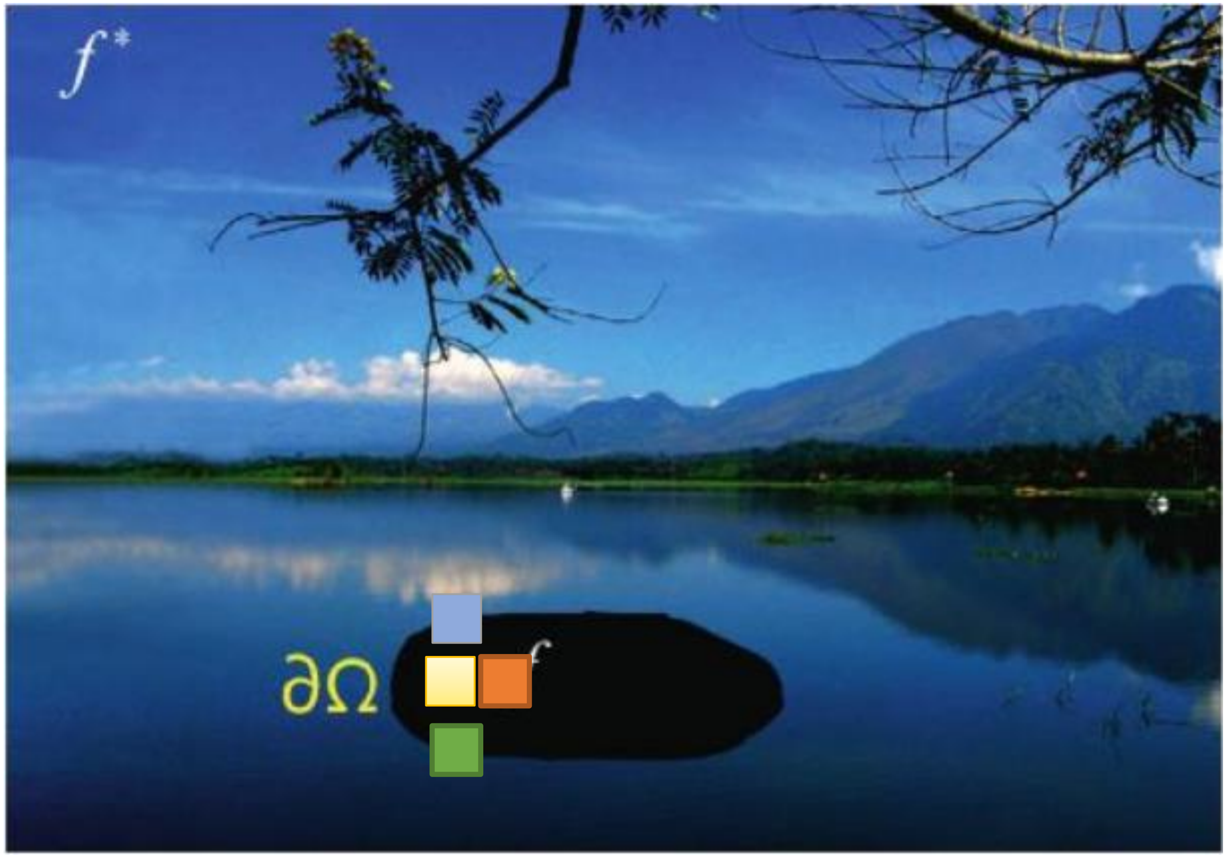


Laplace Image inpainting

- Part of the image is missing, we want to fill the hole.

Unknown values we are looking for

$$\begin{pmatrix} L \\ \hline 0 \;|\; I_{k\times k} \end{pmatrix} \begin{pmatrix} \vdots \\ f_{1+k} \\ \vdots \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ \hline c_1 \\ \vdots \\ c_k \end{pmatrix}$$

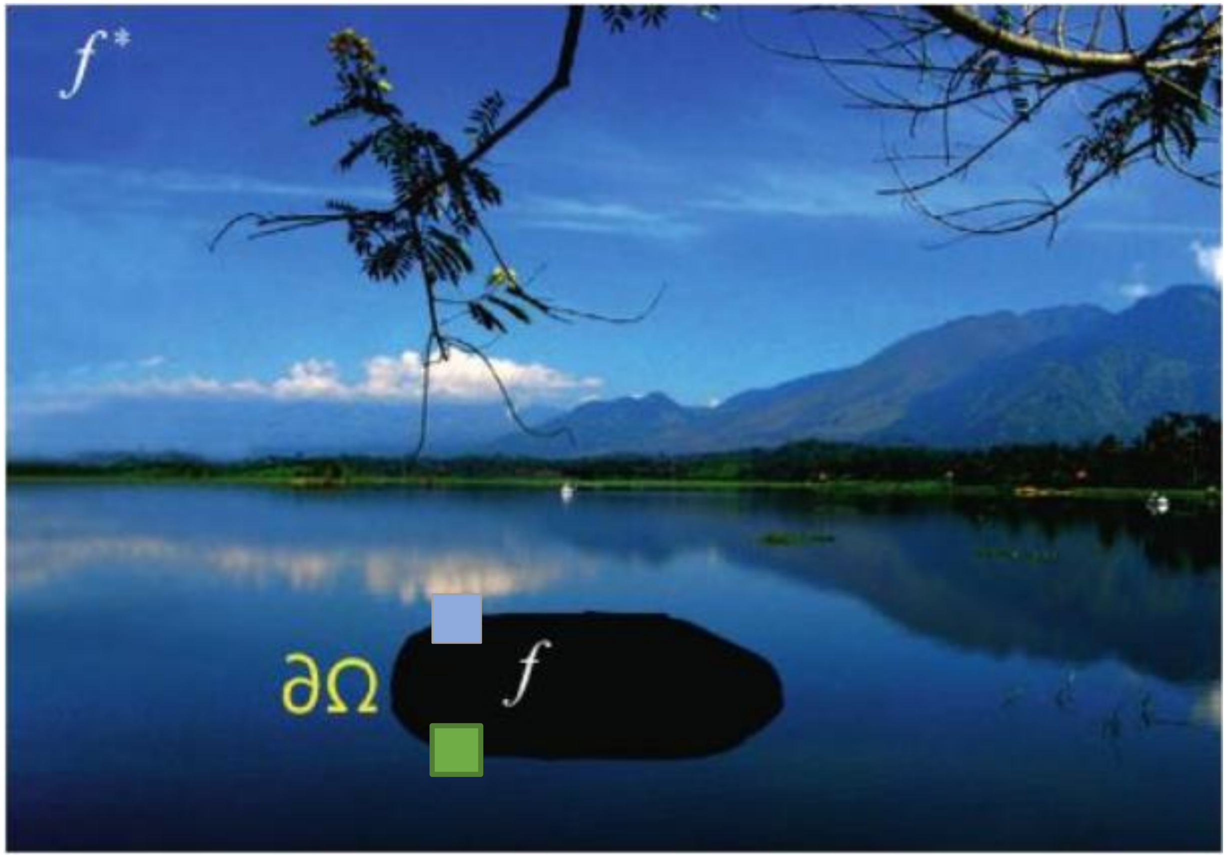


Laplace Image inpainting

- Part of the image is missing, we want to fill the hole.

Known values at the boundary

$$\begin{pmatrix} & L & \\ \hline 0 & | & I_{k\times k} \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ \hline f_1 \\ \vdots \\ f_k \end{pmatrix}$$



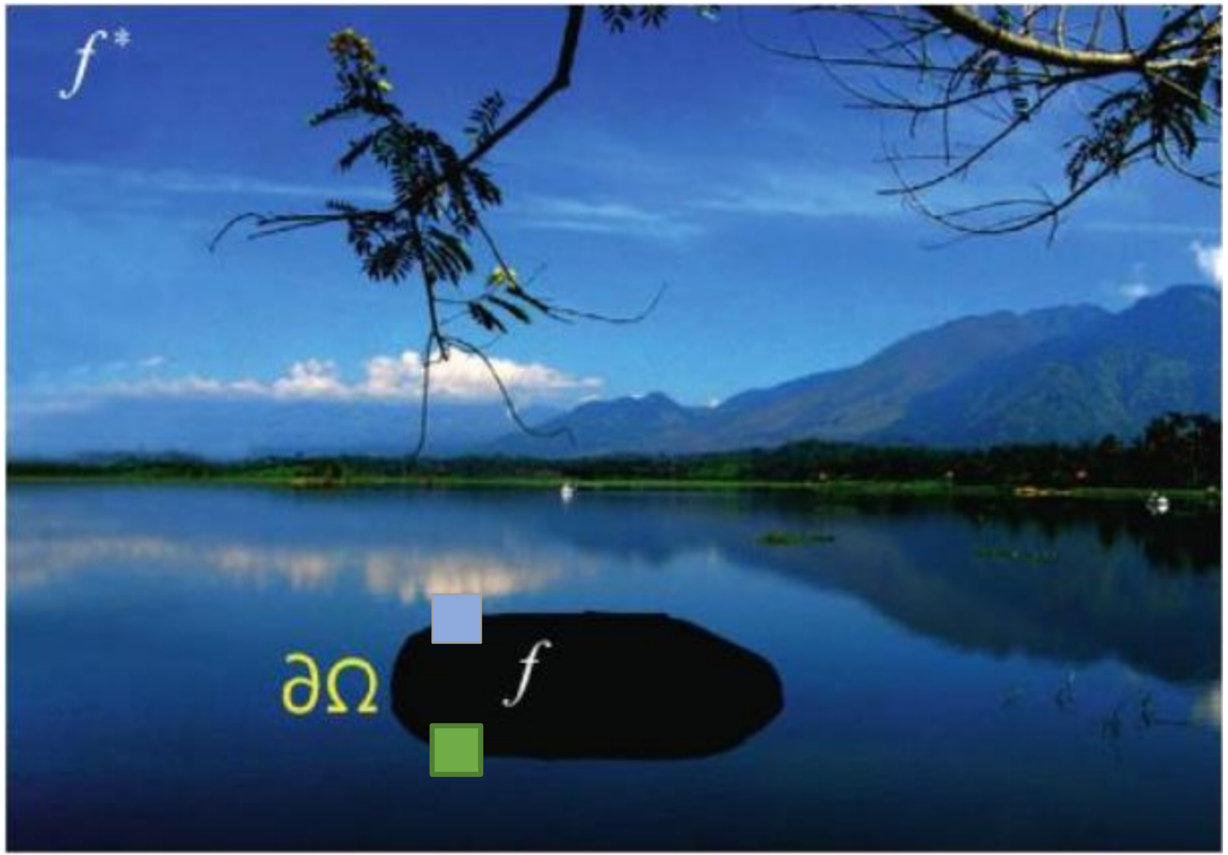
Laplace Image inpainting

- Solving this equation results a smooth hole filling with respect to the values at the boundary.

Tries to smooth the hole

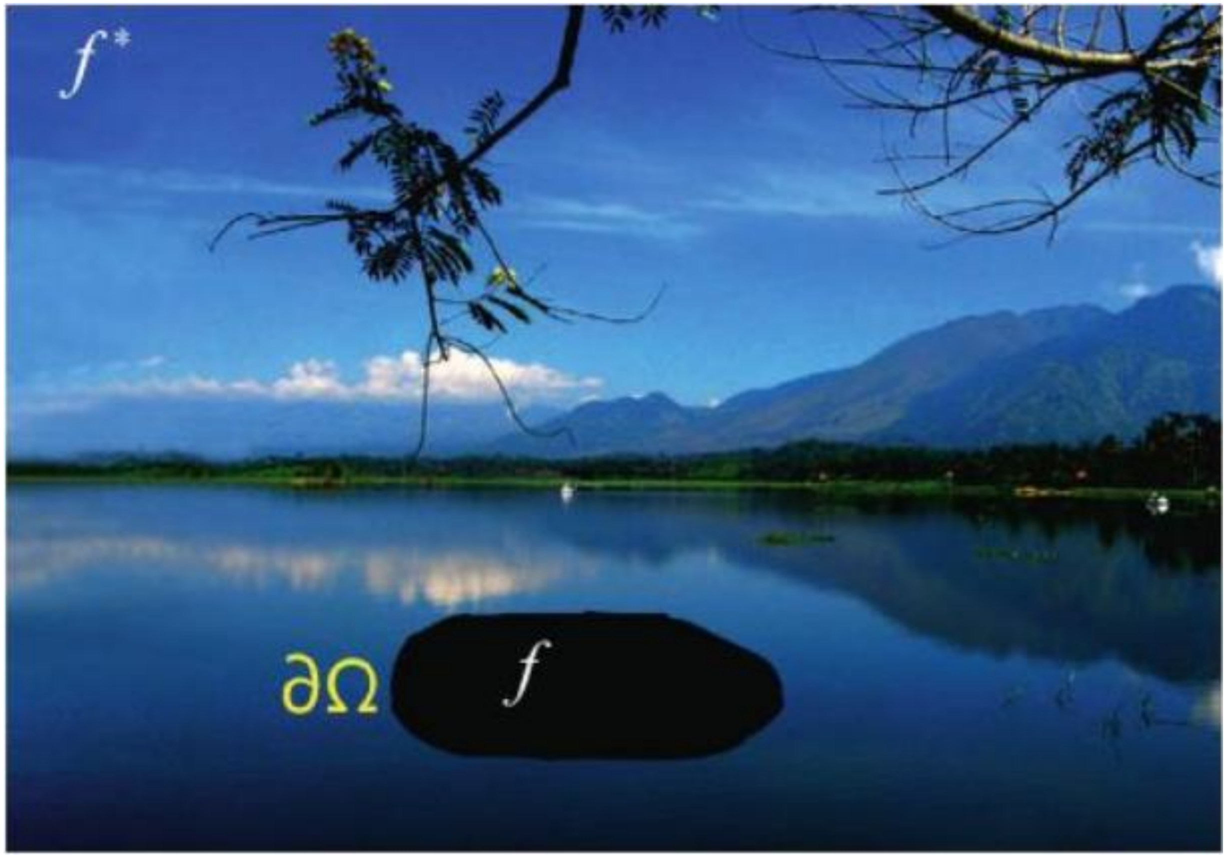
$$\left(\begin{array}{c|c} \multicolumn{2}{c}{\boxed{L}} \\ \hline 0 & \boxed{I_{k\times k}} \end{array}\right)\begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \left(\begin{array}{c} 0 \\ \hline c_1 \\ \vdots \\ c_k \end{array}\right)$$

Tries to interpolate the boundary



Laplace Image inpainting

- Solving this equation results a smooth hole filling with respect to the values at the boundary.



Laplace Image inpainting

- Simple Example
  - We are looking for the value of f_4

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let's form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let’s form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0	1	0	1	-4	1	0	1	0

0

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let’s form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0	1	0	1	-4	1	0	1	0
1	0	0	0	0	0	0	0	0

0

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let's form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0	1	0	1	-4	1	0	1	0
1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0

0

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let’s form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	
0	1	0	1	-4	1	0	1	0	0
1	0	0	0	0	0	0	0	0	
0	1	0	0	0	0	0	0	0	
0	0	1	0	0	0	0	0	0	

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let’s form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0	1	0	1	-4	1	0	1	0
1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0

0

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let’s form the matrix

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	
0	1	0	1	-4	1	0	1	0	0
1	0	0	0	0	0	0	0	0	
0	1	0	0	0	0	0	0	0	
0	0	1	0	0	0	0	0	0	
0	0	0	1	0	0	0	0	0	
0	0	0	0	0	1	0	0	0	

f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Laplace Image inpainting

- Let’s form the matrix

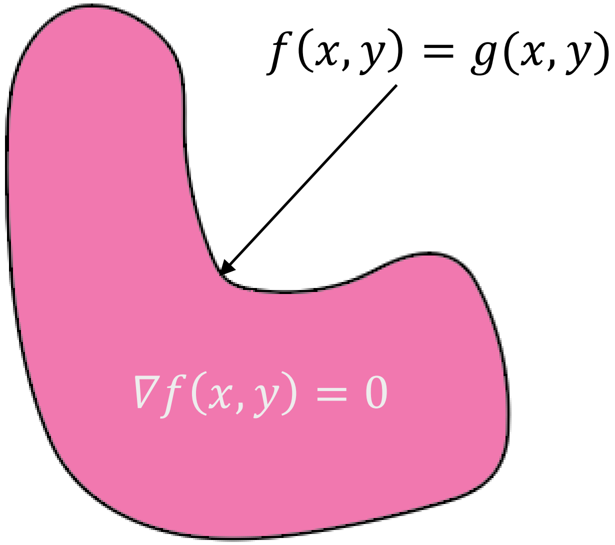
f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0	1	0	1	-4	1	0	1	0
1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0
0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1

0

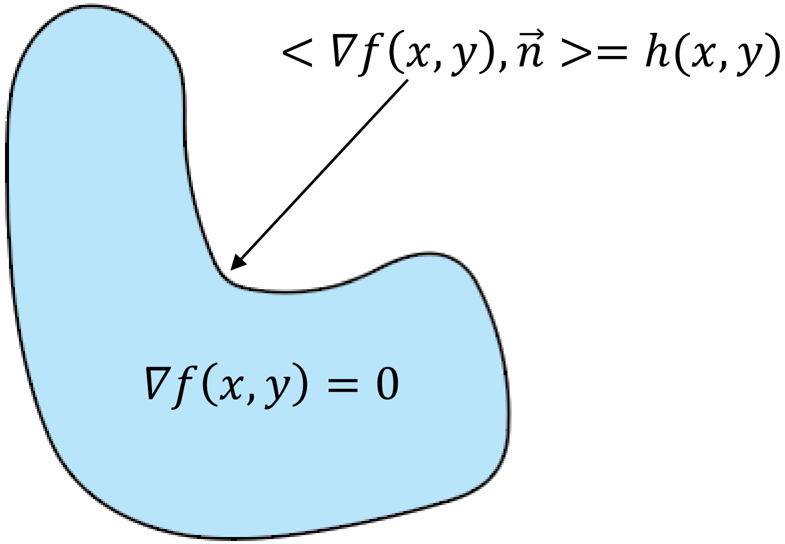
f_0	f_1	f_2
f_3	f_4	f_5
f_6	f_7	f_8

Boundaries

- Another type of boundary constraint exist



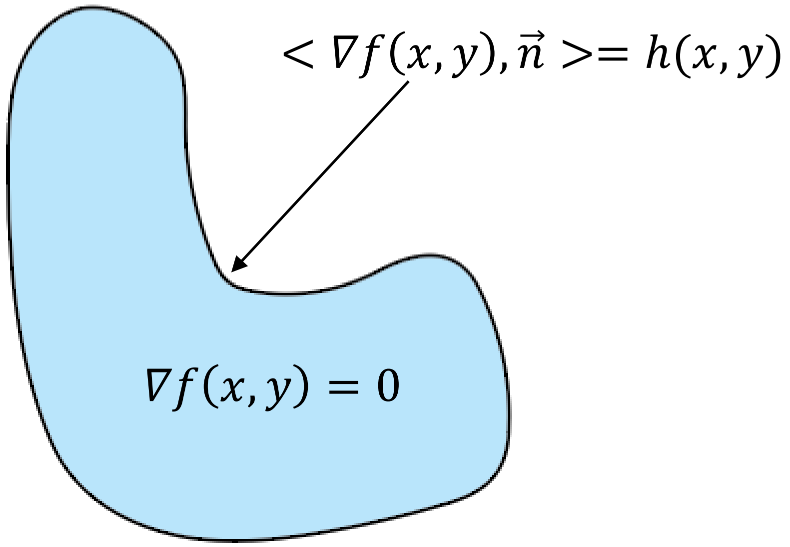
Dirichlet



Neuman

Boundaries

- Another type of boundary constraint exist



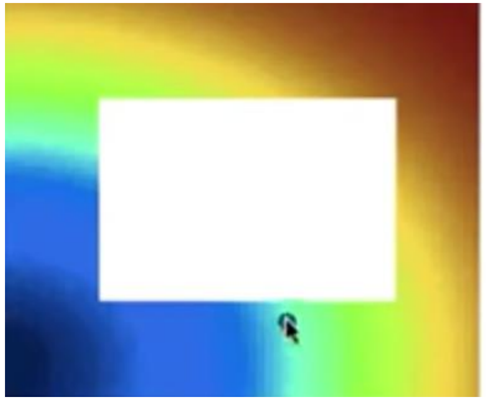
Neuman

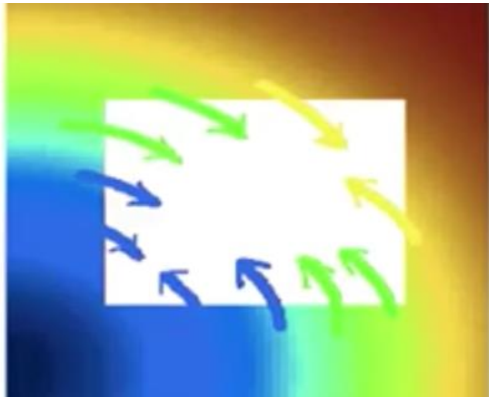
Boundaries

- We want to respect to the flow (direction) of the boundary and that's why we bring vector $\vec{n}$ to the play.

$$< \nabla f(x,y), \vec{n} >= h(x,y)$$

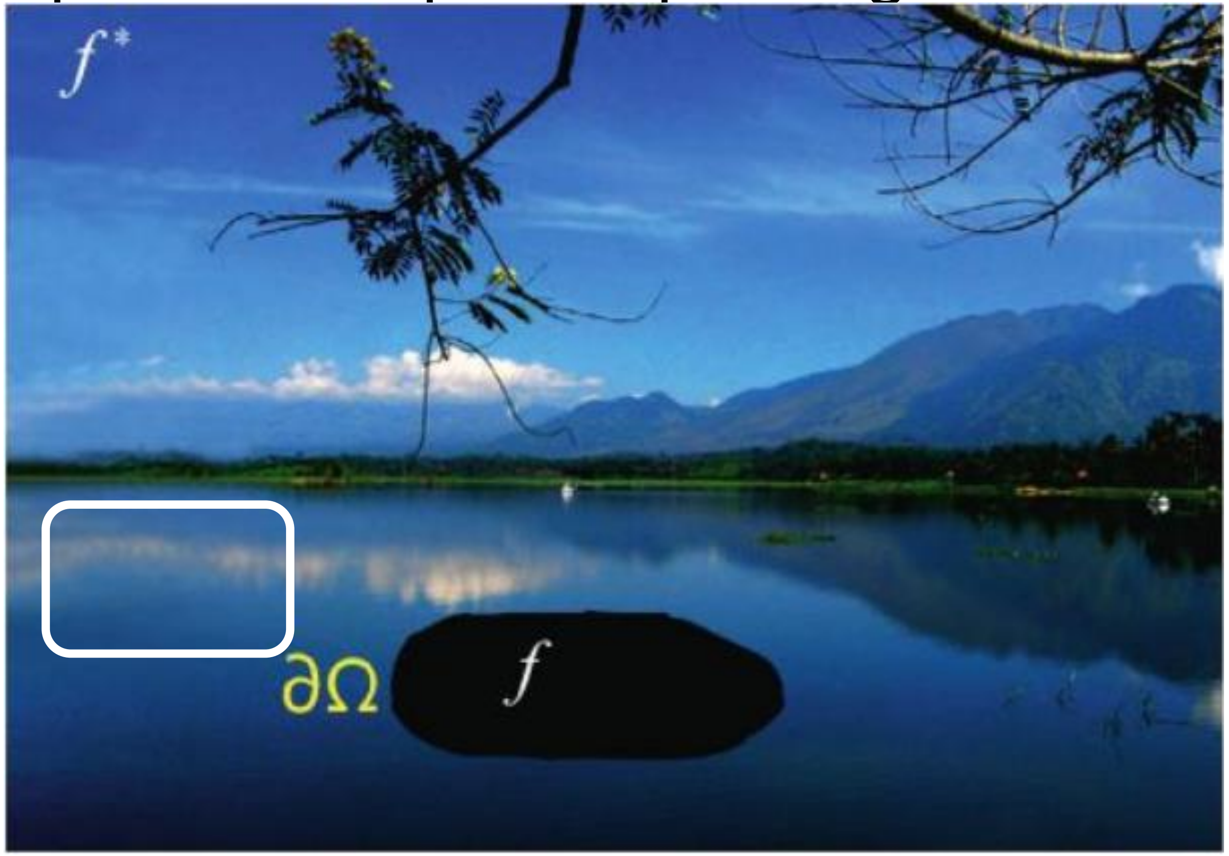
Normal derivative along the boundary





Poisson Image Inpainting

- There are many important details, natural noises, etc that are smoothed out by Laplacian in Laplace inpainting

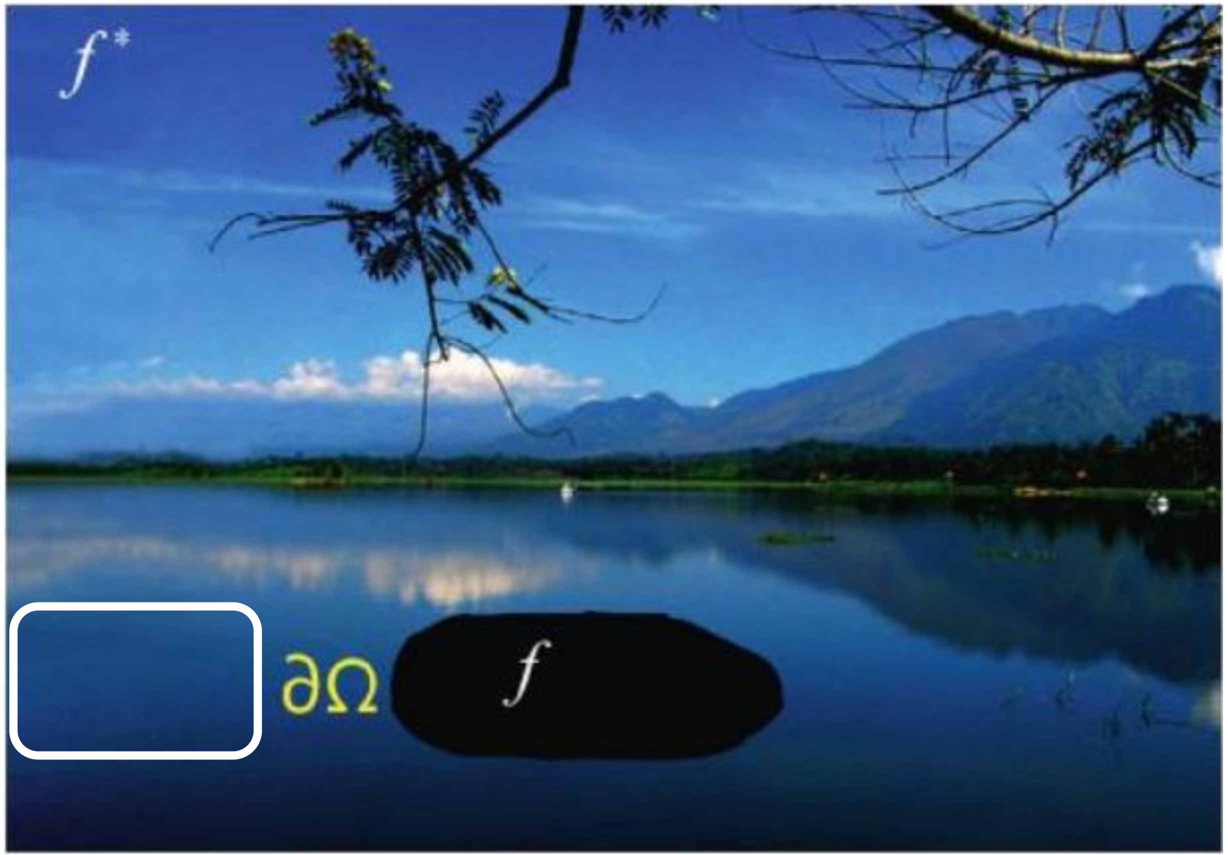


Poisson Image Inpainting

- The idea of Poisson is to borrow such details from a similar patch (given by search or interaction).

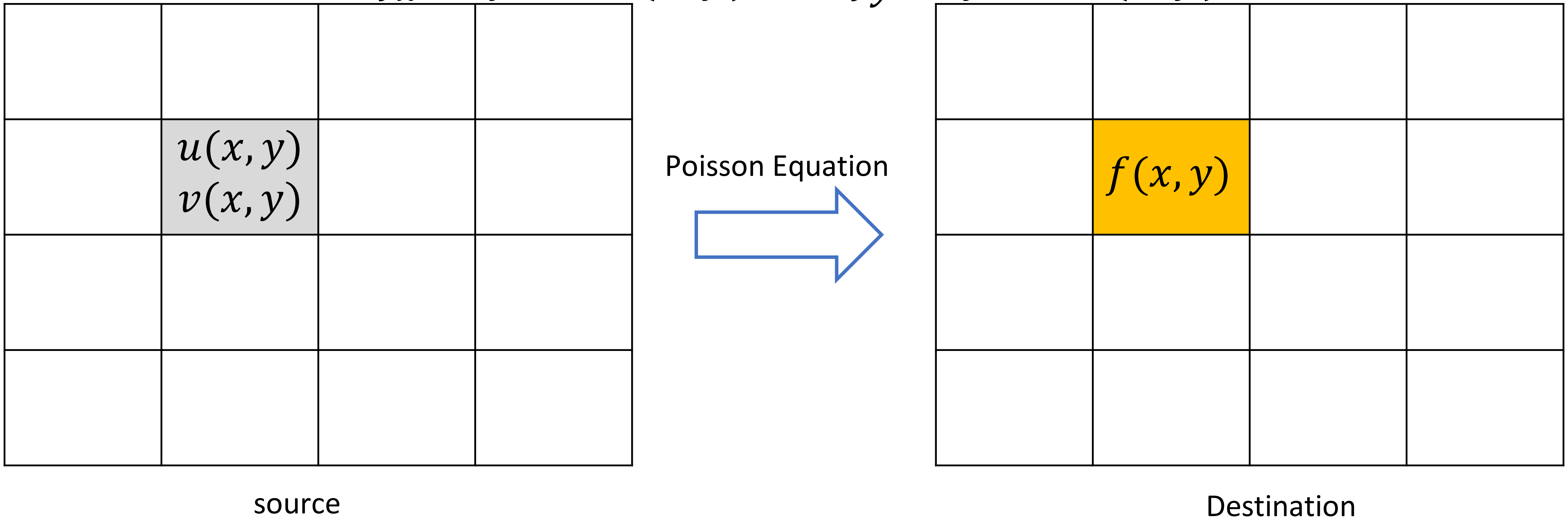
Poisson Image Inpainting

- In Poisson, details are respected by making sure that the reconstructed patch has similar derivative to a given patch source patch.



Poisson Image Inpainting

- In Poisson, details are respected by making sure that the reconstructed patch $f_x(x,y)$ has similar derivative to a given patch source patch. $f_x(x,y) = u(x,y)$ and $f_y(x,y) = v(x,y)$.



Poisson Image Inpainting

- In Poisson, details are respected by making sure that the reconstructed patch $f_x(x, y)$ has similar derivative to a given patch source patch. $f_x(x, y) = u(x, y)$ and $f_y(x, y) = v(x, y)$.

Poisson Image Inpainting

- Review: answer to Poisson equation is

$$\Delta f(x,y) = div[u(x,y), v(x,y)]$$

$$\begin{pmatrix} 1 & & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & & \vdots & & & \\ & & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ div\ \boldsymbol{g} \\ | \end{pmatrix}$$

Poisson Image Inpainting

- Review: answer to Poisson equation is

$$\Delta f(x, y) = div[\boxed{u(x, y), v(x, y)}]$$

Flatten it to a vector

$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots \\ & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots \\ & & 1 & 1 & -4 & 1 & 1 & \cdots \\ & & & & \vdots & & & \\ & & & 1 & 1 & -4 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ div\ \boldsymbol{g} \\ | \end{pmatrix}$$

Poisson Image Inpainting

- Similar to Laplace equation, we need boundary condition

$$\begin{pmatrix} L \\ -------- \\ 0 \quad | \quad I_{k\times k} \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ div\ \boldsymbol{g} \\ | \\ --- \\ c_1 \\ \vdots \\ c_k \end{pmatrix}$$

Poisson Image Inpainting

- Similar to Laplace equation, we need boundary condition

What should be the value of $\boldsymbol{g}$?

$$\left(\begin{array}{ccc} & L & \\ \hline 0 & | & I_{k\times k}\end{array}\right)\begin{pmatrix} f_1\\ \vdots \\ f_{n-k}\\ \vdots \\ f_n\end{pmatrix} = \left(\begin{array}{c} | \\ \boxed{div\ \boldsymbol{g}} \\ | \\ \hline c_1 \\ \vdots \\ c_k\end{array}\right)$$

Poisson Image Inpainting

- Similar to Laplace equation, we need boundary condition

$$
\left(\begin{array}{ccc} & L & \\ \hline 0 & | & I_{k\times k} \end{array}\right)
\begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix}
=
\left(\begin{array}{c} | \\ \boxed{div\ \boldsymbol{g}} \\ | \\ \hline c_1 \\ \vdots \\ c_k \end{array}\right)
$$

What should be the value of $\boldsymbol{g}$?

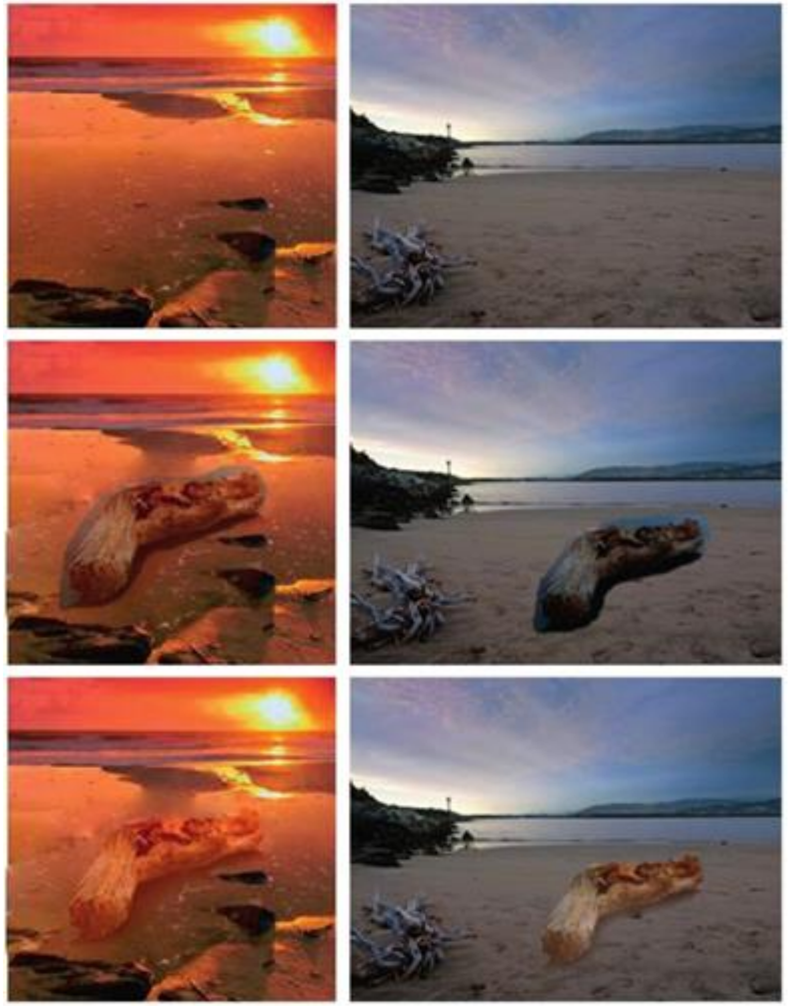
Poisson Image Blending

source

Destination

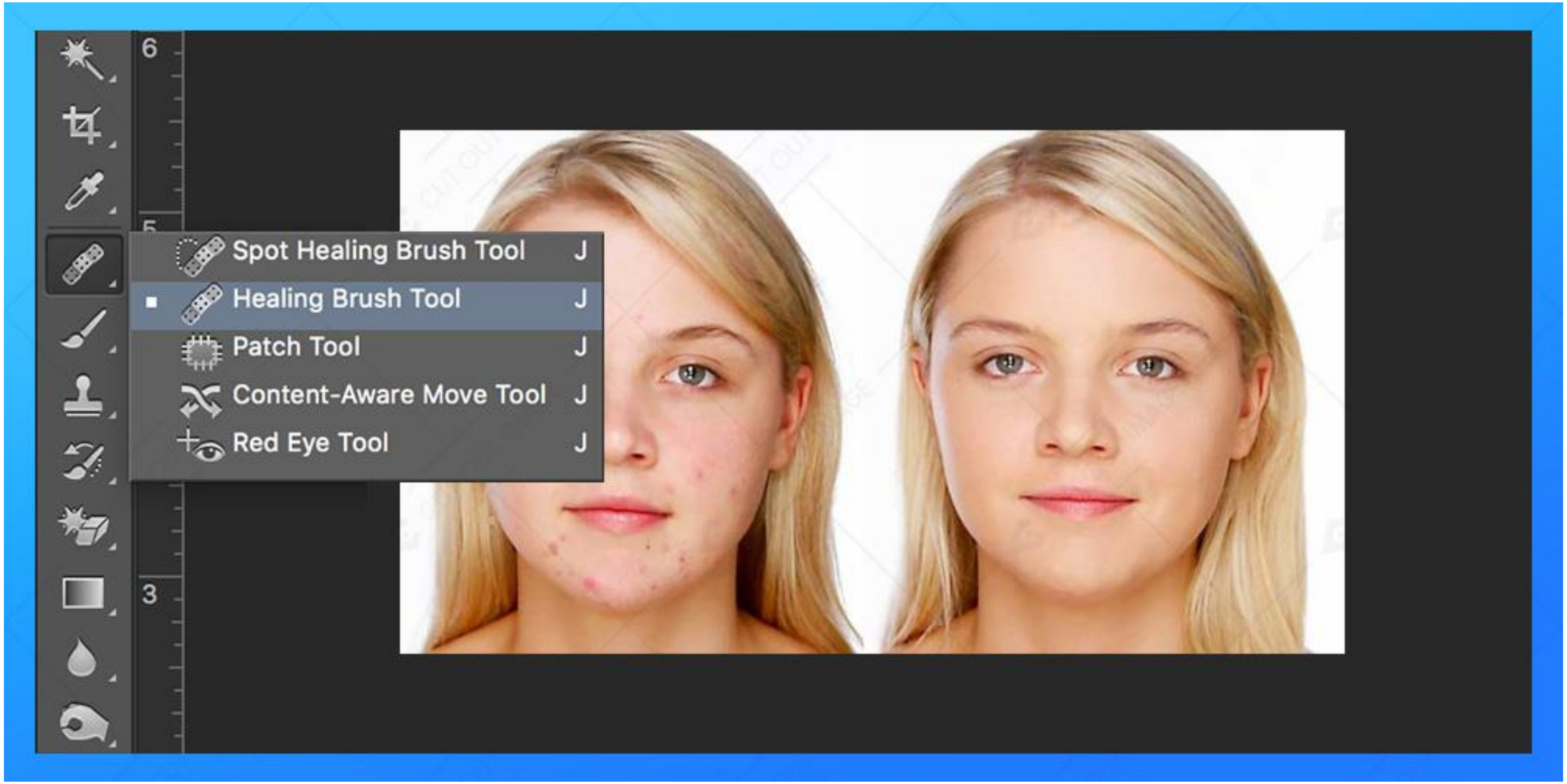
Copy Paste

Poisson

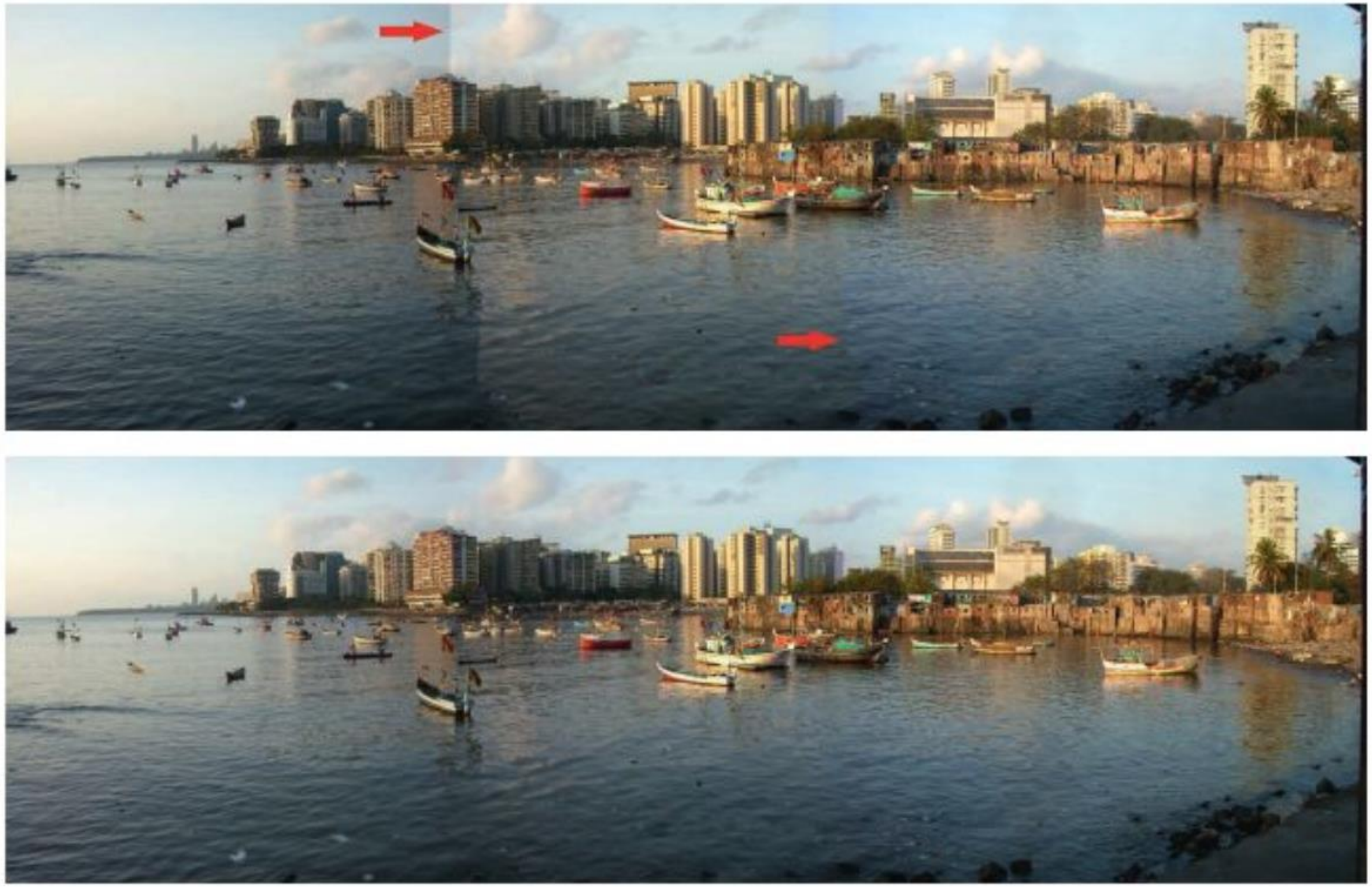


Healing Brush (photo shop)

Healing Brush (photo shop)



Panorama Stitching



How to solve?

$$\begin{pmatrix} L \\ ------- - \\ 0 \quad | \quad I_{k\times k} \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ -- - \\ c_1 \\ \vdots \\ c_k \end{pmatrix}$$

How to solve?

$$\underbrace{\begin{pmatrix} L \\ \hline 0 \;\big|\; I_{k\times k} \end{pmatrix}}_{A} \begin{pmatrix} f_1 \\ \vdots \\ f_{n-k} \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \\ \hline c_1 \\ \vdots \\ c_k \end{pmatrix}$$

How to solve?

$$\underbrace{\left(\begin{array}{ccc} & L & \\ \hline 0 & | & I_{k\times k}\end{array}\right)}_{A}\begin{pmatrix} f_1\\ \vdots \\ f_{n-k}\\ \vdots\\ f_n\end{pmatrix} = \underbrace{\left(\begin{array}{c} | \\ 0 \\ | \\ \hline c_1 \\ \vdots \\ c_k\end{array}\right)}_{b}$$

How to solve?

$$Af = b$$

How to solve?

$$Af = b$$

Normal equation

$$f = (A^T A)^{-1} A^T b$$

Assignment 2

- Poisson Blending (Matlab)