

Poisson Image Editing

Jack Breeding, Ehsan Ahmadi, Sek Cheong

University of Wisconsin, Madison

{jbreeding, eahmadi2, sucheong}@wisc.com

February 16, 2019

Overview

1 Motivation

- Make local changes during image editing can be difficult
- Traditional methods take a lot of time and leave noticeable seams
- Laplacian pyramid approach can blend the edges but have color mismatch

2 Approach

- Change in gradient domain
- Reconstruct image from gradient

3 Mathematical Background

- Definition
- Basic Idea

Motivation

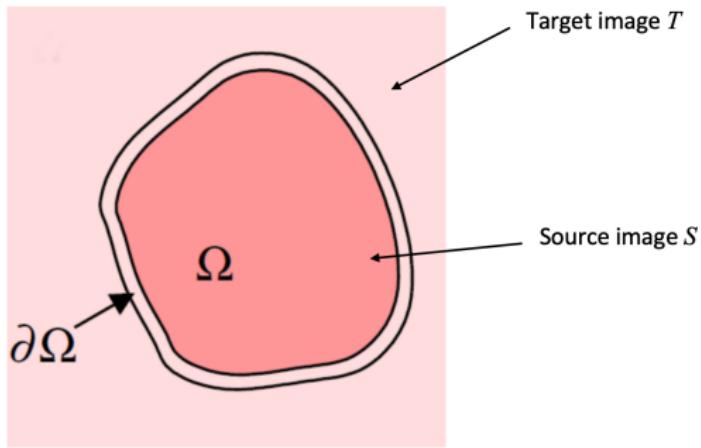
- Make local changes during image editing can be difficult
- Traditional methods take a lot of time and leave noticeable seams
- Laplacian pyramid approach can blend the edges but have color mismatch



Approach

- Change in gradient domain
- Reconstruct image from gradient

Definition



- The source image $S(x, y)$ defined over a closed region Ω
- The boundary of this region denoted as $\partial\Omega$
- The target image $T(x, y)$ defined some rectangular region in \mathbb{R}^2

Basic Idea

- Want to construct the composite image, $I(x, y)$, which should agree with $T(x, y)$ and look like $S(x, y)$
- $I(x, y)$ should exactly agree with $T(x, y)$
- $I(x, y)$ “look like” $S(x, y)$ inside Ω
- If we place directly S over T and smooth over the edges, the result maybe unacceptable, due to color mismatch

Basic Idea



source image



target image

Basic Idea



Figure: Simple cloning, background of source does not agree with the target

Basic Idea



source images



simple cloning

Basic Idea



source images



Poisson seamless cloning

How to Solve

- Make the interior of Ω “look like” $S(x, y)$
- Also want the color to match between $S(x, y)$ and $T(x, y)$

How to Solve

- Transfer the “edges” of the $S(x, y)$ to into Ω
- Computer colors inside Ω as close as possible with pixels from $T(x, y)$ surrounding Ω
- We want the gradient of the composite image, $\nabla I(x, y)$, inside Ω as close possible to $\nabla S(x, y)$
- Subject to the constraint the result must match existing values of $T(x, y)$ on the boundary $\partial\Omega$

How to solve

Mimimize:

$$\min_{I(x,y) \in \Omega} \iint_{\Omega} \|\nabla I(x,y) - \nabla S(x,y)\|^2 dx dy \quad (1)$$

Subject to:

$$I(x,y) = T(x,y), \quad (x,y) \in \partial\Omega \quad (2)$$

Euler-Lagrange Equation

A fundamental equation of **Calculus of Variations** states that, if J is defined by an integral of the form:

$$J = \int F(x, f, f_x) dx$$

Then J has a stationary value if the following differential equation is satisfied:

$$\frac{\partial F}{\partial f} - \frac{d}{dx} \frac{\partial F}{\partial f_x} = 0$$

Euler-Lagrange Equation

In our case, we have:

$$\begin{aligned} F(x, y) &= \|\nabla I(x, y) - \nabla S(x, y)\|^2 \\ &= \left(\frac{\partial I}{\partial x} - \frac{\partial S}{\partial x} \right)^2 + \left(\frac{\partial I}{\partial y} - \frac{\partial S}{\partial y} \right)^2 \end{aligned} \tag{3}$$

$I(x, y)$ that solves Equation 1 is a solution of the **Euler-Lagrange equation**:

$$\frac{\partial F}{\partial I} - \frac{d}{dx} \frac{\partial F}{\partial I_x} - \frac{d}{dx} \frac{\partial F}{\partial I_y} = 0, \quad (x, y) \in \Omega \tag{4}$$

Possion Equation

Plugging in Equation 3 into Equation 4:

$$2\left(\frac{\partial^2 I}{\partial x^2} - \frac{\partial^2 S}{\partial x^2}\right) + 2\left(\frac{\partial^2 I}{\partial y^2} - \frac{\partial^2 S}{\partial y^2}\right) \quad (5)$$

We have:

$$\nabla^2 I(x, y) = \nabla^2 S(x, y), \quad (x, y) \in \Omega \quad (6)$$

Where:

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Subject to:

$$I(x, y) = T(x, y), \quad (x, y) \in \partial\Omega \quad (7)$$

Note, an equation of the form 6 is called a **Possion equation**, and a constraint of the form of Equation 7 is called a **Dirichlet boundary condition**.

Discrete Poisson solver

- In Equation 6 the Laplacian of the new image $I(x, y)$ is equal to the Laplacian of the source image $S(x, y)$ inside Ω .
- More powerful to mim. the difference using a *guidance vector field* $\vec{S} = (S_x(x, y), S_y(x, y))$ at every pixel (x, y) , Equation 6 change to:

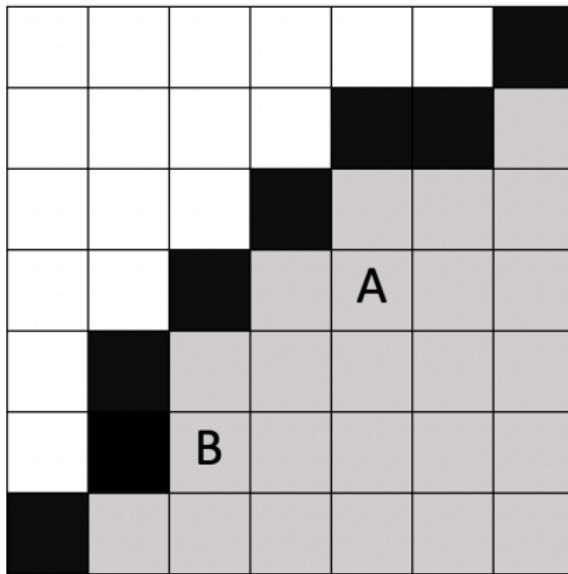
$$\nabla^2 I(x, y) = \operatorname{div} \begin{bmatrix} S_x(x, y) \\ S_y(x, y) \end{bmatrix} = \frac{\partial S_x}{\partial x} + \frac{\partial S_y}{\partial y}, \quad (x, y) \in \Omega \quad (8)$$

- Create discrete version of Equation 6 and 7
- Use 4 neighborhood operator to approximiate the Laplacian operator

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Discrete Poisson solver

- Each pixel $p = (x, y)$ in Ω generates a linear equation for unknown values in $I(x, y)$
- Two cases, for the 4-neighborhood $N(p)$:



Discrete Poisson solver

- ① $N(p) \subset \Omega$ The neighborhoods of p are fully contained in Ω and the Laplacian:

$$\begin{aligned} I(x+1, y) + I(x-1, y) + I(x, y+1) + I(x, y-1) - 4S(x, y) = \\ S(x+1, y) + S(x-1, y) + S(x, y+1) + S(x, y-1) - 4S(x, y) \end{aligned}$$

- ② $N(p) \not\subset \Omega$ Not all neighborhoods of p are contained in Ω , at least one neighborhood is in $\partial\Omega$:

$$\begin{aligned} \left(\sum_{q \in N(p) \cap \Omega} I(q) \right) + \left(\sum_{q \in N(p) \cap \partial\Omega} T(q) \right) - 4I(x, y) = \\ S(x+1, y) + S(x-1, y) + S(x, y+1) + S(x, y-1) - 4S(x, y) \end{aligned}$$

Discrete Poisson solver

- This reduces to the sparse linear systems:

$$\begin{pmatrix} 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 1 & -4 & 1 & 1 & \cdots & 0 \\ \vdots & & & & & \ddots & 0 & & \\ \vdots & & & & & & \ddots & 0 & \\ \vdots & & & & & & & \ddots & 0 \\ \vdots & & & & & & & & \ddots \\ 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ b_1 \\ b_2 \\ 0 \\ \vdots \\ \vdots \\ b_3 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

References



John Smith (2012)

Title of the publication

Journal Name 12(3), 45 – 678.

The End