Poisson Image Editing Project

Qu Yi(Albert)
The Chinese University of Hongkong, Shenzhen
Yi Qu@link.cuhk.edu.cn

Abstract

This report presents the Poisson Image Editing project, which includes the implementation and application of Poisson image editing algorithms. A detailed mathematical derivation of the algorithm is provided in a dedicated section.

1. Introduction

Poisson image editing is a technique used to seamlessly blend an object or texture from a source image into a target image. This project involves implementing the algorithm in Python and demonstrating its effectiveness with sample images.

2. Project Overview

2.1. Source Code

The primary implementation of the Poisson image editing algorithm is contained in the `poisson_image_editing.py` script. This script performs the necessary computations to blend the source and target images seamlessly. The implementation of the code has referenced Zhou's work. [2]

2.2. Images

The project uses the following images:

- `source.png`: The source image containing the object or texture to be blended.
- `target.png`: The target image into which the source image will be blended.
- `result.png`: The resulting image after applying the Poisson image editing algorithm.

2.3. Running the Code

To run the code, execute the following command in your terminal:

```
python poisson_image_editing.py -s <source image> -t <target image> [-m <mask image>]
```

Ensure that the `<source image>` and `<target image>` images are placed in the same directory with `poisson_image_editing.py`.

- If the mask image is not specified, a window will pop up for you to draw the mask on the source image:

  The green region will be used as the mask. Press 's' to save the result, press 'r' to reset.

- After the mask is defined, a window will pop up for you to adjust the mask position on the target image: The mask corresponds to the region of source image that will be copied, you can move the mask to put the copied part into desired location in the target image. **Press 's' to save the result, press 'r' to reset.**

Then the Poisson image editing process will start. The blended image will be named as `target_result.png`, in the same directory as the source image.

3. Mathematical Derivation

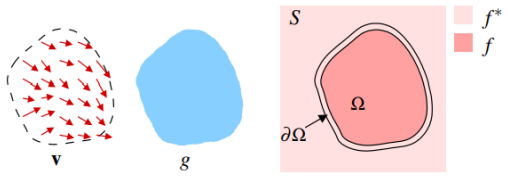


Figure 1. Guided Interpolation Sample

3.1. Problem Formulation

The goal is to blend a source image region S into a target image T such that the gradients (changes in intensity) of the source image are preserved while seamlessly fitting into the target image. This can be formulated as solving the following Poisson equation:

$$\nabla^2 f = \nabla \cdot \mathbf{v} \quad \text{in} \quad \Omega$$

where:

- ∇^2 is the Laplacian operator.
- f is the function representing the pixel intensities in the blended region.
- $\mathbf{v}$ is the vector field representing the gradients of the source image.
- Ω is the region of interest where the blending occurs.

We start with the minimization problem given by:

$$\min_f \iint_\Omega |\nabla f - v|^2 \, d\Omega \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega} \tag{1}$$

This can be interpreted as finding the function f that minimizes the difference between its gradient and a given vector field v, subject to the boundary condition f on $\partial\Omega$.

The algorithm used in this section can be found in Perez et.al's paper. [1]

3.2. Mathematical Optimization

By considering the Euler-Lagrange equation for this functional, we get:

$$\Delta f = \operatorname{div} v \quad \text{over}\, \Omega \quad \text{with} \quad f|_{\partial\Omega} = f^*|_{\partial\Omega} \tag{2}$$

Here, Δf denotes the Laplacian of f, and $\operatorname{div} v$ denotes the divergence of v.

To solve this equation numerically, we discretize the problem, leading to a system of linear equations. In the discrete case, the Laplacian operator is represented by a Laplacian matrix L, and we can write the discrete version of the equation as:

$$Lf = b \tag{3}$$

where L is the Laplacian matrix and b is the vector representing the divergence of v in the discrete setting.

In summary, the derivation follows these steps:

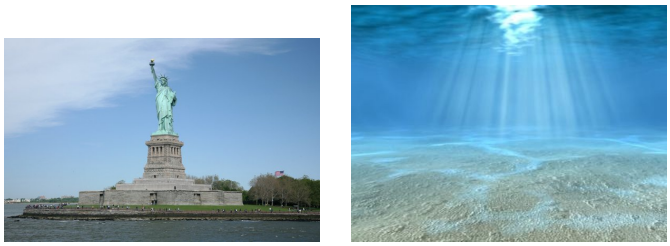
1. Formulate the minimization problem.
2. Apply the Euler-Lagrange equation to derive the PDE.
3. Discretize the PDE to obtain the linear system.

This derivation establishes the connection from the original minimization problem to the solution of a linear system involving the Laplacian matrix.

4. Results

Here are the images used and the result obtained:

4.1. Test 1



source1 target1

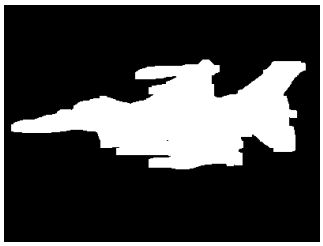
Figure 2. Test 1: Source Image and Target Image

Figure 3. Test 1: Result Image

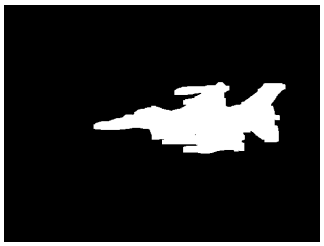
4.2. Test 2

source2 target2

Figure 4. Test 2: Source Image and Target Image



Mask 2



Target mask 2

Figure 5. Test 2: Mask and Target mask Images

Figure 6. Test 2: Result Image

4.3. Test 3

source3

target3

Figure 7. Test 3: Source Image and Target Image

Figure 8. Test 3: Result Image

4.4. Test 4

source4

target4

Figure 9. Test 4: Source Image and Target Image

Figure 10. Test 4: Result Image

5. Conclusion

The Poisson image editing algorithm effectively blends source and target images by preserving the source image gradients and fitting them seamlessly into the target image. The implementation demonstrates the practical application of this technique.

References

[1] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Transactions on Graphics (TOG)*, 22(3):313–318, 2003.

[2] Ziqi Zhou. Fast poisson image editing. `https://github.com/Trinkle23897/Fast-Poisson-Image-Editing`, 2023. Accessed: 2024-06-25.