POISSON BLENDING AND THE POISSON EQUATION

A PREPRINT

Zehao Zhang
School of Data Science
Fudan University
220 Handan Road, Shanghai 200433
20307130201@fudan.edu.cn

June 9, 2022

ABSTRACT

In this paper we will revise the concept of Poisson image blending and the Poisson equation behind. We will illustrate the algorithm for solving the Poisson equation by discretization and applying conjugate gradient on a sparse linear system. At last we will display some of the results. Codes available at https://github.com/ForeverHaibara/Poisson-Blending.

Keywords Computer Vision · Poisson Equation · Conjugate Gradient

1 Introduction

Image blending is one of the topics in computer vision, the task of which is to place two pictures together. The most direct approach is cut-and-paste, yet the result is oftentimes rigid and unsatisfying. Despite the rapid growth of deep learning has gradually overshadowed most of the classical methods, the conventional image blending strategies might still be interesting to work with. And in this paper, we will reproduce the method of Poisson blending.

2 Poisson Blending

Proposed in [2], Poisson blending is a simple image-blending technique. As the name suggest, it involves the application of Poisson equations. The key idea is simple: if we combine two images into one, then the result must be smooth and possess the features of both.

Mathematically, we can formulate the problem as follows. We treat an image as a function $F(x, y)$ on the Cartesian coordinate where $F(x, y)$ represents the corresponding pixel color or RGBA values. Now given two images, background A and foreground B, our task is to transfer the image B in some target region Ω to the same place in A. This is done by constructing a new image C such that

$$A(x, y) = C(x, y), \quad \forall (x, y) \notin \mathrm{int}(\Omega) \tag{1}$$

and

$$C = \arg\min_{C} \iint_{\Omega} \|\nabla C - \nabla B\|^2 \mathrm{d}S \quad \text{with constraint (1).} \tag{2}$$

The first equation, where the notation $\mathrm{int}(\Omega)$ stands for the interior of Ω, requires that the image remains unchanged outside and on the boundary of Ω. So one can imagine that the resulting image C is seamlessly attached to the background A. The second equation, a minimization problem, obviously suggests that $\nabla C \approx \nabla B$ given the boundary condition. As we know the edge of a digital image often corresponds to large gradients, this helps C preserve, or imitate, the edge features of B.

2.1 Guidance Field

More generally, we can replace the ∇B in the equation (2) by any ∇F, i.e.

$$C = \arg\min_{C} \iint_{\Omega} \|\nabla C - \nabla F\|^2 \mathrm{d}S \quad \text{with constraint (1).} \tag{3}$$

The given ∇F is called the guidance field. It, both literally and intuitively, means that ∇C should approximate ∇F. The solution image C is therefore *guided* by F starting from the boundary conditions (1). One of the choices of ∇F is

$$\nabla F(x,y) = \begin{cases} \nabla A(x,y) & \text{if } \|\nabla A(x,y)\| > \|\nabla B(x,y)\|, \\ \nabla B(x,y) & \text{otherwise,} \end{cases} \tag{4}$$

where we select the stronger gradient of A and B. Other different choices for $\nabla F(x,y)$ will possibly lead to different effects, like texture transferring, local coloring, etc. as reported in [2]. In the following sections we will use the generalized equation (3) together with the guidance field choice (4) rather the condition (2).

2.2 Poisson Equation

Until now, we have not yet got any clue that the problem is related to the Poisson equation. In fact, it lies in the solution to the minimization problem equation (3). If we assume $\partial\Omega$ is piecewise smooth, which is a mild condition in application, we state the following theorem.

Theorem 1. *If equation (3) has minimizer C and both C and F are twice continuously differentiable, then $\Delta C = \Delta F$ in the interior of Ω where Δ is the Laplacian operator.*

Proof. Let C be the minimizer and for any twice continuously differentiable V such that $V(x,y) = 0\ \forall (x,y) \in \partial\Omega$, we have

$$\iint_{\Omega} \|\nabla(C + tV) - \nabla F\|^2 \mathrm{d}S \geqslant \iint_{\Omega} \|\nabla C - \nabla F\|^2 \mathrm{d}S \qquad \forall t \in \mathbb{R}$$

by the definition of the minimizer. Take the difference and we obtain

$$t^2 \iint_{\Omega} \|\nabla V\|^2 \mathrm{d}S + 2t \iint_{\Omega} \nabla V^T (\nabla C - \nabla F)\, \mathrm{d}S \geqslant 0.$$

The left hand side in the inequality above is a quadratic function with respect to the variable t and can be negative as long as $\iint_{\Omega} \nabla V^T (\nabla C - \nabla F) \mathrm{d}S \neq 0$. If we denote $R = C - F$, it requires that for arbitrarily given V,

$$0 = \iint_{\Omega} \nabla V^T \nabla R \mathrm{d}S = \iint_{\Omega} \left(\frac{\partial V}{\partial x}\frac{\partial R}{\partial x} + \frac{\partial V}{\partial y}\frac{\partial R}{\partial y} \right) \mathrm{d}S.$$

Now we recall the Green's theorem that

$$\oint_{\partial\Omega} V\frac{\partial R}{\partial x}\mathrm{d}y - V\frac{\partial R}{\partial y}\mathrm{d}x = \iint_{\Omega} \left(\frac{\partial V}{\partial x}\frac{\partial R}{\partial x} + V\frac{\partial^2 R}{\partial x^2} + \frac{\partial V}{\partial y}\frac{\partial R}{\partial y} + V\frac{\partial^2 R}{\partial y^2} \right) \mathrm{d}S.$$

Hence by the fact that $V(x,y) = 0$ on $\partial\Omega$ we yield

$$0 = \iint_{\Omega} V \left(\frac{\partial^2 R}{\partial x^2} + \frac{\partial^2 R}{\partial y^2} \right) \mathrm{d}S.$$

If $\Delta R = \frac{\partial^2 R}{\partial x^2} + \frac{\partial^2 R}{\partial y^2} \neq 0$ at some interior (x_0, y_0) of Ω, then in a neighborhood of (x_0, y_0) the sign of ΔR does not change. Hence we can pick up some V such that V is positive in this neighborhood but strictly zero outside and on the boundary. This leads to a contradiction. Thus we must require ΔR is zero everywhere inside Ω and $\Delta C = \Delta F$ as desired.

□

Back to our problem, the Laplacian relation $\Delta C = \Delta F$ plus the boundary condition $C(x,y) = A(x,y)$ on $\partial\Omega$ forms an exact Poisson equation. We shall note that, although the selection of ∇F in equation (4) does not ensure twice differentiable continuity, we will still use the equation $\Delta C = \Delta F$.

2.3 Discretization

Before we dive into the discrete Poisson equation, we shall clarify the discretized definition of each term. In the discretized formulation of the model, images (or image functions) are only observed at the integer points, $\mathbb{Z} \times \mathbb{Z}$. And integer coordinates (i_1, j_1) and (i_2, j_2) are neighboring if and only if $|i_1 - i_2| + |j_1 - j_2| = 1$. And we write $\mathrm{N}(x, y)$ to stand for the set of four coordinates that are neighboring to (x, y),

$$\mathrm{N}(x, y) = \{(x-1, y),\ (x+1, y),\ (x, y-1),\ (x, y+1)\}. \tag{5}$$

One would question whether there are not four neighbors in the case where the pixel lies on the border of the image, but borders can be handled by padding the image with the same RGB values.

Discrete target region $\hat{\Omega}$ is defined by all the integer points inside, $\hat{\Omega} = \Omega \cap (\mathbb{Z} \times \mathbb{Z})$. The boundary is thus defined by points in $\hat{\Omega}$ but has $0 \sim 3$ neighbors in $\hat{\Omega}$, that is $\partial\hat{\Omega} = \left\{(x, y) :\ (x, y) \in \hat{\Omega},\ \mathrm{Card}\{\hat{\Omega} \cap \mathrm{N}(x, y)\} \leqslant 3\right\}$. The corresponding interior is given by the difference set, $\mathrm{int}(\hat{\Omega}) = \hat{\Omega} \setminus \partial\hat{\Omega}$.

For discrete gradients, we define $\nabla F_{(i_1,j_1),(i_2,j_2)}$ by $\nabla F_{(i_1,j_1),(i_2,j_2)} = F(i_2, j_2) - F(i_1, j_1)$ when (i_1, j_1) and (i_2, j_2) are neighbors.

2.4 Discrete Poisson Equation

The key of the discrete Poisson equation is given by the central difference formula of order 2,

$$\Delta F(x, y) \approx \frac{-4F(x, y) + F(x-h, y) + F(x+h, y) + F(x, y-h) + F(x, y+h)}{h^2}. \tag{6}$$

When we select $h = 1$, it becomes the combination of neighboring values and itself,

$$\Delta F(x, y) \approx \frac{1}{h^2}\left(-4F(x, y) + \sum_{(i,j)\in \mathrm{N}(x,y)} F(i, j)\right). \tag{7}$$

The same also applies on ΔC. In the Poisson equation $\Delta C = \Delta F$, the denominator h^2 cancels out. Therefore, for all (x, y) in $\mathrm{int}(\hat{\Omega})$ we have

$$-4C(x, y) + \sum_{(i,j)\in \mathrm{N}(x,y)} C(i, j) = -4F(x, y) + \sum_{(i,j)\in \mathrm{N}(x,y)} F(i, j). \tag{8}$$

This forms a linear system with all $C(x, y)$ where $(x, y) \in \mathrm{int}(\hat{\Omega})$. Values $C(x, y)$ on the boundary $\partial\hat{\Omega}$ are excluded since they have known values $C(x, y) = A(x, y)$. Therefore we can rewrite it as

$$4C(x, y) - \sum_{(i,j)\in \mathrm{N}(x,y)\setminus\partial\hat{\Omega}} C(i, j) = 4F(x, y) - \sum_{(i,j)\in \mathrm{N}(x,y)} F(i, j) + \sum_{(i,j)\in \mathrm{N}(x,y)\cap\partial\hat{\Omega}} A(i, j), \quad \forall (x, y) \in \mathrm{int}(\hat{\Omega}). \tag{9}$$

However, what we actually know is the guidance field, a gradient, ∇F rather the primitive F. We can fix (9) by replacing $F(x, y) - F(i, j)$ by $\nabla F_{(x,y),(i,j)}$ and finally yield

$$4C(x, y) - \sum_{(i,j)\in \mathrm{N}(x,y)\setminus\partial\hat{\Omega}} C(i, j) = \sum_{(i,j)\in \mathrm{N}(x,y)} \nabla F_{(i,j),(x,y)} + \sum_{(i,j)\in \mathrm{N}(x,y)\cap\partial\hat{\Omega}} A(i, j), \quad \forall (x, y) \in \mathrm{int}(\hat{\Omega}). \tag{10}$$

The linear system (10) is sparse and symmetric. Sparsity is due to the fact that each equation only involves at most four neighboring pairs, while symmetricity is because neighboring pairs are commutative. Further, the system is positive semidefinite guaranteed by the Gerschgorin circle theroem.

2.5 Conjugate Gradient

Considering that the Poisson system is large and sparse, one would often turn to sparse, iterative methods rather a dense, accurate one. Also, thanks to the characteristic of human eyes, small perturbation on the images does not really affect much. Hence time-saving iterative methods at the expense of accuracy is very attractive. We suggest the conjugate

gradient [1], a widely applied method for positive semidefinite matrices that has the following iterative schedule when solving for $Ux = b$.

$$\begin{aligned} \alpha_k &= \frac{r_k^T r_k}{p_k^T U p_k} \\ x_{k+1} &= x_k + \alpha p_k \\ r_{k+1} &= r_k - \alpha U p_k \\ \beta_{k+1} &= \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} \\ p_{k+1} &= r_{k+1} + \beta_{k+1} p_k \end{aligned} \tag{11}$$

The initial values are $x_0 = 0$ and $r_0 = p_0 = b$. The approximant x_k will converge to the exact solution x_*.

3 Experiments

3.1 Time

The time for solving the Poisson equation, namely the sparse linear system is plotted in figure 1. Here we have used 200 iterations and the data points are averaged on 12 runs with 95% confidence interval marked out. As far as we are concerned, the computation time is almost linear with the number of nonzero entries in the matrix. Recall that the nonzero entries are around five times the number of the matrix length! Only 3 seconds is needed to solve a system with around 2×10^5 pixels on a 16GB i7-1065G7 CPU, an equivalence to 10^6 nonzero entries.

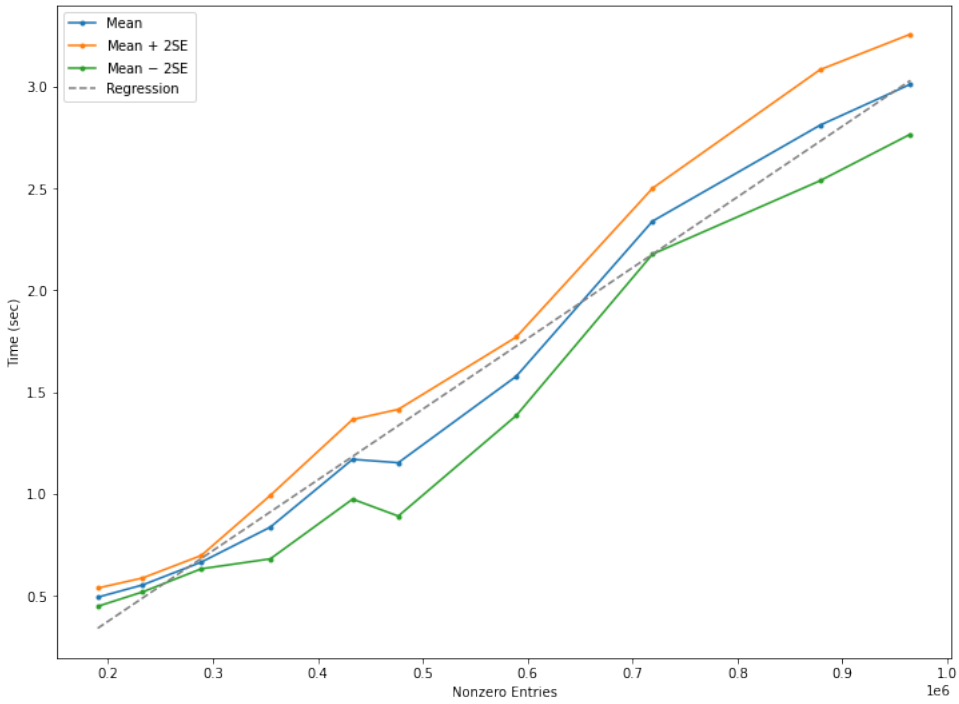

Figure 1: Execution time for solving linear systems with different sizes averaged on 12 runs each.

3.2 Results

Figure 2 illustrates two examples of Poisson blending and comparisons with trivial cut-and-paste strategy. In the first example, a moon is added to the forest-like background. Cut-and-paste method has preserved a stiff boundary that overshadows the woods, while the method of Poisson blending successfully place the moon on the back and has further rendered the clouds upwards.

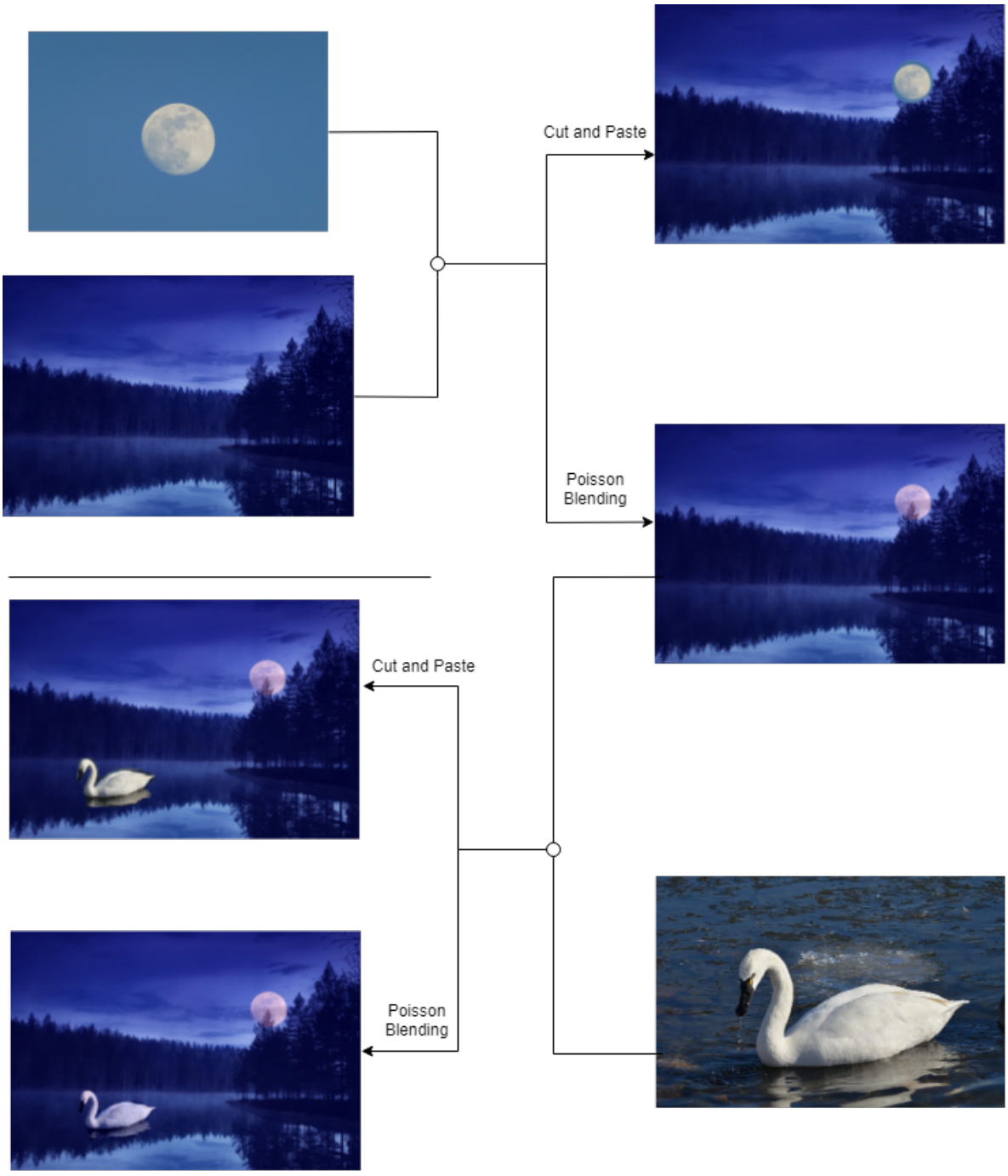

Figure 2: Two comparisons between cut-and-paste and Poisson blending. Above: Adding a moon. Below: Adding a swan.

Besides handling boundaries smoothly, Poisson blending has shown strong ability in color transferring. In the second example of Figure 2 we move a swan and its reflection in the water on to the river. Compared with the cut-and-paste, Poisson blending colors the swan that matches the nearby environment and the dim moonlight. Figure 3 is another example where the balloon we transplant is recolored in the style of an oil paint.

Moreover, we find by experience that when the background has sharp gradient and complicated textures, the foreground image tends to transparent. On the one hand, this feature can be applied to draw something smoky like rainbows in Figure 4. On the other, it indicates challenges when dealing with solid entities.

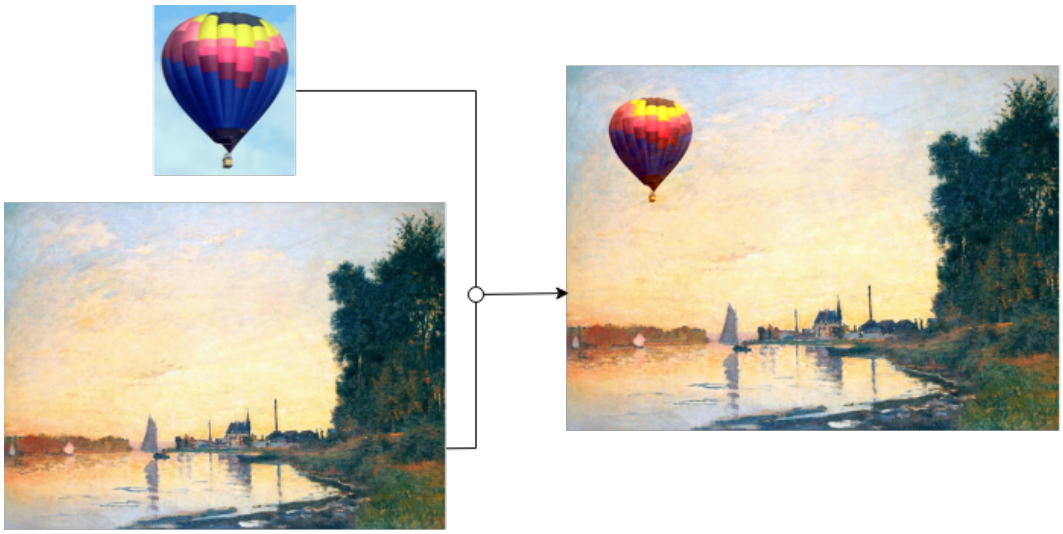

Figure 3: Place a balloon onto an artwork.

Figure 4: Add a rainbow to a waterfall. The rainbow is firstly cropped out from the foreground.

4 Conclusions

Poisson blending is a dedicate method of seamlessly combining two images that apparently outperforms the cut-and-paste strategy. Yet Poisson blending is not perfect. As illustrated above, it faces seemingly inevitable transparency when placing targets onto sharp backgrounds. Also, difficulties like shadows and shadings are hard to overcome. Still, we shall here be content with the charm of Poisson blending, artistically and mathematically.

Acknowledgements The author is grateful to Meiyue Shao, who delivered two terms of insightful numerical algorithm courses that inspires this work. All images in this work are collected from the Internet.

References

[1] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

[2] J. Martino, Gabriele Facciolo, and Enric Meinhardt-Llopis. Poisson image editing. *Image Processing On Line*, 5:300–325, 11 2016.