Assignment-matlab 2

Haowei Kan

September 28, 2020

1 Introduction

Image editing tasks concern either global changes or local changes confined to a selection. Here we are interested in achieving local changes. With classic tools to achieve that, changes in the selected regions result in visible seams.

In this experiment, we will introduce a generic interpolation machinery based on solving Poisson equations, and a cloning tool, which is an application of this machinery for seamless editing of image regions will be shown.

2 Algorithm

2.1 Generic machinery

We know that slow gradients of intensity can be superimposed on an image with barely noticeable effect, and a scalar function on a bounded domain is uniquely defined by its values on the boundary and its Laplacian in the interior. So, given methods for crafting the Laplacian of an unknown function over some domain, and its boudary conditions, the Poisson equation can be solved numerically to achieve seamless filling of that domain. When applied to the image editting, it can be replicated independently in each of the channels of a color image.

Solving the Poisson equation also has an alternative interpretation as a minimization problem: it computes the function whose gradient is the closest, in the L_2-norm, to some prescribed vector field – the *guidance* vector field – under given boundary conditions inwards, while following the spatial variations of the guidance filed as closely as possible.

2.2 Discrete Poisson solver

For discrete images, the problem can be discretized naturally using the underlying discrete pixel grid. Let S be the set of all the pixels of an image, Ω be a subset of S with boundary $\partial\Omega$. For each pixel p in S, let N_p be the set of its 4-connected neighbors which are in S, and let $\langle p, q\rangle$ denote a pixel pair s.t. $q \in N_p$. The boundary of Ω is now $\partial\Omega = \{p \in S \setminus \Omega : N_p \cap \Omega \neq \emptyset\}$. Let f_p be the value of f at p. The task is to compute the set of intensities $f|_\Omega = \{f_p, p \in \Omega\}$.

Then the discrete optimization problem will be:

$$\min_{f|_\Omega} \sum_{\langle p,q\rangle \cap \Omega \neq \emptyset} (f_p - f_q - v_{pq})^2, \text{ with } f_p = f_p^*, \text{ for all p} \in \partial\Omega \tag{1}$$

where v_{pq} can be thought as the guided vector field in discrete problem.

Its solution satisfies the following simultaneous linear equations:

$$\text{for all } p \in \Omega, \ |N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial\Omega} f_q^* + \sum_{q \in N_p} v_{pq} \tag{2}$$

So we can calculate unknown values f_p on Ω by solving these linear equations.

2.3 Mixed seamless cloning

To apply the Poisson image editting to seamless cloning, we can think the each channel of destinated area on the background is the unknown set Ω that need to be computed by discrete poisson solver. And the v_{pq} is related to the cloning area taken from source image.

So all we need to determine for the linear equations is the values of v_{pq}. In this experiment, we choose the mixing gradients to solve the problem, which is

$$v_{pq} = \begin{cases} f_p^* - f_q^* & \text{if } |f_p^* - f_q^*| > |g_p - g_q| \\ g_p - g_q & \text{otherwise} \end{cases} \tag{3}$$

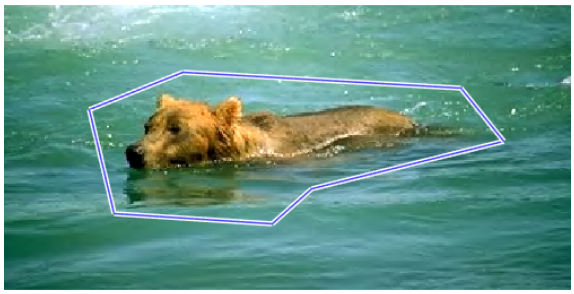
for all $\langle p, q\rangle$, where g_p is the value of pixel p in source image.

2.4 Real-time cloning

In this experiment, we are required to acheive real-time cloning when moving the destinated area on background. Therefore, we need to consider decreasing the run-time of our algorithm. Notice that the coefficient matrix will be unchanged when moving the destinated area if we don't drag it out of the background. So we can store the coefficient matrix and reuse it in real-time computation. Here are some further measures we took:

1. use the sparse matrix to store the coefficients of the linear equations
2. pre-decomposite the coefficient matrix through Cholesky method

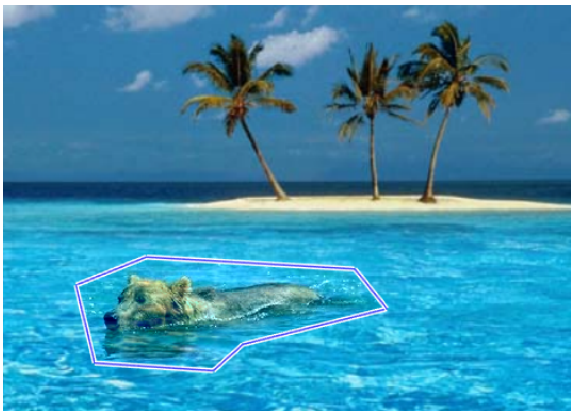
3 Result



(a) source image

(b) mixed seamless cloning 1

(c) mixed seamless cloning 2



(d) mixed seamless cloning 3

4 Why Cholesky method

Notice that the coefficient matrix we get is a real symmetrix positive definite matrix. So the Cholesky method is the best choice. And here is a table of the range of runtime of LU, QR and Cholesky method for a 15845×15845 sparse matrix in our real-time calculation.

method	LU	QR	Chol
range of runtim(s)	(0.008, 0.011)	(0.019, 0.022)	(0.004, 0.006)

This verifies that Cholesky method for this problem gives the best performance.

A Code

All the code of this assignment can be downloaded from https://github.com/mathendy/MS-USTC/tree/master/Code%20Training/matlab-2_assignment