CS276 Fall 2019

Computational Photography

Instructor: Jingyi Yu

Assignment 2: Poisson Image Editing

Due October 29th 11:59 p.m.

In this project, you will need to implement two recent papers on Poisson Image Editing. Part of the project involves using Dijkstra's shortest path algorithm. This is a LONG project, so start early.

Image composition is the process of creating a new image by pasting an object or a region from a source image onto a target image. Poisson image editing that was introduced in class has been an effective approach for seamless image composition. As the first part of this project, you will need to provide a user-friendly system for seamless image composition.

Part I

1. Your program should able to display the source and the target image (and a mask image if necessary). Your program should clone the area of interest from the source image (denoted by the mask image) onto the target image at the specified offset, displaying the final composited image and saving the result as the specified output filename. For the mask, you can obtain one with GUI, or manually create it.

2. You will need to use the Poisson Image Editing technique described in class to compute the final composition. Basically, you need to first form a composited gradient field and then solve for the image that has the closest gradient field to the

composited one. This requires solving a large yet sparse matrix system, as discussed in class. You can either use matlab or the attached C++ matrix library.

Part II

3. As for the second part, you will need to implement the optimal boundary condition for Poisson image editing, as is described in the Drag-and-Drop Pasting paper. You program should specify both the exterior boundary and the interior boundary. You should then implement a shortest path algorithm (Dijkstra) for computing the optimal boundary.

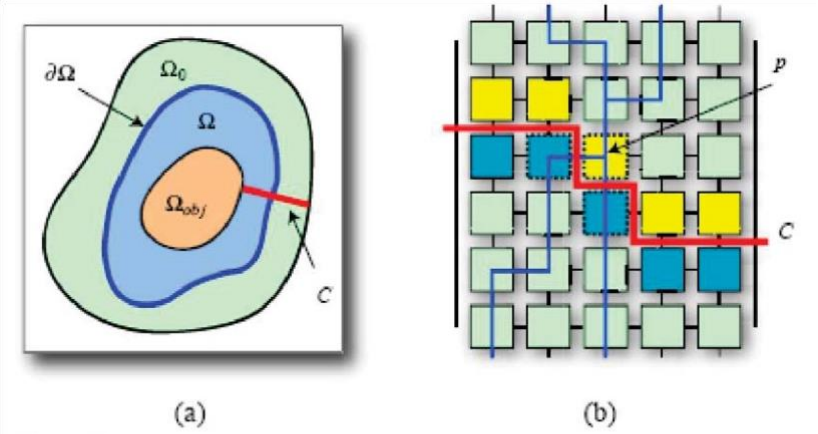

Figure 3: Boundary optimization. (a) The region of interest Ω_0 is pasted onto the target image, which completely encloses the object of interest Ω_{obj}. The optimized boundary $\partial\Omega$ lies inside $\Omega_0 \backslash \Omega_{obj}$. The cut C is shown in red. (b) Zoom-in view of the cut C. The yellow and blue pixels are on different sides of C. The dashed yellow pixel p is adjacent to two blue pixels. Two shortest paths, shown as blue lines, are simultaneously computed.

3.1. You will first need to build up a graph G that consists of pixels within the band while the edges represent 4-connectivity relationships between neighboring pixels.

3.2. The algorithm starts as an iterative process. Initialize the contour to be the exterior boundary, and compute the optimal k, which can be found as

$$k = \frac{1}{|\partial\Omega|} \sum_{p \in \partial\Omega} (f_t(p) - f_s(p))$$

3.3. You then need to find the improved contour given k. To do so, you then need to define a cut C by breaking the band connectivity in G. C can be defined as the shortest straight line segment among all pixels pairs connecting the exterior and the interior boundary by computing the Euclidian distance. The line segment C can then be rasterized into a pixel list in a 2D image plane using basic line rasterization algorithms such as the Bresenham's algorithm. Sample code is provided in the lecture notes on how to rasterize a line using Bresenham's.

3.4. For each pixel on the cut on one side, compute the shortest paths to from the pixel to its adjacent pixel on the other side of the cut using Dijkstra's algorithm.
3.5. Repeat the previous computation for all pixels on the one side of the cut C, and get a set of Path. The optimized boundary is assigned as one that gives the globally minimum cost.
3.6. Once the contour is determined, use the same code from Part I to do Poisson Image Editing.

Material needed to submit:

1. A report containing your results and explanations of your implemented algorithms. Upload your report to Gradescope.
2. Your C++ or Matlab project. (I should be able to run you code and reproduce your result)
3. Minimum requirement: Create a jagged mask manually (not a rectangular shape) where it hits the boundary of the target image (for including corner cases). All codes should be zipped and be sent to cp2020fall@126.com with email subject "CS276_P2_name_ID"(same for the .zip name format).

References

1. Poisson image editing, P`EREZ, P., GANGNET, M., AND BLAKE, A. SIGGRAPH 2003.
2. Drag-and-drop Pasting. J. Jia, J. Sun, C-K. Tang, and H-Y. Shum. SIGGRAPH 2006.