CS 633 Computational Geometry Fall 2019

Voronoi Stippling

Prof. Jyh-Ming Lien

The goal of this assignment is to deepen your understanding on various implementations of 2-d Voronoi Diagram. You are given an implementation of 2-d Voronoi Diagram using well-known image-based wave propagation method. You task is to (1) understand the implementations and (2) improve the provided implementation.

What to submit: You need to turn in a report in LaTeX (see the template in report folder). Your report should include two main sections: a summary of what the code does, and your improvement. In the second section, you should include all the example outputs (visual and/or statistical results). In the last section you should report known bugs, and known limitations.

How to submit: Zip your entire folder including code and example input and output in a single file called "your_net_id_PA02.zip". For example, my GMU net id is jmlien, so my submission will be jmlien_PA02.zip.

Due: Oct 30, 2019. At 11:59 pm.

1 Part 1: Understand the implementations (40 pts)

1.1 What should you do?

Your goal is to get a full grasp of what the code does. Your summary should provide algorithms for computing Voronoi diagram, Centroidal Voronoi tessellation (CVT), and stippling methods.

Hint 1: The implementation is based on the paper by Secord, Adrian. "Weighted voronoi stippling." Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering. ACM, 2002. It is highly recommend that you read the paper first.

Hint 2: To compile hedcuter code, please use cmake or the solution file in folder *hedcuter/code/vc_files*. This code requires OpenCV 4.0. Installing OpenCV is straight forward on OSX and Linux. By default, it requires 64 bits installation of OpenCV on Windows. The solution file also uses two environment variables called "OPENCV_INCLUDEDIR" and "OPENCVX64_LIBRARYDIR" the point to the include and library folders on the system. Therefore, make sure that you have those variable defined before you compile. You can consult OpenCV documents (*docs.opencv.org*) or (*https://www.learnopencv.com/install-opencv-4-on-windows/* if you encounter problems.

2 Part 2: Improve "hedcuter" code (60 pts)

2.1 What should you do?

Provide at least three improvements (each will worth 20 points) to the hedcuter code. Below are some possible improvements that you can do. In your report, you should show the improvement

either visually or/and statistically for timing/performance results using images of your own, i.e. do not use the images in hedcuter/images. Extra bonus of 20 points will be given for an additional improvement beyond the 3rd improvement.

1. Improve the distribution of the disks to avoid unnatural clustering of the disks. One idea is to use higher image resolution (using subpixels) for computing the centroids of Voronoi cells.

2. Improve the computation efficiency. One way of doing this is via the implementation of Fortune's algorithm. Another way is GPU. You can try the method by Hoff III, Kenneth E., et al. "Fast computation of generalized Voronoi diagrams using graphics hardware." Proceedings of the 26th annual conference on Computer graphics and interactive techniques, 1999. The implementation should be pretty simple if you know OpenGL. Quasi-Newton Method

3. Add functionality to generate colorful disks. For example, you can implement functions that are not available in hedcuter code but provided in the voronoi code.

4. Using feature extraction, such as lines, to get better looking results.

References