CIE 6004(Fall 2019) Assignment 3

Author

Name: Jian Zhang
Student ID: 219012058
MSc Program: Data Science

November 21, 2019

Contents

1 Introduction

This report is about the Assignment 3 of "Image Processing and Computer Vision". We mainly discuss texture synthesis(A.A. Efros and T.K. Leung, Texture Synthesis by Non-parametric Sampling, 1999), image segmentation with k-means method and face detection(Viola-Jones Face Detector). Details of algorithm are mostly based on paper, while for some parts of the implementation, I get ideas from GitHub. And my code is based on Python.

The report includes the description of the problems, the algorithm to solve them and the results of the resolution.

2 Exercise

2.1 Texture

Implement the texture sythesis algorithm based on 'Texture Synthesis by Non-parametric Sampling, A.A. Efros and T.K. Leung, 1999'. Use the image 'brick.png'(100 * 100) to synthesize a 1000 * 1000 image.

Algorithm details from https://people.eecs.berkeley.edu/ efros/research/NPS/alg.html

Consider the patch size, we know that n = 88 is the solution for best synthesis. However, 88 cannot be divisible by 1000, and we can synthesize an image slightly bigger than 1000 * 1000 and crop it to 1000 * 1000.

To generate a synthesized image of 1000 * 1000 is quite slow. We can have techniques to speed up the computation of the matching score in every searching. Suppose that the P head is consist of the patch to be filled and its some borders, and E is the example image. To compute the matching score (the moving square difference) more efficiently, we can do it using 'valid' convolutions,

$$S = \hat{P}.^2 * ones(size(E)) + E.^2 * ones(size(\hat{P})) - 2\hat{P} * E$$

Figure 1: 'valid' convolutions

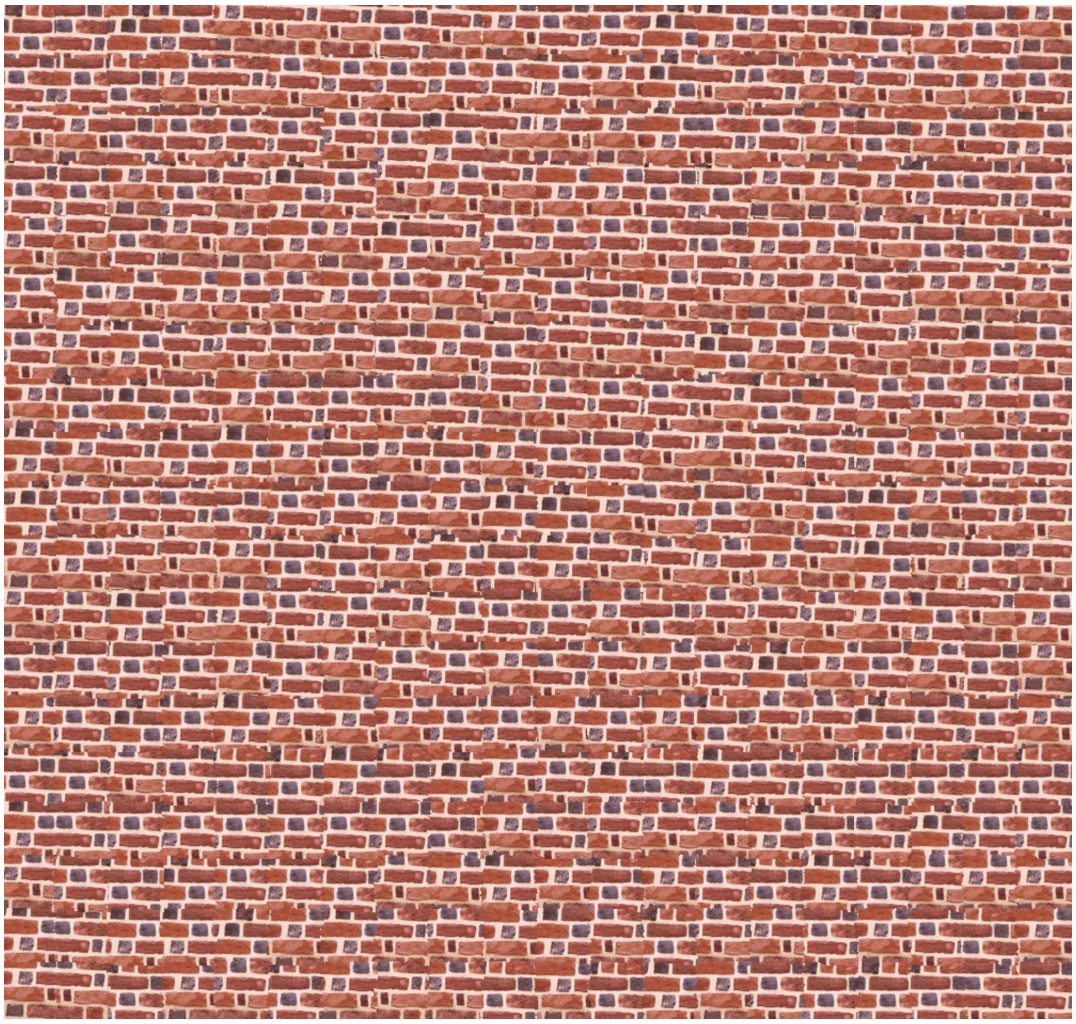

Figure 2: synthesis brick image

2.2 Segmentation

Use K-means clustering algorithm for image segmentation. We use (r, g, b, x, y) as feature vector for every pixel.

We test code with the image tanglang.jpg or dayun.jpg, and choose 5 segments, such as sky, water, buildings and trees.

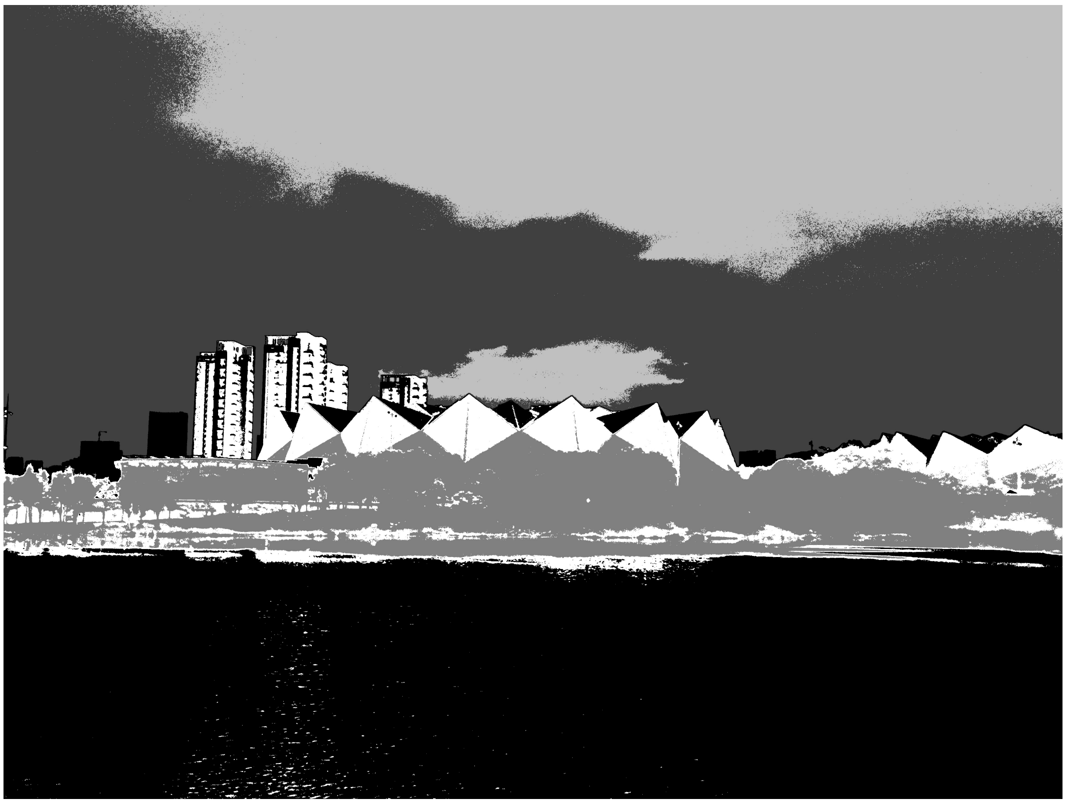

Figure 3: K-means Dayun

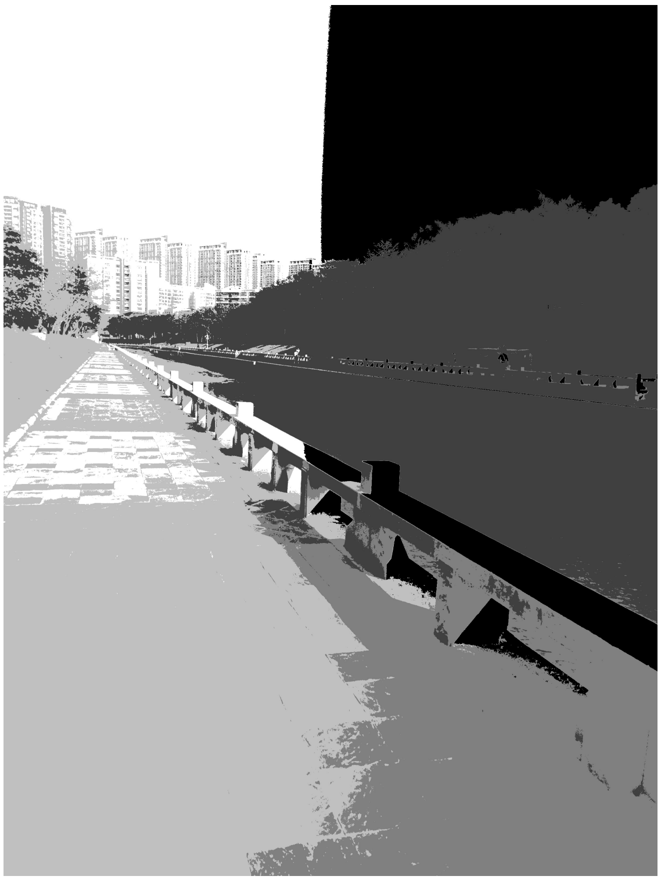

Figure 4: K-means Tanglang

2.3 Face Detection

Implement a simple Viola-Jones Face Detector:(a) First, compute enough candidate Haar features on positive dataset lfw1000 and nagative dataset nonface, and integral image can improve efficiency.(b) Select features by AdaBoost. We use AdaBoostClassifier from sklearn.ensemble (c) Further construct a cascaded classifier and test on Solvay.jpg or Big3.jpg.

Implement a function to generate haar features, here, we create 56,000 features for each 64 * 64 size image.

Based on the function, we create features for train set and test set. For human face, we use lfw1000 dataset as positive images(all the faces are in 64 * 64 size), and we set the label of these images one.

Then, we build negative images for training. We select some images from nonface dataset, and randomly pick 64 * 64 windows from those images. For these images in 64 * 64 size, we create features for them and set the label of these images zero.

We train the data with AdaBoostClassifier(default parameters) from sklearn.ensemble. Then we predict on the test image. It turns out that we get high accuracy in test set.

The following figure is the feature importance of model, we construct a cascaded classifier based on important features.

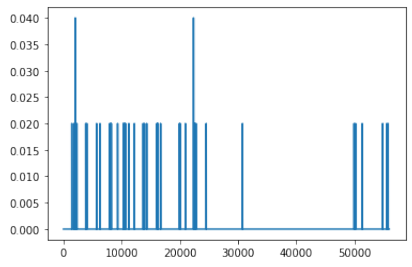

Figure 5: Feature Importance of AdaBoostClassifier Model

For cascaded classifier, I fit adaboost model with 2, 10, 50, 200, 2000 features and finally with 56,000 features.

We apply cascaded classifier on 'Solvay.jpg' and 'Big3.jpg'. For each image, we stride windows in the image, if the window is classified as a face, then we save the location.